

Kernel Methods in Computer-Aided Constructive Drug Design

by

William W.L. Wong

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2009

©William W.L. Wong 2009

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

A drug is typically a small molecule that interacts with the binding site of some target protein. Drug design involves the optimization of this interaction so that the drug effectively binds with the target protein while not binding with other proteins (an event that could produce dangerous side effects). Computational drug design involves the geometric modeling of drug molecules, with the goal of generating similar molecules that will be more effective drug candidates. It is necessary that algorithms incorporate strategies to measure molecular similarity by comparing molecular descriptors that may involve dozens to hundreds of attributes. We use kernel-based methods to define these measures of similarity. Kernels are general functions that can be used to formulate similarity comparisons.

The overall goal of this thesis is to develop effective and efficient computational methods that are reliant on transparent mathematical descriptors of molecules with applications to affinity prediction, detection of multiple binding modes, and generation of new drug leads. While in this thesis we derive computational strategies for the discovery of new drug leads, our approach differs from the traditional ligand-based approach. We have developed novel procedures to calculate inverse mappings and subsequently recover the structure of a potential drug lead. The contributions of this thesis are the following:

1. We propose a vector space model molecular descriptor (VSMMD) based on a vector space model that is suitable for kernel studies in QSAR modeling. Our experiments have provided convincing comparative empirical evidence that our descriptor formulation in conjunction with kernel based regression algorithms can provide sufficient discrimination to predict various biological activities of a molecule with reasonable accuracy.

2. We present a new component selection algorithm KACS (Kernel Alignment Component Selection) based on kernel alignment for a QSAR study. Kernel alignment has been developed as a measure of similarity between two kernel functions. In our algorithm, we refine kernel alignment as an evaluation tool, using recursive component elimination to eventually select the most important components for classification. We have demonstrated empirically and proven theoretically that our algorithm works well for finding the most important components in different QSAR data sets.
3. We extend the VSMMD in conjunction with a kernel based clustering algorithm to the prediction of multiple binding modes, a challenging area of research that has been previously studied by means of time consuming docking simulations. The results reported in this study provide strong empirical evidence that our strategy has enough resolving power to distinguish multiple binding modes through the use of a standard k -means algorithm.
4. We develop a set of reverse engineering strategies for QSAR modeling based on our VSMMD. These strategies include:
 - (a) The use of a kernel feature space algorithm to design or modify descriptor image points in a feature space.
 - (b) The deployment of a pre-image algorithm to map the newly defined descriptor image points in the feature space back to the input space of the descriptors.
 - (c) The design of a probabilistic strategy to convert new descriptors to meaningful chemical graph templates.

The most important aspect of these contributions is the presentation of strate-

gies that actually generate the structure of a new drug candidate. While the training set is still used to generate a new image point in the feature space, the reverse engineering strategies just described allows us to develop a new drug candidate that is independent of issues related to probability distribution constraints placed on test set molecules.

ACKNOWLEDGEMENT

I would like to express my deepest thanks to my supervisor, Professor Forbes J. Burkowski, for his thoughtful guidance, continual encouragement and support during the completion of this thesis. Without his help and his time devoted, this thesis would not have been done.

I would like to express my appreciation to Professors Kristin Bennett, Ming Li, Brendan McConkey and Mu Zhu for their careful reading of this thesis, and their valuable suggestions and insight.

I wish to give my sincere gratitude to my parents, Diana Luk and Richard Wong, for their endless support. I would like to express my special thanks to my parents in law, Pik Wa Lo and John Fung, for their help and support.

Special thanks also go to professors and staff in the David R. Cheriton School of Computer Science. I wish to thank Dr. Kevin Lanctot for his helps in improving my drug discovery knowledge. I also want to thank all my friends at Waterloo especially Michael Kwok, James She and Gary Yeung who made my stay here so enjoyable.

Lastly, but most importantly, I would like to thank my beloved wife Zeny and my son Jeremy. They are powerful source of energy and inspiration. Zeny, I thank you so much for your encouragement, help and being always with me.

Contents

List of Tables	xiii
List of Figures	xvi
1 Introduction	1
1.1 Drug Discovery	1
1.2 Computer-Aided Drug Design	3
1.2.1 Structure-Based Design	5
1.2.2 Ligand-Based Design	6
1.3 Contributions of the thesis	7
1.3.1 Contributions to the traditional ligand-based approach . . .	8
1.3.2 Contributions to improve the traditional ligand-based approach	13
2 Molecular Modeling and Computational Modeling	19
2.1 Molecular Modeling	19
2.2 Computational Modeling	21

2.2.1	Quantitative Structure-Activity Relationship	21
2.2.2	Historical Background	22
2.2.3	Deriving a QSAR Equation: Kernel Methods	23
3	The Vector Space Model Molecular Descriptor	36
3.1	The vector space model molecular descriptor	37
3.2	VSM and VSMMD compared	39
3.3	Analysis based on VSMMD	41
3.4	Data Fusion for VSMMD	42
3.5	Experimental Results	43
3.5.1	Data	43
3.5.2	Implementation details	44
3.5.3	QSAR results	45
3.6	Discussion	46
3.7	Conclusion	51
4	Component Selection	53
4.1	Introduction	53
4.1.1	Filter Methods	55
4.1.2	Wrapper Methods	56
4.1.3	Other Methods	57
4.1.4	Component Selection for Computer-Aided Drug Design	58

4.2	Kernel Alignment Component Selection (KACS)	59
4.2.1	Kernel Alignment	59
4.2.2	SVM based Recursive Component Elimination (RCE)	60
4.2.3	Kernel Alignment Component Selection Algorithm	61
4.2.4	Theoretical Analysis	63
4.2.5	Empirical Analysis	78
4.2.6	Random Subspace Kernel Alignment Component Selection (RSKACS)	79
4.3	Experimental Results - Experiment 1:	
	Binary Response QSAR Data	82
4.3.1	Data	82
4.3.2	Implementation Details	83
4.3.3	Results and Discussion	83
4.4	Experimental Results - Experiment 2:	
	Continuous Response QSAR Data	88
4.4.1	Data Set	88
4.4.2	Results and Discussion	89
4.5	Conclusion	91
5	Multiple Binding Modes	93
5.1	Introduction	93
5.2	The kernel k -means algorithm	94

5.3	Experimental Results	97
5.3.1	Data	97
5.3.2	Implementation Details	100
5.3.3	Test Results	100
5.3.4	Discussion	102
5.4	Conclusion	104
6	Inverse QSAR	105
6.1	Introduction	107
6.2	VSMMD Inverse-QSAR Approach	109
6.2.1	Vector Space Model Molecular Descriptor (VSMMD)	109
6.2.2	Designing descriptor image point in feature space	110
6.2.3	The Pre-Image Problem	117
6.2.4	The Need for a Nonlinear Kernel	122
6.2.5	Recovering the molecule	123
6.2.6	Reversible VSMMD	124
6.3	Experimental Results	134
6.3.1	Data	134
6.3.2	Implementation details	136
6.3.3	Verification of the Inverse Mapping - Test Result	136
6.3.4	Inverse-QSAR Test Results	138
6.3.5	Discussion	140
6.4	Conclusion	143

7 Conclusions	146
Bibliography	149

List of Tables

3.2.1 General format of the fragment dictionary (A_T =Atom Type, B_T =Bond Type)	40
3.5.2 σ values derived using cross-validation. TypeID is the fragment type ID defined in Table 3.2.1	46
3.5.3 r^2 statistics using SVR on the testing set	47
3.6.4 Comparison of the residual statistics r^2 comparing the VSMMD descriptor set and the Cerius2 2.5D descriptor set (both using SVR) .	48
3.6.5 Comparison of the residual statistics r^2 comparing the VSMMD descriptor set using SVR and the various descriptor sets reported by Sutherland <i>et al.</i> using PLS	49
3.6.6 Comparison of the residual statistics r^2 comparing the VSMMD with the graph kernel and the pharmacophore kernel (all using SVR) (*supplied program failed to provide results)	51
4.2.1 UCI data sets	78
4.4.2 r^2 statistics using SVR on the testing set	90

5.3.1 Free Energy of Binding (FEB) of ten p38 MAP Kinase Inhibitors reported by Hauser <i>et al.</i> [41]	99
5.3.2 <i>k</i> -means clustering result for anilinopyrazole data based on 100 runs.	101
5.3.3 <i>k</i> -means clustering result for p38 MAP Kinase Inhibitors data based on 100 runs.	102

List of Figures

1.1.1	Drug Discovery Process	2
1.2.2	General steps involved in ligand-based drug design.	7
1.3.3	Kernel-based traditional ligand-based drug design.	12
1.3.4	Our ligand-based constructive drug design approach.	17
2.1.1	An example of three-point pharmacophore.	22
2.2.2	The implicit function ϕ maps points in the input space over to the feature space.	25
2.2.3	An example of the Support Vector Machine.	32
3.1.1	Labels for atoms and bonds in a molecule	38
3.1.2	Processing steps for the VSMMD.	39
3.6.3	Location of highly ranked descriptor (circled) in selected COX-2 molecules.	50
4.2.1	KACS Results for 4 different UCI data sets	80
4.3.2	The results of the KACS Algorithm on the HIA data set	84

4.3.3	The leave-one out error generated by the KACS, the RSKACS and the SVMRCE algorithm on the HIA data set	85
4.3.4	The results of the KACS Algorithm on the Pgp data set	85
4.3.5	The leave-one out error generated by the KACS, the RSKACS and the SVMRCE algorithm on the Pgp data set	86
4.3.6	The results of the KACS Algorithm on the TdP data set	87
4.3.7	The leave-one out error generated by the KACS, the RSKACS and the SVMRCE algorithm on the TdP data set	88
5.3.1	Type A and Type B compound of anilinopyrazole, a CDK2 inhibitor [85]	98
5.3.2	Two different binding modes reported of anilinopyrazole, a CDK2 inhibitor [85]	98
5.3.3	Two different binding modes reported for p38Map inhibitor [41] .	99
6.2.1	Overall concept for the VSMMD inverse-QSAR approach.	110
6.2.2	Deriving a new image in kernel feature space.	113
6.2.3	Minimum enclosing and maximum excluding hypersphere in the feature space.	115
6.2.4	The pre-image Problem.	118
6.2.5	Kwok and Tsang pre-image strategy.	120
6.2.6	Simplified VSMMD with an aromatic ring treated as a super atom.	125
6.2.7	An example of a chemical structure template.	126
6.2.8	The De Bruijn graph D for VSMMD shown in Figure 6.2.6. . . .	127

6.2.9	The expanded graph M	128
6.2.10	Some possible Euler circuits.	129
6.2.11	An example in edges traversal.	131
6.2.12	Six highest probability Euler Circuits for VSMMD shown in Fig. 6.2.4 and the corresponding chemical structure templates.	132
6.2.13	A case where the pre-image vector did not form a fully connected De Bruijn Graph.	135
6.3.14	Verification test result.	137
6.3.15	Ten highest active compounds in the COX-2 training set.	139
6.3.16	The pre-image VSMMD of the center of the minimum enclosing and maximum excluding hyperspheres.	139
6.3.17	Two Euler circuits with the highest probability for the pre-image VSMMD in Fig. 6.3.16 and the corresponding chemical structure templates.	140
6.3.18	Matching molecule in the test set.	141
6.3.19	The closest matching molecule in the test set for the generated chemical template across 8 data sets.	142

Chapter 1

Introduction

1.1 Drug Discovery

Drugs are molecules that are used as medications or as components in medicines to diagnose, cure, mitigate, prevent, or treat diseases. To determine whether a drug is useful, several complicated processes have to be undertaken. These processes include:

1. investigations of biological activities of the drugs;
2. determination of the drug's absorption, transport and distribution properties;
3. studies of the metabolic transformation of the drugs into other chemicals and their excretion.

As shown in Figure 1.1.1, the drug discovery process in general can be divided into three main stages: target identification, lead discovery, and clinical trials.



Figure 1.1.1: Drug Discovery Process

- **Target Identification.** A drug discovery process begins with a known target (usually a protein) that is related to a disease of interest. The target typically has already been identified and shown to have importance in the disease mechanism through biological or genetic investigations.
- **Lead Discovery.** The second step in the process is to discover a lead molecule. This process begins with the development of an assay to look for a modulator. A modulator can be an inhibitor, an antagonist, or an agonist of the target's activity. Following a high-throughput screening (HTS) of a large number of small molecules, one or several small molecules can be identified. Elaboration of the initial small list of molecule hits through medicinal chemistry is next used to improve the potency so as to produce a potential lead molecule. Among approximately 10^{60} possible molecules [57], only a small portion will be considered for drug development. To speed up the whole screening process, computational techniques such as quantitative structure activity relationship (QSAR), computer-aided drug design and structure-based drug design are widely used to rapidly generate hundreds of derivative compounds from a common scaffold in the hit-to-lead optimization stage. The next step is to optimize the lead molecule to get a candidate drug. In this

process, chemists have to select the exact compounds that fulfill all the requirements related to potency, absorption, bioavailability, metabolism and safety.

- **Clinical Trials.** From the selected candidate molecules, drugs are produced and given to animals for the preclinical safety studies. After this, an officially proven new drug is required to pass three clinical trials on human beings. In phase 1, studies on healthy subjects are conducted to confirm safety. In phase 2, studies are conducted on patients to confirm efficacy. Finally in phase 3, large studies on patients are conducted to gather information about safety and efficacy at the population level.

Drug discovery is a time consuming and costly process. On average, it takes 12 - 15 years to release a new drug to the market. The average cost for developing a new drug is about 600–800 million dollars [1]. Among 10,000 drugs that are applied on animals, only tens of them are left for human clinical trials. With good fortune, one of them can be successfully put into the market. In many cases, none of them can qualify.

1.2 Computer-Aided Drug Design

Computer-Aided Drug Design (CADD) is a specialized discipline that uses computational knowledge-based methods to aid the drug discovery process. There are several key areas where CADD plays an important role in the traditional drug

discovery process. Genomics and bioinformatics support genetic methods of target identification and validation. Chemoinformatics techniques enable researchers to process millions of virtual compounds for selection for synthesis and screening. This allows researchers to make better decisions faster in the arena of drug lead identification and optimization. ADMET (absorption, distribution, metabolism, excretion, and toxicology) modeling aids researchers to identify a bioavailable drug with suitable drug metabolism properties.

CADD methods offer significant benefits for drug discovery. They include:

- **Time and Cost Savings.** Virtual screening, lead optimization and predictions of bioavailability and bioactivity can guide experimental research. Only the most promising drug candidates will be tested and experimental dead-ends can be avoided early based on the results of CADD.
- **Provide Insight.** CADD provides deep insight on drug-receptor interactions. For example: molecular models of drug compounds can reveal intrinsic, atomic scale binding properties that are difficult to envision for researchers.

In general, CADD can be divided into two different strategies: structure-based and ligand-based. Structure-based design usually starts with the structure of a receptor site, such as the active site in a protein. Ligand-based design relies on a set of ligands with known activities interacting with the same receptor and is particularly valuable if no structural information is available for the binding site.

1.2.1 Structure-Based Design

The preliminary step in structure-based drug design is to determine the three dimensional structure of a target molecule. This can be achieved by X-ray crystallography or NMR spectroscopy experiments. Or, it can be approximated by some computational methods such as comparative modeling¹ and ab initio modeling². Given the three dimensional image of a target molecule (receptor), one needs to identify the location of its binding site. The actual binding site can be located by comparing with known protein-ligand complexes or through homology comparisons to related complexes. With a well determined binding site, a ligand can be determined.

Usually, a ligand can be determined either through de novo design³ or through a database search for a molecule that matches to the binding site. Docking methods are then used to evaluate the quality of the ligand. There are three tasks that a docking procedure is required to address:

- characterizing the binding site;
- positioning the ligand into the binding site;
- evaluating the strength of interaction for a specific ligand-receptor complex.

¹Comparative protein modelling uses previously solved structures as starting points to determine the three-dimensional structure of a protein

²Ab initio protein modelling methods seek to build three-dimensional protein models based on physical principles rather than previously solved structures.

³De novo design involves creating new ligand from scratch by connecting possible atoms or molecular fragments for a particular receptor and evaluating their quality.

1.2.2 Ligand-Based Design

Ligand-based design methods are based on the fact that ligands similar to an active ligand are more likely to be active than random ligands. Ligand-based design usually starts with a database containing a set of molecules with known activities interacting with the same receptor. A domain expert is required to construct this set of molecules from the database. The domain expert is also responsible for dividing the set of molecules into training and testing sets if necessary. The second step of the ligand-based design is molecular modeling. Ligand-based approaches commonly consider descriptors based on chemistry, shape, electrostatic, and interaction points (e.g., pharmacophore points) to assess similarity. A pharmacophore is an explicit geometric hypothesis of the critical features of a ligand [65]. Features usually include hydrogen-bond donors and acceptors, charged groups and hydrophobic patterns. The hypothesis can be used to screen databases for candidate compounds and also can be used to refine existing leads. In Chapter 2, we will discuss this concept in more detail.

The third step of the ligand-based design usually involves setting up a computational model to identify the most promising molecule as a lead molecule for further experimental investigation. A common method is the Quantitative Structure Activity Relationships (QSAR) modeling for identifying a lead molecule. The concept of QSAR is based on the fact that the biological properties of a compound can be expressed as functions of its physicochemical parameters. Popular physicochemical parameters of a compound include its solubility, lipophilicity, electronic effects, and ionization. The goal of QSAR studies is to predict the activity of new compounds

based solely on these parameters.

The last step is to identify the lead molecule from the results of the computational model. Figure 1.2.2 is a schematic presentation that summarizes the general steps involved in ligand-based drug design.

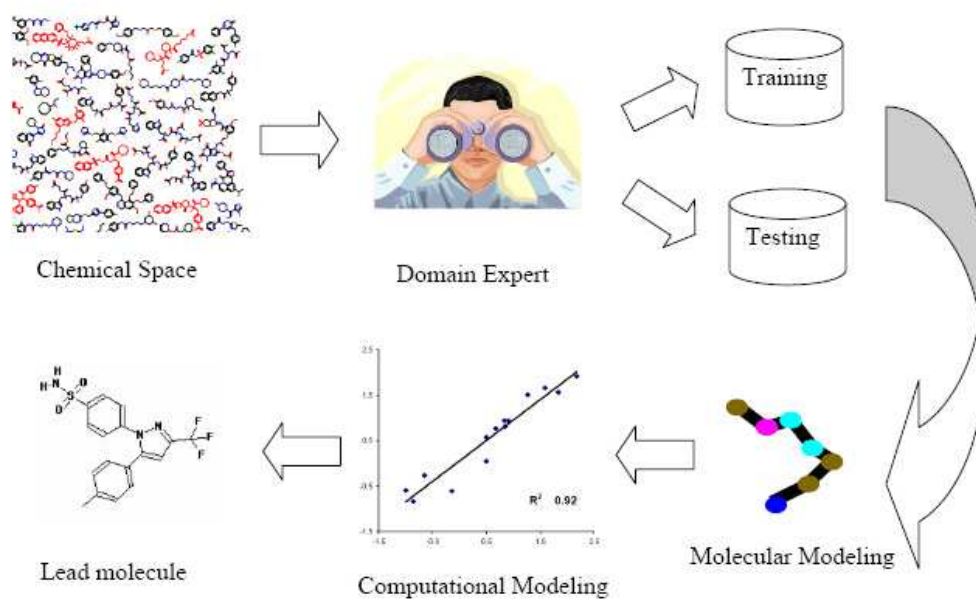


Figure 1.2.2: General steps involved in ligand-based drug design.

1.3 Contributions of the thesis

The structural conformation and physicochemical properties of both the ligand and its receptor site determine the level of binding affinity that is observed in such an interaction. If the structural properties of the receptor site are known (for example, there is crystallographic data), techniques involving approximations of

potential functions can be applied to estimate or to compare binding affinities of various ligands. When this information is sparse or not available, as is the case for many membrane proteins, it becomes necessary to estimate affinities using only the properties of the ligand. In this case, one strives to design a drug by improving the chemical structure of a ligand that has been observed to successfully interact with the protein. So, in this thesis, we focus on the development of ligand-based prediction algorithms. We assume that all the training samples are processed using only ligand properties with no explicit dependency on the nature of the protein binding site.

1.3.1 Contributions to the traditional ligand-based approach

Following the traditional ligand-based approach that we have described in Figure 1.2.2, the main concerns of this approach can be categorized into the following issues: formation of descriptors, component selection, and computational modeling issues.

- **Formation of descriptors:** The manipulation and analysis of chemical structural information of a ligand is made possible through the use of a molecular descriptor. These are numerical values that characterize properties of the ligand. For example, they may represent both the reaction centers and their relationship to one another in a three-dimensional setting.
- **Component selection:** High dimensionality of descriptors poses a challenge for statistical learning algorithms used to formulate predictors. The minimum

number of samples required to ensure a high level of prediction confidence rapidly increases as the dimensionality of the descriptors increases. We need computational strategies to do component selection, keeping only those components in the initial version of a molecular descriptor that are useful for later study.

- **Computational modeling issues:** A good computational model should address the following issues:
 1. Ligand binding affinities depend on molecular descriptors in a nonlinear fashion.
 2. Data sets for ligand-based approach are usually not balanced. It is a typical case that training data contains a large number of samples in the negative class (ligands not binding to the protein) and a much smaller number of samples in the positive class (ligands that show a high affinity binding).
 3. Many data are poor in quality. A poor quality data set may be due to errors in wet lab observations. A good computational model should be robust when applied to such a noisy data set.

In this thesis, we use kernel methods to implicitly define a nonlinear mapping from the input space of molecular descriptors to a high dimensional feature space of image points. Thus, a linear predictor defined in the feature space defines a nonlinear predictor for the input space of descriptors. Kernel methods are also known to behave well in comparison to other statistical or machine learning methods when

dealing with high-dimensional noisy data [87]. Furthermore, kernel methods can be modified to handle unbalanced data. This approach addresses the issues related to computational modeling.

The contributions of this thesis to the traditional ligand-based approach are the following:

- **Contribution 1:** We propose a vector space model molecular descriptor (VSMMD) based on a vector space model that is suitable for kernel studies in QSAR modeling. Our experiments have provided convincing comparative empirical evidence that our descriptor in conjunction with kernel based regression algorithms can provide sufficient discrimination to predict various biological activities of a molecule with reasonable accuracy. This contribution addresses the issues related to formation of a descriptor.
- **Contribution 2:** We present a new component selection algorithm KACS (Kernel Alignment Component Selection) based on kernel alignment for a QSAR study. Kernel alignment has been developed as a measure of similarity between two kernel functions. In our algorithm, we refine kernel alignment as an evaluation tool, using recursive component elimination to eventually select the most important components for classification. We have demonstrated empirically and proven theoretically that our algorithm works well for finding the most important components in different QSAR data sets. This contribution addresses the issues related to component selection.

Figure 1.3.3, an extension of Figure 1.2.2, includes the use of kernel methods in the traditional ligand-based drug design. The kernel-based traditional ligand-based

drug design involves the following steps:

1. The approach starts with a database of molecules. In this step, we say that the molecules are in a chemical space ⁴.
2. A domain expert is involved to select a set of molecules with known activities interacting with the same receptor. The domain expert is also responsible for dividing the set of molecules into training and testing sets.
3. Initial molecular descriptors are generated for the training set.
4. A component selection algorithm is used to select the most important components of the initial descriptors. Once these component positions are determined for the descriptor vector, these components are used across all initial descriptors of the training set to get the final descriptors.
5. Kernel methods are used to map the input space of the final descriptors to high dimensional feature space of image points.
6. A predictor can be defined and trained in the feature space.
7. Generate initial molecular descriptors for the testing set. Final descriptors are then constructed using the best component selection strategy as determined by the training set. The predictor defined in step 6 will be used to assess the testing set.

⁴In this thesis, chemical space is defined as the space that encompasses all the small molecules that could in principle be created.

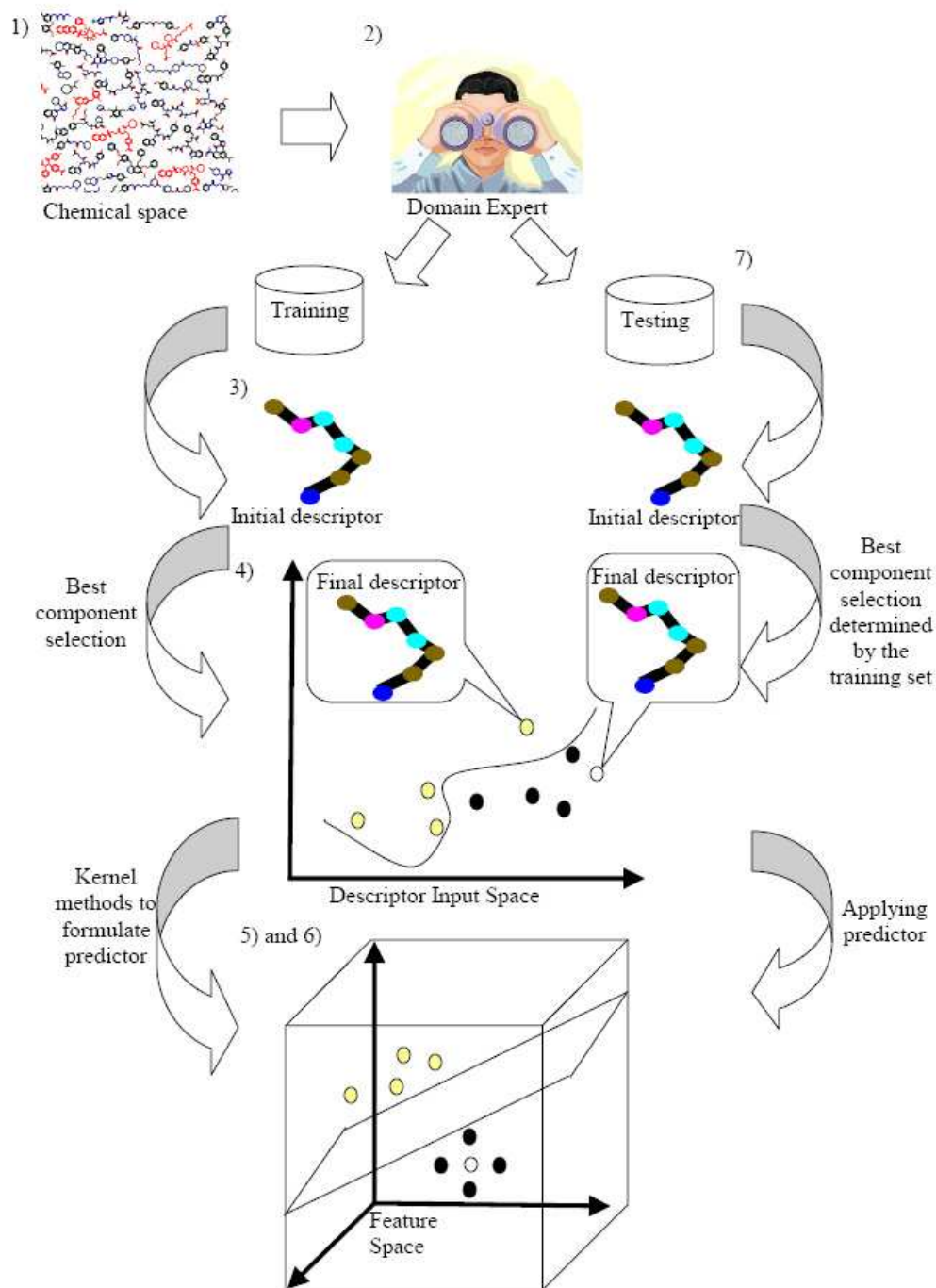


Figure 1.3.3: Kernel-based traditional ligand-based drug design.

1.3.2 Contributions to improve the traditional ligand-based approach

The traditional ligand-based approach involves several relevant issues and problems that have not yet been addressed. This includes issues related to: multiple binding modes, library dependence, and reverse engineering of a molecular descriptor.

- **Multiple binding modes.** The ligand may show evidence of multiple binding modes within the same binding site. The same ligand can bind in distinct orientations or conformations in the binding site. The information of binding affinities for each mode are usually combined and packed as a whole in the training set. To disentangle this information is a challenge and thus complicates the machine learning strategy.
- **Library dependence.**

Statistical learning theory [95] asserts the success of a predictor only when the test sample is drawn from a data source that has the same probability distribution as that characterizing the training set. More precisely, the components of the descriptors must have the same probability distribution in both training and testing set. In traditional ligand-based drug design, we can define various data sets: training data set, validation data set, testing data set and application data set. The first two data sets are used to derive the predictor and the testing data set is used to get a measure of success for the predictor. Note that in all these cases, we know the affinities of the ligand and it is likely that the probability distribution constraint can be met. The

application data set is the set of ligands on which the predictor will be used in the future to get a new drug candidate. For this set, we do not know the affinities. It is usually difficult for one to guarantee the probability distribution constraint. In this case, the predictor is often applied to molecules that have very little relationship to the training data.

- **Reverse engineering of a molecular descriptor.** Ideally, a good molecular descriptor should not only be able to capture important molecular properties, it should also be able to represent these properties in a transparent fashion. In particular, we want descriptors that facilitate the implementation of recovery algorithms that can accept a newly computed descriptor as input and subsequently compute a new ligand corresponding to this new descriptor.

The contributions of this thesis to improve the traditional ligand-based approach are the following:

- **Contribution 3:** Traditionally, the assumption of similar analogs binding to the same binding site in a similar binding model is often employed in QSAR studies [55]. In the QSAR modeling process, data from molecules with alternate binding modes in a binding site are completely ignored or treated as outliers. However, the ability of predicting the correct binding modes of active compounds identified from high-throughput or virtual screening will facilitate the drug discovery process. In conjunction with a kernel based clustering algorithm, we extend the VSMMD to the prediction of multiple binding modes. The results reported in this study provide strong empirical evidence

that our strategy has enough resolving power to distinguish multiple binding modes through the use of a standard k -means algorithm. This contribution addresses the issue related to the multiple binding modes.

- **Contribution 4:** We have developed a set of reverse engineering strategies for QSAR modeling based on our VSMMD. These strategies include:
 1. The use of a kernel feature space algorithm to design or modify descriptor image points in a feature space.
 2. The deployment of a pre-image algorithm to map the newly defined descriptor image points in the feature space back to the input space of the descriptors.
 3. The design of a probabilistic strategy to convert new descriptors into meaningful chemical graph templates.

The most important aspect of these contributions is the presentation of strategies that actually generate the structure of a new drug candidate. While the training set is still used to generate a new image point in the feature space, the reverse engineering strategies just described allow us to develop a new drug candidate that is independent of issues related to the probability distribution constraints placed on the testing or application set. This contribution addresses the issues related to library dependence and reverse engineering of a molecular descriptor.

Figure 1.3.4 summarizes our ligand-based constructive drug design approach. Our approach involves the following steps:

1. The approach starts with a database of molecules in chemical space.

2. A domain expert is involved to select a set of molecules with known activities interacting with the same receptor.
3. Initial molecular descriptors are generated for the training set.
4. A component selection algorithm is used to select the most important components of the initial descriptors. Once these component positions are determined for the descriptor vector, these components are used across all initial descriptors of the training set to get the final descriptors.
5. Kernel methods are used to map the final descriptors in the input space to a high dimensional feature space of image points.
6. A predictor can be defined and trained in the feature space. A test set may be used to gain confidence in the predictor.
7. A new image point in the feature space can be defined using an appropriate kernel feature space algorithm.
8. A kernel pre-image method is used to calculate the inverse mapping of the new image point back to the input space of the descriptors and thus derive the descriptor for the new drug lead.
9. A probabilistic algorithm is used to convert the descriptor for the drug lead to a meaningful chemical graph template.
10. Based on the chemical graph template, lead molecules can be identified by means of high-throughput screening.

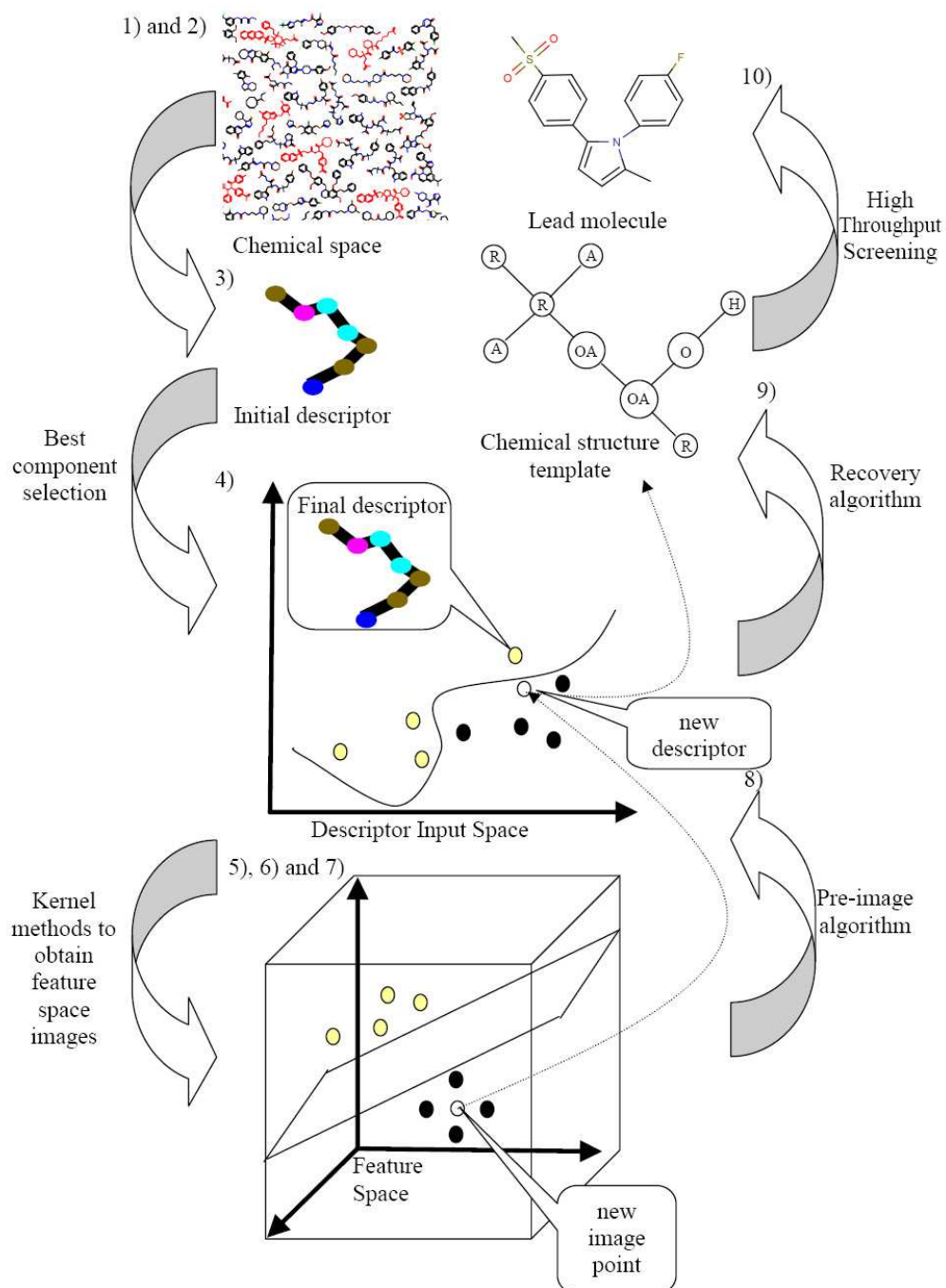


Figure 1.3.4: Our ligand-based constructive drug design approach.

The overall goal of this thesis is to develop effective and efficient computational methods that are reliant on transparent mathematical descriptors of molecules with applications to affinity prediction, detection of multiple binding modes, and generation of new drug leads. While in this thesis we derive computational strategies for the discovery of new drug leads, our approach differs from the traditional ligand-based approach. We have developed novel procedures to calculate an inverse mapping and subsequently recover the structure of a potential drug lead.

The layout of this thesis is as follows: In chapter 2, we will introduce some relevant prior studies on molecular descriptors, QSAR, and kernel methods. In chapter 3, we introduce our vector space model molecular descriptor (VSMMD). In chapter 4, we will focus on component selection and our newly proposed method Kernel Alignment Component Selection (KACS) will be presented. In chapter 5, we extend the VSMMD in conjunction with a kernel based clustering algorithm to the prediction of multiple binding modes. In chapter 6, we present our reverse engineering strategies for QSAR modeling based on our VSMMD. Conclusion and future initiatives are summarized in chapter 7.

Chapter 2

Molecular Modeling and Computational Modeling

In this chapter, we will overview some relevant prior studies on the two key components in traditional ligand-based drug design: They are molecular modeling and computational modeling.

2.1 Molecular Modeling

The manipulation and analysis of chemical structural information of a ligand is made possible through the use of molecular descriptors. These are numerical values that characterize properties of the ligand. A successful traditional ligand-based drug design approach critically depends on the accurate definition and the appropriate use of molecular descriptors [87]. In this section, we will overview some existing descriptors.

Molecular descriptors can be categorized as empirical descriptors or theoretical descriptors [87]. The empirical descriptors are those that can be obtained from some experimentally measured data that usually include thermodynamic, kinetic, chemical and physical data. The theoretical descriptors are usually derived from the chemical structure of a molecule and are expressed with an explicit mathematical formula that is based on fundamental physical attributes of the molecule.

In general, theoretical descriptors can be classified into different classes based on their complexity or the method of calculation. They include constitutional descriptors, topological descriptors, geometrical descriptors, charge-distribution-related descriptors, molecular-orbital-related descriptors, temperature dependent descriptors, solvation descriptors, and mixed descriptors. We will limit our overview to a brief description of the first three types of descriptors.

The simplest class of theoretical descriptors is the set of constitutional descriptors. They are constructed based on simple counts of features, such as hydrogen bond donors, hydrogen bond acceptors, aromatic rings, number of rotatable bonds, and molecular weight.

The topological descriptors are single valued functions that can be calculated from the two-dimensional graphical representation of molecules. The descriptor describes the structure according to the molecule's size, degree of branching, and overall shape. Example of topological descriptors include the Wiener index [110], the branching index [65], and chi molecular connectivity indices [52].

Geometrical descriptors are derived from the three-dimensional structure of the molecules. The properties of a molecule depend on how its atoms can be positioned

in space to produce its three-dimensional structures or conformations. The quality of a geometrical descriptor depends heavily on the quality of these conformations. Molecular surface area is one of the more useful geometrical descriptors. Other popular geometrical descriptors include molecular volume and the solvent-excluded volume.

A pharmacophore is defined as a set of critical features in a molecule together with their relative spatial orientations. These features usually include hydrogen-bond donors and acceptors, charged groups and hydrophobic centers. We often refer to these features as pharmacophore points. The spatial relationships between the features are usually specified by all possible inter-point distances between chosen pharmacophore points. The pharmacophore is usually derived from three-dimensional computed conformations of a molecule and is an abstract representation of the molecule. A simple example of a three-point pharmacophore based on inter-point distances is given in Figure 2.1.1. Three-point pharmacophores have traditionally been used as descriptors for many applications such as high-throughput screening and QSAR modeling [65].

2.2 Computational Modeling

2.2.1 Quantitative Structure-Activity Relationship

In ligand-based design, a computational model is required to identify the most promising molecule as a lead molecule for further experimental investigation. In a more general setting, we strive to establish the quantitative dependency between

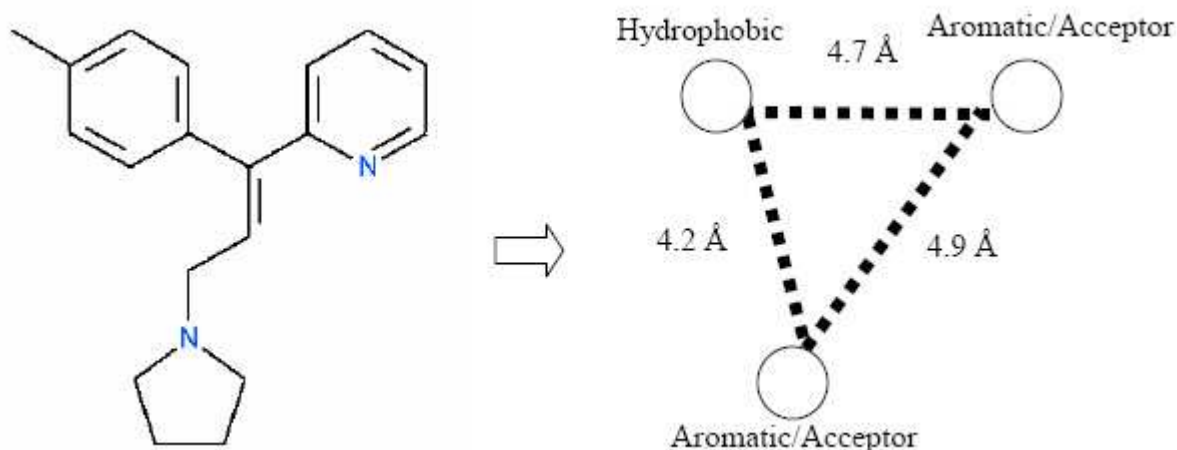


Figure 2.1.1: An example of three-point pharmacophore.

the molecular properties of a ligand and its binding affinity. To restate this goal using current terminology: we want to analyze the Quantitative Structure-Activity Relationship (QSAR) of the ligand with respect to a particular type of receptor. A common approach for a QSAR analysis is the use of a machine learning strategy that processes sample data to learn a function that will predict binding affinities. The input of such a function is a molecular descriptor that was described in the previous section.

2.2.2 Historical Background

The concept of QSAR was first introduced [88] in 1868, when a QSAR was modeled [18] using the equation $\sigma = f(C)$ where the physiological activity σ was expressed as a function of the chemical structure C .

Later quantitative approaches combined different physicochemical parameters in

a linear additive manner [40] followed by Free and Wilson who formulated structure-activity dependencies using the equation: $A_B = u + \sum_i a_i x_i$ where A_B is the biological activity, u is the average contribution of the unsubstituted parent molecule of a particular series (for example, a scaffold), the a_i values are the contributions of various structural features, and the x_i values denote the presence or absence of a particular fragment [27]. Since then, QSAR has remained a thriving research area in drug design.

More recently developed QSAR modeling approaches include HQSAR [69], Inverse QSAR [13] and Binary QSAR [29]). The accuracy of QSAR modeling is greatly improved by using sophisticated statistical and machine learning methods, for example, partial least square (PLS) [21] and support vector machines (SVM) [104].

2.2.3 Deriving a QSAR Equation: Kernel Methods

Various mathematical methods have been used to derive QSAR models. These methods can be categorized into linear and non-linear methods.

- **Linear Methods.** A linear method defines a linear relationship between the activity level and the molecular descriptors to establish the structure-activity relationship. There are several commonly used linear methods in QSAR modeling. They include multiple linear regression (MLR) [115, 103, 45], partial least squares (PLS) [99], and linear discriminant analysis (LDA) [34].
- **Non-linear Methods.** Non-linear methods establish the structure-activity

relationship using a non-linear function. Common non-linear methods include: Bayes classifier [99], k -nearest neighbors [47], neural networks [96] and kernel methods [4]. These methods generally provide more accurate results, especially for a large and diverse dataset.

Kernel Methods are a relatively new class of algorithms for pattern analysis in data sets. The idea behind kernel methods is to have a systematic methodology for incorporating nonlinearities into the predictor function [10]. In this section, we will next describe the theory behind kernel methods.

Kernels

Kernel-based learning algorithms work by embedding the data into a Hilbert space, often called the feature space, followed by a search for linear relations within this Hilbert space. Kernels are functions that can be used to formulate similarity comparisons. They provide a general framework to represent data, subject to certain mathematical conditions. Data are not represented individually by kernels. Instead, data relations are represented through a set of pair-wise comparisons.

More formally, suppose we have n training data pairs $((x_1, y_1), \dots, (x_n, y_n)) \in X \times Y$ where X is the input space, and Y is vector of the output values. In the process of machine learning, we want to be able to generalize to previously unseen data points. In the case of binary classification, given some new input $x \in X$, we want to predict the corresponding $y \in \{+1, -1\}$. In other words, we want to choose y such that (x, y) is in some sense similar to the training examples. In order to achieve this, we require similarity measures in X and in Y . Since $y \in \{+1, -1\}$, to

find the similarity measure in Y is relatively easy. On the other hand, we require a function to measure the similarity in X :

$$\begin{aligned} k : X \times X &\rightarrow \mathbb{R}, \\ (x, x_i) &\rightarrow k(x, x_i) \end{aligned} \tag{2.2.1}$$

satisfying, for all $x, x_i \in X : i = 1 \dots n$,

$$k(x, x_i) = \langle \phi(x), \phi(x_i) \rangle, \tag{2.2.2}$$

where ϕ maps points into a dot product space FS , called the feature space. The similarity measure k is called a kernel, and ϕ is its feature map. Figure 2.2.2 illustrates the overall mapping concept.

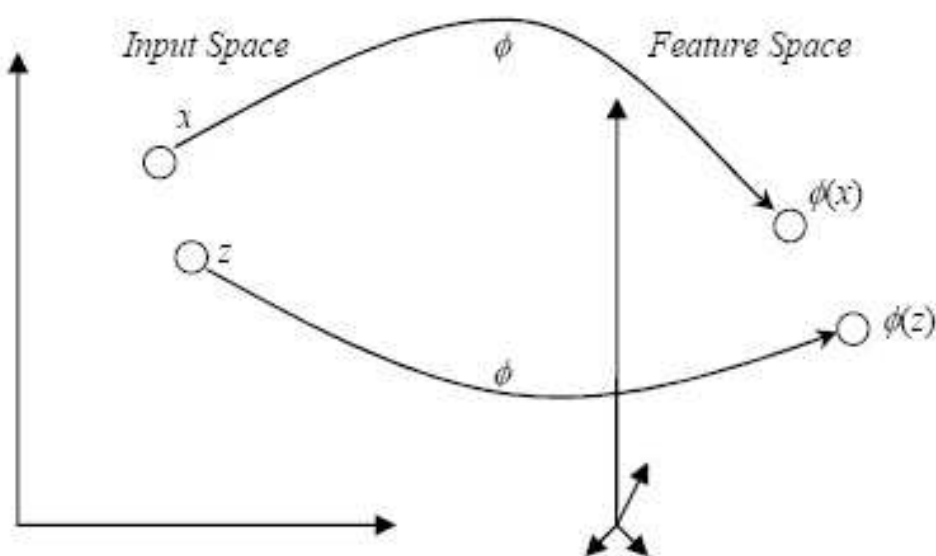


Figure 2.2.2: The implicit function ϕ maps points in the input space over to the feature space.

By using a kernel function, the embedding in the Hilbert space is actually performed implicitly, that is by specifying the inner product between each pair of

points rather than by giving their coordinates explicitly. This approach has several advantages, the most important being the fact that often the inner product in the embedding space can be computed much more easily using a kernel rather than using the coordinates of the image points themselves.

In machine learning, using kernels is a strategy for converting a linear classifier algorithm into a non-linear one by using a non-linear function to map the original observations into a higher-dimensional feature space; this makes a linear classification in the new feature space equivalent to a non-linear classification in the original input space.

We also consider the matrix K with elements $K_{ij} = k(x_i, x_j)$, called the kernel matrix or the Gram matrix. Because of the fact that it is built from inner products, it is always a symmetric positive semi-definite matrix, and since it specifies the inner products between all pairs of points, it completely determines the relative positions between those points in the embedding space. According to Mercer's theorem [89], every symmetric, positive semi-definite bilinear function can be represented as an inner product in a feature space. This property allows one to develop feature space algorithms by simply using positive semi-definite kernel functions. Thus the exact form of the feature images $\phi(x)$ does not have to be known.

Understanding the feature space FS and the norm associated with a kernel often helps in understanding kernel methods and in designing new kernels.

Let k be a kernel on a space X . Let us assume that the prediction function sought by kernel-based algorithms is a linear function in the feature space:

$$f(x) = \langle w, \phi(x) \rangle, \quad (2.2.3)$$

for some vector $w \in FS$. The w is called the weight vector and can be expressed as a linear combination of the training points:

$$w = \sum_{i=1}^n \alpha_i \phi(x_i), \quad (2.2.4)$$

where the vector α , containing these α_i as its entries, will be referred to as the dual vector. With equation (2.2.4), we can express f as follows:

$$f(x) = \langle w, \phi(x) \rangle = \left\langle \sum_{i=1}^n \alpha_i \phi(x_i), \phi(x) \right\rangle = \sum_{i=1}^n \alpha_i k(x_i, x). \quad (2.2.5)$$

The corresponding norm in the feature space is:

$$\|f\|_F^2 = \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j k(x_i, x_j). \quad (2.2.6)$$

In the remaining section, we will use $f(\cdot)$ to denote a function. Note that this is different from $f(x)$ which is $f(\cdot)$ evaluated at x . The space is a vector space and it contains any linear combination of these functions. We will ensure that the space also contains functions such as $k(x_i, \cdot)$.

More formally, we start with a real-valued symmetric positive semi-definite kernel:

$$k : X \times X \rightarrow \mathbb{R}. \quad (2.2.7)$$

Our ϕ map takes points in X and maps them into a space of functions where each function maps X into \mathbb{R} . This space of functions is defined as:

$$\mathbb{R}^X := \{f | X \rightarrow \mathbb{R}\}, \quad (2.2.8)$$

and the kernel mapping is defined via:

$$\begin{aligned} \phi : X &\rightarrow \mathbb{R}^X \\ x &\rightarrow k(x, \cdot). \end{aligned} \quad (2.2.9)$$

Another way to express this:

$$\phi(x)(\cdot) = k(x, \cdot). \quad (2.2.10)$$

One may think of \mathbb{R}^X as a very large space. Within \mathbb{R}^X , we focus on a smaller infinite subset that is an inner product space derived from our given kernel function. Our inner product space contains the functions $k(x_i, \cdot)$ for all $x_i \in X$. As a vector space, it will also contain all linear combinations of these functions such as:

$$f(\cdot) = \sum_{i=1}^n \alpha_i k(x_i, \cdot) \quad (2.2.11)$$

where the x_i are arbitrary points in X and all $\alpha_i \in \mathbb{R}$. Since FS is a Hilbert space, a dot product can be defined for two elements $f(\cdot) = \sum_{i=1}^n \alpha_i k(x_i, \cdot)$ and $g(\cdot) = \sum_{j=1}^m \beta_j k(z_j, \cdot)$ by

$$\langle f, g \rangle = \sum_{i=1}^n \sum_{j=1}^m \alpha_i \beta_j k(x_i, z_j). \quad (2.2.12)$$

We should think of this inner product as $\langle f(\cdot), g(\cdot) \rangle$.

One important property of this construction is that the value of $f(x)$ of a function $f \in FS$ can be expressed as a dot product in FS : $f(x) = \langle f, k(x, \cdot) \rangle$. By taking $g(\cdot) = k(z, \cdot)$, we have

$$\langle f, g \rangle = \langle f, k(z, \cdot) \rangle = \sum_{i=1}^n \alpha_i k(x_i, z) = f(z). \quad (2.2.13)$$

We rewrite equation (2.2.13) as

$$f(x) = \langle f, k(x, \cdot) \rangle. \quad (2.2.14)$$

Equation (2.2.14) says that we can get the value of $f(\cdot)$ at x by evaluating the inner product of two functions namely f itself and the function $k(x, \cdot)$. We say that k is the *representer of evaluation*.

Note that $f(\cdot)$ is any function in our vector space. We can set $f(\cdot) = k(z, \cdot)$ in $f(x) = \langle f, k(x, \cdot) \rangle$, then we can derive the following reproducing property valid for any $x, z \in X$:

$$\langle k(z, \cdot), k(x, \cdot) \rangle = k(z, x). \quad (2.2.15)$$

For this reason, the feature space FS is usually called the reproducing kernel Hilbert space (RKHS) associated with k . The equation (2.2.13) leads to the following Theorem:

Theorem 2.2.1. *For any kernel k on a space X , there exists a Hilbert space FS and a mapping $\phi : X \rightarrow FS$ such that*

$$k(x, z) = \langle \phi(x), \phi(z) \rangle, \quad (2.2.16)$$

for $x, z \in X$. Given two points $u, v \in FS$, $\langle u, v \rangle$ represents the dot product in the Hilbert space.

For more details about RKHS, readers can refer to [89].

By using a kernel, we have the following advantages:

1. **Representation independently:** The kernel matrix does not depend on the nature of the objects to be analyzed. They can be images, molecules, or sequences, and the representation of a data set is always a real-valued square matrix. This independence enables one to develop more flexible algorithms for different studies.
2. **Complexity:** The size of the matrix does not grow as the dimension of the input data increases.

3. **Easy comparison:** Kernels can act as a tool to measure similarity.

There are several common kernels. They include:

1. linear kernel: $k(x, x_i) = x^T x_i$, the inner product between vectors;
2. Gaussian kernel: $k(x, x_i) = e^{-\|x-x_i\|^2/2\sigma}$; and
3. polynomial kernel: $k(x, x_i) = (\langle x, x_i \rangle + c)^d$ where d is the polynomial degree.

There are some other kernels defined in the research literature for different purposes. They include: string kernels [89], and tree kernels [89].

Kernel methods are a class of algorithm for which the data to be analyzed only enter the algorithm through the kernel function. Common kernel methods include the support vector machines (SVM).

Support Vector Machine

The Support Vector Machine (SVM) algorithm [89] is relatively new supervised learning technique originally introduced by Vapnik [104]. The SVM has been successively extended by a number of other researchers. In a classification problem, the SVM separates a given set of binary labeled training data with a hyper-plane that is maximally distant from them. This hyper-plane is called the maximal margin hyper-plane (Figure 2.2.3). The weight vector w and threshold b for the hard margin SVM are chosen to optimize the following problem:

$$\min_{w,b} \frac{1}{2} \|w\|^2, \quad (2.2.17)$$

subject to:

$$y_i(\langle w, \phi(x_i) \rangle - b) \geq 1, \quad i = 1, \dots, n, \quad (2.2.18)$$

where y_i is the target value for data x_i . The corresponding dual problem can be derived as:

$$\max_{\alpha} \left[\sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j k(x_i, x_j) \right], \quad (2.2.19)$$

subject to:

$$\sum_{i=1}^n y_i \alpha_i = 0 \text{ and } 0 \leq \alpha_i \text{ for } i = 1, \dots, n, \quad (2.2.20)$$

where α_i is the dual variable, and $k(x_i, x_j)$ can be obtained from the kernel matrix K . In practice, the data may not be separable in this fashion. By introducing slack variables $\zeta_i \geq 0$, equation (2.2.17) becomes:

$$\min_{w,b,\zeta} \left[\frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \zeta_i \right], \quad (2.2.21)$$

subject to:

$$y_i(\langle w, \phi(x_i) \rangle - b) \geq 1 - \zeta_i, \quad i = 1, \dots, n, \quad (2.2.22)$$

where the parameter C controls the trade-off between the width of the margin and the size of the slack variables.

By using equation (2.2.21), we allow some negative samples inside the positive side of the maximal margin hyper-plane and some positive samples outside the positive side of the maximal margin hyper-plane. The dual problem of equation (2.2.21) is:

$$\max_{\alpha} \left[- \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j k(x_i, x_j) \right], \quad (2.2.23)$$

subject to:

$$\sum_{i=1}^n y_i \alpha_i = 0, \quad \sum_{i=1}^n \alpha_i = 1 \quad \text{and} \quad 0 \leq \alpha_i \leq C, \quad \text{for } i = 1, \dots, n. \quad (2.2.24)$$

For more details about SVM, readers can refer to [89].

SVM have remarkable robust performance with respect to sparse and noisy data. This makes SVM the strategy of choice in a number of applications ranging from text categorization to protein function prediction.

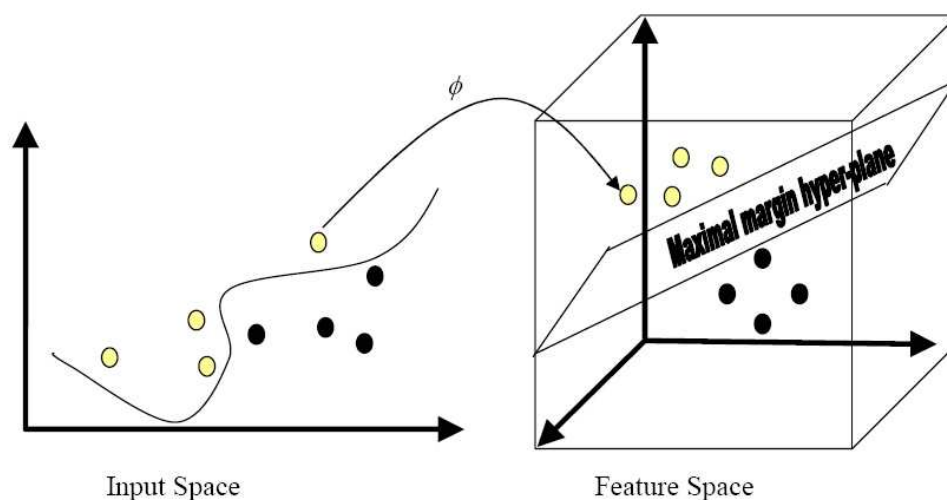


Figure 2.2.3: An example of the Support Vector Machine.

Support Vector Regression

To perform regression, we made use of a linear ε -insensitive Support Vector Regression (SVR). In ε -insensitive SVR, the goal is to find a function $f(d)$ that has at most ε deviation from the activity values of the training data. Suppose we are

working with n training data samples. We apply the mapping indicated in equation (2.2.9) repeatedly to generate $\phi(x_1), \phi(x_2), \dots, \phi(x_n)$. The weight vector w and threshold b for the linear ε -insensitive SVR are chosen to optimize the following problem:

$$\min_{w, b, \xi, \hat{\xi}} \left[\frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\xi_i + \hat{\xi}_i) \right], \quad (2.2.25)$$

subject to:

$$\left. \begin{aligned} (\langle w, \phi(x_i) \rangle + b) - y_i &\leq \varepsilon + \xi_i \\ y_i - (\langle w, \phi(x_i) \rangle + b) &\leq \varepsilon + \hat{\xi}_i \\ \xi_i, \hat{\xi}_i &\geq 0 \end{aligned} \right\} \quad i = 1, \dots, n. \quad (2.2.26)$$

where y_i is the target value for data x_i , $\hat{\xi}_i$ and ξ_i are the slack variables representing upper and lower constraints on the outputs of the regression system, and C is a constant. The constant $C > 0$ determines the trade off between flatness of the regression line and the amount up to which deviations larger than $\varepsilon > 0$ are tolerated. The corresponding dual problem can be derived as:

$$\max_{\alpha, \hat{\alpha}} \left[\sum_{i=1}^n (\hat{\alpha}_i - \alpha_i) y_i - \varepsilon \sum_{i=1}^n (\hat{\alpha}_i + \alpha_i) - \frac{1}{2} \sum_{i,j=1}^n (\hat{\alpha}_i - \alpha_i) (\hat{\alpha}_j - \alpha_j) k(x_i, x_j) \right], \quad (2.2.27)$$

subject to:

$$\sum_{i=1}^n (\hat{\alpha}_i - \alpha_i) = 0 \text{ and } 0 \leq \alpha_i, \hat{\alpha}_i \leq C, \text{ for } i = 1, \dots, n. \quad (2.2.28)$$

where $\hat{\alpha}_i$ and α_i are the dual variables corresponding to $\hat{\xi}_i$ and ξ_i , and $k(x_i, x_j)$ can be obtained from the kernel matrix K . After making the substitution $\beta_i = \hat{\alpha}_i - \alpha_i$ and then using the Karush-Kuhn-Tucker relations [91], we can rewrite the dual problem as:

$$\max_{\beta} \left[\sum_{i=1}^n \beta_i y_i - \varepsilon \sum_{i=1}^n |\beta_i| - \frac{1}{2} \sum_{i,j=1}^n \beta_i \beta_j k(x_i, x_j) \right], \quad (2.2.29)$$

subject to:

$$\sum_{i=1}^n \beta_i = 0. \quad (2.2.30)$$

After we find β^* as a solution of the above optimization problem, we set up the regression function $f(x) = \sum_{j=1}^n \beta_j^* k(x_j, x) + b^*$, where $b^* = -\varepsilon + y_i - \sum_{j=1}^n \beta_j^* k(x_j, x_i)$ for i with $0 < \beta_i^* < C$. For more details about SVR, readers can refer to Smola *et al.* [91].

Kernel Methods in Computational Biology

More recently, kernel methods have been widely used in computational biology. Support Vector Machines (SVM) have been found to be very useful in this area. They offer versatile tools to process and analyze data and can offer state-of-art performance under different situations [87]. Common applications include:

1. protein remote homology detection;
2. receptor classification and protein function annotation;
3. microarray gene expression analysis;
4. proteomics/protein expression analysis.

The use of kernel methods in computational biology is advisable for the following reasons:

1. The biological data are usually contained in high dimensional spaces.
2. The data are complicated by noise.

3. Kernel methods can handle the variable length data that is often seen in biological applications.

Kernel Methods in QSAR modeling

Currently, kernel methods are popular tools in QSAR modeling and are used to predict attributes such as: activity towards a therapeutic target, ADMET properties (absorption, distribution, metabolism, excretion, and toxic effects), and adverse drug reactions. Various kernel methods based on different molecular representations have been proposed for QSAR modeling [4], they include the SMILES strings kernel [100], graph kernels [28, 72, 80] and a pharmacophore kernel [71].

Chapter 3

The Vector Space Model Molecular Descriptor

The manipulation and analysis of chemical structural information of a ligand is made possible through the use of a molecular descriptor. A good molecular descriptor should not only be able to capture important molecular properties, it should also be able to facilitate the derivation of computational model. An ideal computational model should predict ligand binding affinities based on molecular descriptors in a nonlinear fashion. It should also be able to handle an imbalanced and noisy data set.

In this chapter, we propose a vector space model molecular descriptor (VSMMD) based on a vector space model that is suitable for kernel studies in QSAR modeling.

3.1 The vector space model molecular descriptor

Our vector space model molecular descriptor (VSMMD) would be categorized as belonging to the constitutional descriptors that provide component counts related to the structure of a molecule. Similar to other fragment based methods, the topological patterns of atoms and bonds in a molecule are encoded into our descriptor. Fragment based methods like HQSAR [69] and ECFP [83] have been used with considerable success in various QSAR studies [30, 43]. Among different kinds of fragment based methods, HQSAR and ECFP provide better performance among others [30, 43]. However, when dealing with a small data set (< 500 data points), both methods often overfit to the training set and lead to poor performance on the test set [30]. Our VSMMD approach is designed to address this issue.

The first step in constructing the VSMMD is to identify the physicochemical properties of each atom in a molecule. Specifically, we affix labels to atoms and bonds as specified in Figure 3.1.1. It should be noted that triple bonds would also be labeled as “=”.

The VSMMD strategy is based on the extraction of molecular fragments that are comprised of bonded atoms. The atom count for a fragment is at least two and at most f_a where f_a is some pre-specified relatively small value such as 2, 3, or 4.

To illustrate the processing of a molecule we describe the steps that are taken in the processing of a molecule (atom count for a fragment limited to 2). Figure 3.1.2 shows one of the pyrrole compounds that is a member of the data set used in [96]. The algorithm goes through the following steps:

hydrogen bond acceptor	A	} Atom Type (A_T)
hydrogen bond donor	D	
positively charged	P	
negatively charged	N	
aromatic rings	O	
halogen atom	H	
none of the above	R	
single bond	-	} Bond Type (B_T)
double bond	=	
Aromatic bond	#	

Figure 3.1.1: Labels for atoms and bonds in a molecule

1. The atoms and bonds are labeled using the 7 atom types and 3 bond types as prescribed by Figure 3.1.1.
2. The molecular descriptor is created by extracting from the molecule a complete set of small fragments.
3. Frequency counts are evaluated for all the fragments so that a multi-set or bag can be generated.

When these steps are completed, the multi-set counts are placed into a vector that has a position for each of the different possible fragments. For example, if fragment size is limited to 2 atoms this vector would have a dimensionality of $7*7*3 = 147$.

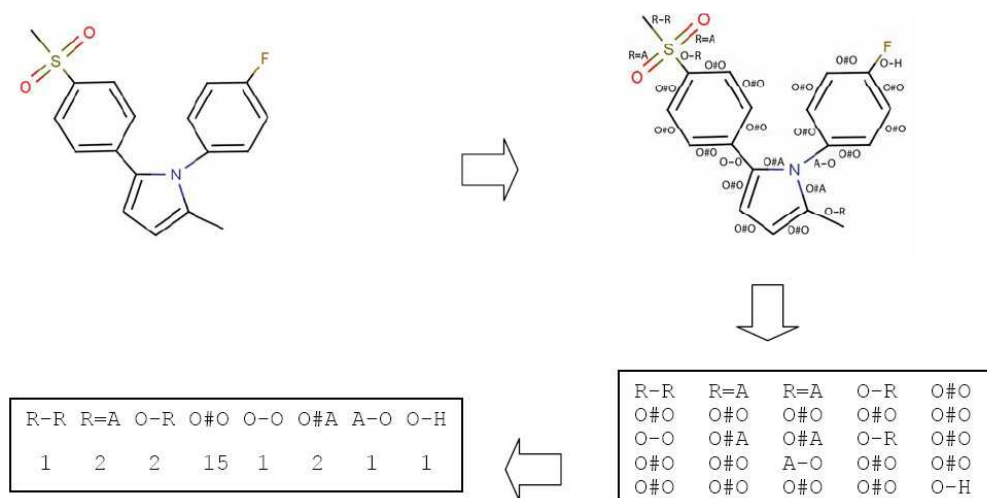


Figure 3.1.2: Processing steps for the VSMMD.

3.2 VSM and VSMMD compared

The motivation for the VSMMD descriptor comes from the “bag-of-words” approach [89] that is based on the vector space model (VSM). Roughly speaking, the atom fragments, extracted in the VSMMD process, correspond to the document words and phrases extracted in the VSM. The molecular descriptor is then analogous to the text document and much of the analysis used in the bag-of-words strategy can be brought over to the VSMMD setting.

In practice, we have found that the success of VSMMD is greatly enhanced by utilizing fragments that contain at least two atoms, for example, $f_a = 2$ or $f_a = 3$. This adoption of a higher level of structure corresponds to the incorporation of phrase structures in the VSM. As the value of f_a increases there is a point of diminished returns due to the combinatorial explosion of fragment possibilities. To help reduce this “curse of dimensionality”, we can remove the vector entries that

Type ID	General Fragment Type	Atom Count(f_a)
1)	$A_T B_T A_T$	2
2)	$A_T B_T A_T B_T A_T$	3
3)	$A_T B_T A_T B_T A_T B_T A_T$	4
4)	$A_T B_T A_T B_T A_T$	4
	B_T	
	A_T	

Table 3.2.1: General format of the fragment dictionary (A_T =Atom Type, B_T =Bond Type)

have been observed to have frequency counts equal to zero across all molecules under consideration.

In the VSM, a language dictionary can be defined using some permanent predefined set of words. In our model, according to the encoding scheme, the dictionary will be the collection of all possible atom type (A_T) and bond type (B_T) labeled graphs that arise from molecular fragments that are restricted to having atom counts of 2, 3, or 4. Table 3.2.1 shows the general format of our dictionary with f_a limited to 4. The last entry of this table represents a four atom fragment in which a central atom is bonded to three other atoms.

In the most general case, a molecular descriptor is represented by a bag of fragments, each an entry in the dictionary.

3.3 Analysis based on VSMMD

We now describe the notation and mathematical setting used in VSMMD. After the molecular descriptor d is generated, we represent the descriptor as a column vector in an m dimensional space using the mapping

$$\phi : d \mapsto \phi(d) = (q(f_1, d), q(f_2, d), \dots, q(f_m, d))^T \in \mathbb{R}^m \quad (3.3.1)$$

where $q(f_i, d)$ is the frequency of the fragment f_i in the descriptor d .

The use of $\phi(\cdot)$ in this last equation is deliberate since we want to view this mapping as the type of kernel function that is used in the bag-of-words strategy described by Shawe-Taylor and Cristianini [89]. Via this mapping, each molecular descriptor is taken over to an m -dimensional vector, where m is the size of the dictionary. Although m could be very large, the typical vector generated in this way is usually quite sparse (just as vectors in the VSM are sparse).

Working with n molecules, we can apply the mapping repeatedly to generate a succession of column vectors: $\phi(d_1), \phi(d_2), \dots, \phi(d_n)$. Computation of the vector space kernel is done by calculating the fragment-descriptor matrix F with rows indexed by the fragments and columns indexed by the descriptors:

$$F = \begin{pmatrix} \phi(d_1) & \dots & \phi(d_n) \end{pmatrix} = \begin{pmatrix} q(f_1, d_1) & \dots & q(f_1, d_n) \\ \vdots & \ddots & \vdots \\ q(f_m, d_1) & \dots & q(f_m, d_n) \end{pmatrix}. \quad (3.3.2)$$

The entry at position (i, j) gives the frequency of fragment f_i in document d_j . Subsequently, we create the kernel matrix as

$$K = F^T F \quad (3.3.3)$$

corresponding to the vector space kernel

$$k(d_i, d_j) = \langle \phi(d_i), \phi(d_j) \rangle = \sum_{z=1}^m q(f_z, d_i) q(f_z, d_j). \quad (3.3.4)$$

With the vector space kernel, we can apply a kernel-based method to generate a predictor of any one of several biological activities, for example: affinity of ligands used as therapeutic agents or ADMET properties. To reduce the row dimensions of the F matrix, it is possible to employ component selection procedures that attempt to eliminate fragment entries that appear to have little or no influence on the prediction algorithm being developed.

3.4 Data Fusion for VSMMD

The VSMMD strategy is based on the extraction of molecular fragments that are comprised of chains of bonded atoms. Consequently, different fragment lengths will give different results. In this section, we give a useful strategy to combine different kernel matrices by computing a weighted average of all kernel matrices generated by different fragment lengths. Suppose we have p kernel matrices K_1, K_2, \dots, K_p . We can compute the average of all kernels normalized by positive constants $\beta_1, \beta_2, \dots, \beta_p > 0$:

$$K = \frac{1}{p} \sum_{j=1}^p \frac{K_j}{\beta_j}. \quad (3.4.5)$$

Note that the choice of β_j may be arbitrary, as long as they are positive. However, a recent study [19] gave theoretical evidence that β_j should be proportional to the trace of K_j divided by the original dimension of the data.

Suppose we want to study four different fragment types as shown in Table 3.2.1. By using equation (3.3.3), we can create four different kernel matrices K_1, K_2, K_3, K_4 corresponding to each fragment type. To perform data fusion of these four kernel matrices, we use equation (3.4.5) with $p = 4$ and β_j is the trace of K_j divided by the total number of different descriptors for that type j fragment.

3.5 Experimental Results

3.5.1 Data

In our QSAR study, eight different data sets were used to test the ability of the VSMMD to predict biological activities. All these data sets contain real valued QSAR inhibitor data.

The eight QSAR data sets are from Sutherland *et al.* [96]. The first data set contains 114 angiotensin converting enzyme (ACE) inhibitors collected by Depriest and colleagues [20] with pIC50 values ranging from 2.1 to 9.9. pIC50 is a measure of the effectiveness of a compound in inhibiting biological function. The second data set contains 111 acetylcholinesterase (AChE) inhibitors assembled by Sugimoto *et al.* [94] and split into training and test data sets by Golbraikh *et al.* [32] with pIC50 values ranging from 4.3 to 9.5. The third data set contains 163 ligands for the benzodiazepine receptor (BZR) assembled from the work of Haefely [37] and his

colleagues and reported by Maddalena et al. [70] with pIC50 values ranging from 5.5 to 8.9. The fourth data set contains 322 cyclooxygenase-2 (COX2) inhibitors collected by Seibert and colleagues [46] and subsequently utilized in a QSAR study by Chavatte *et al.* [12] with each inhibitor having pIC50 values ranging from 5.5 to 8.9. The fifth data set contains 397 dihydrofolate reductase inhibitors (DHFR) collected by Queener and colleagues [9] and set up as data sets by Sutherland and Weaver [97] with each inhibitor having pIC50 values for rat liver enzyme ranging from 3.3 to 9.8. The final three data sets were prepared by Klebe and his colleagues [8, 31, 60]. They include 66 inhibitors of glycogen phosphorylase b (GPb) with pKi values ranging from 1.3 to 6.8, 76 thermolysin inhibitors (THER) having pKi values ranging from 0.5 to 10.2 and 88 thrombin inhibitors (THR) with pKi values ranging from 4.4 to 8.5.

In all our experiments the data were separated into the same training and testing sets used by Sutherland *et al.* [96]. Leave-one-out cross validation was used to obtain the best parameters for model generation.

3.5.2 Implementation details

To identify the physicochemical properties of each atom, we implemented our descriptor generation program using the chemical development kit (CDK) [92, 93] programmed in Java. As illustrated in Figure 3.1.2, we traversed each molecule once and the kernel matrix K was generated in a few seconds for each complete data set.

For classification, we used SVMlight [48] to perform the support vector regres-

sion experiments.

3.5.3 QSAR results

To perform regression, we made use of a linear ε -insensitive Support Vector Regression (SVR) algorithm as mentioned in the previous chapter. In our experiments, ε was set to 0.1, and the Gaussian kernel [89]

$$k(d_i, d_j) = e^{-\langle \phi(d_i), \phi(d_j) \rangle / 2\sigma^2}, \quad (3.5.6)$$

was used for the SVR. Two parameters, σ and C , had to be evaluated through cross-validation. A simple grid search was used to select the parameters σ and C . In the grid search, we considered only the Gaussian kernels with parameter σ equal to values ranging from 1 to 60 in steps of 1. The parameter C was chosen from values ranging from 100 to 1000 in steps of 100. For all eight data sets, we obtained the best σ and C by leave-one-out cross validation using only the training set. Table 3.5.2 gives the value of σ for all eight data sets across various descriptor models.

The prediction accuracy was assessed using the designated test sets and subsequently residual r^2 statistics [42] were obtained. A residual r^2 value close to one indicates a better prediction. The residuals, for all the eight data sets, across various descriptor models are given in Table 3.5.3.

TypeID	ACE	AchE	BZR	COX2	DHFR	GPB	THER	THR
1	1	5	6	35	9	1	3	3
2	2	2	35	15	2	8	5	5
3	1	1	15	2	20	2	1	1
4	2	1	35	2	10	1	60	60
1 ∪ 2	1	1	35	10	1	7	2	2
1 ∪ 2 ∪ 3	1	1	15	10	6	2	1	1
1 ∪ 2 ∪ 3 ∪ 4	3	2	15	2	3	1	1	1

Table 3.5.2: σ values derived using cross-validation. TypeID is the fragment type ID defined in Table 3.2.1

3.6 Discussion

For the eight data sets, Sutherland *et al.* [96] tried 7 different descriptor sets. They include CoMFA [16], CoMSIA basic [60], CoMSIA Extra [59], EVA [26], HQSAR [69], Cerius2 2D and Cerius2 2.5D. Of these descriptor sets, only the Cerius2 2.5D set was available for experimental comparison. Their experiments show that none of these descriptor sets can completely outperform the others. For a detailed description of the Cerius2 2.5D descriptor, the reader may consult Sutherland *et al.* [96].

To compare our VSMMD results with those of Sutherland *et al.*, we worked with the given Cerius2 2.5D descriptors and then performed a regression using SVR. In order to provide a fair comparison, full cross-validation studies were run on both VSMMD and Cerius2 2.5D descriptors. For Cerius2 2.5D, we performed leave-one-out cross validation on the training set only to obtain the optimal value

TypeID	ACE	AchE	BZR	COX2	DHFR	GPb	THER	THR
1	0.45	0.48	0.32	0.35	0.53	0.23	0.39	0.16
2	0.39	0.5	0.38	0.31	0.5	0.76	0.33	0.24
3	0.4	0.35	0.31	0.31	0.59	0.56	0.29	0.05
4	0.39	0.35	0.23	0.26	0.57	0.26	0.28	0.14
1 \cup 2	0.4	0.48	0.38	0.34	0.5	0.73	0.35	0.24
1 \cup 2 \cup 3	0.42	0.36	0.38	0.32	0.57	0.55	0.38	0.25
1 \cup 2 \cup 3 \cup 4	0.4	0.22	0.35	0.41	0.61	0.25	0.36	0.29

Table 3.5.3: r^2 statistics using SVR on the testing set

for the two parameters C and σ . For the VSMMD, we performed leave-one-out cross validation on the training set only to obtain the optimal value for the three parameters C , σ and the fragment type ID as defined in Table 3.2.1. Table 3.6.4 summarizes the comparison between our results and those of Sutherland *et al.* Both residual r^2 statistics and the cross-validated residual statistics q^2 are shown.

In Table 3.6.4, it can be seen that by using our VSMMD descriptor, we obtained a higher residual r^2 indicating that our descriptor set performs significantly better than the Cerius2 2.5D descriptor in five out of eight data sets. As mentioned in Sutherland *et al.*, Cerius2 2.5D descriptors perform reasonably well on these two data sets and so these empirical results indicate that the VSMMD descriptor set is very suitable for kernel models.

Sutherland *et al.* also utilized various model-building methods, including Partial Least Squares (PLS) [112], methods related to Genetic Algorithms [84, 21], and neural networks [78, 56]. Table 3.6.5 summarizes the comparison between our

		ACE	Ache	BZR	COX2	DHFR	GPb	THER	THR
Cerius2	r^2	0.45	0.31	0.21	0.32	0.51	0.26	0.39	0.3
	q^2	0.66	0.15	0.15	0.56	0.53	0.35	0.7	0.35
	$std(q^2)$	0.3539	0.2542	0.286	0.2355	0.3969	0.2299	0.3798	0.2638
VSMMD	r^2	0.39	0.48	0.32	0.41	0.57	0.76	0.38	0.29
	q^2	0.74	0.23	0.41	0.42	0.6	0.57	0.31	0.22
	$std(q^2)$	0.0535	0.1673	0.3603	0.3508	0.062	0.29	0.1565	0.1601

Table 3.6.4: Comparison of the residual statistics r^2 comparing the VSMMD descriptor set and the Cerius2 2.5D descriptor set (both using SVR)

results and the best results reported by Sutherland *et al.*

Table 3.6.5 gives evidence that by using the VSMMD descriptor, we obtained a higher residual r^2 in four out of eight data sets and performed robustly across all eight data sets.

We also compared the VSMMD kernel with the 2D graph spectrum kernel [72] and the 3D pharmacophore kernel [71] based on a regression using SVR utilizing the same data set. For the 2D graph spectrum kernel, we performed 10-fold cross validation, on the training set only, to obtain the optimal value for the three parameters C , σ , and the fragment length. For the 3D pharmacophore kernel, we performed 10-fold cross validation, on the training set only, to obtain the optimal value for the two parameters C and σ . For the VSMMD, we performed leave-one-out cross validation on the training set only to obtain the optimal value for the three parameters C , σ and the fragment type ID as defined in Table 3.2.1. Table 3.6.6 summarizes the comparison of our kernel results with others.

Table 3.6.6 gives evidence that by using the VSMMD kernel, we obtained a

	ACE	AchE	BZR	COX2	DHFR	GPb	THER	THR
VSMMD	0.39	0.48	0.32	0.41	0.57	0.76	0.38	0.29
Cerius 2.5D	0.51	0.16	0.20	0.27	0.49	0.04	0.07	0.28
CoMFA	0.49	0.47	0.00	0.29	0.59	0.42	0.54	0.63
CoMSIA	0.52	0.44	0.08	0.03	0.52	0.46	0.36	0.55
EVA	0.36	0.28	0.16	0.17	0.57	0.49	0.36	0.11
HQSAR	0.30	0.37	0.17	0.27	0.63	0.58	0.53	-0.25

Table 3.6.5: Comparison of the residual statistics r^2 comparing the VSMMD descriptor set using SVR and the various descriptor sets reported by Sutherland *et al.* using PLS

higher residual r^2 in seven out of eight data sets. Furthermore, the computational time for the VSMMD kernel is much faster than the 3D pharmacophore kernel.

In order to investigate whether our descriptors have the ability to capture the important properties of a molecule, we performed a more detailed analysis of the COX-2 data set. Reviewing the data set, we considered the molecules that had the top 10 highest activity values. For each molecule, we obtained the five descriptors with the highest rank using inverse document frequency [89]. These ten molecules, together with the location of the highest ranked descriptor (circled in red) are displayed in Figure 3.6.3. Among the ten molecules, the cyclopentene ring was the highest ranked descriptor in seven of the high affinity molecules. From medicinal chemistry studies, we know that cyclopentene derivatives are one of the first series of diaryl-substituted cycles successfully investigated as COX-2 inhibitors [66, 82]. This empirical evidence demonstrates that our descriptors are able to specify important properties of the molecule.

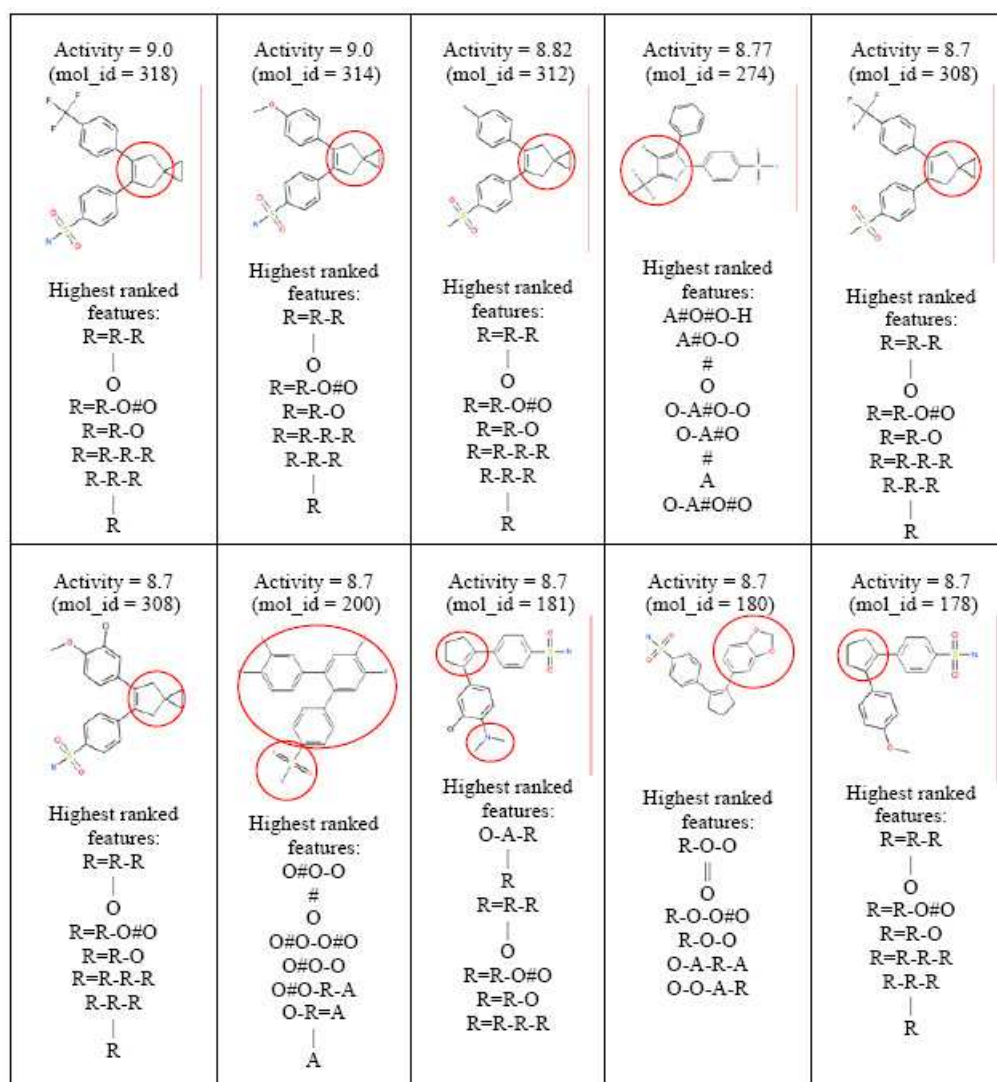


Figure 3.6.3: Location of highly ranked descriptor (circled) in selected COX-2 molecules.

		ACE	AchE	BZR	COX2	DHFR	GPb	THER	THR	
2D	r^2	0.2	0.06	0.11	0.27	0.48	0.07	0.16	0.19	
	Graph	q^2	0.4	0.06	0.31	0.44	0.53	0.21	0.12	0.16
		$std(q^2)$	0.017	0.026	0.093	0.092	0.148	0.074	0.06	0.08
3D	r^2	0.39	0.36	0.3	0.32	0.63	0.48	0.28	NA*	
	pharma-	q^2	0.69	0.276	0.42	0.47	0.66	0.48	0.16	NA*
	cophore	$std(q^2)$	0.056	0.119	0.046	0.134	0.057	0.191	0.09	NA*
VSMMD	r^2	0.39	0.48	0.32	0.41	0.57	0.76	0.38	0.29	
	q^2	0.74	0.23	0.41	0.42	0.6	0.57	0.31	0.22	
	$std(q^2)$	0.0535	0.1673	0.3603	0.3508	0.062	0.29	0.1565	0.1601	

Table 3.6.6: Comparison of the residual statistics r^2 comparing the VSMMD with the graph kernel and the pharmacophore kernel (all using SVR) (*supplied program failed to provide results)

3.7 Conclusion

In this chapter, we have introduced a constitutional descriptor, the vector space model molecular descriptor (VSMMD) that is similar to the vector space model used by researchers in information retrieval. The significance of this model rests on its mathematical setting. Molecular descriptors are converted to vectors that represent the molecules as points in a descriptor input space. Using kernel method strategies these points can be mapped into a feature space that is suitable for various machine learning techniques such as classification by support vector machines or regression by support vector regression algorithms.

Furthermore, since the VSMMD descriptor is directly based on molecular fragments, it is possible to use a ranking procedure to determine those fragments that

are most important in the prediction of high affinity molecules. This interpretability is very important in drug design efforts since it gives valuable information about the structure of high affinity drug candidates leading to a more focused search in high-throughput screening exercises.

Based on these encouraging empirical results, we are confident that this vector space modeling of molecular descriptors has the potential to provide a versatile mathematical setting for further developments in this area.

Chapter 4

Component Selection

High dimensionality of descriptors poses a challenge for statistical learning algorithms to formulate predictors. The minimum number of samples required to ensure a high level of prediction confidence rapidly increases as the dimensionality of the descriptors increases. We need computational strategies to do component selection, keeping only those components in the initial version of a molecular descriptor that are useful for later study.

In this chapter, we present a new component selection algorithm KACS (Kernel Alignment Component Selection) based on kernel alignment for QSAR studies.

4.1 Introduction

Component selection, sometimes called “feature section” is an essential data pre-processing step that is needed in machine learning, wherein a subset of the descriptor components is selected for participation in a learning algorithm. The idea of

component selection is to remove irrelevant or redundant components from a set of computed components contained in the initial version of a descriptor. The final objective is to utilize these improved descriptors as training data in the construction of a good predictor or classifier for particular machine learning problems. With a good selection of components, the following benefits can be achieved:

1. improving the prediction accuracy of the predictor,
2. reducing computational time and storage requirements, and
3. providing a better understanding and clear interpretation of the underlying process that generated the data.

In many modern scientific research studies, component selection is an indispensable step of data analysis. In general, a subset of components is chosen based on the following two criteria:

1. find a subset of the available components that gives the smallest expected generalization error; or
2. find a smallest subset of the available components that is below some pre-specified maximum allowable generalization error.

In this chapter, our research studies mainly focus on selecting a subset of components that is subject to the first criterion. That is, our objective is to select a subset of components that minimizes the expected error.

Currently, many methods have been developed for component selection. In general, there are three main approaches: filter methods, wrapper methods, and embedded methods. We will review these three approaches in the following section.

4.1.1 Filter Methods

Filter methods usually select subsets of components as a pre-processing step, using an approach that is independent of the classifier's objectives. In other words, the components are selected with regard to some predefined relevance measure that is independent of classifier performance. Common measures include the correlation index and mutual information.

For example, Pearson's correlation coefficient is commonly used to eliminate redundant components. It is a statistical method used to measure the linear relationship between two variables. The correlation is also commonly used in conjunction with principle component analysis [108] to rank the importance of the components. Pearson's correlation coefficient is given by:

$$r = \frac{(\sum xy) - n\bar{x}\bar{y}}{\sqrt{(\sum x^2) - n\bar{x}^2}\sqrt{(\sum y^2) - n\bar{y}^2}}, \quad (4.1.1)$$

where x and y are two random variables, \bar{x} and \bar{y} are their corresponding means.

Mutual information, sometimes called information gain, is also widely used in QSAR applications [107]. The information of the component is defined in terms of entropy of the component treated as a random variable.

The mutual information of two discrete random variables X and Y can be

defined by:

$$I(X; Y) = \sum_{y \in Y} \sum_{x \in X} p(x, y) \log \left(\frac{p(x, y)}{p_1(x)p_2(y)} \right) \quad (4.1.2)$$

where $p(x, y)$ is the joint probability distribution function of X and Y , while $p_1(x)$ and $p_2(y)$ are the marginal probability distribution functions of X and Y respectively.

Based on this notion, various measures relating the information shared between two components or between a component and its related activity can be defined. These strategies are commonly used in QSAR studies to rank the components.

4.1.2 Wrapper Methods

Wrapper methods utilize some particular learning machine as a black box to score subsets of variables according to their predictive power [35]. Some common models include simulated annealing, genetic algorithms, and sequential forward/backward elimination.

In Simulated Annealing (SA) [58], a function usually based on Boltzmann's distribution, is utilized to minimize the error of the model built using a subset of components. The SA strategy usually starts from a random configuration of subsets and then attempts to find a better subset of the components by altering the subset currently considered to be the best. Next, SA evaluates the prediction error of the new subset. If the new solution is better than the current best solution, it will update the current best solution. If the new solution is slightly worse than the current best solution, it might still be retained based on a comparison that

involves the Boltzmann distribution. The iterative procedure continues until no further enhancement can be found or until the stopping criteria is satisfied.

The genetic algorithm, similar to SA, relies on a guided random process to explore the space of component subsets. Venkatraman *et al.* [107] provide a good discussion on how a genetic algorithm can be applied to component selection.

In addition to the stochastic algorithms mentioned above, the sequential component forward / backward selection (SCFS / SCBS) algorithm [114, 113] operates in a deterministic manner. Both SCFS and SCBS use a greedy approach to search the component subsets. For SCFS, we begin with one component that leads to the best prediction, then each component is individually added to the current subset and the errors of the resulting models are quantified. The component that is the best in reducing the error is incorporated into the subset. We repeat the process until some stopping criteria has been achieved. For SCBS, we begin with the full set of components, then each component is individually eliminated from the current subset and the errors for the resulting models are quantified. The component that leads to a model with the highest error is removed from the current subset. The SCBS process is repeated until some stopping criterion has been met.

4.1.3 Other Methods

In addition to purely filter or wrapper-based component selection methods, some methods utilize fusion of these two approaches. An example of this “embedded method” involves the combining of the correlation index strategy with the greedy algorithm [33].

4.1.4 Component Selection for Computer-Aided Drug Design

In the literature, over 1000 different descriptors have been proposed [102]. Selecting appropriate descriptor components to represent a molecule is a critical and important step. It has been shown in [87] that the quality of an inferred model strongly depends on the selected components. Thus, component selection for computational drug design remains a topic of high importance in the QSAR community.

Filter based component selection algorithms are becoming popular tools in QSAR studies. Some examples:

1. Merkwirth *et al.* [75] created a cluster of components using the correlation coefficient of candidate components and then retained only one representative for each cluster;
2. Guha *et al.* [33] used a random forest model to provide an alternate measure of the importance of a component;
3. Liu [67] used information gain to select components for different data sets; and
4. Venkatraman *et al.* [107] combined a genetic algorithm with mutual information to select components.

Among the entire wrapper based component selection algorithms, genetic algorithms have been often used with a wide range of mapping methods. Examples

include neural networks [33], nearest neighbor methods [6] and random forest methods [33]. Simulated Annealing has also been used in component selection for QSAR studies. For example, Sutter *et al.* [98] employed generalized simulated annealing to select an optimal set of components. Variants of forward/backward recursive component selection methods have been used in numerous QSAR studies. For example, Xue *et al.* applied both forward [114] and backward recursive component selection [113] methods to different QSAR data sets.

4.2 Kernel Alignment Component Selection (KACS)

In this section, we present a new component selection algorithm KACS (Kernel Alignment Component Selection) for a QSAR study based on kernel alignment [17]. Kernel alignment has been developed as a measure of similarity between two kernel functions. In our algorithm, we refine kernel alignment as an evaluation tool, using recursive component elimination to eventually select the most important components for classification. Theoretical and empirical analyses follow after the algorithm is presented.

4.2.1 Kernel Alignment

Kernel alignment has been developed as a measure of similarity between two kernel matrices [17]. Suppose we are working with a data space X containing n training samples stored as column vectors: x_1, x_2, \dots, x_n . Given kernel functions $k_1(x, y)$ and $k_2(x, y)$ we can construct their corresponding Gram (or kernel) matrices K_1 and

K_2 defined as $\{K_1\}_{ij} = k_1(x_i, x_j)$ and $\{K_2\}_{ij} = k_2(x_i, x_j)$. The empirical alignment of matrix K_1 with matrix K_2 , with respect to the sample data X , is denoted by $KA(K_1, K_2)$ and is given by:

$$KA(K_1, K_2) = \frac{\langle K_1, K_2 \rangle_F}{\sqrt{\langle K_1, K_1 \rangle_F \langle K_2, K_2 \rangle_F}}, \quad (4.2.3)$$

where $\langle K_1, K_2 \rangle_F$ is the Frobenius inner product which measures the similarity between two different kernel matrices K_1 and K_2 . It is defined as:

$$\langle K_1, K_2 \rangle_F = \sum_{i=1}^n \sum_{j=1}^n k_1(x_i, x_j) k_2(x_i, x_j). \quad (4.2.4)$$

In the case of binary classification, for a given new input $x \in X$, we want to predict the corresponding response y , where $y = +1$ or -1 . An “ideal” kernel matrix can be formed by calculating:

$$K_{ideal} = yy^T. \quad (4.2.5)$$

The similarity measure between K and this ideal can be computed as:

$$KA(K, K_{ideal}) = \frac{\langle K, K_{ideal} \rangle_F}{\sqrt{\langle K, K \rangle_F \langle K_{ideal}, K_{ideal} \rangle_F}} = \frac{\langle K, K_{ideal} \rangle_F}{n \sqrt{\langle K, K \rangle_F}} = \frac{y^T K y}{n \|K\|_F}, \quad (4.2.6)$$

where $\|K\|_F$ is the Frobenius norm of the kernel matrix K .

Kernel alignment can be viewed as the cosine of the angle between two kernel matrices K_1 and K_2 . It can also be considered as a Pearson correlation coefficient between the random variables $K_1(x, z)$ and $K_2(x, z)$.

4.2.2 SVM based Recursive Component Elimination (RCE)

SVM based Recursive Component Elimination (SVM-RCE) is an iterative sequential backward component selection algorithm proposed by Guyon *et al.* [36] for

gene selection using a support vector machine (SVM). Initially the SVM classifier is trained with the full component set. The significance of a component is characterized by the weight that the SVM optimization assigns to that component. Guyon proposed using the weight magnitude from the SVM as a ranking criterion. At each iteration step, the component with the smallest ranking criterion is eliminated. By doing so, the components that contribute least to the maximization of the separation margin are removed. SVM-RCE is a wrapper based approach. At each step, the SVM must be trained in order to obtain this ranking criterion. In Guyon's studies, a linear-SVM classifier was used.

Later, Yu *et al.* [116] modified the SVM-RCE algorithm by using a nonlinear SVM classification system of polynomial kernels for prediction of drug activity. Xue *et al.* [113] further extended the SVM-RCE strategy by using a nonlinear SVM classification system with a Gaussian kernel.

4.2.3 Kernel Alignment Component Selection Algorithm

Instead of using a wrapper based approach, we propose a filter based component selection algorithm using kernel alignment as a ranking criterion. In general, this will be a faster computation.

To begin our approach, we recompute the kernel K_{ideal} by centering the response as follows:

$$K_{ideal} = y^* y^{*\top} \text{ where } y^* = y - \bar{y} \quad (4.2.7)$$

where \bar{y} represents the mean of the y target values in the training set. We start with a full set of components. Next, we recursively remove the component that produced the maximum difference between the kernel alignment value evaluated with the reduced component set and the kernel alignment value evaluated with the current component set, that is, we remove the component that maximizes the difference $KA(K_{removed}, K_{ideal}) - KA(K_{current}, K_{ideal})$. A valid stopping condition can be formulated based on a combination of changes in the kernel alignment value and changes in the leave-one-out error in each step. A reasonable stopping condition would be:

- Stops the algorithm when $KA(K_{removed}, K_{ideal}) - KA(K_{current}, K_{ideal}) < 0$.

Consequently, we stopped with the component set that maximized the kernel alignment.

The following steps summarize our algorithm:

- 1 **Kernel Alignment Component Selection Algorithm**
- 2 Input: training set X , label set y , stopping condition sc .
- 3 Compute the kernel alignment value $KA(K_{current}, K_{ideal})$ for X ;
- 4 *for* $p = 1$ to the number of components remaining in X
- 5 Compute the Kernel matrix $K_{removed(p)}$ with component p removed;
- 6 Compute the kernel alignment value $KA(K_{removed(p)}, K_{ideal})$;
- 7 *endfor*

- 8 Remove the component providing the maximum $KA(K_{removed(p)}, K_{ideal}) - KA(K_{current}, K_{ideal})$;
- 9 Update X ;
- 10 Repeat step 3 to step 9 until some stopping condition sc has been met;
- 11 return X ;

4.2.4 Theoretical Analysis

In the Kernel Alignment Component Selection Algorithm, for each iteration, we removed the component that let the remaining components maximize

$$KA(K_{removed(p)}, K_{ideal}) - KA(K_{current}, K_{ideal}). \quad (4.2.8)$$

In this subsection, we will show that maximizing (4.2.8) will lower the generalization error bound in an SVM classifier.

Suppose we have n training data pairs $((x_1, y_1), \dots, (x_n, y_n)) \in X \times Y$ where X is the input space and Y is the set of corresponding output values. In the case of binary classification, $x_i \in \mathbb{R}^m$ and $y_i \in \{+1, -1\}$. Partition the training data into two groups c^+ and c^- , where c^+ contains the samples with output value $y = +1$ and c^- contains the samples with output value $y = -1$. We will use the analysis of variance (ANOVA) technique to study the source of variation of the training data.

The within-class sum of squares matrix SS_w , is:

$$SS_w = \sum_{x_i \in c^+} (x_i - \bar{x}_+)(x_i - \bar{x}_+)^T + \sum_{x_i \in c^-} (x_i - \bar{x}_-)(x_i - \bar{x}_-)^T \quad (4.2.9)$$

where \bar{x}_+ and \bar{x}_- are the mean vectors for the groups c^+ and c^- , respectively.

The between-classes sum of squares matrix SS_B , is:

$$SS_B = n_+(\bar{x}_+ - \bar{x})(\bar{x}_+ - \bar{x})^T + n_-(\bar{x}_- - \bar{x})(\bar{x}_- - \bar{x})^T \quad (4.2.10)$$

where n_+ and n_- are the sample sizes of group c^+ and c^- , respectively, and \bar{x} is the mean vector for the entire training data set.

The total sum of squares matrix SS_T , is

$$SS_T = \sum_{x_i \in c^+} (x_i - \bar{x})(x_i - \bar{x})^T + \sum_{x_i \in c^-} (x_i - \bar{x})(x_i - \bar{x})^T = SS_B + SS_w \quad (4.2.11)$$

By using a kernel function, $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$, we get the implicit definition of ϕ , where ϕ maps a sample point x into a feature space FS . The similarity function $k(.,.)$ is called a kernel, and ϕ is called its implied mapping. The feature space FS is also called the reproducing kernel Hilbert space (RKHS) associated with k . In the feature space FS , the norm of $\phi(x)$ is given by:

$$\|\phi(x)\| = \sqrt{\|\phi(x)\|^2} = \sqrt{\langle \phi(x), \phi(x) \rangle} = \sqrt{k(x, x)}. \quad (4.2.12)$$

The norm may be used to calculate the length of the line joining two images $\phi(x_1)$ and $\phi(x_2)$:

$$\begin{aligned} \|\phi(x_i) - \phi(x_j)\|^2 &= \langle \phi(x_i) - \phi(x_j), \phi(x_i) - \phi(x_j) \rangle \\ &= \langle \phi(x_i), \phi(x_i) \rangle - 2 \langle \phi(x_i), \phi(x_j) \rangle + \langle \phi(x_j), \phi(x_j) \rangle \\ &= k(x_i, x_i) - 2k(x_i, x_j) + k(x_j, x_j). \end{aligned} \quad (4.2.13)$$

The mean \bar{x}^ϕ of the data set in the feature space is:

$$\bar{x}^\phi = \frac{1}{n} \sum_{i=1}^n \phi(x_i). \quad (4.2.14)$$

The analysis is similar to that just described for the input space. In the feature space, the within-class sum of squares matrix SS_w^ϕ , is:

$$SS_w^\phi = \sum_{x_i \in c^+} (\phi(x_i) - \bar{x}_+^\phi)(\phi(x_i) - \bar{x}_+^\phi)^\top + \sum_{x_i \in c^-} (\phi(x_i) - \bar{x}_-^\phi)(\phi(x_i) - \bar{x}_-^\phi)^\top \quad (4.2.15)$$

where \bar{x}_+^ϕ and \bar{x}_-^ϕ are the respective mean vectors for the images of classes c^+ and c^- in the feature space. The between-classes sum of squares matrix SS_B^ϕ , is:

$$SS_B^\phi = n_+(\bar{x}_+^\phi - \bar{x}^\phi)(\bar{x}_+^\phi - \bar{x}^\phi)^\top + n_-(\bar{x}_-^\phi - \bar{x}^\phi)(\bar{x}_-^\phi - \bar{x}^\phi)^\top \quad (4.2.16)$$

and the total sum of squares matrix SS_T^ϕ , is:

$$SS_T^\phi = \sum_{i=1}^n (\phi(x_i) - \bar{x}^\phi)(\phi(x_i) - \bar{x}^\phi)^\top = SS_B^\phi + SS_w^\phi \quad (4.2.17)$$

Since the feature space can have arbitrary high dimensions, the matrices SS_w^ϕ , SS_B^ϕ and SS_T^ϕ are almost always singular [109] and cannot be explicitly computed in terms of the kernel matrix [77]. In order to express the sum of squares matrices in terms of kernel matrix, we utilize the following identities:

Identity 4.2.1. *The trace of the between-classes sum of squares matrix SS_B^ϕ is given by: $\text{trace}(SS_B^\phi) = \frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - \frac{1}{n} \sum_{x_i, x_j} k(x_i, x_j)$*

Proof.

$$\begin{aligned}
\text{trace}(SS_B^\phi) &= \text{trace} \left[n_+(\bar{x}_+^\phi - \bar{x}^\phi)(\bar{x}_+^\phi - \bar{x}^\phi)^\top + n_-(\bar{x}_-^\phi - \bar{x}^\phi)(\bar{x}_-^\phi - \bar{x}^\phi)^\top \right] \\
&= n_+(\bar{x}_+^\phi - \bar{x}^\phi)^\top(\bar{x}_+^\phi - \bar{x}^\phi) + n_-(\bar{x}_-^\phi - \bar{x}^\phi)^\top(\bar{x}_-^\phi - \bar{x}^\phi) \\
&= n_+ \left[\|\bar{x}_+^\phi\|^2 - 2 \langle \bar{x}_+^\phi, \bar{x}^\phi \rangle + \|\bar{x}^\phi\|^2 \right] + n_- \left[\|\bar{x}_-^\phi\|^2 - 2 \langle \bar{x}_-^\phi, \bar{x}^\phi \rangle + \|\bar{x}^\phi\|^2 \right] \\
&= n_+ \left[\frac{1}{n_+^2} \sum_{x_i, x_j \in c^+} k(x_i, x_j) - 2 \frac{1}{n_+ n} \sum_{x_i \in c^+, x_j} k(x_i, x_j) + \frac{1}{n^2} \sum_{x_i, x_j} k(x_i, x_j) \right] \\
&\quad + n_- \left[\frac{1}{n_-^2} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - 2 \frac{1}{n_- n} \sum_{x_i \in c^-, x_j} k(x_i, x_j) + \frac{1}{n^2} \sum_{x_i, x_j} k(x_i, x_j) \right] \\
&= \frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) - 2 \frac{1}{n} \sum_{x_i \in c^+, x_j} k(x_i, x_j) + \frac{n_+}{n^2} \sum_{x_i, x_j} k(x_i, x_j) \\
&\quad + \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - 2 \frac{1}{n} \sum_{x_i \in c^-, x_j} k(x_i, x_j) + \frac{n_-}{n^2} \sum_{x_i, x_j} k(x_i, x_j) \\
&= \frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - \frac{2}{n} \left[\sum_{x_i \in c^+, x_j} k(x_i, x_j) + \sum_{x_i \in c^-, x_j} k(x_i, x_j) \right] \\
&\quad + \frac{n_+ + n_-}{n^2} \sum_{x_i, x_j} k(x_i, x_j) \\
&= \frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - \frac{2}{n} \sum_{x_i, x_j} k(x_i, x_j) + \frac{1}{n} \sum_{x_i, x_j} k(x_i, x_j) \\
&= \frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - \frac{1}{n} \sum_{x_i, x_j} k(x_i, x_j)
\end{aligned} \tag{4.2.18}$$

□

Identity 4.2.2. *The trace of the within-class sum of squares matrix SS_w^ϕ is given*

$$\text{by: } \text{trace}(SS_w^\phi) = \sum_{x_i} k(x_i, x_i) - \frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) - \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j)$$

Proof.

$$\begin{aligned}
\text{trace}(SS_w^\phi) &= \text{trace} \left[\sum_{x_i \in c^+} (\phi(x_i) - \bar{x}_+^\phi)(\phi(x_i) - \bar{x}_+^\phi)^\top + \sum_{x_i \in c^-} (\phi(x_i) - \bar{x}_-^\phi)(\phi(x_i) - \bar{x}_-^\phi)^\top \right] \\
&= \sum_{x_i \in c^+} (\phi(x_i) - \bar{x}_+^\phi)^\top (\phi(x_i) - \bar{x}_+^\phi) + \sum_{x_i \in c^-} (\phi(x_i) - \bar{x}_-^\phi)^\top (\phi(x_i) - \bar{x}_-^\phi) \\
&= \sum_{x_i \in c^+} \left[\|\phi(x_i)\|^2 - 2 \langle \phi(x_i), \bar{x}_+^\phi \rangle + \|\bar{x}_+^\phi\|^2 \right] + \sum_{x_i \in c^-} \left[\|\phi(x_i)\|^2 - 2 \langle \phi(x_i), \bar{x}_-^\phi \rangle + \|\bar{x}_-^\phi\|^2 \right] \\
&= \left[\sum_{x_i \in c^+} k(x_i, x_i) - \frac{2}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) \right] \\
&\quad + \left[\sum_{x_i \in c^-} k(x_i, x_i) - \frac{2}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) + \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) \right] \\
&= \sum_{x_i \in c^+} k(x_i, x_i) + \sum_{x_i \in c^-} k(x_i, x_i) - \frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) - \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) \\
&= \sum_{x_i} k(x_i, x_i) - \frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) - \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j)
\end{aligned} \tag{4.2.19}$$

□

Identity 4.2.3. *The trace of the total sum of squares matrix SS_T^ϕ is given by:*

$$\text{trace}(SS_T^\phi) = \sum_{x_i} k(x_i, x_i) - \frac{1}{n} \sum_{x_i, x_j} k(x_i, x_j) .$$

Proof.

$$\begin{aligned}
\text{trace}(SS_T^\phi) &= \text{trace} \left[\sum_{i=1}^n (\phi(x_i) - \bar{x}^\phi) (\phi(x_i) - \bar{x}^\phi)^\top \right] \\
&= \sum_{i=1}^n (\phi(x_i) - \bar{x}^\phi)^\top (\phi(x_i) - \bar{x}^\phi) \\
&= \sum_{i=1}^n \left[\|\phi(x_i)\|^2 - 2 \langle \phi(x_i), \bar{x}^\phi \rangle + \|\bar{x}^\phi\|^2 \right] \\
&= \sum_{x_i}^n k(x_i, x_i) - \frac{2}{n} \sum_{x_i, x_j}^n k(x_i, x_j) + \frac{1}{n} \sum_{x_i, x_j}^n k(x_i, x_j) \\
&= \sum_{x_i}^n k(x_i, x_i) - \frac{1}{n} \sum_{x_i, x_j}^n k(x_i, x_j) \\
&= \text{trace}(SS_B^\phi) + \text{trace}(SS_B^\phi)
\end{aligned} \tag{4.2.20}$$

□

Now, we will show that the kernel alignment value $KA(K, K_{ideal})$ has a particular relationship with $\text{trace}(SS_B^\phi)$ and $\text{trace}(SS_T^\phi)$:

Corollary 4.2.4. $KA(K, K_{ideal}) \geq \frac{\text{trace}(SS_B^\phi)}{\sqrt{n^2 - n \text{trace}(SS_T^\phi)}}$

Proof. In order to prove this inequality, we first must prove the following two lemmas:

Lemma 4.2.5. $\langle K, K_{ideal} \rangle_F = 4 \frac{n+n_-}{n} \text{trace}(SS_B^\phi)$

Proof. From (4.2.18), we have

$$\begin{aligned}
\text{trace}(SS_B^\phi) &= \frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - \frac{1}{n} \sum_{x_i, x_j}^n k(x_i, x_j) \\
&= \frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - \frac{1}{n_+ + n_-} \sum_{x_i, x_j}^n k(x_i, x_j) \\
&= \frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) \\
&\quad - \frac{1}{n_+ + n_-} \left[\sum_{x_i, x_j \in c^+} k(x_i, x_j) + \sum_{x_i, x_j \in c^-} k(x_i, x_j) + 2 \sum_{x_i \in c^+, x_j \in c^-} k(x_i, x_j) \right] \\
&= \frac{n_-(n_+ + n_-)}{n_+ n_-(n_+ + n_-)} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{n_+(n_+ + n_-)}{n_+ n_-(n_+ + n_-)} \sum_{x_i, x_j \in c^-} k(x_i, x_j) \\
&\quad - \frac{n_+ n_-}{n_+ n_-(n_+ + n_-)} \left[\sum_{x_i, x_j \in c^+} k(x_i, x_j) + \sum_{x_i, x_j \in c^-} k(x_i, x_j) + 2 \sum_{x_i \in c^+, x_j \in c^-} k(x_i, x_j) \right] \\
&= \left[\frac{n_+ n_-}{n_+ n_-(n_+ + n_-)} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{n_-^2}{n_+ n_-(n_+ + n_-)} \sum_{x_i, x_j \in c^+} k(x_i, x_j) \right] \\
&\quad + \left[\frac{n_+ n_-}{n_+ n_-(n_+ + n_-)} \sum_{x_i, x_j \in c^-} k(x_i, x_j) + \frac{n_+^2}{n_+ n_-(n_+ + n_-)} \sum_{x_i, x_j \in c^-} k(x_i, x_j) \right] \\
&\quad - \frac{n_+ n_-}{n_+ n_-(n_+ + n_-)} \left[\sum_{x_i, x_j \in c^+} k(x_i, x_j) + \sum_{x_i, x_j \in c^-} k(x_i, x_j) + 2 \sum_{x_i \in c^+, x_j \in c^-} k(x_i, x_j) \right] \\
&= \frac{n_-^2}{n_+ n_-(n_+ + n_-)} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{n_+^2}{n_+ n_-(n_+ + n_-)} \sum_{x_i, x_j \in c^-} k(x_i, x_j) \\
&\quad - 2 \frac{n_+ n_-}{n_+ n_-(n_+ + n_-)} \sum_{x_i \in c^+, x_j \in c^-} k(x_i, x_j) \\
&= \frac{1}{(n_+ + n_-)} \left[\frac{n_-}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{n_+}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - 2 \sum_{x_i \in c^+, x_j \in c^-} k(x_i, x_j) \right]
\end{aligned} \tag{4.2.21}$$

Without loss of generality, let us assume that the first n_+ data samples belong to class c^+ and the remaining n_- data samples belong to c^- .

From (4.2.7), our centered response variable is:

$$y_i^* = \begin{cases} +1 - \frac{n_+ - n_-}{n} & \text{for } i = 1, \dots, n_+ \\ -1 - \frac{n_+ - n_-}{n} & \text{for } i = n_+ + 1, \dots, n. \end{cases} \quad (4.2.22)$$

the corresponding ideal kernel matrix is $K_{ideal} = y^* y^{*\top}$, and its corresponding entries are:

$$k_{ideal}(x_i, x_j) = \begin{cases} 4 \left(\frac{n_-}{n}\right)^2 & \text{if } x_i \in c^+ \text{ and } x_j \in c^+ \\ 4 \left(\frac{n_+}{n}\right)^2 & \text{if } x_i \in c^- \text{ and } x_j \in c^- \\ -4 \left(\frac{n_+ n_-}{n^2}\right) & \text{if } (x_i \in c^+ \text{ and } x_j \in c^-) \text{ or } (x_i \in c^- \text{ and } x_j \in c^+). \end{cases} \quad (4.2.23)$$

Thus, it can be shown that

$$\begin{aligned} \langle K_{ideal}, K_{ideal} \rangle_F &= \sum_{i=1}^n \sum_{j=1}^n k_{ideal}^2(x_i, x_j) \\ &= n_+^2 \left(4 \left(\frac{n_-}{n}\right)^2\right)^2 + n_-^2 \left(4 \left(\frac{n_+}{n}\right)^2\right)^2 + 2n_+ n_- \left(-4 \left(\frac{n_+ n_-}{n^2}\right)\right)^2 \\ &= \left(\frac{4n_+ n_-}{n}\right)^2 \end{aligned} \quad (4.2.24)$$

From (4.2.4), we have

$$\begin{aligned}
\langle K, K_{ideal} \rangle_F &= \sum_{i=1}^n \sum_{j=1}^n k(x_i, x_j) k_{ideal}(x_i, x_j) \\
&= 4 \left(\frac{n_-}{n} \right)^2 \sum_{x_i, x_j \in c^+} k(x_i, x_j) + 4 \left(\frac{n_+}{n} \right)^2 \sum_{x_i, x_j \in c^-} k(x_i, x_j) - 8 \left(\frac{n_+ n_-}{n^2} \right) \sum_{x_i \in c^+, x_j \in c^-} k(x_i, x_j) \\
&= 4 \frac{n_+ n_-}{n} \left(\frac{n_-}{n_+ n} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{n_+}{n_- n} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - \frac{2}{n} \sum_{x_i \in c^+, x_j \in c^-} k(x_i, x_j) \right) \\
&= 4 \frac{n_+ n_-}{n} \left(\frac{1}{n} \left(\frac{n_-}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{n_+}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - 2 \sum_{x_i \in c^+, x_j \in c^-} k(x_i, x_j) \right) \right) \\
&= 4 \frac{n_+ n_-}{n} \text{trace}(SS_B^\phi)
\end{aligned} \tag{4.2.25}$$

□

Lemma 4.2.6. *When using a Gaussian RBF kernel, $\langle K, K \rangle_F \leq n^2 - n \text{ trace}(SS_T^\phi)$*

Proof. For a Gaussian RBF kernel, the kernel matrix K will be normalized such that all the elements in the matrix satisfy $0 < k(x_i, x_j) \leq 1$. Under this kernel, the images of all points have norm 1 in the resulting feature space since $k(x_i, x_i) = 1$. Consequently, the feature space distance between two points is:

$$\begin{aligned}
\|\phi(x_i) - \phi(x_j)\|^2 &= \langle \phi(x_i) - \phi(x_j), \phi(x_i) - \phi(x_j) \rangle \\
&= \langle \phi(x_i), \phi(x_i) \rangle - 2 \langle \phi(x_i), \phi(x_j) \rangle + \langle \phi(x_j), \phi(x_j) \rangle.
\end{aligned} \tag{4.2.26}$$

Rearranging the equation, we have:

$$\begin{aligned}
\langle \phi(x_i), \phi(x_j) \rangle &= \frac{1}{2} \langle \phi(x_i), \phi(x_i) \rangle + \frac{1}{2} \langle \phi(x_j), \phi(x_j) \rangle - \frac{1}{2} \|\phi(x_i) - \phi(x_j)\|^2 \\
&= \frac{1}{2} k(x_i, x_i) + \frac{1}{2} k(x_j, x_j) - \frac{1}{2} \|\phi(x_i) - \phi(x_j)\|^2 \\
&= 1 - \frac{1}{2} \|\phi(x_i) - \phi(x_j)\|^2
\end{aligned} \tag{4.2.27}$$

Since $0 < k(x_i, x_j) \leq 1$, we can obtain the following inequality:

$$\langle K, K \rangle_F = \sum_{x_i, x_j}^n k^2(x_i, x_j) \leq \sum_{x_i, x_j}^n k(x_i, x_j) = \sum_{x_i, x_j}^n \langle \phi(x_i), \phi(x_j) \rangle \quad (4.2.28)$$

By substituting equation (4.2.27) into inequality (4.2.28), we get

$$\begin{aligned} \langle K, K \rangle_F &\leq \sum_{x_i, x_j}^n \langle \phi(x_i), \phi(x_j) \rangle \\ &= \sum_{x_i, x_j}^n \left[1 - \frac{1}{2} \|\phi(x_i) - \phi(x_j)\|^2 \right] \\ &= n^2 - \frac{1}{2} \sum_{x_i, x_j}^n \|\phi(x_i) - \phi(x_j)\|^2 \\ &= n^2 - \frac{1}{2} \sum_{x_i, x_j}^n \|\phi(x_i) + \bar{x}^\phi - \bar{x}^\phi - \phi(x_j)\|^2 \\ &= n^2 - \frac{1}{2} \sum_{x_i, x_j}^n \|(\phi(x_i) - \bar{x}^\phi) - (\phi(x_j) - \bar{x}^\phi)\|^2 \\ &= n^2 - \frac{1}{2} \sum_{x_i, x_j}^n \left[\|\phi(x_i) - \bar{x}^\phi\|^2 - 2 \|(\phi(x_i) - \bar{x}^\phi)\| \|(\phi(x_j) - \bar{x}^\phi)\| + \|(\phi(x_j) - \bar{x}^\phi)\|^2 \right] \\ &= n^2 - \frac{1}{2} \sum_{x_i, x_j}^n \|\phi(x_i) - \bar{x}^\phi\|^2 + \sum_{x_i, x_j}^n (\phi(x_i) - \bar{x}^\phi)^\top (\phi(x_j) - \bar{x}^\phi) - \frac{1}{2} \sum_{x_i, x_j}^n \|(\phi(x_j) - \bar{x}^\phi)\|^2 \\ &= n^2 - \sum_{x_i, x_j}^n \|\phi(x_i) - \bar{x}^\phi\|^2 \\ &= n^2 - n \sum_{x_i}^n \|\phi(x_i) - \bar{x}^\phi\|^2 \\ &= n^2 - n \sum_{x_i}^n (\phi(x_i) - \bar{x}^\phi)^\top (\phi(x_i) - \bar{x}^\phi) \\ &= n^2 - n \text{ trace}(SS_T^\phi) \end{aligned} \quad (4.2.29)$$

□

Using equation (4.2.3), we have

$$KA(K, K_{ideal}) = \frac{\langle K, K_{ideal} \rangle_F}{\sqrt{\langle K, K \rangle_F \langle K_{ideal}, K_{ideal} \rangle_F}} = \frac{\text{trace}(SS_B^\phi)}{\sqrt{\langle K, K \rangle_F}} \quad (4.2.30)$$

Substitute (4.2.24), (4.2.25) and (4.2.29) into (4.2.30) to get:

$$KA(K, K_{ideal}) \geq \frac{\text{trace}(SS_B^\phi)}{\sqrt{n^2 - n \text{trace}(SS_T^\phi)}}.$$

So the corollary is proven. \square

Theorem 4.2.7. *In an SVM classifier, the upper bound of the generalization error can be reduced by maximizing the kernel alignment value $KA(K, K_{ideal})$.*

Proof. The upper bound of an estimate of the leave-one-out (LOO) generalization error of an SVM classifier is [104]:

$$Error_{loo} \leq \frac{4R^2}{\gamma^2} \quad (4.2.31)$$

where R is the radius of the smallest hypersphere enclosing the training samples in the feature space FS and γ is the separation margin. Consider a hard margin SVM [89], let α^* be the optimal solution for the following dual problem:

$$\begin{aligned} \max \left(\frac{1}{2} W(\alpha) \right) &= \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i^* y_j^* k(x_i, x_j) \\ \text{subject to: } &\sum_{i=1}^n y_i^* \alpha_i = 0 \text{ and } \alpha_i \geq 0 \end{aligned} \quad (4.2.32)$$

Suppose we obtain a sub-optimal solution α^{sub} of (4.2.32) as:

$$\alpha^{sub} = \begin{cases} 1/n_+ & \text{if } x_i \in c^+ \\ 1/n_- & \text{if } x_i \in c^- \end{cases}$$

then

$$\begin{aligned}
\frac{1}{2}W(\alpha^{sub}) &= \sum_{i=1}^n \alpha_i^{sub} - \frac{1}{2} \sum_{i,j}^n \alpha_i^{sub} \alpha_j^{sub} y_i^* y_j^* k(x_i, x_j) \\
&= 2 - \frac{1}{2} \sum_{x_i, x_j}^n \alpha_i^{sub} \alpha_j^{sub} y_i^* y_j^* k(x_i, x_j) \\
&= 2 - \frac{1}{2} \left[\sum_{x_i, x_j \in c^+} \alpha_i^{sub} \alpha_j^{sub} y_i^* y_j^* k(x_i, x_j) + \sum_{x_i, x_j \in c^-} \alpha_i^{sub} \alpha_j^{sub} y_i^* y_j^* k(x_i, x_j) + \right. \\
&\quad \left. + 2 \sum_{x_i \in c^+, x_j \in c^-} \alpha_i^{sub} \alpha_j^{sub} y_i^* y_j^* k(x_i, x_j) \right] \\
&= 2 - \frac{1}{2} \left[\frac{1}{n_+^2} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{1}{n_-^2} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - 2 \frac{1}{n_+ n_-} \sum_{x_i \in c^+, x_j \in c^-} k(x_i, x_j) \right] \\
&= 2 - \frac{1}{2} \left[\frac{n_-^2}{n_+^2 n_-^2} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{n_+^2}{n_+^2 n_-^2} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - 2 \frac{n_+ n_-}{n_+^2 n_-^2} \sum_{x_i \in c^+, x_j \in c^-} k(x_i, x_j) \right].
\end{aligned} \tag{4.2.33}$$

Since $\sum_{x_i, x_j}^n k(x_i, x_j) = \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \sum_{x_i, x_j \in c^-} k(x_i, x_j) + 2 \sum_{x_i \in c^+, x_j \in c^-} k(x_i, x_j)$, we

have

$$\begin{aligned}
& \frac{1}{2}W(\alpha^{sub}) \\
&= 2 - \frac{1}{2} \left[\frac{n_-^2}{n_+^2 n_-^2} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{n_+^2}{n_+^2 n_-^2} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - 2 \frac{n_+ n_-}{n_+^2 n_-^2} \sum_{x_i \in c^+, x_j \in c^-} k(x_i, x_j) \right] \\
&= 2 - \frac{1}{2} \left[\frac{n_-^2}{n_+^2 n_-^2} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{n_+^2}{n_+^2 n_-^2} \sum_{x_i, x_j \in c^-} k(x_i, x_j) \right. \\
&\quad \left. - \frac{n_+ n_-}{n_+^2 n_-^2} \left[\sum_{x_i, x_j}^n k(x_i, x_j) - \sum_{x_i, x_j \in c^+} k(x_i, x_j) - \sum_{x_i, x_j \in c^-} k(x_i, x_j) \right] \right] \\
&= 2 - \frac{1}{2} \frac{1}{n_+ n_-} \left[\frac{n_-^2}{n_+ n_-} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{n_+^2}{n_+ n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - \sum_{x_i, x_j}^n k(x_i, x_j) \right. \\
&\quad \left. + \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \sum_{x_i, x_j \in c^-} k(x_i, x_j) \right] \\
&= 2 - \frac{1}{2} \frac{1}{n_+ n_-} \left[\frac{n_-^2}{n_+ n_-} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \sum_{x_i, x_j \in c^+} k(x_i, x_j) \right. \\
&\quad \left. + \frac{n_+^2}{n_+ n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) + \sum_{x_i, x_j \in c^-} k(x_i, x_j) - \sum_{x_i, x_j}^n k(x_i, x_j) \right] \\
&= 2 - \frac{1}{2} \frac{1}{n_+ n_-} \left[\frac{n_-(n_+ + n_-)}{n_+ n_-} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{n_-(n_+ + n_-)}{n_+ n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - \sum_{x_i, x_j}^n k(x_i, x_j) \right] \\
&= 2 - \frac{1}{2} \frac{(n_+ + n_-)}{n_+ n_-} \left[\frac{1}{n_+} \sum_{x_i, x_j \in c^+} k(x_i, x_j) + \frac{1}{n_-} \sum_{x_i, x_j \in c^-} k(x_i, x_j) - \frac{1}{(n_+ + n_-)} \sum_{x_i, x_j}^n k(x_i, x_j) \right] \\
&= 2 - \frac{(n_+ + n_-)}{2n_+ n_-} \text{trace}(SS_B^\phi).
\end{aligned}$$

(4.2.34)

Since $\gamma = \frac{1}{\|W(\alpha^*)\|}$ and $W(\alpha^*) \geq W(\alpha^{sub})$, we have

$$\gamma^2 \leq \frac{1}{4 - \frac{(n_+ + n_-)}{n_+ n_-} \text{trace}(SS_B^\phi)} \quad (4.2.35)$$

Now, consider the problem of finding the smallest hypersphere that encloses the training data [68]. Suppose α^* is the optimal solution for the following dual problem:

$$\begin{aligned} \max(W(\alpha)) &= \sum_{i=1}^n \alpha_i k(x_i, x_i) - \sum_{i,j}^n \alpha_i \alpha_j k(x_i, x_j) \\ \text{subject to: } &\sum_{i=1}^n \alpha_i = 1 \text{ and } \alpha_i \geq 0 \end{aligned} \quad (4.2.36)$$

Consider the following sub-optimal solution of (4.2.36):

$$\alpha^{sub} = \frac{1}{n_+ + n_-}$$

then

$$\begin{aligned} W(\alpha^{sub}) &= \sum_{i=1}^n \alpha_i^{sub} k(x_i, x_i) - \sum_{i,j}^n \alpha_i^{sub} \alpha_j^{sub} k(x_i, x_j) \\ &= \frac{1}{n_+ + n_-} \sum_{x_i}^n k(x_i, x_i) - \frac{1}{(n_+ + n_-)^2} \sum_{x_i, x_j}^n k(x_i, x_j) \\ &= \frac{1}{n} \left[\sum_{x_i}^n k(x_i, x_i) - \frac{1}{n} \sum_{x_i, x_j}^n k(x_i, x_j) \right] \\ &= \frac{1}{n} \text{trace}(SS_T^\phi) \end{aligned} \quad (4.2.37)$$

The radius, R , of the smallest hypersphere enclosing the training samples in the feature space FS is:

$$R^2 = W(\alpha^*) \quad (4.2.38)$$

Since $W(\alpha^*) \geq W(\alpha^{sub})$, we have the following inequality:

$$R^2 \geq \frac{1}{n} \text{trace}(SS_T^\phi). \quad (4.2.39)$$

From (4.2.31), we notice that $\frac{4R^2}{\gamma^2}$ is the upper bound of $Error_{loo}$ for R^2 and γ^2 satisfying the inequalities as stated in (4.2.35) and (4.2.39). That is, we have

$$Error_{loo} \leq \frac{4R^2}{\gamma^2} \Rightarrow Error_{loo} \leq \min \left(\frac{4R^2}{\gamma^2} \right). \quad (4.2.40)$$

To minimize $\frac{4R^2}{\gamma^2}$, we minimize R^2 and maximize γ^2 . Consider (4.2.39), the minimum of R^2 is:

$$R^2 = \frac{1}{n} \text{trace}(SS_T^\phi) \quad (4.2.41)$$

Similarly, the maximum of γ^2 is seen in (4.2.35) as:

$$\gamma^2 = \frac{1}{4 - \frac{(n_+ + n_-)}{n_+ n_-} \text{trace}(SS_B^\phi)} \quad (4.2.42)$$

Substitute (4.2.41) and (4.2.42) into equation (4.2.40), we obtain:

$$\begin{aligned} Error_{loo} &\leq \min \left(\frac{4R^2}{\gamma^2} \right) \\ &= \frac{4 \left[\frac{1}{n} \text{trace}(SS_T^\phi) \right]}{\left[\frac{1}{4 - \frac{n}{n_+ n_-} \text{trace}(SS_B^\phi)} \right]} \\ &= \frac{4 \text{trace}(SS_T^\phi) \left[4 - \frac{n}{n_+ n_-} \text{trace}(SS_B^\phi) \right]}{n} \end{aligned} \quad (4.2.43)$$

Substitute (4.2.29) and (4.2.30) into (4.2.43) we get:

$$\begin{aligned} Error_{loo} &\leq \frac{4 \text{trace}(SS_T^\phi) \left[4 - \frac{n\sqrt{\langle K, K \rangle_F}}{n_+ n_-} [KA(K, K_{ideal})] \right]}{n} \\ &\leq \frac{4 \left[n - \frac{\langle K, K \rangle_F}{n} \right] \left[4 - \frac{n\sqrt{\langle K, K \rangle_F}}{n_+ n_-} [KA(K, K_{ideal})] \right]}{n} \end{aligned} \quad (4.2.44)$$

Now, R^2 and γ^2 are always non-negative for a kernel which maps the input data onto a unit hypersphere. Thus, $n - \frac{\langle K, K \rangle_F}{n}$ and $4 - \frac{n\sqrt{\langle K, K \rangle_F}}{n_+ n_-} [KA(K, K_{ideal})]$ are

always non-negative. So, for a fixed $\langle K, K \rangle_F$, increasing the $KA(K, K_{ideal})$ results in lowering the upper bound of the $Error_{loo}$. \square

For each iteration in the Kernel Alignment Component Selection Algorithm, we removed the component that gives the maximum $KA(K_{removed(p)}, K_{ideal}) - KA(K_{current}, K_{ideal})$. In other words, among all the reduced component sets, we kept the subset that maximizes the kernel alignment value. This reduced set will replace the current set for the next iteration. According to Theorem 4.2.7, the operation will produce a lower error bound.

4.2.5 Empirical Analysis

The purpose of this subsection is to study the effect of kernel alignment on the generalization error.

The experiments use 4 data sets from the UCI repository. Table 4.2.1 gives a brief summary on the UCI data set.

Data Set	Train size (n)	Components (m)	Classes
Breast Cancer	699	9	2
Sonar	208	60	2
German Credit	1000	24	2
Votes	435	16	2

Table 4.2.1: UCI data sets

For each data set, the components were normalized. The Kernel Alignment Component Selection Algorithm was run using the full component set as the initial

input set. In the experiment, a Gaussian Kernel with default parameter $\sigma = 2$ was used. The kernel alignment values and the leave-one-out errors were calculated and stored for each of the iterations of the algorithm. In Figure 4.2.1, we show 8 plots (two for each of the four data sets) to illustrate the results of component selection for each data set. The upper plot shows the kernel alignment value for each reduced component set (labeled with the number of component left in the data set). The lower plot shows the leave-one-out validation error vs. the number of components left in the data set.

In Figure 4.2.1, we observe a general trend across all 4 data sets: When the kernel alignment value decreases, the leave-one-out error increases. This provides further evidence that Theorem 4.2.7 holds true for a practical data set.

4.2.6 Random Subspace Kernel Alignment Component Selection (RSKACS)

For the Kernel Alignment Component Selection Algorithm, we began with the full set of components, and then each component was individually eliminated from the current subset. The major concerns about this algorithm are:

1. It is computationally intensive when the number of components is large.
2. The resulting component subset may represent a local maximum but not necessarily a global maximum.

To address these points, we consider a randomized approach called Random Subspace Kernel Alignment Component Selection (RSKACS) based on the random

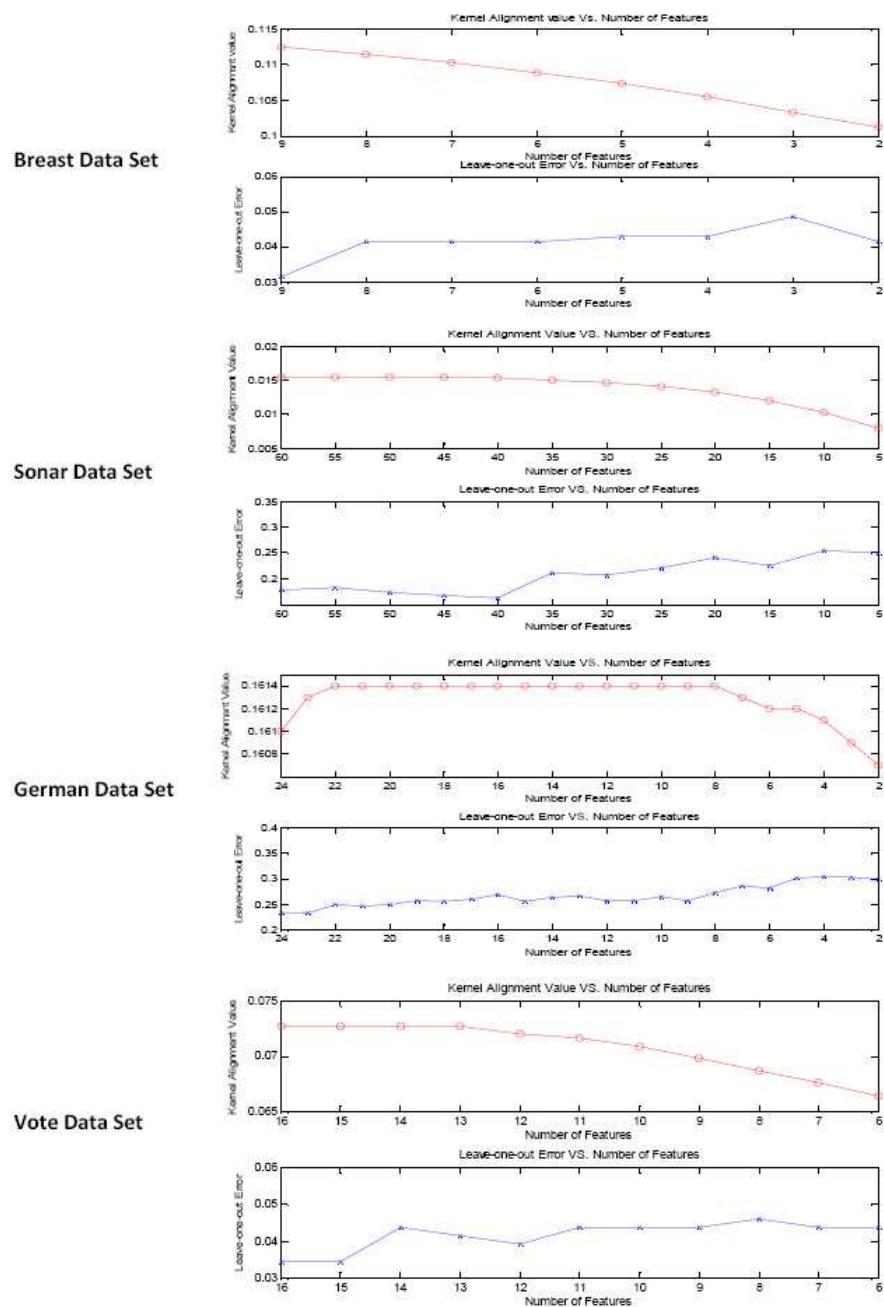


Figure 4.2.1: KACS Results for 4 different UCI data sets

subspace method introduced by Ho [44, 64].

Suppose we have a large number of components. Instead of starting the KACS algorithm with the full set of components, the RSKACS approach starts the algorithm with a much smaller subset that is randomly selected from the full set. With a smaller starting set, KACS will perform more efficiently. We randomly select t different initial subsets. To ensure that we work with most of the components in the dataset, we assume that t is reasonably large.

For each initial subset, denoted by X_i^{sub} , where $i = 1, \dots, t$, we apply KACS on X_i^{sub} . The KACS algorithm will generate a reduced subset for each initial subset until the stopping condition takes effect. The reduced subset containing the highest kernel alignment value will be the final subset of the RSKACS. The final selection criteria can be based on a combination of kernel alignment value, leave-one-out error and the size of the selected subset. This can be further investigated by the user of the RSKACS algorithm. Here, for simplicity, we only use the kernel alignment value as our final selection criteria.

The following pseudo-code summarizes our algorithm:

- 1 **Random Subspace Kernel Alignment Component Selection Algorithm**
- 2 Input: training set X , label set y , stopping condition sc , number of selections t , size of initial subspace s .
- 3 for $i = 1: t$
- 4 Generate a random component subset $X_i^{sub} \subset X$ with size s .

- 5 Apply the KACS Algorithm on X_i^{sub} with the given y , and sc .
- 6 Compute the kernel alignment value $KA(K, K_{ideal})$ for X_i^{sub} ;
- 7 *endfor*
- 8 return X_i^{sub} that give the maximum kernel alignment value;

4.3 Experimental Results - Experiment 1: Binary Response QSAR Data

4.3.1 Data

In our QSAR study, three different data sets were used to test the performance of the KACS algorithm. The three QSAR data sets are from Xue *et al.* [113]. The first data set contains the report of human intestinal absorption (HIA) of chemical agents. HIA is an important indicator of drug absorption. In a total of 196 molecules, 131 molecules have been classified as absorbable and the remaining 65 are classified as non-absorbable. The second data set reports on P-glycoprotein (Pgp) substrates. There are 116 substrates and 86 non-substrates of Pgp in the data set. The third data set reports on compounds that induce torsades depointes (TdP). TdP is an uncommon adverse drug reaction responsible for the withdrawal of some marketed drugs. In a total of 361 molecules, 85 of them are classified as TdP positive and the remaining 276 are classified as TdP negative.

The descriptors used in this experiments were selected by Xue *et al.* [113]. They use 159 descriptors that contain simple molecular properties, molecular connectiv-

ity and shape, 3D geometrical properties, electro-topological state and quantum chemical properties.

4.3.2 Implementation Details

To compare our proposed KACS and RSKACS with the SVMRCE, we used MATLAB and SVMlight [48] to implement the KACS and the Gaussian version of SVMRCE.

4.3.3 Results and Discussion

For the three data sets, Xue *et al.* [113] used 5 fold cross-validation to compare the prediction accuracy of SVMRCE classification with a SVM that did not use any component selection method. The data was randomly divided into five subsets for the purpose of cross-validation. To provide a fair comparison, we used leave-one-out cross-validation to ensure that all component selection algorithms were trained and tested using the same data. The parameter $\sigma = 2$ was also fixed for all component selection algorithms. For each of the iterations of the algorithm, only one component was removed. For each of the data sets, component values were normalized. Leave-one-out errors were calculated at each iteration step for the purpose of comparison. For the RSKACS, we set $t = 500$ and $s = j + 1$, where j is the current number of iteration in our experiments.

Figure 4.3.2 summarizes the results of the KACS Algorithm on the HIA data set. We observe that when the kernel alignment value decreases, the leave-one-out error increases. The maximum kernel alignment value is 0.2041 with 85 components

left and the leave-one-out error at the corresponding kernel alignment value is at a minimum which is 0.1633.

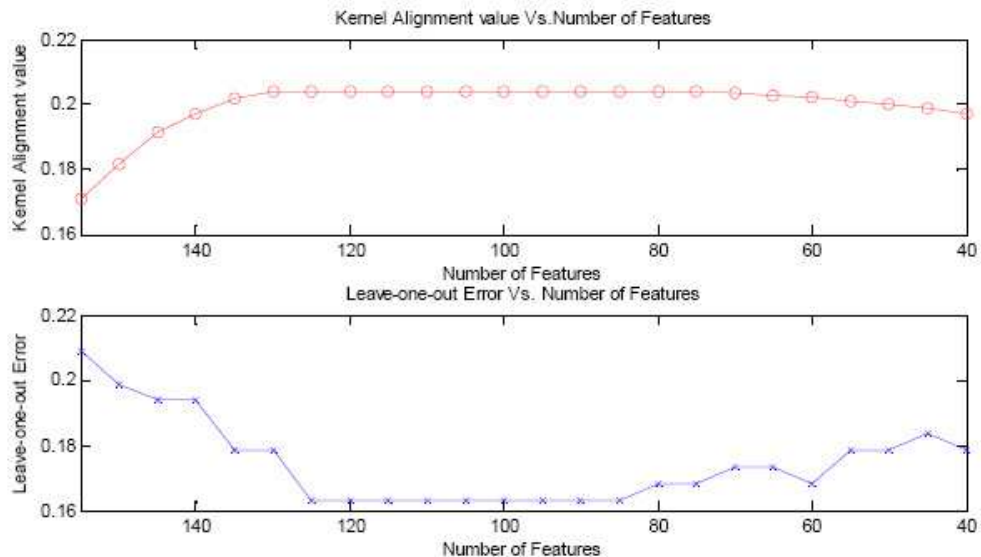


Figure 4.3.2: The results of the KACS Algorithm on the HIA data set

Figure 4.3.3 compares the leave-one out error generated by the KACS algorithm, RSKACS algorithm and the SVMRCE algorithm on the HIA data set. It can be seen that a lower leave-one-out error is achieved by our KACS algorithm for all the iterations, indicating that the algorithm performs significantly better than the SVMRCE algorithm on the HIA data set. The RSKACS algorithm also performs very well for all the iterations.

Figure 4.3.4 summarizes the results of the KACS Algorithm on the Pgp data set. The maximum kernel alignment value is 0.0638 with 55 components left. The corresponding leave-one-out error is at a minimum which is 0.2090.

Figure 4.3.5 compares the leave-one-out error generated by the KACS algorithm,

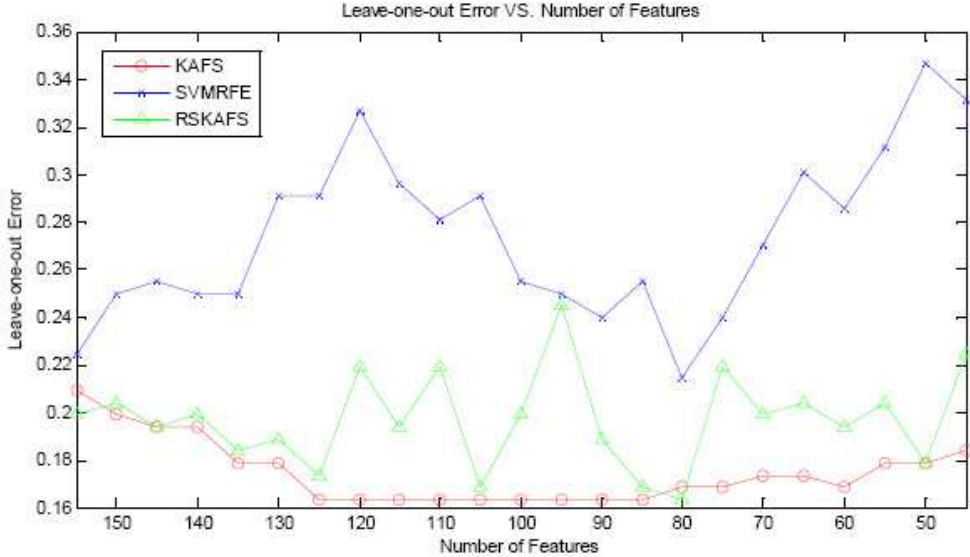


Figure 4.3.3: The leave-one out error generated by the KACS, the RSKACS and the SVMRCE algorithm on the HIA data set

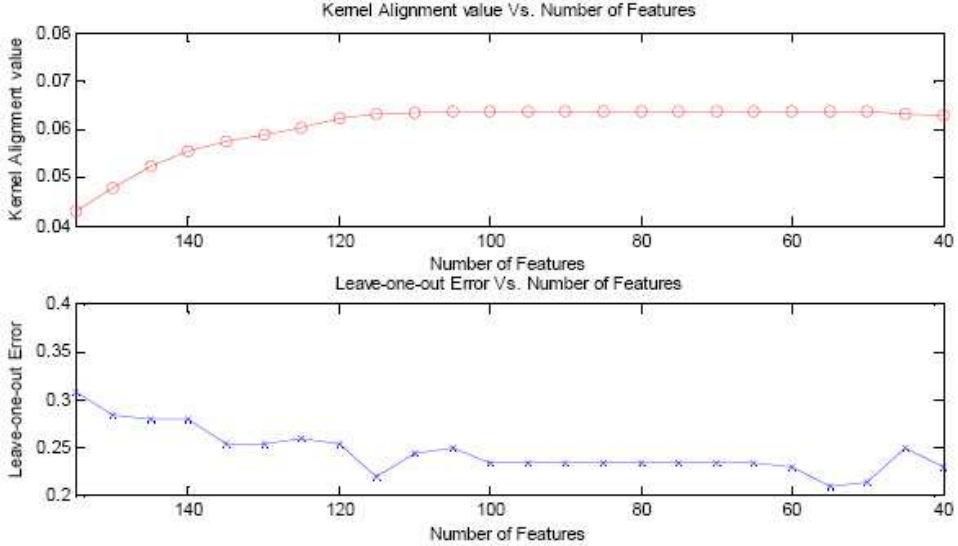


Figure 4.3.4: The results of the KACS Algorithm on the Pgp data set

the RSKACS algorithm and the SVMRCE algorithm on the Pgp data set. It can be seen that in the first 50 iterations, both algorithms perform almost the same by selecting roughly the same set of components for elimination. However, after 50 iterations, the KACS algorithm performs significantly better than the SVMRCE algorithm in the Pgp data set by finding a better set of components. Due to the random effect on the RSKACS algorithm, the leave-one-out error generated by the RSKACS algorithm fluctuated from 0.21 to 0.33 across all iterations.

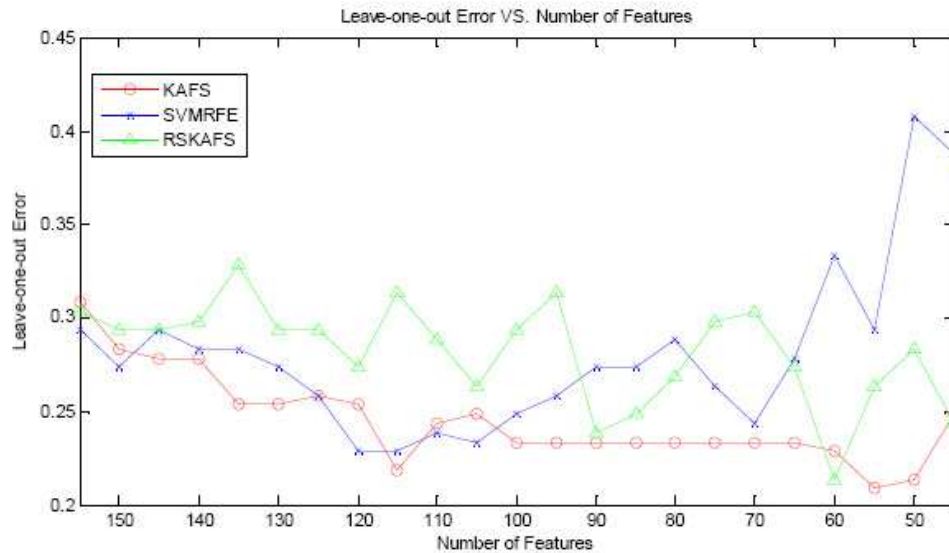


Figure 4.3.5: The leave-one out error generated by the KACS, the RSKACS and the SVMRCE algorithm on the Pgp data set

Figure 4.3.6 summarizes the results of the KACS Algorithm on the TdP data set. We observed that when the kernel alignment value decreases, the leave-one-out error typically increases. The maximum kernel alignment value is 0.3090 with 40 components left and the leave-one-out error at the corresponding kernel alignment

value is at a minimum which is 0.1662.

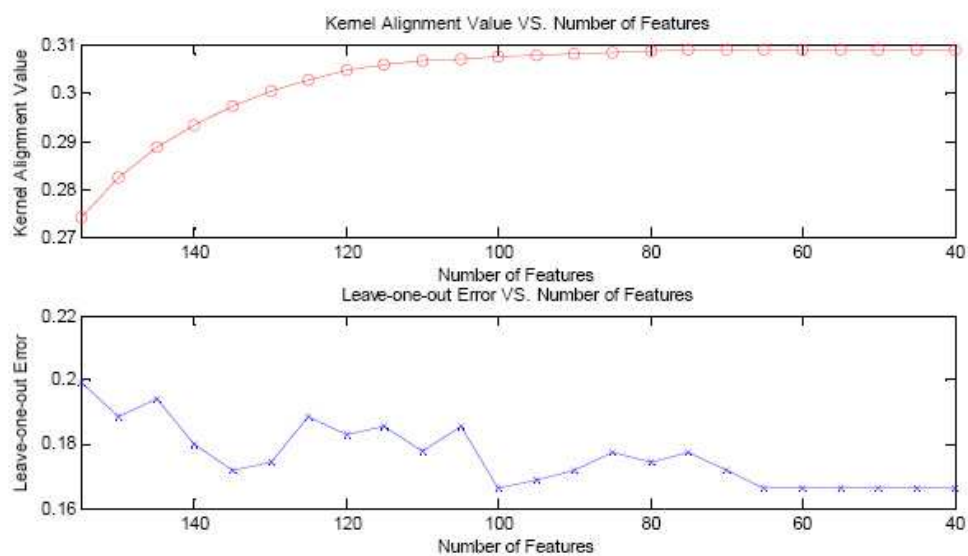


Figure 4.3.6: The results of the KACS Algorithm on the TdP data set

Figure 4.3.7 compares the leave-one out error generated by the KACS algorithm, the RSKACS algorithm and the SVMRCE algorithm on the Tdp data set. It can be seen that by using our KACS algorithm and RSKACS algorithm, we obtained a lower leave-one-out error for most of the iterations after the first 15 iterations, indicating that our KACS algorithm and RSKACS algorithm perform significantly better than the SVMRCE algorithm on the TdP data set.

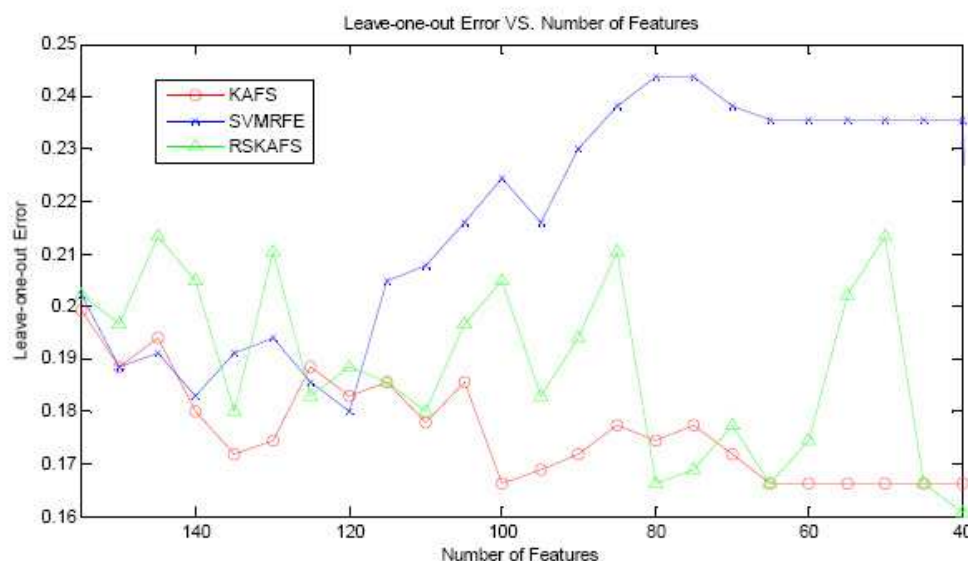


Figure 4.3.7: The leave-one out error generated by the KACS, the RSKACS and the SVMRCE algorithm on the TdP data set

4.4 Experimental Results - Experiment 2: Continuous Response QSAR Data

4.4.1 Data Set

In this QSAR study, four different data sets were used to test the ability of the KACS to select important components. All these data sets contain continuous response QSAR inhibitor data.

The four QSAR data sets were selected from Sutherland *et al.* [96]. The first data set contains 114 angiotensin converting enzyme (ACE) inhibitors collected by Depriest and colleagues [20] with pIC50 values ranging from 2.1 to 9.9. The second data set contains 322 cyclooxygenase-2 (COX2) inhibitors collected by Seibert and

colleagues [46] and subsequently utilized in a QSAR study by Chavatte *et al.* [12] with inhibitors having pIC50 values ranging from 5.5 to 8.9. The third data set contains 397 dihydrofolate reductase inhibitors (DHFR) collected by Queener and colleagues [9] and set up as data sets by Sutherland and Weaver [97] with inhibitors having pIC50 values for rat liver enzyme ranging from 3.3 to 9.8. The final data set were prepared by Klebe and his colleagues [60] including 88 thrombin inhibitors (THR) with pKi values ranging from 4.4 to 8.5.

In all our experiments the data were separated into the same training and testing sets used in Sutherland *et al.* [96]. Leave-one-out cross validation was used to obtain the parameters for the model.

4.4.2 Results and Discussion

The descriptor used in this experiment is our VSMMD descriptor. As reported in an earlier chapter, our modeling mechanisms have produced very effective algorithms to predict drug-binding affinities. For regression analysis, we made use of a linear ε -insensitive Support Vector Regression (SVR) algorithm as mentioned in the previous chapter. In our experiments, ε was set to 0.1, fragment type ID of the VSMMD was set to 1 and the Gaussian kernel was used for the SVR. Two parameters, σ and C , had to be evaluated through cross-validation. For all four data sets, we choose the best σ and C by cross-validation using only the training set.

The Kernel Alignment Component selection algorithm was run on the full component set using a Gaussian Kernel with the selected σ . For regression, we modified

the K_{ideal} by centering the response as:

$$K_{ideal} = y^* y^{*\top} \text{ where } y^* = y - \bar{y}.$$

In this equation, \bar{y} represents the mean evaluated over the training set of target values. For the reduced model generated by the KACS algorithm, we performed leave-one-out cross validation on the training set only, to obtain the optimal values for the two parameters C and σ .

For the RSKACS, we set $t = 500$ and $s = \frac{1}{2}m$, where m is the total number of components in our experiments. For the reduced model generated by the RSKACS algorithm, we performed leave-one-out cross validation on the training set only, to obtain the optimal values for the two parameters C and σ .

The prediction accuracy was assessed using the designated test sets and residual r^2 statistics [42] were obtained. A residual r^2 value close to one indicates a good prediction.

The residuals r^2 as well as the cross-validation residual q^2 , for all the four data sets are given in Table 4.4.2.

	Full Model ($m=147$)			KAFS Reduced Model				RSKAFS Reduced Model			
	r^2	q^2	Std(q^2)	r^2	q^2	Std(q^2)	m	r^2	q^2	Std(q^2)	m
ACE	0.45	0.63	0.087	0.5	0.68	0.041	52	0.47	0.63	0.055	78
COX2	0.35	0.37	0.134	0.42	0.41	0.093	133	0.42	0.35	0.169	98
DHFR	0.53	0.57	0.068	0.56	0.58	0.097	129	0.57	0.61	0.081	65
THR	0.16	0.2	0.201	0.24	0.23	0.134	24	0.17	0.21	0.146	115

Table 4.4.2: r^2 statistics using SVR on the testing set

From Table 4.4.2, the residual r^2 statistics based on the KACS and RSKACS methods increased across all four data sets.

In these two experiments, we have demonstrated that both the KACS algorithm and RSKACS are capable of automatic selection of important components in both binary QSAR data and real value QSAR data. The reductions of some overlapping and redundant molecular components in a descriptor enhance the performance of kernel based machine learning algorithms.

4.5 Conclusion

While the use of component selection techniques has appeared several times in research studies dealing with quantitative structure-activity relationships, we have further developed a new filter based component selection algorithm that is suitable for kernel based prediction algorithms. Our contributions include:

1. We have utilized a kernel alignment value as a relevance measure that is independent of the performance of the classifier.
2. We have developed a filter based kernel alignment component selection algorithm.
3. We have shown theoretically and empirically that our algorithm kept the component subset that maximized the kernel alignment value, which can produce a lower generalization error bound in an SVM classifier.

Empirical results show that our algorithm works well for finding the most important components in different QSAR data sets. The prediction accuracies are substantially increased and compare favorably with those from the earlier studies.

Chapter 5

Multiple Binding Modes

The ligand may show evidence of multiple binding modes within the same binding site. The same ligand can bind in distinct orientations or conformations in the binding site. The information of binding affinities for each mode are usually combined and packed as a whole in the training set. To disentangle this information is a challenge and thus complicates the machine learning strategy.

In this chapter, we extend the VSMMD to the prediction of multiple binding modes through the use of a standard k -means algorithm in the feature space.

5.1 Introduction

Most of the QSAR studies mentioned earlier rely on an assumption stated as follows: “similar analogs bind to the same binding site in a similar binding mode.” [54, 55] Usually, multiple binding modes of a binding site are completely ignored or treated as outliers in the modeling process. However, it is well known that reliable predic-

tion of correct binding modes of active compounds identified from high-throughput or virtual screening will facilitate the drug discovery process and provide results that are more accurate.

In this chapter, we describe the use of our reported novel descriptors, the vector space model molecular descriptor (VSMMD), based on a vector space model that is suitable for kernel studies in QSAR modeling, together with a kernel feature space algorithm to identify the binding modes in kernel feature space. Two different data sets:

1. Anilinopyrazoles as CDK2 inhibitors;
2. 6,9-diarylpurin-8-ones as inhibitors in the ATP binding site of p38 MAP kinase;

were chosen for the experiment. The inhibitors in both data sets can adopt multiple binding modes depending on the substituents. Previous studies [85, 41] have shown that it was difficult for docking programs to identify the alternative binding mode. Our experiments provide convincing empirical evidence that our VSMMD can provide sufficient information to identify different binding modes of a molecule with high accuracy.

5.2 The kernel k -means algorithm

In order to visualize the different binding modes, we applied the k -means algorithm to cluster image points in the RKHS. Clustering is one of the most important and

widely used methods of unsupervised learning. The algorithm has the ability to partition data into dissimilar groups of similar items.

Suppose we are working with n molecules (d_1, \dots, d_n) , and we wish to find an assignment of each point to one of a finite number g of classes. In other words, we seek a map:

$$f : (d_1, d_2, \dots, d_n) \rightarrow (1, 2, \dots, g). \quad (5.2.1)$$

This cluster mapping function f should be chosen to optimize:

$$f = \arg \min_{f:(d_1, d_2, \dots, d_n) \rightarrow (1, 2, \dots, g)} \sum_{i, j: f(d_i)=f(d_j)} \|\phi(d_i) - \phi(d_j)\|^2 \quad (5.2.2)$$

where ϕ maps into inner product space feature space FS using:

$$k(d_i, d_j) = \langle \phi(d_i), \phi(d_j) \rangle. \quad (5.2.3)$$

According to Shawe-Taylor and Cristianini [89], the solution of the clustering optimization function equation (5.2.2) can be found in the form:

$$f(d_i) = \arg \min_{1 \leq z \leq g} \|\phi(d_i) - C_z\| \quad (5.2.4)$$

where C_z is the centroid of the points assigned to cluster z . A detailed proof can be found in [89]. We can also derive the generalization form of equation (5.2.4) as:

$$f(.) = \arg \min_{1 \leq z \leq g} \|\phi(.) - C_z\| \quad (5.2.5)$$

The k -means algorithm keeps a set of cluster centroids C_1, C_2, \dots, C_g that are initialized randomly and it then seeks to minimize the following expression:

$$\sum_{i=1}^n \|\phi(d_i) - C_{f(d_i)}\|^2 \quad (5.2.6)$$

By adapting both f as well as the centers, the algorithm will converge to a solution in which C_z is the center of mass of points assigned to cluster z and will satisfy the criterion of equation (5.2.4).

The algorithm alternates between updating f to adjust the assignment of points to clusters and updating the C_z giving the positions of the centers in a two stage iterative procedure. The first stage simply assigns points to the cluster whose cluster center is closest. The second stage repositions the center of each cluster to be the center of mass of the points assigned to that cluster. Note that each stage can only decrease the value of expression (5.2.6) [89]. Since the number of possible clusters is finite, it follows that after a finite number of iterations the algorithm will converge to a stable clustering assignment.

Shawe-Taylor and Cristianini [89] provide a dual form solution to this problem. Let A be an $n \times g$ matrix indicating the containment of a data point in a cluster:

$$A_{iz} = \begin{cases} 1 & \text{if } d_i \text{ is in cluster } z; \\ 0 & \text{otherwise.} \end{cases} \quad (5.2.7)$$

Note that each row of A contains exactly one 1, while the column sums give the number of points assigned to the different clusters. Using A , we can compute the coordinates of the centroids C_z using FAD , where F is the $m \times n$ matrix of training molecules as in (3.3.2), and D is a diagonal $g \times g$ matrix with diagonal entries equal to the inverse of the column sums of A . Now we can compute the distance between a new point $\phi(d)$ and the centroid by:

$$\begin{aligned} \|\phi(d) - C_z\|^2 &= \|\phi(d)\|^2 - 2\langle \phi(d), C_z \rangle + \|C_z\|^2 \\ &= k(d, d) - 2(k^T AD)_z + (DA^T F^T F AD)_{zz} \end{aligned} \quad (5.2.8)$$

where k is the vector of inner products between $\phi(d)$ and the training samples. Hence, the cluster to which $\phi(d)$ should be assigned is given by:

$$\arg \min_{1 \leq z \leq g} \|\phi(d) - C_z\|^2 = \arg \min_{1 \leq z \leq g} ((DA^T KAD)_{zz} - 2(k^T AD)_z) \quad (5.2.9)$$

where K is the kernel matrix of the training set. This provides the rule for classifying new data. The update rule consists in reassigning the entries in the matrix A according to the same rule in order to redefine the cluster.

5.3 Experimental Results

5.3.1 Data

Two different data sets were used to test the ability of the VSMMD to identify multiple binding modes. The first data set contains 63 CDK2 inhibitors, anilino-pyrazole, collected by Sato *et al.* [85]. The compounds are divided into two groups (Type A and Type B) based on the substituents at the 5-position of the pyrazole ring (See Figure 5.3.1). Sato *et al.* [85] found two different binding modes (See Figure 5.3.2) from two representative compounds, one from each group, among the 65 compounds in their studies. The type-A representative compound is in binding mode 1, and the type B representative compound is in binding mode 2.

The second data set contains ten p38 MAP Kinase Inhibitors prepared by Hauser *et al.* [41]. Two different binding modes have been reported (See Figure 5.3.3). The free energy of binding (FEB) is reported for the two different binding modes in Table 5.3.1.

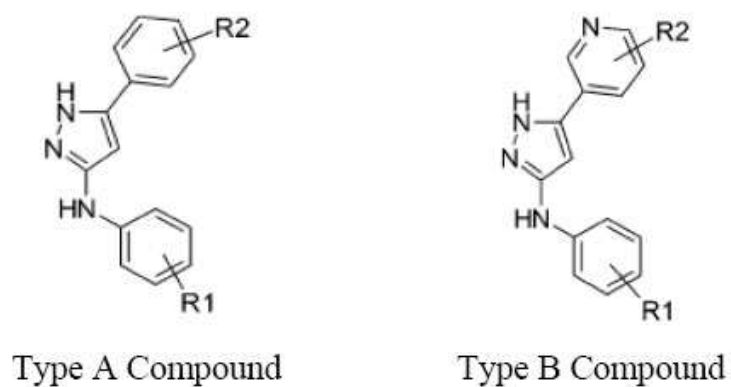


Figure 5.3.1: Type A and Type B compound of anilinopyrazole, a CDK2 inhibitor [85]

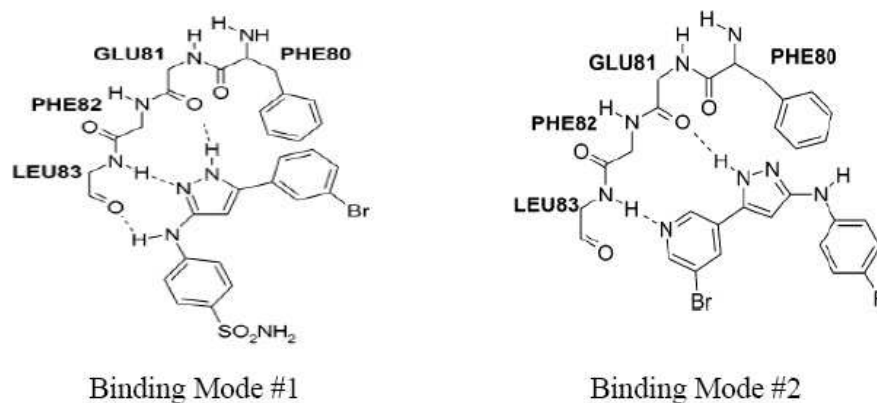


Figure 5.3.2: Two different binding modes reported of anilinopyrazole, a CDK2 inhibitor [85]

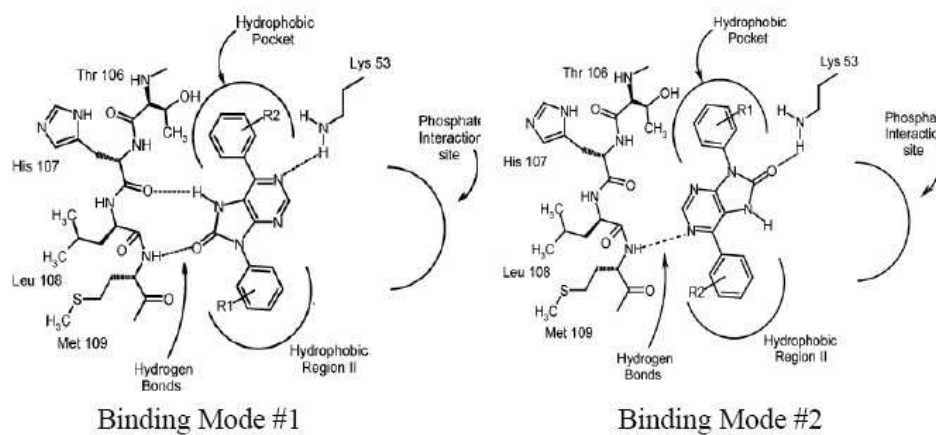


Figure 5.3.3: Two different binding modes reported for p38Map inhibitor [41]

Molecule ID	IC ₅₀ (μ M)	Best binding mode	FEB of binding mode 1	FEB of binding mode 2
21	0.9	1/2	-8.7	-8.7
22	7.8	1	-9.4	-9.2
23	2.7	1	-8.8	-8.7
24	17.5	1/2	-8.2	-8.2
25	18.1	1	-9.4	-8
26	0.5	2	-8.7	-8.8
27	1.6	2	-8.7	-8.9
28	5.2	2	-8.8	-9.2
29	7.8	1	-8.5	-6.9
30	100	2	-9.2	-9.4

Table 5.3.1: Free Energy of Binding (FEB) of ten p38 MAP Kinase Inhibitors reported by Hauser *et al.* [41]

5.3.2 Implementation Details

For the kernel k -means algorithm, we used MATLAB to perform the clustering experiments.

5.3.3 Test Results

To test our descriptor’s ability to facilitate the prediction of multiple binding modes, we generated the VSMMD kernels of different lengths using equation (3.3.3), and then we used the data fusion equation (3.4.5) to average the kernels to form a final kernel matrix K . Finally, we applied the kernel k -means algorithm to the final kernel matrix K .

Since the optimization of k -means algorithm is not convex, there is the possibility that the algorithm will converge to a local minima. In order to reduce the effect of local minima, the reported results are based on 100 runs of the k -means algorithm. Since we believed that there are two binding modes associated with each data set, for our experiment, we used $k = 2$. The result of the k -means algorithm will classify the molecules into either one of the two clusters, each cluster representing a binding mode. Table 5.3.2 summarizes the clustering result for the anilinopyrazole data set. Table 5.3.3 summarizes the clustering result for the p38 MAP Kinase Inhibitors data set.

Compound	IC50	Best binding mode	Clustering binding mode	Compound	IC50	Best binding mode	Clustering binding mode
1	5.2	1	1	33	7.63	1	1
2	5.2	1	1	34	7.76	1	2
3	5.23	1	1	35	7.9	1	2
4	5.32	1	1	36	7.97	1	1
5	5.36	1	1	37	7.97	1	1
6	5.36	1	1	38	8	1	2
7	5.51	1	1	39	8.04	1	1
8	5.53	1	2	40	8.11	1	1
9	5.56	1	2	41	8.76	1	1
10	5.57	1	1	42	9.48	1	1
11	5.65	1	2	43	5.35	2	2
12	5.65	1	2	44	5.36	2	2
13	5.67	1	2	45	5.38	2	2
14	5.69	1	1	46	5.42	2	2
15	5.73	1	1	47	5.51	2	2
16	5.81	1	1	48	5.54	2	2
17	5.81	1	1	49	5.62	2	2
18	5.89	1	2	50	5.67	2	2
19	5.95	1	2	51	5.68	2	2
20	6.03	1	1	52	5.86	2	2
21	6.04	1	1	53	5.86	2	2
22	6.06	1	2	54	5.89	2	2
23	6.22	1	2	55	5.94	2	2
24	6.22	1	1	56	5.98	2	2
25	6.31	1	1	57	6.04	2	2
26	6.39	1	2	58	6.26	2	2
27	6.45	1	2	59	6.28	2	2
28	6.49	1	2	60	6.49	2	2
29	6.78	1	1	61	6.76	2	2
30	6.9	1	1	62	6.76	2	1
31	7.21	1	1	63	6.79	2	2
32	7.61	1	1				

Table 5.3.2: k -means clustering result for anilinopyrazole data based on 100 runs.

Molecule	IC50	Best binding mode	Clustering binding mode
21	0.9	1/2	1
22	7.8	1	2
23	2.7	1	1
24	17.5	1/2	1
25	18.1	1	2
26	0.5	2	2
27	1.6	2	2
28	5.2	2	2
29	7.8	1	1
30	100	2	2

Table 5.3.3: k-means clustering result for p38 MAP Kinase Inhibitors data based on 100 runs.

5.3.4 Discussion

For the anilinopyrazole data set, Sato *et al.* [85] identified two representative compounds in their experiments: compound 33 and compound 47. According to their crystal structure experiments [101], compound 33 has binding mode 1 while compound 47 has binding mode 2. Sato *et al.* [85] also performed docking simulations using various docking programs, including FlexX [81, 7, 61], GOLD [49, 50, 51] and LigandFit [105]. When type-A compounds were docked, docking programs could find the correct binding mode. However, no docking programs found the alternative binding mode of type-B compounds without a template constraint¹. For example,

¹A template is the desired position of another ligand in the receptor. It provides an example for the docking program to perform constrained docking.

in order to reproduce binding mode 2 using the docking program GOLD, the program required a template constraint, based on the crystal structure of compound 47, to get a successful docking simulation.

As shown in Table 5.3.2, our feature space clustering experiments, based only on ligand information, predict that compound 33 has binding mode 1 and compound 47 has binding mode 2. More significantly, most of the type-B compounds have binding mode 2 as suggested by our result. As mentioned by Sato *et al.* [85]: “... the binding mode of type-B compounds was more complicated than what we could expect from the experimental evidences.” In terms of statistics, for binding mode 1, we have an accuracy value of 68.52%; for binding mode 2, we have an accuracy value of 94.14%; and so the weighted average accuracy is 74.60%. With our high accuracy in predicting binding mode 2, one can apply our methodology first before performing a docking simulation to determine whether a template constraint is necessary.

For the p38 MAP Kinase Inhibitors, Hauseret *et al.* [41] produced a series of 6,9-diarylpurin-8-ones as inhibitors in the ATP binding site of p38 MAP kinase. According to their crystal structure experiments, some compounds belong to binding mode 1 while others belong to binding mode 2. The binding mode of inhibitors in the ATP binding site of p38 MAP kinase is determined by both hydrogen bonds and a lipophilic interaction. To determine which binding mode a compound belongs to, a balance of both types of interaction should be made. Hauseret *et al.* [41] also performed a docking simulation using various docking programs, including FlexX and Autodock. According to Hauser *et al.* [41], Autodock performs well in the

case of p38 MAP kinase giving similar docking results for the structurally similar inhibitors of the purin-8-one series. In contrast, the docking program FlexX does not predict a consistent binding mode.

As shown in Table 5.3.3, our clustering experiment correctly identified all compounds (Compounds 26, 27, 28, and 30) that belong to binding mode 2. In terms of statistics, for binding mode 1, we have the accuracy value of 66.7%; for binding mode 2, we have the accuracy value of 100%; the weighted average accuracy is 80%. It should be noted that clustering applied directly to the input space data did not successfully predict any alternate binding modes.

5.4 Conclusion

In conjunction with a kernel based clustering algorithm, we extended the VSMMD to the prediction of multiple binding modes, a challenging area of research that has been previously studied by means of time consuming docking simulations. The results reported in this chapter provided strong empirical evidence that our strategy has enough resolving power to distinguish multiple binding modes through the use of a kernel k -means algorithm. This has various applications. For example, using our techniques as a pre-processing step prior to a docking exercise would help to designate the template that could be used to hunt for an alternative binding mode.

Chapter 6

Inverse QSAR

A common assumption in supervised learning is that the components of the descriptors from the training and testing data are independently and identically (i.i.d.) drawn from same probability distribution. However, this assumption is difficult to guarantee, especially in traditional ligand-based drug design. In traditional ligand-based drug design, we can define various data sets: training data set, validation data set, testing data set and application data set. The first two data sets are used to derive the predictor and the testing data set is used to get a measure of success for the predictor. Note that in all these cases, we know the affinities of the ligand and it is likely that the probability distribution constraint can be met. The application data set is the set of ligands on which the predictor will be used in the future to get a new drug candidate. For this set, we do not know the affinities and it is difficult for one to guarantee the probability distribution constraint.

Let us consider a regression problem of learning $f(X)$ from training data, where X contains the descriptors of the data. We assume that $Y = f(X) + \varepsilon$,

where $E(\varepsilon) = 0$ and $Var(\varepsilon) = \sigma_\varepsilon^2$. Suppose the application data descriptors independently follow a probability distribution with density $p_a(X)$ and the training data descriptors independently follow a probability distribution with density $p_t(X)$. We can derive an expression for the expected error of a regression model $\hat{f}(X)$ as:

$$\begin{aligned} Errr &= E \left(\int \left(\hat{f}(X) - f(X) \right)^2 p_a(X) dX \right) \\ &= \int \left(E\hat{f}(X) - f(X) \right)^2 p_a(X) dX + E \left(\int \left(E\hat{f}(X) - \hat{f}(X) \right)^2 p_a(X) dX \right) \end{aligned} \quad (6.0.1)$$

The first term is the bias and the second term is the variance. The situation where the training and application distributions are different is referred to as the situation under the covariate shift or the sample selection bias [95]. When $p_t(X) \neq p_a(X)$, two difficulties arise in a learning process:

- Prediction can be inaccurate if the region with high application data density has low training data density.
- Cross-validation can be heavily biased, because the generalization error is over-estimated in the high training data density region and it is under-estimated in the high application data density region.

In traditional ligand-based drug design this is rarely, if ever, discussed. The predictor is often applied to application molecules that have very little relationship to the training data. In these cases, the predictor is optimistically treated as if it actually incorporates an algorithm that has some firm and direct relationship to the biological context of the problem.

In this chapter, we try to sidestep such concerns. We develop a set of reverse engineering strategies for QSAR modeling, based on our VSMMD, that actually generate the structure of a new drug candidate. While the training set is still used to generate a new image point in the feature space, the reverse engineering strategies allow us to develop a new drug candidate that is independent of issues related to probability distribution constraints placed on the descriptors in testing or application set.

6.1 Introduction

In the previous chapters, we strive to establish the quantitative dependency between the molecular properties of a ligand and its binding affinity or binding modes. In contrast to this approach, the inverse-QSAR problem seeks to find a new molecular descriptor from which one can recover the structure of a molecule that possesses a desired activity or property. Surprisingly, there are very few papers providing solutions to this problem [22]. It is a difficult problem because the molecular descriptors involved with the inverse-QSAR algorithm must adequately address the forward QSAR problem for a given biological activity if the subsequent recovery phase is to be meaningful. In addition, one should be able to construct a feasible molecule from such a descriptor. The difficulty of recovering the molecule from its descriptor is the major limitation of most inverse-QSAR methods.

Most of the proposed techniques are stochastic in nature (see papers such as [90, 106, 62]). A limited number of deterministic approaches have been developed including Kier and Hall's [39, 38, 53] approach based on a count of paths, and a

strategy based on signature descriptors (See Faulon *et al.* [14, 25, 23, 24]).

The key to an effective method lies in the use of a descriptor that facilitates the reconstruction of the corresponding molecular structure. Ideally, such a descriptor should be informative, have good correlative abilities in QSAR applications, and most importantly, the construction of the descriptor should be computationally efficient. A descriptor should also have a low degeneracy, that is, it should lead to a limited number of solutions when a molecular recovery algorithm is applied.

Currently, kernel methods are popular tools in QSAR modeling and are used to predict attributes such as activity towards a therapeutic target, ADMET properties (absorption, distribution, metabolism, excretion, and toxic effects), and adverse drug reactions. Various kernel methods based on different molecular representations have been proposed for QSAR modeling [4]. They include the SMILES string kernel [100], graph kernels [28, 72, 80] and a pharmacophore kernel [71]. However, none of these kernel methods have been used for the inverse-QSAR problem.

In this chapter, we investigate the reversibility of our previously reported descriptor, the vector space model molecular descriptor (VSMMD). VSMMD is based on a vector space model that is suitable for kernel studies in QSAR modeling. Our approach to the inverse-QSAR problem consists of first deriving a new image point in the kernel feature space and then finding the corresponding pre-image descriptor in the input space. Then, we use a recovery algorithm to generate a chemical structure template suitable for high throughput screening. In section 6.2, we provide a detailed description of our inverse-QSAR approach using our VSMMD approach. In section 6.3, we present the experimental results of our descriptors in the vector

space setting.

6.2 VSMMD Inverse-QSAR Approach

Our inverse-QSAR approach can be described in five steps. The first two steps are to perform a QSAR analysis. In the first step, we generate a VSMMD for each compound in the training set. A component selection algorithm is then used to select the most important components of the initial VSMMD. Once these component positions are determined for the descriptor vector, these components are used across all initial descriptors of the training set to get the final descriptors. Then, in the second step, we use a kernel function to map the VSMMD to a feature space typically used for classification or regression analysis. The third step is to design or to generate a new point in the kernel feature space using a kernel feature space algorithm (e.g. the centroid of highly active compounds). In the fourth step, we map this point from the feature space back to the input space using a pre-image approximation algorithm. In the last step, the molecular structure template will be built by our VSMMD molecule recovery algorithm. Figure 6.2.1 illustrates the overall processing.

6.2.1 Vector Space Model Molecular Descriptor (VSMMD)

The first two steps for the inverse-QSAR approach are to perform forward QSAR analysis. In the first step, we generate a VSMMD for each compound in the training set as described in section 3.2. Next, the KACS algorithm described in section 4.2.

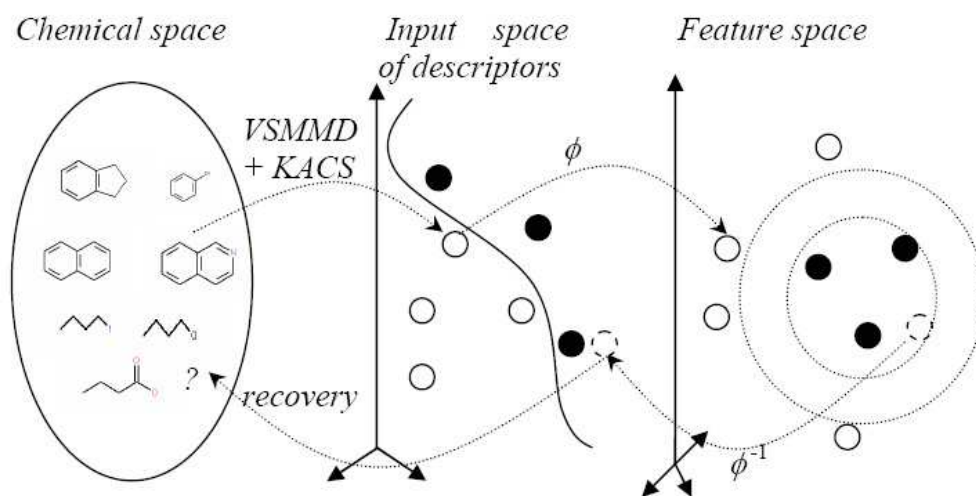


Figure 6.2.1: Overall concept for the VSMMD inverse-QSAR approach.

is used to select the most important components of the initial VSMMD. Once these component positions are determined for the descriptor vector, these components are used across all initial descriptors of the training set to get the final descriptors. Then, in the second step, we use a kernel function to map the VSMMD to the feature space for classification or regression analysis as described in section 3.3.

6.2.2 Designing descriptor image point in feature space

Suppose we have a set of n molecular descriptors S , designated as $S = \{d_1, d_2, \dots, d_n\}$ where each d_i is in the input space X . Let us assume we are using a Gaussian vector space kernel as defined in (3.5.6). Under this kernel, any point $d_i \in X$, is implicitly mapped to an image $\phi(d_i)$ in the feature space FS . With this kernel mapping, we can define the set $\phi(S) = \{\phi(d_1), \phi(d_2), \dots, \phi(d_n)\} \in FS$.

In this sub-section, we will evaluate various properties of the data set $\phi(S)$. We

provide a set of elementary algorithms to do various calculations such as distance between two descriptor image points in the feature space.

The feature space centroid derived from highly active compounds

Using equation (4.2.12), we can obtain the norm of a descriptor image point $\phi(d)$:

$$\|\phi(d)\| = \sqrt{\|\phi(d)\|^2} = \sqrt{\langle\phi(d), \phi(d)\rangle} = \sqrt{k(d, d)}. \quad (6.2.2)$$

A special case of the norm is the length of the line joining two images $\phi(d_1)$ and $\phi(d_2)$, which can be computed using:

$$\begin{aligned} \|\phi(d_i) - \phi(d_j)\|^2 &= \langle\phi(d_i) - \phi(d_j), \phi(d_i) - \phi(d_j)\rangle \\ &= \langle\phi(d_i), \phi(d_i)\rangle - 2\langle\phi(d_i), \phi(d_j)\rangle + \langle\phi(d_j), \phi(d_j)\rangle \\ &= k(d_i, d_i) - 2k(d_i, d_j) + k(d_j, d_j). \end{aligned} \quad (6.2.3)$$

The norm described by (6.2.3) represents the distance between two descriptor image points in the feature space. We define the centroid ϕ_s of the molecule data set S in the feature space as:

$$\phi_s = \frac{1}{n} \sum_{i=1}^n \phi(d_i). \quad (6.2.4)$$

The norm of the centroid can be calculated using only the evaluations of the kernel on the inputs:

$$\begin{aligned} \|\phi_s\|^2 = \langle\phi_s, \phi_s\rangle &= \left\langle \frac{1}{n} \sum_{i=1}^n \phi(d_i), \frac{1}{n} \sum_{j=1}^n \phi(d_j) \right\rangle \\ &= \frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n \langle\phi(d_i), \phi(d_j)\rangle \\ &= \frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n k(d_i, d_j). \end{aligned} \quad (6.2.5)$$

Note that the result is the average of the entries in the kernel matrix. The inner product between a descriptor image point $\phi(d)$ and the centroid ϕ_s is given by:

$$\begin{aligned}\langle \phi(d), \phi_s \rangle &= \left\langle \phi(d), \frac{1}{n} \sum_{i=1}^n \phi(d_i) \right\rangle = \frac{1}{n} \sum_{i=1}^n \langle \phi(d), \phi(d_i) \rangle \\ &= \frac{1}{n} \sum_{i=1}^n k(d, d_i).\end{aligned}\tag{6.2.6}$$

Using equation (6.2.3), we can calculate the distance between $\phi(d)$ and the centroid ϕ_s in the feature space by:

$$\begin{aligned}\|\phi(d) - \phi_s\|^2 &= \|\phi(d)\|^2 - 2 \langle \phi(d), \phi_s \rangle + \|\phi_s\|^2 \\ &= k(d, d) - \frac{2}{n} \sum_{i=1}^n k(d, d_i) + \frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n k(d_i, d_j).\end{aligned}\tag{6.2.7}$$

Recall that the kernel-based learning algorithms work by embedding the data into the feature space, and searching for a linear relationship within this feature space. With this linear relationship, it makes sense to derive a new descriptor image point using the centroid point of the highly active compound's image points which will share the general properties of all highly active compounds. If the centroid point can be mapped from the feature space back to the input space, we can obtain the descriptor of a new candidate molecule. Figure 6.2.2 illustrates this idea. It should be stressed that when a nonlinear kernel (such as the Gaussian kernel) is used, then a point in the feature space bears a nonlinear relationship to the point in the input space. The centroid in the feature space would rarely, if ever, map back to the centroid in the input space. We have chosen to use the centroid in the feature space because there is more assurance of generating a new point in the feature space that in some sense represents a point of reasonable interpolation in

this high dimensional space. Picking an arbitrary point in the feature space runs a higher risk of extrapolation which may be difficult to avoid especially when a small training set is spread over the higher dimensional feature space in some rarefied manner.

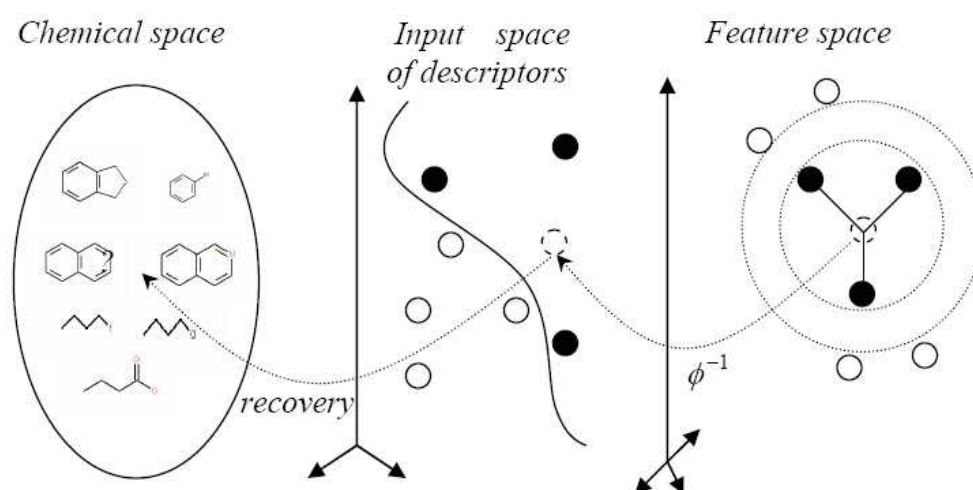


Figure 6.2.2: Deriving a new image in kernel feature space.

The inverse in the input space is called the pre-image. We will discuss pertinent details in the pre-image subsection.

There are several studies that use a feature space centroid to generate new data. Kwok and his colleagues used a feature space centroid to generate a new data point for hand-written digit recognition [63] and speech processing [74, 73]. In both applications, the pre-image has been shown to be robust and meaningful. In the next subsection we describe another strategy for the derivation of a new feature space point.

Minimum Enclosing and Maximum Excluding Hypersphere

In the last subsection, we derived a new descriptor image point using the centroid of feature space images derived from the highly active compounds. In this subsection, we use the highly active compounds to derive two hyperspheres with the same center. The center of the hyperspheres is then mapped back from the feature space to the input space to generate the descriptor of a new candidate molecule.

Suppose we can identify a subset $G \subseteq S$ where G contains the descriptors of molecules in the chemical space with the highest activity. We let $|G|$ represent the number of descriptors in G . In an ideal situation, the feature space images of G will be spherically separable from all the other descriptor images mapped over from S . With this assumption, we can derive two hyperspheres, sharing the same center a , such that the images of all descriptors derived from highly active molecules are enclosed by the inner hypersphere H_1 and all the remaining images are excluded by the outer hypersphere H_2 . Let r_1 be the radius of the inner hypersphere and let r_2 be the radius of the outer hypersphere. Consequently, we have:

$$\begin{aligned} \|a - \phi(d_i)\|^2 &\leq r_1^2 \text{ for } d_i \in G, \\ \|a - \phi(d_i)\|^2 &\geq r_2^2 \text{ for } d_i \notin G. \end{aligned} \tag{6.2.8}$$

Figure 6.2.3 illustrates this idea.

Following the development of Liu and Zheng's minimum enclosing and maximum excluding machine (MEMEM) [68], we want the inner hypersphere H_1 as small as possible for a good description of the highly active class. In the meantime, we want the outer hypersphere H_2 as large as possible. In other words, we try

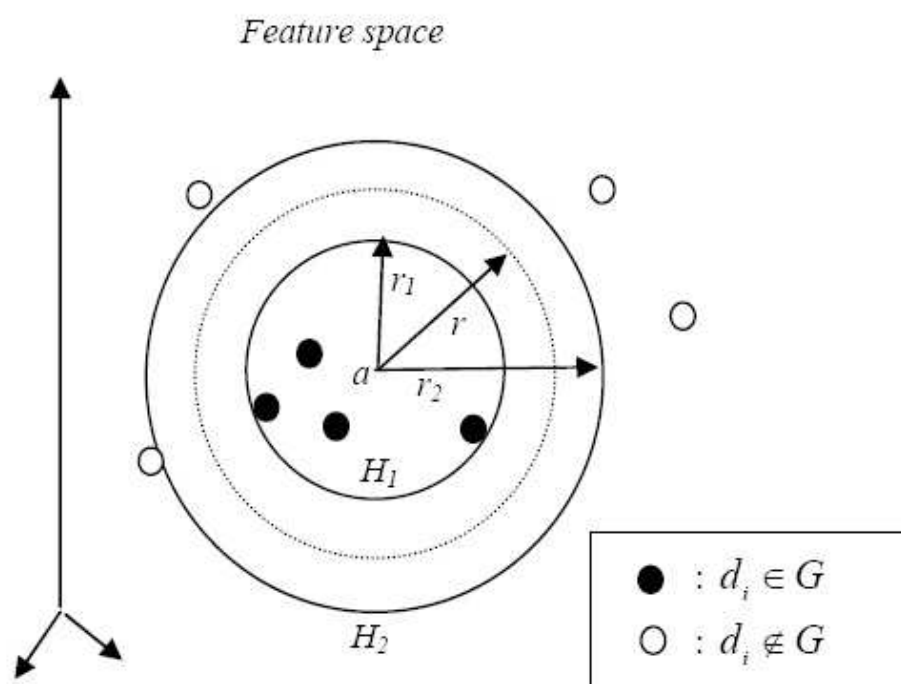


Figure 6.2.3: Minimum enclosing and maximum excluding hypersphere in the feature space.

to maximize the area between two hyperspheres H_2 and H_1 . Note that the area between two hyperspheres H_2 and H_1 is proportional to the quantity $(r_2^2 - r_1^2)$. Let $\Delta r^2 = \frac{1}{2}(r_2^2 - r_1^2)$ and $r^2 = \frac{1}{2}(r_1^2 + r_2^2)$, we can formulate the objective function to have r_1^2 as small as possible and Δr as large as possible by minimizing the quantity $r_1^2 - (r_2^2 - r_1^2) = r^2 - 3\Delta r^2$, or equivalently

$$\frac{1}{3}r^2 - \Delta r^2. \quad (6.2.9)$$

Replacing the constant $\frac{1}{3}$ by η , which controls the trade off between the importance of the inner hypersphere and the outer hypersphere. The objective function will become:

$$\min_{a, r^2, \Delta r^2} \{ \eta r^2 - \Delta r^2 \}, \quad (6.2.10)$$

subject to:

$$\begin{aligned} \|a - \phi(d_i)\|^2 &\leq r^2 - \Delta r^2 \text{ for } d_i \in G, \text{ and} \\ \|a - \phi(d_i)\|^2 &\geq r^2 + \Delta r^2 \text{ for } d_i \notin G. \end{aligned} \quad (6.2.11)$$

Our analysis will require Lagrange multipliers α_i and labels y_i such that

$$\begin{aligned} y_i &= 1 \text{ for } d_i \in G \\ y_i &= -1 \text{ for } d_i \notin G. \end{aligned} \quad (6.2.12)$$

The dual problem of equation (6.2.9) can be obtained [68] as:

$$\max_{\alpha_i} \left\{ -\frac{1}{\eta} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j k(d_i, d_j) + \sum_{i=1}^n \alpha_i y_i k(d_i, d_i) \right\} \quad (6.2.13)$$

subject to:

$$\sum_{i=1}^n \alpha_i y_i = \eta, \quad \sum_{i=1}^n \alpha_i = 1, \text{ and } \alpha_i \geq 0 \text{ for } i = 1 \text{ to } n. \quad (6.2.14)$$

In practice, the data may not be separable in this fashion. By introducing slack variables $\zeta_i \geq 0$, equation (6.2.9) becomes:

$$\min_{a, r^2, \Delta r^2, \zeta_i} \left\{ \eta r^2 - \Delta r^2 + C \sum_{i=1}^n \zeta_i \right\} \quad (6.2.15)$$

subject to:

$$\begin{aligned} \|a - \phi(d_i)\|^2 &\leq r^2 - \Delta r^2 + \zeta_i \text{ for } d_i \in G, \\ \|a - \phi(d_i)\|^2 &\geq r^2 + \Delta r^2 - \zeta_i \text{ for } d_i \notin G, \\ \text{and } \Delta r^2 &\geq 0. \end{aligned} \quad (6.2.16)$$

By using equation (6.2.15), we allow some negative samples inside the inner hypersphere and some positive samples outside the outer hypersphere. The corresponding dual problem of equation (6.2.15) obtained by [68] is:

$$\max_{\alpha_i, \beta} \left\{ -\frac{1}{\eta} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j k(d_i, d_j) + \sum_{i=1}^n \alpha_i y_i k(d_i, d_i) \right\} \quad (6.2.17)$$

subject to

$$\sum_{i=1}^n \alpha_i y_i = \eta, \quad \sum_{i=1}^n \alpha_i - \beta = 1, \quad \beta \geq 0, \quad \text{and } C \geq \alpha_i \geq 0 \text{ for } i = 1 \text{ to } n, \quad (6.2.18)$$

where β is the Lagrange multiplier associated with the constraint $\Delta r^2 \geq 0$, and C is some constant to be determined with a validation data set.

The center a of the hyperspheres is then mapped back from the feature space to the input space to generate the descriptor of a new candidate molecule.

6.2.3 The Pre-Image Problem

In Section 2.2.3, we illustrated how a point in the input space is mapped to the feature space via the implicit function ϕ . In this section, we are interested in

finding how a point in the feature space can be mapped back to the input space. Formally, this is called the pre-image problem of reconstructing patterns from their representation in feature space (see Fig. 6.2.4).

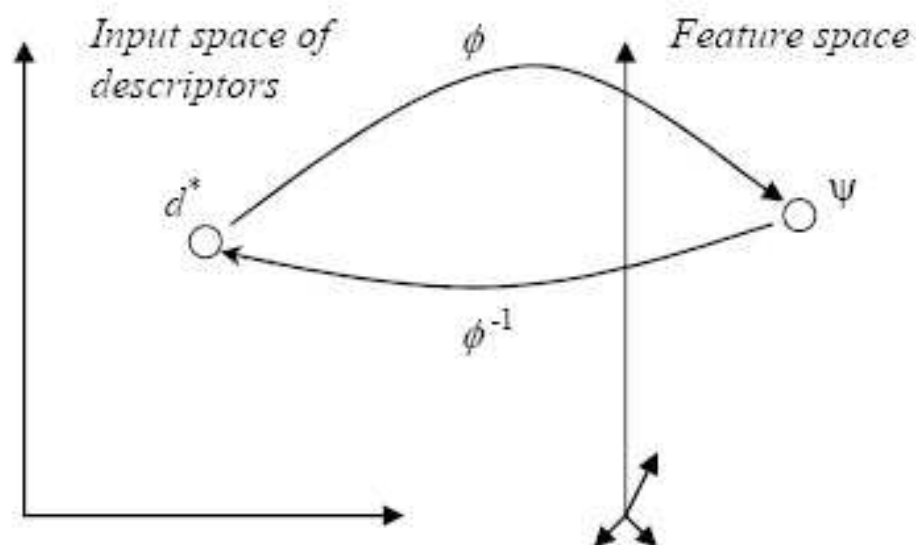


Figure 6.2.4: The pre-image Problem.

Let Ψ be a point in the Reproducing Kernel Hilbert Space (RKHS) FS . The pre-image of $\Psi \in FS$ is a point $d^* \in X$ (the original input space). Formally,

$$\Psi = \phi(d^*). \quad (6.2.19)$$

The problem of finding the pre-image d^* is equivalent to the problem of finding the inverse function of ϕ defined in equation (6.2.19):

$$d^* = \phi^{-1}(\Psi). \quad (6.2.20)$$

However, the problem of finding $\phi^{-1}(\cdot)$ is a typical ill-posed problem. A problem is said to be ill-posed if the solution is not unique, does not exist, or is not a continuous function of the data [76].

One possible way to overcome this problem is to look for \hat{d}^* an approximation of the pre-image such that $\phi(\hat{d}^*)$ is as close as possible to Ψ . Formally, we search for $\hat{d}^* \in X$, such that

$$\hat{d}^* = \arg \min_{d \in X} \|\phi(d) - \Psi\|_F^2. \quad (6.2.21)$$

Typically, we will have Ψ defined as some linear combination of implicit mappings from the input space. Consequently, expanding equation (6.2.21), we get:

$$\hat{d}^* = \arg \min_{d \in X} \left\| \phi(d) - \sum_{i=1}^n \alpha_i \phi(d_i) \right\|_F^2, \quad (6.2.22)$$

which can be rewritten as:

$$\hat{d}^* = \arg \min_{d \in X} \left[k(d, d) - 2 \sum_{i=1}^n \alpha_i k(d, d_i) + \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j k(d_i, d_j) \right]. \quad (6.2.23)$$

Using equation (6.2.23), an inversion problem turns out to be an optimization problem. There are several algorithms that attempt to solve this optimization problem. Schölkopf *et al.* [86] proposed an iterative fixed point algorithm strategy. Kwok and Tsang [63] proposed another method that exploits the correspondence between distances in the input space and the feature space. Also, a standard gradient optimization method can be used to find an approximation of the pre-image [86]. Note that all these methods are only guaranteed to find a local optimum.

In this thesis, we are going to follow Kwok and Tsang [63] and use their approach to approximate the pre-image. Their algorithm is based on the notion of distance

constraints. They assume that there exists a simple relationship between distances in the input space and distances in feature space. Figure 6.2.5 illustrates these relationships.

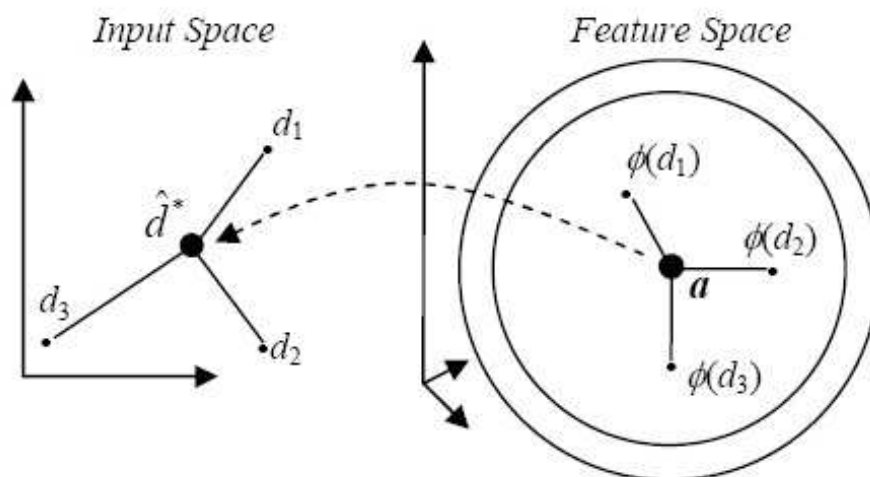


Figure 6.2.5: Kwok and Tsang pre-image strategy.

Suppose we have derived the center a of the hyperspheres using equation (6.2.17). It was shown that the center a of the hyperspheres is a linear combination of the training samples [68] i.e.: $a = \frac{1}{\eta} \sum_{i=1}^n \alpha_i y_i \phi(d_i)$. Recall that η is defined in equation (6.2.17). The norm of the center can be calculated using only the evaluations of the kernel on the training sample inputs:

$$\begin{aligned} \langle a, a \rangle &= \left\langle \frac{1}{\eta} \sum_{i=1}^n \alpha_i y_i \phi(d_i), \frac{1}{\eta} \sum_{j=1}^n \alpha_j y_j \phi(d_j) \right\rangle \\ &= \frac{1}{\eta^2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j k(d_i, d_j). \end{aligned} \quad (6.2.24)$$

For each training sample d_i , we can derive $dist_{a,i}^F = \|a - \phi(d_i)\|^2$ representing the

square of the distance between the training sample image d_i and the center a of the hyperspheres:

$$\begin{aligned}
 dist_{a,i}^F &= \|a - \phi(d_i)\|^2 = \langle a, a \rangle - 2 \langle a, \phi(d_i) \rangle + \langle \phi(d_i), \phi(d_i) \rangle \\
 &= \langle a, a \rangle - 2 \left\langle \frac{1}{\eta} \sum_{j=1}^n \alpha_j y_j \phi(d_j), \phi(d_i) \right\rangle + \langle \phi(d_i), \phi(d_i) \rangle \\
 &= \frac{1}{\eta^2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j k(d_i, d_j) - \frac{2}{\eta} \sum_{j=1}^n \alpha_j y_j k(d_j, d_i) + k(d_i, d_i).
 \end{aligned} \tag{6.2.25}$$

Using the Gaussian kernel as specified in equation (3.5.6) and an observation made by Kwok and Tsang [63], the corresponding input space distance $dist_{a,i}$ between the center and training sample d_i can be found using:

$$dist_{a,i} = -\frac{1}{\sigma} \log\left(1 - \frac{1}{2} dist_{a,i}^F\right). \tag{6.2.26}$$

To speed up the algorithm (as observed by Kwok and Tsang [63]), only the p closest training sample images of a will be considered. From (6.2.25), we can identify the p closest neighbours of a in the feature space. Using (6.2.26), we can convert these p closest neighbour distances in the feature space to their corresponding input space distances. Let \mathbf{b} be the vector representing these input space distances with

$$\mathbf{b} = [dist_{a,1}, dist_{a,2}, \dots, dist_{a,p}]^T. \tag{6.2.27}$$

Their corresponding descriptors in the input space are $d_1, d_2, \dots, d_p \in \mathbb{R}^m$, and the centroid is defined as $\bar{d} = \frac{1}{p} \sum_{i=1}^p d_i$. Let $D = [d_1, d_2, \dots, d_p]$ be a $m \times p$ matrix. To establish the centroid at the origin, we let

$$A = \left(I - \frac{1}{p} \mathbf{1}\mathbf{1}^T \right) D^T, \tag{6.2.28}$$

where I is a $p \times p$ identity matrix, and $\mathbf{1}$ is a p dimensional vector with each component equal to 1.

Assuming matrix D is of rank q , we obtain the singular value decomposition (SVD) of A^T as:

$$A^T = USV^T = UZ, \quad (6.2.29)$$

where $U = [u_1, u_2, \dots, u_q]$ is a $m \times q$ matrix with orthonormal columns composed of the u_i 's, and $Z = [z_1, z_2, \dots, z_p]$ is a $q \times p$ matrix with the i -th column z_i being the projection of d_i on to u_j 's such that the squared distance of d_i to the origin is equal to $\|z_i\|^2$. Let $\mathbf{b}_0 = [\|z_1\|^2, \|z_2\|^2, \dots, \|z_p\|^2]^T$, the pre-image \hat{d}^* of the center a can be obtained by:

$$\hat{d}^* = -\frac{1}{2}US^{-1}V^T(\mathbf{b} - \mathbf{b}_0) + \bar{d}. \quad (6.2.30)$$

6.2.4 The Need for a Nonlinear Kernel

The nonlinear implicit mapping provided by the kernel operation allows us to generate an inner product in the feature space by computing a kernel function that has arguments taken from the input space. More significantly, when a nonlinear kernel is used, linear operations in the feature space correspond to nonlinear operations in the input space. This is important because the nonlinear mapping will involve various cross products of components within a vector of the input space. As a consequence, linear structures within the feature space correspond to nonlinear or .warped. structures in the input space.

To illustrate this, we ran a small experiment with the COX2 training set (described in the later section): As described earlier, the new feature space point a , generated by extracting the center of the enclosing hypersphere, was mapped back to the input space to get its pre-image \hat{d}^* . We then formed the set S_{fe} of points in

the input space (taken from the training set) that produced the ten closest neighbours of a under the kernel mapping. This set S_{fe} was compared with the set S_{in} containing the 10 closest neighbours of \hat{d}^* in the input space (these neighbours derived using the Euclidean metric). Because of the warping effect, pre-images of close neighbours in the feature space are not necessarily the closest neighbours of the pre-image \hat{d}^* in the input space, in fact, the intersection of S_{in} and S_{fe} is only 3 descriptors. More significant: the average affinity of molecules in S_{in} is 8.03 while the average affinity of molecules in S_{fe} is 8.73. This provides empirical evidence that the nonlinear mapping provided by the kernel function helps us to select input space descriptors that are more significant when considering their corresponding affinities.

6.2.5 Recovering the molecule

In order to solve the recovery problem for chemical structures, we have to investigate a way to derive a graph representing the 2D structure of a molecule that has \hat{d}^* as its descriptor. There are several related studies that attempt to find such a graph, for example, Bakir *et al.* [5] who worked with a stochastic search algorithm. However, this recovery problem is not well studied from a computational viewpoint. Akutsu and Fukagawa [2, 3] proposed a dynamic programming algorithm for inferring a chemical structure from a descriptor. However, the algorithms are not practical for a large data set. Previous studies focused on creating a real chemical structure, which defined too many constraints on the problem due to the complexity of chemical structure. In our study, we do not attempt to recover a real chemical structure;

instead, we generate a chemical structure template with physiochemical properties only. This simplifies the problem and makes it practical for real data sets.

6.2.6 Reversible VSMMD

For illustration and without loss of generality, we will assume that the atom count associated with the VSMMD is two, and we further assume all the aromatic rings are replaced by “super atoms” containing all the rings’ physicochemical properties. Figure 6.2.6 illustrates a simplified VSMMD using the same example as in Figure 3.1.1.

From Figure 6.2.6, we observe that the VSMMD model contains only the physiochemical properties of the chemical structure. As a result, for the recovery problem, we do not attempt to recover the entire chemical structure. Instead, we attempt to generate a chemical structure template with physiochemical properties only. A chemical structure template converted from the molecule shown in Figure 6.2.6 is illustrated in Figure 6.2.7.

In the next subsection, we define the notion of a structure template and show how it can be derived.

Forming the De Bruijn Graph

A De Bruijn graph is a graph whose nodes are labeled by strings over some alphabet and whose edges indicate some relations between the strings in nodes. In this subsection, we use the De Bruijn graph as a data structure to derive a chemical structure template.

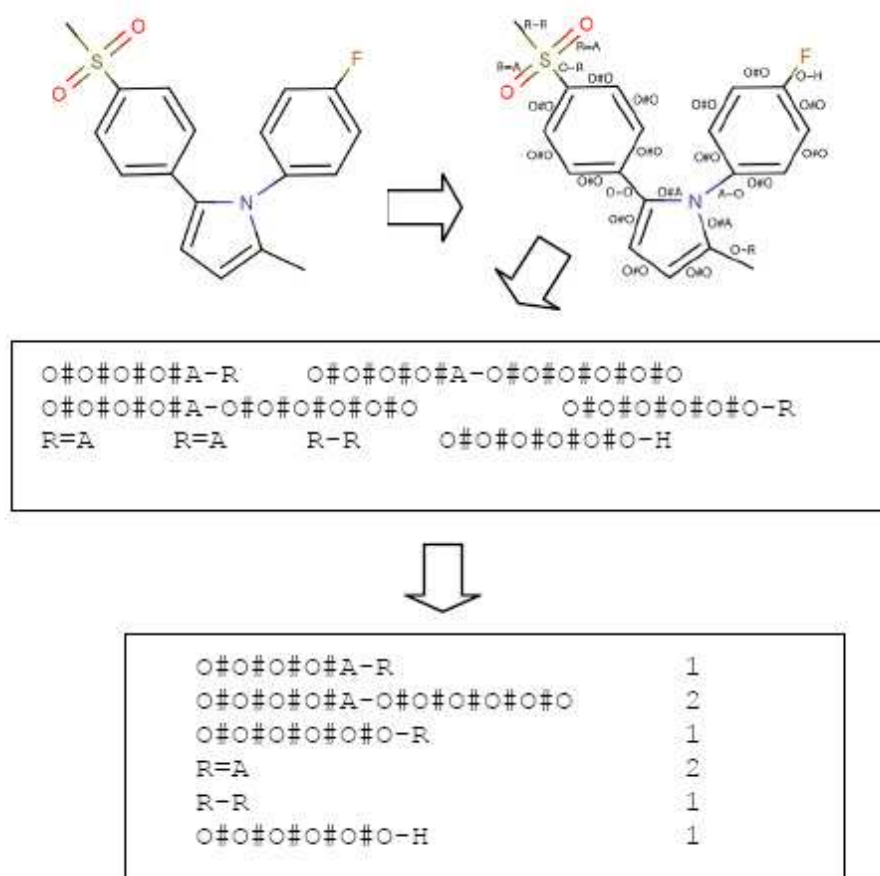


Figure 6.2.6: Simplified VSMMD with an aromatic ring treated as a super atom.

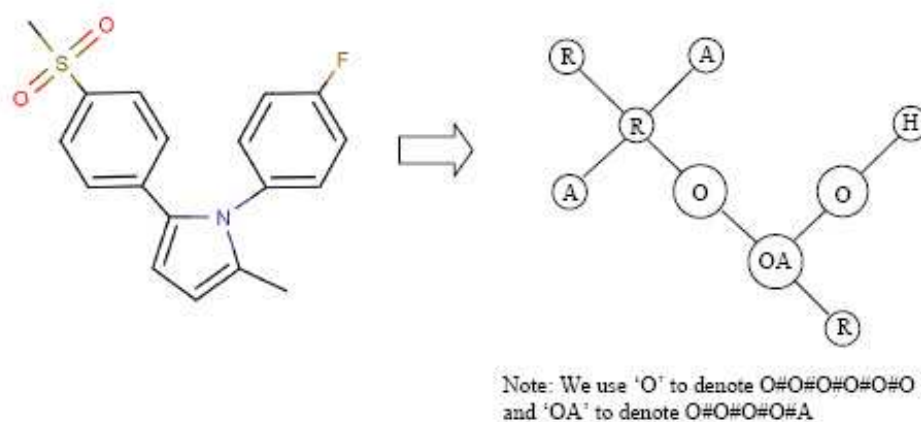


Figure 6.2.7: An example of a chemical structure template.

If we consider a molecule to be comprised of molecular fragments then it is clear that there is a hierarchical organization of these fragments. A linear fragment with an atom count of three can be seen as containing two smaller fragments each with an atom count of two and of course the two fragments overlap in the central atom. If we restrict a fragment to have an atom count of two, then it will contain two elementary fragments, namely two atoms, each labeled with their atom types.

Informally: The purpose of a De Bruijn graph is to provide a data structure that shows how small fragments combine to build larger fragments. Since we wish to handle ring structures using the simplification of a “super atom”, we will abuse these concepts slightly and consider the fragments under discussion to be fragments within a template as describe in the previous section.

Suppose we are dealing with fragments that have an atom count designated as f_a . The De Bruijn graph D is constructed in the following way: We provide a vertex for each fragment that has an atom count equal to $f_a - 1$. In our simplified

case, $f_a = 2$ and each vertex will represent an atom labeled with a physicochemical property. We then add a bi-directional edge from vertex a to vertex b if the fragments associated with these vertices are within a larger fragment with atom count equal to f_a . Each edge is weighted with a value representing the number of times that this larger type of fragment occurs in the template.

Although we have been referencing the template in describing the construction of D , it should be clear that it is possible to accomplish the generation of D by processing the descriptor that represents this template. Figure 6.2.8 illustrates the De Bruijn graph D generated from the VSMMD shown in Figure 6.2.6 .

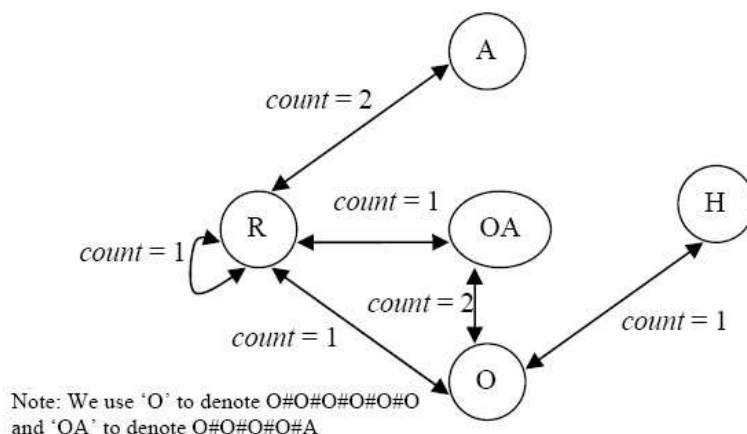
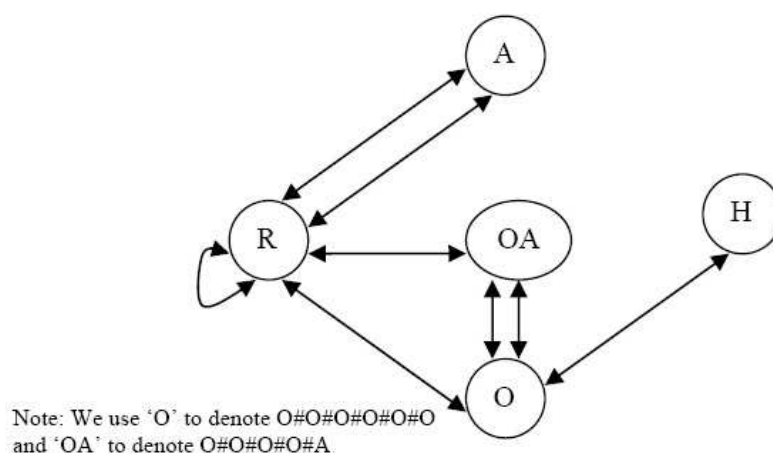


Figure 6.2.8: The De Bruijn graph D for VSMMD shown in Figure 6.2.6.

The De Bruijn graph can be *expanded* by replacing each edge carrying weight c with c unweighted edges, each with the same direction as the original edge. Figure 6.2.9 illustrate this expansion. Let M be the resulting unweighted De Bruijn graph.

From the VSMMD, we know the exact number of vertices that should appear in the chemical structure template. With this information, we can derive a chemical

Figure 6.2.9: The expanded graph M .

structure template from the De Bruijn graph by finding an Euler circuit of M .

All possible Euler Circuits

An Euler circuit is a circuit on the graph such that each edge is traversed exactly once. Each traversal of an edge corresponds to the consumption of one instance of a component in the VSMMD. The problem of finding an Eulerian circuit of a graph is well known and there exists a linear time algorithm for its derivation [111]. The following is the pseudocode of the Euler circuit recursive algorithm.

- 1 **EULER**(q)
- 2 Path \leftarrow none
- 3 For each unmarked edge e leaving q do
- 4 Mark(e)


```

5   Path ← EULER(oppositevertex(e)) || Path1
6   endfor
7   Return Path

```

Each Euler circuit will represent the extraction of a unique chemical structure template from the De Bruijn Graph M . Figure 6.2.10 shows a subset of all the Euler circuits that can be generated. The circuit labeled with a ‘*’ corresponds to the chemical structure template illustrated in Figure 6.2.6. The total number of possible Euler circuits for the chemical structure in Figure 6.2.5 is 2700.

```

R->A->R->O->OA->O->H->O->R->R->R->A->R->OA->O->OA->R
R->A->R->O->OA->O->H->O->R->R->R->OA->O->OA->R->A->R
R->A->R->O->OA->O->H->O->OA->R->A->R->OA->O->R->R->R
R->A->R->O->OA->O->H->O->OA->R->A->R->R->OA->O->R->R
R->A->R->O->OA->O->H->O->OA->R->A->R->R->R->OA->O->R
R->A->R->O->OA->O->H->O->OA->R->OA->O->R->A->R->R->R *
R->A->R->O->OA->O->H->O->OA->R->OA->O->R->R->A->R->R
R->A->R->O->OA->O->H->O->OA->R->OA->O->R->R->R->A->R
R->A->R->O->OA->O->H->O->OA->R->R->A->R->OA->O->R->R
R->A->R->O->OA->O->H->O->OA->R->R->A->R->R->OA->O->R

```

Figure 6.2.10: Some possible Euler circuits.

In order to generate all possible chemical structures associated with the VS-MMD, we have to find all Euler circuits. Different chemical structure templates correspond to the different possible orderings when traversing outgoing edges of each vertex. This produces a factorial explosion with respect to the number of outgoing edges of each vertex. Thus, finding all Euler circuits is not feasible.

¹|| represents concatenation

To overcome this, we have developed an algorithm that generates Eulerian circuits by doing a guided walk of the graph. During the walk we choose an outgoing edge in a probabilistic fashion. The choice is dependent on statistics that are gathered from the descriptors in the training set. To accomplish this, we have built a statistical model that is used to estimate the probability of an Euler circuit.

Let E denote a path of N edges, that is, $E = e_1, e_2, \dots, e_N$. Then, by the probability chain rule, we can obtain:

$$P(E) = P(e_1) \prod_{i=2}^N P(e_i | e_1, \dots, e_{i-1}) \quad (6.2.31)$$

To estimate the conditional probabilities $P(e_i | e_1, e_2, \dots, e_{i-1})$, we need training data consisting of a large number of Euler circuits each corresponding to some particular molecular template. One can obtain these conditional probability distributions from the training data by keeping statistics on the dependency between the next edge to traverse and the history of the previously traversed edges. Seen as probabilities of traversal, we of course use normalized values so that the probabilities of all possible “next-edges” sum to 1.0.

To simplify the statistical model, independence assumptions are made so that each edge depends only on the last t edges. Consequently, we have a Markov model that provides an approximation of how the fragments, each labeled with physicochemical properties, are connected within the template. More precisely, our model predicts traversal of e_i based on previously traversed edges $e_{i-1}, e_{i-2}, \dots, e_{i-t}$. Formally, this is described as:

$$P(E) = P(e_1)P(e_2|e_1)P(e_3|e_1, e_2) \cdots \prod_{i=t+1}^N P(e_i|e_{i-t}, \cdots, e_{i-1}) \quad (6.2.32)$$

If we could handle unlimited amounts of training data, the maximum likelihood estimate of $P(e_i|e_{i-t}, \cdots, e_{i-1})$ would be:

$$P(e_i|e_{i-t}, \cdots, e_{i-1}) = \frac{c(e_{i-t}, e_{i-t+1}, \cdots, e_{i-1}, e_i)}{c(e_{i-t}, e_{i-t+1}, \cdots, e_{i-1})} \quad (6.2.33)$$

where $c(e_{i-t}, \cdots, e_{i-1}, e_i)$ is the number of times the edge sequence $e_{i-t}, \cdots, e_{i-1}, e_i$ is seen in the training data.

As an example, consider Figure 6.2.11 with $t = 1$. In order to determine the edge to be traversed next when at the node labeled “R”, we consult the associated probabilities: $P(\text{R-O} \mid \text{A-R})$ and $P(\text{R-A} \mid \text{A-R})$. We traverse the edge with the largest probability first.

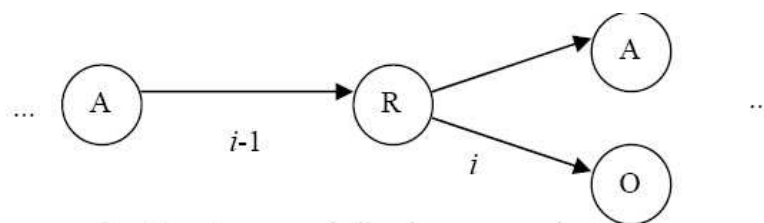


Figure 6.2.11: An example in edges traversal.

A threshold h can also be used as a cutoff to limit the number of edges that the algorithm should examine in an effort to sidestep the factorial explosion that can occur without this limitation. With this understanding, we can compute the overall probability of each Euler circuit using equation (6.2.32).

With $t = 1$ and $h = 2$, a total of 102 Euler circuits are generated. The six Euler circuits with highest probability are shown in Figure 6.2.12. They corresponded to two unique chemical templates. Templates 1 and 3 are the exact templates derived from the chemical structure shown in Figure 6.2.6.

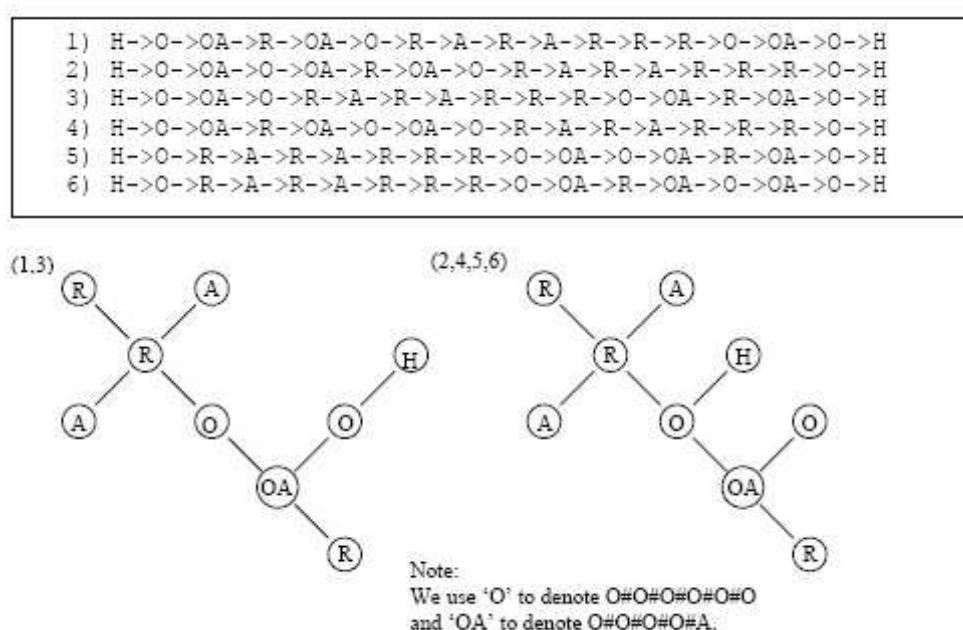


Figure 6.2.12: Six highest probability Euler Circuits for VSMMD shown in Fig. 6.2.4 and the corresponding chemical structure templates.

There are several related research papers that attempt to retrieve the order of elements that are part of a larger structure using Eulerian circuits. Cortes *et al.* [15] retrieve the order of words in documents using an Eulerian circuit approach. Pevzner *et al.* [79] assembly DNA fragments using an Eulerian circuit.

As mentioned in [63], in general, there was no exact pre-image in the input space; The pre-image returned by the algorithm was an approximation and so it

was compromised by approximation errors. Because of these approximation errors, the following problems may exist:

- The pre-image vector may consist of non-integer components.
- The pre-image vector may not form a fully connected De Bruijn Graph.

Our solution to overcome the first problem is to round the components to obtain integer counts. To deal with the case where the graph is not connected, the all-possible Euler circuits algorithm is called at each vertex whose outgoing edges are not all marked. The resulting path is the concatenation of the paths returned by different calls to the all-possible Euler algorithm. More precisely, a bidirectional edge with the largest conditional probability based on previously traversed edges in the path, (using the same Markov model that we set up in the previous section), is added to connect two Euler paths together.

Consider the pre-image example given in Figure 6.2.13 (a), the corresponding expanded De Bruijn Graph is given in Figure 6.2.13 (b). The all-possible Euler algorithm is called at each vertex whose outgoing edges are not all marked. The highest probability Euler circuits for the disjoint De Bruijn Graph are given in Figure 6.2.13 (c). To determine the concatenation location of the two Euler circuits, our algorithm considers all the possible connections between the two disjoint parts of the graph. All the possible connections are illustrated in Figure 6.2.13 (d). These possible connections are evaluated using the same Markov model that we set up before to calculate the Euler circuits. Among all the possible connections, the $P(R-O|R-R)$ value gives the highest probability. Thus a bi-directional edge between

node ‘R’ and node ‘O’, is added to connect the two disjointed parts together. The final De Bruijn Graph, the concatenated Euler circuit and the corresponding graph template are shown in Figure 6.2.13 (e).

6.3 Experimental Results

6.3.1 Data

In our previous work [11], eight different data sets were used to test the ability of the VSMMD to predict biological activities. All these data sets contain real valued QSAR inhibitor data. The eight QSAR data sets are from Sutherland *et al.* [96]. For illustrative purposes, we chose one data set from these eight data sets to demonstrate the effectiveness of the recovery algorithm when applied to our VSMMD.

The data set we chose contains 322 cyclooxygenase-2 (COX2) inhibitors collected by Seibert and colleagues [46] and subsequently utilized in a QSAR study by Chavatte *et al.* [12] with each inhibitor having pIC50 values ranging from 5.5 to 8.9. We chose this data set because training samples in the COX2 data set were presented using diagrams of molecular structures. This allowed us to compare our generated chemical structure templates with the given molecules in the data set. The same inverse-QSAR procedure was applied to the remaining seven data sets, the closest matching molecule in the test set for the generated chemical template was shown in the last section.

In our experiments, the data were separated into the same training and testing

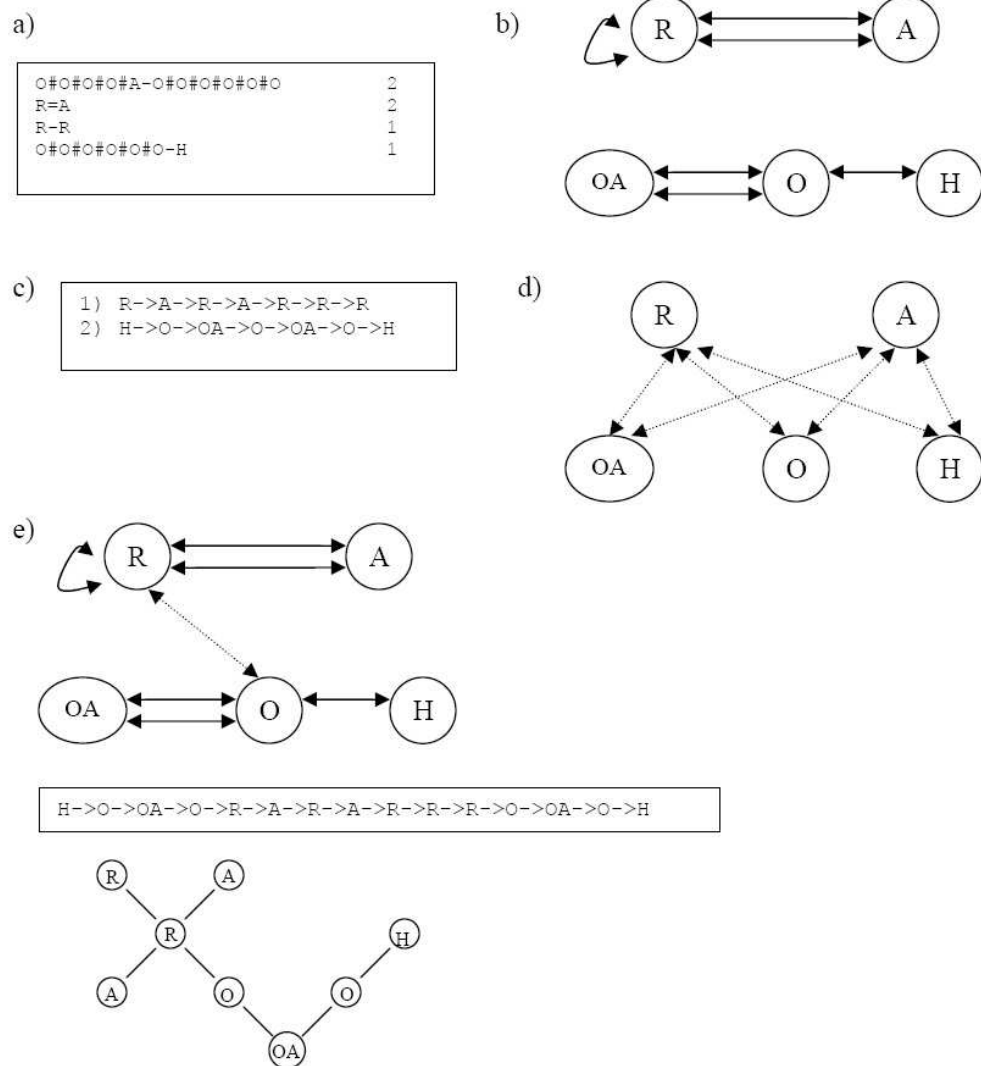


Figure 6.2.13: A case where the pre-image vector did not form a fully connected De Bruijn Graph.

sets as specified by Sutherland *et al.* [96].

6.3.2 Implementation details

For the pre-image algorithm and the feature space algorithm, we used MATLAB to perform the required calculations. For the recovery phase, we implemented the Probabilistic Euler Paths algorithm in Java.

6.3.3 Verification of the Inverse Mapping - Test Result

To verify our proposed inverse approach, we picked one molecule in the COX2 training set randomly as shown in Figure 6.3.14(a). We then generated the VSMMD for each of the compounds in the training set. The corresponding VSMMD for the chosen molecule is shown in Figure 6.3.14(b). Next, we implicitly mapped this VSMMD to the kernel feature space using a Gaussian kernel function as stated in equation (3.5.6). Instead of generating a new point in the feature space, we used the pre-image approximation algorithm to compute the pre-image of this feature space point. The corresponding pre-image is shown in Figure 6.3.14(c). Finally, we applied our VSMMD recovery algorithm to obtain a chemical template. With $t = 1$ and $h = 3$, we generated the Euler circuit with the highest probability and the corresponding chemical structure templates are shown in Figure 6.3.14(d). From this result, we observed that our approach is able to generate a chemical template corresponding to the original chosen molecule.

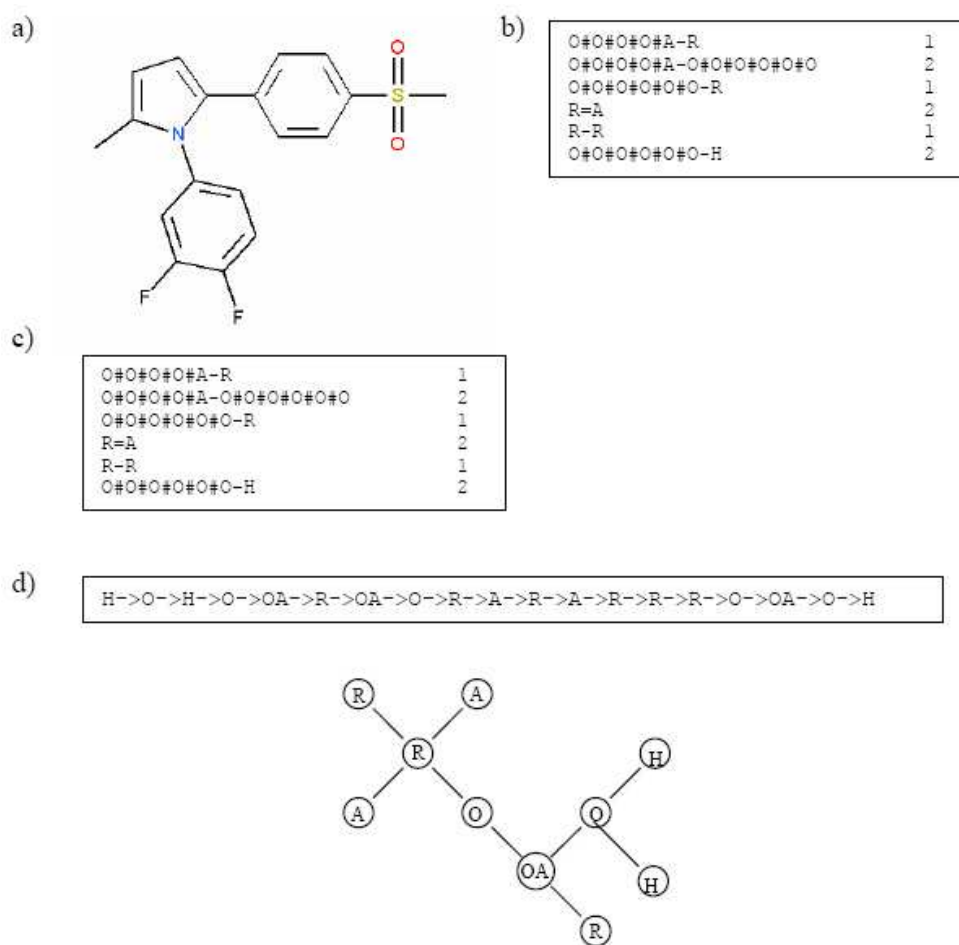


Figure 6.3.14: Verification test result.

6.3.4 Inverse-QSAR Test Results

Recall that our inverse-QSAR approach contains five steps. The first two steps are to perform QSAR analysis. In the first step, we generated the VSMMD for the compounds in the training set. The KACS algorithm is then used to select the most important components of the initial VSMMD. Once these component positions are determined for the descriptor vector, these components are used across all initial descriptors of the training set to get the final descriptors. Then, in the second step, we implicitly mapped the VSMMD to the kernel feature space using an appropriate kernel function for classification or regression analysis. The results of the forward QSAR can be found in our previous chapter.

The third step was to design or to generate a new point in the kernel feature space using a kernel feature space algorithm. To demonstrate our approach, we formed a new point in the feature space by using the ten highest active compounds in the training set. The center of the minimum enclosing and maximum excluding hypersphere was obtained. Figure 6.3.15 shows these ten compounds.

In the fourth step, we mapped the feature space point back into the input space using the pre-image approximation algorithm. In this case, we used the Kwok and Tsang algorithm [63] described in Section 6.2.4 to map the center of the minimum enclosing and maximum excluding hypersphere back into the input space. Figure 6.3.16 illustrates the derived VSMMD.

The last step concerned building the molecular structure template using our VSMMD recovery algorithm described in Section 6.2.5. Since the center of the minimum enclosing and maximum excluding hypersphere was derived from the ten

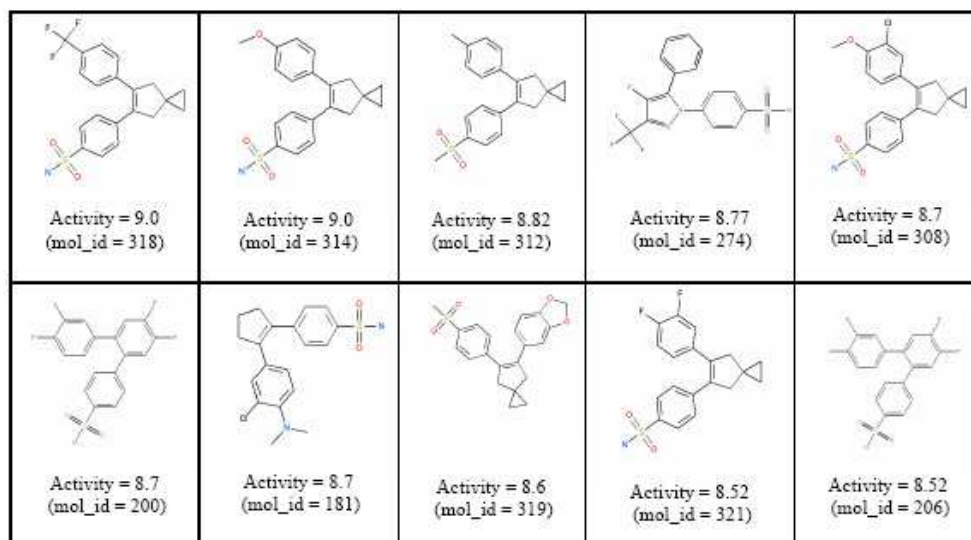


Figure 6.3.15: Ten highest active compounds in the COX-2 training set.

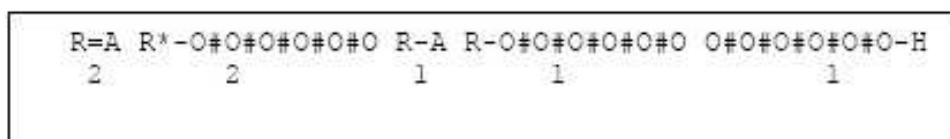


Figure 6.3.16: The pre-image VSMMD of the center of the minimum enclosing and maximum excluding hyperspheres.

highest active compounds in the training set, we assumed that the new derived compounds should look similar to these ten compounds. With this assumption the path probability was calculated. Setting $t = 1$ and $h = 3$, we generated two Euler circuits and the corresponding chemical structure template is shown in Figure 6.3.17.

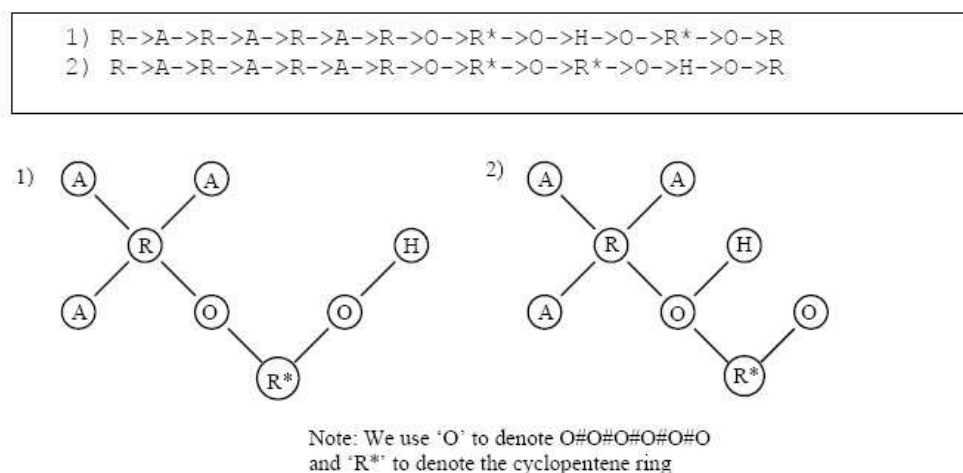


Figure 6.3.17: Two Euler circuits with the highest probability for the pre-image VSMMD in Fig. 6.3.16 and the corresponding chemical structure templates.

6.3.5 Discussion

In order to investigate whether the pre-image VSMMD is reasonable, we performed a more detailed analysis of the COX-2 data set. In the pre-image VSMMD as shown in Figure 16, the cyclopentene ring can be found in one of the descriptors. From medicinal chemistry studies, we know that cyclopentene derivatives are one of the first series of diaryl-substituted cycles that have been well known as COX-2 inhibitors [66, 82]. This empirical evidence demonstrates that our generated

pre-image VSMMD is able to capture important properties of the ten most active molecules.

When we perform a high throughput screening on the test set using the generated chemical structure template, the following molecule was identified as an exact match to the template.

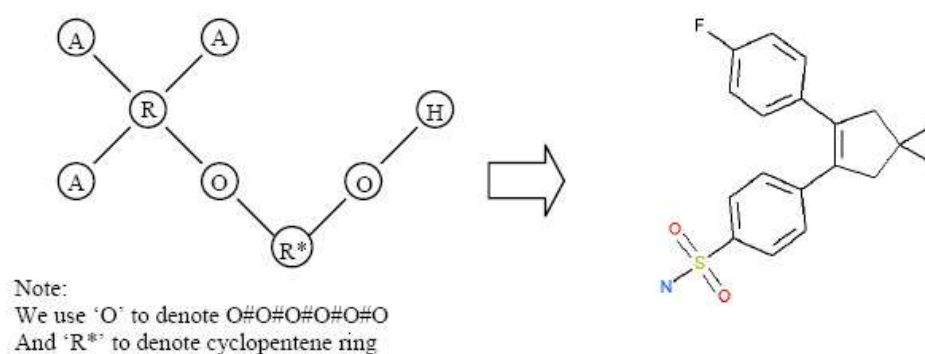


Figure 6.3.18: Matching molecule in the test set.

The molecule in Figure 6.3.18 has pIC₅₀ values of 8.52, and it was one of the highest active molecules in the *testing* set. From this result, we demonstrated that our strategy was able to generate a high affinity molecule using only data in the training set. We were able to claim that the generated molecule was a high affinity molecule because it appeared as such in the testing set. In practice, the success of the algorithm would have to be assessed by using a wet lab procedure to determine the affinity of the generated molecule.

The inverse-QSAR procedure was applied to all eight data sets; the closest matching molecule in the test set for the generated chemical template across 8 data sets is shown in Figure 6.3.19. A quantitative evaluation of the generated molecules

was performed by implicitly mapping the VSMMD of each molecule to the kernel feature space for regression analysis. The regression results are also shown in Figure 6.3.19.

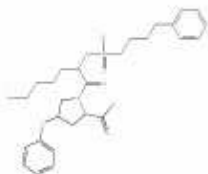
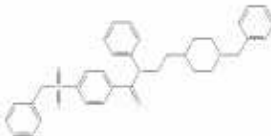
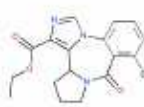
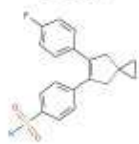
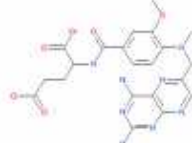
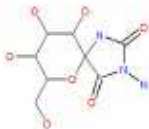
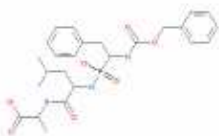
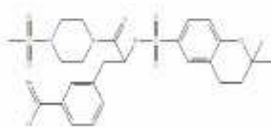
<p>ACE</p>  <p>Given Activity=9.11 Predicted Activity=9.15</p>	<p>Ache</p>  <p>Given Activity=9.22 Predicted Activity=9.46</p>	<p>BZR</p>  <p>Given Activity=8.77 Predicted Activity=8.40</p>
<p>COX2</p>  <p>Given Activity=8.52 Predicted Activity=8.21</p>	<p>DHFR</p>  <p>Given Activity=8.96 Predicted Activity=8.38</p>	<p>GPB</p>  <p>Given Activity=6.8 Predicted Activity=5.24</p>
<p>THER</p>  <p>Given Activity=10.17 Predicted Activity=6.77</p>	<p>THR</p>  <p>Given Activity=8.13 Predicted Activity=7.03</p>	

Figure 6.3.19: The closest matching molecule in the test set for the generated chemical template across 8 data sets.

6.4 Conclusion

In kernel-based learning, the usual assumption is that the data pairs $\{(x_i, y_i)\}_{i=1}^n$, in the training set, come from a source that provides these samples in an independent identically distributed (i.i.d.) fashion according to an unknown probability distribution $P(x, y)$. Furthermore, the test examples are assumed to come from the same distribution [95]. In an ideal situation, the collection of molecular descriptors in the training set follow a probability distribution that is only determined by the interactions between ligands and the binding site. In practise this does not happen. The selection of members in the training set may involve a significant amount of bias due to human involvement in its creation:

- Selection of members of the training set may be restricted by rules that exclude molecules that are not “drug like”.
- Since the training set involves molecules that have been assessed for binding affinity, they had to be synthesized and may be part of a suite of molecules for which the synthesis was not overly complicated.
- Furthermore, the molecular descriptors in the training set may show various types of repetition, (for example, the repeated occurrence of some type of scaffold). This may or may not be intended.

As a consequence of these issues, the learning algorithm will produce a predictor that is taking into account both a biological process and the human activity intrinsic to the formation of the training set. More significantly, there is the demand

that future test molecules come from the same probability distribution. Statistical learning theory will guarantee certain generalization bounds, but only if these demands are met. In effect, the theory tells us that if test samples come from a source, such as a virtual screening library that is not characterized by the same rules of formation as the training set - then all bets are off.

In the constructive approach that has been described in this chapter, it is clear that we are also limited by the information that is intrinsic to a training set. But beyond this, the strategy significantly differs from virtual screening. Instead of trying to find a new molecule in a database that should exhibit the same $P(x, y)$ characteristics, we side step this requirement (which may be difficult to guarantee) and we build a new drug candidate using only the information that is strictly contained in the training set itself.

While molecular fragments have been used in research studies for dealing with quantitative structure-activity relationship problems, we have further evolved this strategy to include a reverse engineering mechanism.

These mechanisms include:

1. The use of a kernel feature space algorithm to design or modify descriptor image points in a feature space.
2. The deployment of a pre-image algorithm to map the descriptor image points in the feature space back to the input space of the descriptors.
3. The design of a probabilistic strategy to convert new descriptors into meaningful chemical graph templates.

As reported in earlier chapters, our modeling has produced very effective algorithms to predict drug-binding affinities and to predict multiple binding modes. We have now extended our modeling approach to the development of algorithms that derive new descriptors and then to facilitate the reverse engineering of such a descriptor. This is a very desirable capability for a molecular descriptor.

Chapter 7

Conclusions

This thesis describes recent developments and applications of kernel based constructive ligand-based drug design. We have reached the following conclusion:

1. The “formation of descriptors” and the “computational modeling” problems stated in Chapter 1 have been given a solution: we demonstrated that VSMMD was suitable for kernel studies in QSAR modeling. Our experiments provided convincing comparative empirical evidence that this kernel method can provide sufficient discrimination to predict various biological activities of a molecule. The prediction ability of VSMMD was compared with other descriptors and other kernel methods.
2. We have given a solution to the “component selection” problem: we developed a new filter based component selection algorithm, Kernel Alignment Component Selection (KACS) based on kernel alignment for QSAR studies. Furthermore, we have proven theoretically that our algorithm works well for

finding the most important components. The reduced component set produced by KACS was compared with the reduced component set produced by SVMRCE. Empirical results showed that our algorithm was able to find the most important descriptor components in different QSAR data sets. The prediction accuracies were substantially increased and compare favorably with those from the earlier studies.

3. The “multiple binding modes” issue has been investigated: In conjunction with a kernel based clustering algorithm, we extended the VSMMD to the prediction of multiple binding modes, a challenging area of research that has been previously studied by means of time consuming docking simulations. The results reported in this study provided strong empirical evidence that our strategy has enough resolving power to distinguish multiple binding modes through the use of a kernel k -means algorithm.
4. The “library dependent” and the “reverse engineering of molecular descriptor” issues have been given a solution: we developed a set of reverse engineering strategies for QSAR modeling based on our VSMMD. The most important aspect of our research is the presentation of strategies that actually generate the structure of a new drug candidate. This is substantially different from methodologies that depend on database screening to get new drug candidates. While our approach can support such an endeavor, it is not our primary goal. In fact, it has not escaped our attention that database screening, done via predictors derived from statistical learning algorithms, is subject to procedural demands that are never discussed in any of the many

papers that advocate the use of such machine learning. We are referring to statistical learning theory that asserts the success of a predictor only when the test sample is drawn from a data source that has the same probability distribution as that characterizing the training set. In most applications of statistical learning to database screening this is rarely, if ever, discussed. The predictor is often applied to molecules in an application data set that have very little relationship to the training data set. In these cases, the predictor is optimistically treated as if it actually incorporates an algorithm that has some firm and direct relationship to the biological context of the problem. In our approach, we have managed to sidestep such concerns. While the training set is still used to generate a new image point in the feature space, the reverse engineering just described allows us to develop a new drug candidate that is independent of issues related to probability distribution constraints placed in the application data set.

In future work, it is of great interest to extend our VSMMD inverse-QSAR algorithm to develop new algorithms that can include the multiple binding modes information for new image point design in the feature space. In our current recovery algorithm, we did not fully utilize the details of bonding information available from the VSMMD. We will develop a new probabilistic model to retain the details of bonding information for a better chemical template structure.

Bibliography

- [1] C. P. Adams and Brantner V. V. Estimating the cost of new drug development: Is it really \$802 million? *Health Affairs*, 25:420–428, 2006.
- [2] T. Akutsu and D. Fukagawa. Inferring a graph from path frequency. *Lecture Notes in Computer Science Volume 3537*, pages 371–382, 2005.
- [3] T. Akutsu and D. Fukagawa. Inferring a chemical structure from a feature vector based on frequency of labeled paths and small fragments. In *In Asia Pacific Bioinformatics Conference 2007*, pages 165–174, 2007.
- [4] C. A. Azencott, A. Ksikes, S. J. Swamidass, J. H. Chen, L. Ralaivola, and P. Baldi. One- to four-dimensional kernels for virtual screening and the prediction of physical, chemical, and biological properties. *Journal of Chemical Information and Modeling*, 47:965–974, 2007.
- [5] G.H. Bakir, A Zien, and K. Tsuda. Learning to find graph pre-images. *Lecture Notes in Computer Science Volume 3175*, pages 253–261, 2004.
- [6] N. Baurin, J. C. Mozziconacci, E. Arnoult, P. Chavatte, C. Marot, and L. Morin-Allory. 2d qsar consensus prediction for high-throughput virtual

- screening. an application to cox-2 inhibition modeling and screening of the nci database. *Journal of Chemical Information and Computer Sciences*, 44:276–285, 2004.
- [7] H. J. Bohm. The development of a simple empirical scoring function to estimate the binding constant for a protein-ligand complex of known three-dimensional structure. *Journal of Computer-Aided Molecular Design*, 8:243–256, Jun 1994.
- [8] M. Bohm, J. Sturzebecher, and G. Klebe. Three-dimensional quantitative structure-activity relationship analyses using comparative molecular field analysis and comparative molecular similarity indices analysis to elucidate selectivity differences of inhibitors binding to trypsin, thrombin, and factor xa. *Journal of Medicinal Chemistry*, 42:458–477, 1999.
- [9] M. C. Broughton and S. F. Queener. Pneumocystis-carinii dihydrofolate-reductase used to screen potential antipneumocystis drugs. *Antimicrobial Agents and Chemotherapy*, 35:1348–1355, 1991.
- [10] F. J. Burkowski. *Structural Bioinformatics: An Algorithmic Approach*. CRC Press, 2008.
- [11] F. J. Burkowski and W. W. L. Wong. Predicting multiple binding modes in qsar studies using a vector space model molecular descriptor in reproducing kernel hilbert space. *International Journal of Computational Biology and Drug Design*, in press, 2008.

- [12] P. Chavatte, S. Yous, C. Marot, N. Baurin, and D. Lesieur. Three-dimensional quantitative structure-activity relationships of cyclo-oxygenase-2 (cox-2) inhibitors: A comparative molecular field analysis. *Journal of Medicinal Chemistry*, 44:3223–3230, 2001.
- [13] S. J. Cho, W. Zheng, and A. Tropsha. Focus-2d: a new approach to the design of targeted combinatorial chemical libraries. *In Pacific Symposium on Biocomputing*, pages 305–316, 1998.
- [14] C. J. Churchwell, M. D. Rintoul, S. Martin, D. P. Visco, A. Kotu, R. S. Larson, L. O. Sillerud, D. C. Brown, and J. L. Faulon. The signature molecular descriptor - 3. inverse-quantitative structure-activity relationship of icam-1 inhibitory peptides. *Journal of Molecular Graphics & Modelling*, 22:263–273, 2004.
- [15] C. Cortes, M. Mohri, and J. Weston. A general regression technique for learning transductions. *In Proceedings of the 22nd international conference on Machine learning*, pages 153–160, Bonn, Germany, 2005. ACM.
- [16] R. D. Cramer, D. E. Patterson, and J. D. Bunce. Comparative molecular-field analysis (comfa) .1. effect of shape on binding of steroids to carrier proteins. *Journal of the American Chemical Society*, 110:5959–5967, 1988.
- [17] N. Cristianini, A. Elisseeff, J. Shawe-Taylor, and J. Kandola. On kernel target alignment. *In Proceedings Neural Information Processing Systems 2001.*, 2001.

- [18] A. Crum-Brown and T. R. Fraser. On the connection between chemical constitution and physiological action. part i. - on the physiological action of the salts of the ammonium bases, derived from strychnia, brucia, thebaia, codeia, morphia, and nicotia. *Transactions of the Royal Society of Edinburgh*, 25:151–203, 1868.
- [19] T. De Bie, Tranchevent L-C., L. M. M. van Oeffelen, and Y. Moreau. Kernel-based data fusion for gene prioritization. *Bioinformatics*, 23:i125–i132, Jul 2007.
- [20] S. A. Depriest, D. Mayer, C. B. Naylor, and G. R. Marshall. 3d-qsar of angiotensin-converting enzyme and thermolysin inhibitors - a comparison of comfa models based on deduced and experimentally determined active-site geometries. *Journal of the American Chemical Society*, 115:5372–5384, 1993.
- [21] W. J. Dunn and D. Rogers. *Genetic partial least squares in QSAR.*, pages 109–130. Academic Press, London, 1996.
- [22] J. L. Faulon, W. Brown, and S. Martin. Reverse engineering chemical structures from molecular descriptors: how many solutions? *Journal of Computer-Aided Molecular Design*, 19:637–650, Sep 2005.
- [23] J. L. Faulon, C. J. Churchwell, and D. P. Visco. The signature molecular descriptor. 2. enumerating molecules from their extended valence sequences. *Journal of Chemical Information and Computer Sciences*, 43:721–734, 2003.
- [24] J. L. Faulon, M. J. Collins, and R. D. Carr. The signature molecular de-

- scriptor. 4. canonizing molecules using extended valence sequences. *Journal of Chemical Information and Computer Sciences*, 44:427–436, 2004.
- [25] J. L. Faulon, D. P. Visco, and R. S. Pophale. The signature molecular descriptor. 1. using extended valence sequences in qsar and qspr studies. *Journal of Chemical Information and Computer Sciences*, 43:707–720, 2003.
- [26] A. M. Ferguson, T. Heritage, P. Jonathon, S. E. Pack, L. Phillips, J. Rogan, and P. J. Snaith. Eva: A new theoretically based molecular descriptor for use in qsar/qspr analysis. *Journal of Computer-Aided Molecular Design*, 11:143–152, 1997.
- [27] S. M. Free and J. W. Wilson. Mathematical contribution to structure-activity studies. *Journal of Medicinal Chemistry*, 7:395–399, 1964.
- [28] H. Frohlich, J. K. Wegner, F.; Sieker, and A. Zell. Optimal assignment kernels for attributed molecular graphs. In *Proceedings of the 22nd international conference on Machine learning*, pages 225–232, Bonn, Germany, 2005. ACM Press.
- [29] H. Gao, C. Williams, P. Labute, and J. Bajorath. Binary quantitative structure-activity relationship (qsar) analysis of estrogen receptor ligands. *Journal of Chemical Information and Modeling*, 39:164–168, Jan 1999.
- [30] P. Gedeck, B. Rohde, and C. Bartels. Qsar - how good is it in practice?: Comparison of descriptor sets on an unbiased cross section of corporate data sets. *Journal of Chemical Information and Modeling*, 46:1924–1936, Sep 2006.

- [31] H. Gohlke and G. Klebe. Drugscore meets comfa: adaptation of fields for molecular comparison (afmoc) or how to tailor knowledge-based pair-potentials to a particular protein. *Journal of Medicinal Chemistry*, 45:4153–4170, 2002.
- [32] A. Golbraikh, P. Bernard, and J. R. Chretien. Validation of protein-based alignment in 3d quantitative structure-activity relationships with comfa models. *European Journal of Medicinal Chemistry*, 35:123–136, 2000.
- [33] R. Guha and P. C. Jurs. Development of linear, ensemble, and nonlinear models for the prediction and interpretation of the biological activity of a set of pdgfr inhibitors. *Journal of Chemical Information and Computer Sciences*, 44:2179–2189, 2004.
- [34] R. Guha and P.C. Jurs. Determining the validity of a qsar model - a classification approach. *Journal of Chemical Information and Modeling*, 45:65–73, 2005.
- [35] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.
- [36] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.
- [37] W. Haefely, E. Kyburz, M. Gerecke, and H. Mohler. Recent advances in the molecular pharmacology of benzodiazepine receptors and in the structure-

- activity relationships of their agonists and antagonists. *Advanced Drug Delivery Reviews*, 14:165–322, 1985.
- [38] L. H. Hall, R. S. Dailey, and L. B. Kier. Design of molecules from quantitative structure-activity relationship models .3. role of higher-order path counts - path-3. *Journal of Chemical Information and Computer Sciences*, 33:598–603, 1993.
- [39] L. H. Hall, L. B. Kier, and J. W. Frazer. Design of molecules from quantitative structure activity relationship models .2. derivation and proof of information-transfer relating equations. *Journal of Chemical Information and Computer Sciences*, 33:148–152, 1993.
- [40] C. Hansch, P. P. Maloney, T. Fujita, and R. M. Muir. Correlation of biological activity of phenoxyacetic acids with hammett substituent constants and partition coefficients. *Nature*, 194:178–180, 1962.
- [41] D. R. J. Hauser, T. Scior, D. M. Domeyer, B. Kammerer, and S. A. Laufer. Synthesis, biological testing, and binding mode prediction of 6,9-diaryl-purin-8-ones as p38 map kinase inhibitors. *Journal of Medicinal Chemistry*, 50:2060–2066, May 2007.
- [42] M. J. R. Healy. The use of r^2 as a measure of goodness of fit. *Journal of the Royal Statistical Society, A* 147:608–609, 1984.
- [43] J. Hert, P. Willett, D. J. Wilton, P. Acklin, K. Azzaoui, E. Jacoby, and A. Schuffenhauer. Comparison of topological descriptors for similarity-based

- virtual screening using multiple bioactive reference structures. *Organic & Biomolecular Chemistry*, 2:3256–3266, 2004.
- [44] T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.
- [45] T.J. Hou, K. Xia, W. Zhang, and X.J. Xu. Adme evaluation in drug discovery. 4. prediction of aqueous solubility based on atom contribution approach. *Journal of Chemical Information and Computer Sciences*, 44:266–275, 2004.
- [46] H. C. Huang, T. S. Chamberlain, K. Seibert, C. M. Koboldt, and P. C. Isakson. Diaryl indenenes and benzofurans - novel classes of potent and selective cyclooxygenase-2 inhibitors. *Bioorganic & Medicinal Chemistry Letters*, 5:2377–2380, 1995.
- [47] P. Itskowitz and A. Tropsha. kappa nearest neighbors qsar modeling as a variational problem: Theory and applications. *Journal of Chemical Information and Modeling*, 45:777–785, 2005.
- [48] T. Joachims. chapter Making large-Scale SVM Learning Practical. MIT-Press, 1999.
- [49] G. Jones, P. Willett, and R. C. Glen. A genetic algorithm for flexible molecular overlay and pharmacophore elucidation. *Journal of Computer-Aided Molecular Design*, 9:532–549, Dec 1995.
- [50] G. Jones, P. Willett, and R. C. Glen. Molecular recognition of receptor

- sites using a genetic algorithm with a description of desolvation. *Journal of Molecular Biology*, 245:43–53, Jan 1995.
- [51] G. Jones, P. Willett, R. C. Glen, A. R. Leach, and R. Taylor. Development and validation of a genetic algorithm for flexible docking. *Journal of Molecular Biology*, 267:727–748, Apr 1997.
- [52] L. B. Kier and L. H. Hall. *Molecular Connectivity in Structure-Activity Analysis*. Research Studies Press, Lectchworth, Hertfordshire, England, 1986.
- [53] L. B. Kier, L. H. Hall, and J. W. Frazer. Design of molecules from quantitative structure activity relationship models .1. information-transfer between path and vertex degree counts. *Journal of Chemical Information and Computer Sciences*, 33:143–147, 1993.
- [54] K. H. Kim. Outliers in sar and qsar: 2. is a flexible binding site a possible source of outliers? *Journal of Computer-Aided Molecular Design*, 21:421–435, 2007.
- [55] K. H. Kim. Outliers in sar and qsar: Is unusual binding mode a possible source of outliers? *Journal of Computer-Aided Molecular Design*, 21:63–86, 2007.
- [56] R. D. King, J. D. Hirst, and M. J. Sternberg. New apporaches to qsar: Neural networks and machine learning. *Perspectives in Drug Discovery and Design*, 1:279–290, 1993.
- [57] P. Kirkpatrick and C. Ellis. Chemical space. *Nature*, 432:823, 2004.

- [58] S. Kirkpatrick, J. Gekatt, and M. P. Vecchi. *Readings in computer vision issues, problems, principles, and paradigms*. Morgan Kaufmann, 1987.
- [59] G. Klebe and U. Abraham. Comparative molecular similarity index analysis (comsia) to study hydrogen-bonding properties and to score combinatorial libraries. *Journal of Computer-Aided Molecular Design*, 13:1–10, 1999.
- [60] G. Klebe, U. Abraham, and T. Mietzner. Molecular similarity indexes in a comparative-analysis (comsia) of drug molecules to correlate and predict their biological-activity. *Journal of Medicinal Chemistry*, 37:4130–4146, 1994.
- [61] B. Kramer, M. Rarey, and T. Lengauer. Evaluation of the flexx incremental construction algorithm for protein-ligand docking. *Proteins*, 37:228–241, Nov 1999.
- [62] V. Kvasnicka and J. Pospichal. Simulated annealing construction of molecular graphs with required properties. *Journal of Chemical Information and Computer Sciences*, 36:516–526, 1996.
- [63] J. T. Y. Kwok and I. W. H. Tsang. The pre-image problem in kernel methods. *IEEE Transactions on Neural Networks.*, 15:1517–1525, 2004.
- [64] C. Lai, M. J. T. Reinders, and L. Wessels. Random subspace method for multivariate feature selection. *Pattern Recognition Letters*, 27(10):1067–1076, 2006.
- [65] A. R. Leach and V. J. Gillet. *An introduction to chemoinformatics*. Kluwer Academic Publishers, Dordrecht, 2003.

- [66] X. Leval, J. Delarge, F. Somers, P. Tullio, Y. Henrotin, B. Pirotte, and J. Dogne. Recent advances in inducible cyclooxygenase (cox-2) inhibition. *Current Medicinal Chemistry*, 7:1041–1062, 2000.
- [67] Y. Liu. A comparative study on feature selection methods for drug discovery. *Journal of Chemical Information and Computer Sciences*, 44:1823–1828, 2004.
- [68] Y. Liu and Y. F. Zheng. Minimum enclosing and maximum excluding machine for pattern description and discrimination. In *18th International Conference on Pattern Recognition*, volume 3, pages 129–132, 2006.
- [69] D. R. Lowis. Hqsar: A new, highly predictive qsar technique. Technical report, 1997.
- [70] D. J. Maddalena and G. A. R. Johnston. Prediction of receptor properties and binding-affinity of ligands to benzodiazepine/gaba(a) receptors using artificial neural networks. *Journal of Medicinal Chemistry*, 38:715–724, 1995.
- [71] P. Mahe, L. Ralaivola, V. Stoven, and J. P. Vert. The pharmacophore kernel for virtual screening with support vector machines. *Journal of Chemical Information and Modeling*, 46:2003–2014, Sep 2006.
- [72] P. Mahe, N. Ueda, T. Akutsu, J. L. Perret, and J. P. Vert. Graph kernels for molecular structure-activity relationship analysis with support vector machines. *Journal of Chemical Information and Modeling*, 45:939–951, Jul 2005.

- [73] B. Mak, J. T. Kwok, and S. Ho. Kernel eigenvoice speaker adaptation kernel eigenvoice speaker adaptation. *IEEE Transactions on Speech and Audio Processing*, 13:984–992, 2005.
- [74] B. K-W Mak, R. W-H Hsiao, S. K-L Ho, and J. T. Kwok. Embedded kernel eigenvoice speaker adaptation and its implication to reference speaker weighting embedded kernel eigenvoice speaker adaptation and its implication to reference speaker weighting. *IEEE Transactions on Audio, Speech, and Language Processing.*, 14:1267–1280, 2006.
- [75] C. Merkwirth, H. A. Mauser, T. Schulz-Gasch, O. Roche, M. Stahl, and T. Lengauer. Ensemble methods for classification in cheminformatics. *Journal of Chemical Information and Computer Sciences*, 44:1971–1978, 2004.
- [76] S. Mika, B. Schölkopf, J.A. Smola, K.-R. Müller, M. Scholz, and G. Rätsch. Kernel pca and de-noising in feature spaces. In *Proceedings of the 1998 conference on Advances in neural information processing systems II*, pages 536–542. MIT Press, 1999.
- [77] C. H. Nguyen and T. B. Ho. Kernel matrix evaluation. In *International Joint Conference on Artificial Intelligence*, 2007.
- [78] K. L. Peterson. *Artificial neural networks and their use in chemistry*. WILEY-VCH, INC, NEW YORK, 2000.
- [79] P. A. Pevzner, H. Tang, and M. S. Waterman. An eulerian path approach to dna fragment assembly. *Proceedings of the National Academy of Sciences*, 98:9748–9753, Aug 2001.

- [80] L. Ralaivola, S. J. Swamidass, H. Saigo, and P. Baldi. Graph kernels for chemical informatics. *Neural Networks*, 18:1093–1110, Oct 2005.
- [81] M. Rarey, B. Kramer, T. Lengauer, and G. Klebe. A fast flexible docking method using an incremental construction algorithm. *Journal of Molecular Biology*, 261:470–489, Aug 1996.
- [82] D. B. Reitz, J. J. Li, M. B. Norton, E. J. Reinhard, J. T. Collins, G. D. Anderson, S. A. Gregory, C. M. Koboldt, and W. E. Perkins. Selective cyclooxygenase inhibitors: Novel 1,2-diarylcyclopentenes are potent and orally active cox-2 inhibitors. *Journal of Medicinal Chemistry*, 37:3878–3881, Nov 1994.
- [83] D. Rogers, R. D. Brown, and M. Hahn. Using extended-connectivity fingerprints with laplacian-modified bayesian analysis in high-throughput screening follow-up. *Journal of Biomolecular Screening*, 10:682–686, 2005.
- [84] D. Rogers and A. J. Hopfinger. Application of genetic function approximation to quantitative structure-activity-relationships and quantitative structure-property relationships. *Journal of Chemical Information and Computer Sciences*, 34:854–866, 1994.
- [85] H. Sato, L. M. Shewchuk, and J. Tang. Prediction of multiple binding modes of the cdk2 inhibitors, anilinopyrazoles, using the automated docking programs gold, flexx, and ligandfit: An evaluation of performance. *Journal of Chemical Information and Modeling*, 46:2552–2562, Nov 2006.

- [86] B. Schölkopf, P. Knirsch, A.J. Smola, and C.J.C Burges. Fast approximation of support vector kernel expansions, and an interpretation of clustering as approximation in feature spaces. In *DAGM-Symposium*, pages 125–132. Springer-Verlag, 1998.
- [87] B. Schölkopf, K. Tsuda, and J. P. Vert. *Kernel methods in computational biology*. MIT Press, Cambridge, Mass, 2004.
- [88] C. D. Selassie. *History of Quantitative Structure-Activity Relationships*, volume 6. John Wiley & Sons, 2003.
- [89] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge University Press, Cambridge, UK, 2004.
- [90] R. P. Sheridan and S. K. Kearsley. Using a genetic algorithm to suggest combinatorial libraries. *Journal of Chemical Information and Computer Sciences*, 35:310–320, 1995.
- [91] J. Smola and B. Schölkopf. A tutorial on support vector regression. Technical report, 1998.
- [92] C. Steinbeck, Y. Han, S. Kuhn, O. Horlacher, E. Luttmann, and E. Willighagen. The chemistry development kit (cdk): an open-source java library for chemo- and bioinformatics. *Journal of Chemical Information and Computer Sciences*, 43:493–500, Mar 2003.
- [93] C. Steinbeck, C. Hoppe, S. Kuhn, M. Floris, R. Guha, and E. L. Willighagen. Recent developments of the chemistry development kit (cdk) - an open-source

- java library for chemo- and bioinformatics. *Current Pharmaceutical Design*, 12:2111–2120, 2006.
- [94] H. Sugimoto, Y. Tsuchiya, H. Sugumi, K. Higurashi, N. Karibe, Y. Iimura, A. Sasaki, S. Araki, Y. Yamanishi, and K. Yamatsu. Synthesis and structure-activity relationships of acetylcholinesterase inhibitors: 1-benzyl-4-(2-phthalimidoethyl)piperidine, and related derivatives. *Journal of Medicinal Chemistry*, 35:4542–4548, Nov 1992.
- [95] M. Sugiyama and K. Müller. Input-dependent estimation of generalization error under covariate shift. *Statistics and Decisions*, 23:249–279, 2005.
- [96] J. J. Sutherland, L. A. O’Brien, and D. F. Weaver. A comparison of methods for modeling quantitative structure-activity relationships. *Journal of Medicinal Chemistry*, 47:5541–5554, Oct 2004.
- [97] J. J. Sutherland and D. F. Weaver. Three-dimensional quantitative structure-activity and structure-selectivity relationships of dihydrofolate reductase inhibitors. *Journal of Computer-Aided Molecular Design*, 18:309–331, 2004.
- [98] J. M. Sutter, S. L. Dixon, and P. C. Jurs. Automated descriptor selection for quantitative structure-activity relationships using generalized simulated annealing. *Journal of Chemical Information and Computer Sciences*, 35:77–84, 1995.
- [99] V. Svetnik, T. Wang, C. Tong, A. Liaw, R.P. Sheridan, and Q.H. Song. Boosting: An ensemble learning tool for compound classification and qsar modeling. *Journal of Chemical Information and Modeling*, 45:786–799, 2005.

- [100] S. J. Swamidass, J. Chen, J. Bruand, P. Phung, L. Ralaivola, and P. Baldi. Kernels for small molecules and the prediction of mutagenicity, toxicity and anti-cancer activity. *Bioinformatics*, 21 supplement 1:i359–i368, 2005.
- [101] J. Tang, L. M. Shewchuk, . Sato, M. Hasegawa, Y. Washio, and N. Nishigaki. Anilinopyrazole as selective cdk2 inhibitors: design, synthesis, biological evaluation, and x-ray crystallographic analysis. *Bioorganic & Medicinal Chemistry Letters*, 13:2985–2988, Sep 2003.
- [102] R. Todeschini and V. Consonni. *Handbook of molecular descriptors*. Wiley-VCH, 2000.
- [103] S. Trohalaki, R. Pachter, K.T. Geiss, and J.M. Frazier. Halogenated aliphatic toxicity qsars employing metabolite descriptors. *Journal of Chemical Information and Computer Sciences*, 44:1186–1192, 2004.
- [104] V. N. Vapnik. *The nature of statistical learning theory*. Springer, New York, 1995.
- [105] C. M. Venkatachalam, X. Jiang, T. Oldfield, and M. Waldman. Ligandfit: a novel method for the shape-directed rapid docking of ligands to protein active sites. *Journal of Molecular Graphics and Modelling*, 21:289–307, Jan 2003.
- [106] V. Venkatasubramanian, K. Chan, and J. M. Caruthers. Evolutionary design of molecules with desired properties using the genetic algorithm. *Journal of Chemical Information and Computer Sciences*, 35:188–195, 1995.

- [107] V. Venkatraman, A. R. Dalby, and Z. R. Yang. Evaluation of mutual information and genetic programming for feature selection in qsar. *Journal of Chemical Information and Computer Sciences*, 44:1686–1692, 2004.
- [108] J. Verdu-andres and D.L. Massart. Comparison of prediction- and correlation-based methods to select the best subset of principal components for principal component regression and detect outlying objects. *Applied Spectroscopy*, 52:1425–1434, 1998.
- [109] L. Wang. Feature selection with kernel class separability. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(9):1534–1546, 2008.
- [110] H. Wiener. Structural determination of paraffin boiling points. *Journal of the American Chemical Society*, 69:17–20, 1947.
- [111] R. J. Wilson. *Introduction to Graph Theory*, volume 4. Addison Wesley, 1996.
- [112] S. Wold, M. Sjostrom, and L. Eriksson. Pls-regression: a basic tool of chemometrics. *Chemometrics and Intelligent Laboratory Systems*, 58:109–130, 2001.
- [113] Y. Xue, Z. R. Li, C. W. Yap, L. Z. Sun, X. Chen, and Y. Z. Chen. Effect of molecular descriptor feature selection in support vector machine classification of pharmacokinetic and toxicological properties of chemical agents. *Journal of Chemical Information and Computer Sciences*, 44(5):1630–1638, 2004.
- [114] Y. Xue, C. W. Yap, L. Z. Sun, Z. W. Cao, J. F. Wang, and Y. Z. Chen. Prediction of p-glycoprotein substrates by a support vector machine approach. *Journal of Chemical Information and Computer Sciences*, 44:1497–1505, 2004.

- [115] X.J. Yao, A. Panaye, J.P. Doucet, R.S. Zhang, H.F. Chen, M.C. Liu, Z.D. Hu, and B.T. Fan. Comparative study of qsar/qspr correlations using support vector machines, radial basis function neural networks, and multiple linear regression. *Journal of Chemical Information and Computer Sciences*, 44:1257–1266, 2004.
- [116] H. Yu, J. Yang, W. Wang, and J. Hah. Discovering compact and highly discriminative features or feature combinations of drug activities using support vector machines. In *IEEE Bioinformatics Conference*, pages 220–228, 2003.