L2 Norm Regularized Feature Kernel Regression For Graph Data

Hongliang Fei, Jun Huan
Department of Electrical Engineering and Computer Science
University of Kansas
Lawrence, KS 66047-7621, USA
{hfei, jhuan}@ittc.ku.edu

ABSTRACT

Features in many real world applications such as Cheminformatics, Bioinformatics and Information Retrieval have complex internal structure. For example, frequent patterns mined from graph data are graphs. Such graph features have different number of nodes and edges and usually overlap with each other. In conventional data mining and machine learning applications, the internal structure of features are usually ignored.

In this paper we consider a supervised learning problem where the features of the data set have intrinsic complexity, and we further assume that the feature intrinsic complexity may be measured by a kernel function. We hypothesize that by regularizing model parameters using the information of feature complexity, we can construct simple yet high quality model that captures the intrinsic structure of the data. Towards the end of testing this hypothesis, we focus on a regression task and have designed an algorithm that incorporate the feature complexity in the learning process, using a kernel matrix weighted L_2 norm for regularization, to obtain improved regression performance over conventional learning methods that does not consider the additional information of the feature. We have tested our algorithm using 5 different real-world data sets and have demonstrate the effectiveness of our method.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications-Data Mining

General Terms

Algorithms, Experimentation

Keywords

Data Mining, Regression, Regularization

1. INTRODUCTION

CIKM'09, November 2–6, 2009, Hong Kong, China.

Data with complex features are becoming abundant in many application domains such as Cheminformatics, Bioinformatics, Information Retrieval and among others. For example in Cheminformatics researchers usually model chemical structures as a graphs and extract frequent subgraphs as features [2]. Such subgraph features have different number of nodes, different number of edges, and usually overlap with each other. In Bioinformatics and Information Retrieval, given a set of protein sequences or documents, if we use frequent subsequences of amino acids or words, each feature has its own complexity such as subsequence length [9].

In this paper we focus on learning from graph data, due to the wide range of applications where graphs are utilized as a modeling tool. In particular, we focus on the *subgraph based graph learning problem* where features are (frequent) subgraphs mined from the training graph data sets and we present each graph as a feature vector. Once we have transformed a graph to a feature vector, mining and learning from graphs is similar to any other type of vectorial data. Typically there are two types of learning tasks: unsupervised and supervised and we focus on the supervised graph learning problem in this paper.

Subgraph based graph learning problem has attracted research interest in the data mining community. For example, Tsuda [20] proposed a graph regression framework in which he employed L_1 norm regularization algorithm Lasso [19] to graph data and conducted forward stepwise feature selection for regression. Saigo [15] applied partial least square regression to graph data and implemented feature selection during the pattern mining process. Yan *et al.*[21] performed a comprehensive study on mining significant graph patterns for graph classification. Fei & Huan [4] studied structure consistency relationship of subgraph features, developed a subgraph feature selection method and employed Support Vector Machine to perform graph classification.

However, none of the existing methods considers the internal structure of subgraph features and utilizes their complexity to construct accurate models. Our current working hypothesis is that complexity of subgraphs should be incorporated into model construction in order to build simple yet high quality model predicting labels of graph data. To illustrate that point, we show an example in Figure 1. There are three graphs G_1, G_2 and G_3 in Figure 1. F_1 and F_2 are frequent subgraph features if we use the minimal support threshold $min\_sup = \frac{2}{3}$. Using F_1 and F_2 as features, the object-feature matrix X, where each row is an graph and each column is a feature, is represented as:

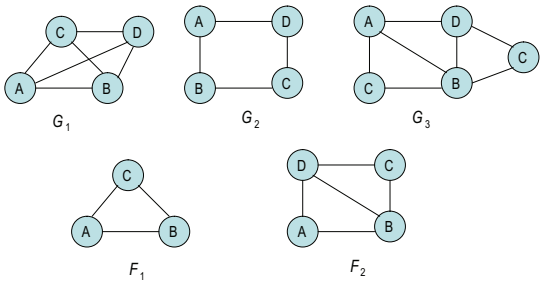

Figure 1: Three graphs G_1, G_2, G_3 and two subgraph features F_1, F_2

$$X = \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}$$

In this example, we can see that no matter what labels the three graphs have, the two features F_1 and F_2 have exactly the same correlation with labels. Lasso [19] based regression method will randomly pick up a feature and assign coefficient to it because F_1 and F_2 have the same correlation with labels. Ridge [7] will assign equal weights to F_1 and F_2 because Ridge shrinks more on the direction where the singular value of X is smaller. In this case, the singular values of X are equal, hence Ridge will shrink the two features with the same ratio. However, intuitively, we would like to assign more weight to F_1 than F_2 since F_1 much simpler than F_2. The reason why current regularization methods fail to do so is that they treat the feature as an atomic element and neglect the internal complexity of features.

In this paper towards the end of incorporating feature complexity in the model selection process, we investigate two approaches. First, we study the approach for measuring the complexity of subgraph features. Second, we investigate the algorithms that select simpler features to construct supervised learning models. Specifically in our study, we utilize graph kernel functions to measure the complexity of graph features. To solve the supervised graph learning problem, we propose a L_2 norm based regularization method for regression using subgraph features. Though we evaluate our algorithms primarily using data sets from chemical structure activity relationship study, these algorithms in principle should be applicable for any types of graph data.

Specifically our contributions in this paper are:

- We proposed a novel regularization framework where we utilize feature complexity to guild model selection process for supervised graph learning problems.
- We extended traditional Ridge regression to L2 norm regularized feature kernel regression, in which we not only realized stability of coefficients but also penalize more on the features with high complexity.
- We performed a comprehensive experimental evaluation. The results demonstrated that L_2 norm based feature kernel regularized regression is an effective method, evaluated these methods on 5 real world data sets and compared the performance of the method to the other State-of-the-art methods including Lasso and Ridge.

The rest of the paper is organized as following. In Section 1.1, we discuss related work. In Section 2 we present background information and in Section 3 we show our detailed methodology. In section 4 we present the experimental study of our algorithm, followed by a conclusion and a discussion of the future work.

1.1 Related Work

Regularization based linear regression is not a new topic. Hoerl and Kennard [7] developed ridge regression based on L2 norm regularization. Tibshirani[19] proposed the Lasso method which is a shrinkage and selection method for linear regression. Lasso minimizes the sum of squared errors, with an upper bound on the L_1 norm of the regression coefficients. Efron & Hastie [3] designed a novel algorithm, Least Angel Regression (LARS), to solve the optimization problem in Lasso efficiently. Zou & Hastie [24] developed a regression framework based upon penalizing on L_1 and L_2 norm of coefficients simultaneously. Recently, a new direction is in feature selection in regularized learning is to explore the relationship of features. Yuan & Lin [22] studied the case when features have a natural group structure and designed a technique to select grouped features called group Lasso. Zhao & Yu [23] integrated a hierarchical relation on features to regression and proposed a method called iCAP. Quanz & Huan [13] assumed a general undirected graph relationships of features and employed the feature graph Laplacian in logistic regression for graph classification.

Though regularized regression has been studied for a long time, none of the existing method considers the special characteristics of graph data and subgraph features and hence may not provide the optimal results for graph regression. We develop a graph regression method incorporating feature information and our experiment study shows that our method works very well on several real-world data sets compared with other regression models.

2. BACKGROUND

Here we introduce basic notations for graph, frequent subgraph mining, graph kernel functions and regularized linear regression.

2.1 Graph Theory

A *labeled graph* G is described by a finite set of nodes V and a finite set of edges $E \subset V \times V$. In most applications, a graph is labeled, where labels are drawn from a label set σ. A labeling function $\lambda : V \cup E \to \Sigma$ assigns labels to nodes and edges. In *node-labeled graphs*, labels are assigned to nodes only and in *edge-labeled graphs*, labels are assigned to edges only. In *fully-labeled graphs*, labels are assigned to nodes and edges. We may use a special symbol to represent missing labels. If we do that, node-labeled graphs, edge-labeled graphs, and graphs without labels are special cases of fully-labeled graphs. Without loss of generality, we handle fully-labeled graphs only in this paper. We do not assume any structure of label set Σ now; it may be a field, a vector space, or simply a set.

Following convention, we denote a graph as a quadruple $G = (V, E, \Sigma, \lambda)$ where V, E, Σ, λ are explained before. A graph $G = (V, E, \Sigma, \lambda)$ is a *subgraph* of another graph $G' = (V', E', \Sigma', \lambda')$, denoted by $G \subseteq G'$, if there exists a 1-1 mapping $f : V \to V'$ such that

- for all $v \in V, \lambda(v) = \lambda'(f(v))$
- for all $(u,v) \in E, (f(u), f(v)) \in E'$
- for all $(u,v) \in E, \lambda(u,v) = \lambda'(f(u), f(v))$

In other words, a graph is a subgraph of another graph if there exits a 1-1 node mapping such that preserve the node labels, edge relations, and edge labels.

The 1-1 mapping f is a *subgraph isomorphism* from G to G' and the range of the mapping f, $f(V)$, is an *embedding* of G in G'.

2.2 Frequent Subgraph Mining

Given a graph database GD, the support of a subgraph G, denoted by sup_G, is the fraction of the graphs in GD of which G is a subgraph, or:

$$sup_G = \frac{|G^{'} \in GD | G \subseteq G^{'}|}{|GD|}$$

Given a user specified minimum support threshold *min_sup* and graph database GD, a *frequent subgraph* is a subgraph whose support is at least *min_sup* (i.e. $sup_G \geq min\_sup$) and the *frequent subgraph mining problem* is to find all frequent subgraphs in GD.

In this paper, we use frequent subgraph mining to extract features in a set of graphs. Each mined subgraph is a feature. Each graph is transformed to a feature vector indexed by the extracted features with values indicate the presence or absence of the feature as did in [8]. We use binary feature vector as contrast to occurrence feature vector (where the value of a feature indicates the number of occurrences of the feature in an object) due to its simplicity. Empirical study shows that there is negligible difference between the two representations in graph classification.

2.3 Graph Kernel Function

Kernel functions are powerful computational tools to analyze large volumes of graph data [6]. The advantage of kernel functions is due to their capability to map a set of data to a high dimensional Hilbert space without explicitly computing the coordinates of the structure. This is done through a special function K. Specifically a binary function $K : X \times X \rightarrow \mathbb{R}$ is a *positive semi-definite* function if

$$\sum_{i,j=1}^{n} c_i c_j K(x_i, x_j) \geq 0 \tag{1}$$

for any $m \in \mathbb{N}$, any selection of samples $x_i \in X$ $(i = [1,n])$, and any set of coefficients $c_i \in \mathbb{R}$ $(i = [1,n])$. In addition, a binary function is symmetric if $K(x,y) = K(y,x)$ for all $x, y \in X$. A symmetric, positive semi-definite function ensures the existence of a Hilbert space $\mathcal{H}$ and a map $\Phi : X \rightarrow \mathcal{H}$ such that

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle \tag{2}$$

for all $x, x' \in X$. $\langle x, y \rangle$ denotes an inner product between two objects x and y. The result is known as the Mercer's theorem and a symmetric, positive semi-definite function is also known as a Mercer kernel function [16], or *kernel* function for simplicity.

Several graph kernel functions have been studied. Recent progresses of graph kernel functions could be roughly divided into two categories. The first group of kernel functions consider the full adjacency matrix of graphs and hence measure the global similarity of two graphs. These include product graph kernels [5], random walk based kernels [10], and kernels based on shortest paths between pair of nodes [11]. The second group of kernel functions try to capture the local similarity of two graphs by counting the shared subcomponents of graphs. These include the subtree kernels [14], cyclic kernels [18], spectrum kernel [2], and recently frequent subgraph kernels [17]. In this paper, we focus on graph random walk based kernels, where we use subgraph as features and kernels are defined on pairwise subgraph features.

2.4 Regularized Linear Regression

In statistics and machine learning, regularization is a powerful tool to prevent overfitting. Regularization usually introduces additional constraints on the model as a form of a penalty for complexity. Consider a typical linear regression problem:

$$Y = X\beta + \epsilon \tag{3}$$

where Y is a $n \times 1$ vector, X is a $n \times p$ matrix and β is a coefficient vector with the size of $p \times 1$ and ϵ is gaussian noise with mean 0 and standard deviation δ. Ordinary Least Square (OLS) minimizes the sum of squared errors $\|Y - X\beta\|^2$, where $\|.\|$ is L_2 norm. But even though the solution of OLS is unbiased estimator,it is well known that OLS often does poorly in both prediction and interpretation and the model is very unstable.

Regularized linear regression not only minimizes the sum of squared errors, but bounds on the norm of regression coefficients. For example, ridge regression [7] minimizes the residual sum of squares subject to a bound on the L2-norm of the coefficients. As a continuous shrinkage method, ridge regression achieves its better prediction performance through a bias variance trade-off. Lasso [19] is a penalized least squares method imposing an L1-penalty on the regression coefficients and does both continuous shrinkage and automatic variable selection simultaneously.

3. METHODOLOGY

Our L_2 Norm Regularized Feature Kernel Regression method has two steps: (1) feature extraction and (2) regression. In the feature extraction step, we mine frequent subgraphs in the training samples as features. We then build a regression model, as discussed below, to predict the numerical labels of testing attribute graphs.

3.1 Notation

In this paper, we use capital letters, such as G, for a single graph and upper case calligraphic letters, such as $\mathcal{G} = G_1, G_2, \ldots, G_n$, for a set of n graphs. We assume each graph $G_i \in \mathcal{G}$ has an associated class label c_i from a label set C. We use $\mathcal{F} = F_1, F_2, \ldots, F_n$ for a set of n features.

3.2 Feature kernel regression framework

In this work we consider combining feature complexity into regression. Also we assume feature intrinsic complexity may be measured by a kernel function. Towards that goal, we build feature kernel first. An advantage of subgraph features is that the kernel function defined on graphs

can also be applied to subgraph features. In our article, we apply Marginalized kernel [10] to the L_2 penalty function. Marginalized kernel for graphs is described as:

$$K_m(G, G') = \sum_h \sum_{h'} K_z(z, z') p(h|G) p(h'|G') \quad (4)$$

where G, G' are two graphs, $z = [G, h]$, and h, h' are hidden variables defined as a sequence of vertex indices, which is generated by random walks on the graph. $K_z(z, z')$ is the kernel between the sequences of vertex and edge labels traversed in the random walk.

To avoid singularity, we add an dirac kernel matrix K_d after marginal feature kernel matrix. That is, $K = K_m + K_d$. This will not jeopardize the kernel feature regression setting because the sum of two kernel matrices is still a valid kernel matrix. The dirac feature kernel matrix is defined as:

$$K_d(F_i, F_j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

To introduce feature complexity to regularization, we construct a weighted complete graph where each node represents a feature F_i and the weight of each edge E_ij equals to a $K(i, j)$ in kernel matrix. With the feature graph, we build graph Laplacian matrix $L = D - K$ to capture complexity of features, where K is the feature kernel matrix and D is a diagonal matrix defined as:

$$d_{ij} = \begin{cases} \sum_{k=1}^{n} K_{ik} & \text{if } i == j \\ 0 & \text{otherwise} \end{cases}$$

We do not normalize the graph Laplacian since we not only consider pairwise features's complexity but each feature's own internal complexity. Suppose that the data set contains n observations and p predictors, with response vector $Y = (y_1, \ldots, y_n)^T$ and the data matrix $X = (\vec{x}_1, \ldots, \vec{x}_p)$, where $\vec{x}_j = (x_{1j}, \ldots, x_{nj})^T$, $j = 1, \ldots, p$. We also assume that the predictors are standardized and the response is centered so that for all j, $\sum_{j=1}^{n} x_{ij} = 0$, $\sum_{j=1}^{n} x_{ij}^2 = 1$ and $\sum_{i=1}^{n} y_i = 0$. The regression function is linear with the following form:

$$Y = X\beta \quad (5)$$

where β is a $n \times 1$ coefficient vector. The Lagrange form of the objective function is:

$$L(\lambda, \beta) = (Y - X\beta)^T (Y - X\beta) + \lambda \beta^T L \beta \quad (6)$$

where $\lambda > 0$ is the regularization parameter.

Our goal is to find β such that equation 6 is minimized. It is nontrivial to solve this optimization problem because the objective function is in quadratic form. Compute the first derivative of equation 6 with respect to β, we have:

$$\frac{\partial L}{\partial \beta} = -2X^T(Y - X\beta) + 2\lambda L\beta \quad (7)$$

then by setting the derivative to zero, we can obtain:

$$\begin{aligned} 0 &= -2X^T(Y - X\beta) + 2\lambda L\beta \\ (X^T X + \lambda K_c)\beta &= X^T Y \\ \hat{\beta} &= (X^T X + \lambda L)^{-1} X^T Y \end{aligned} \quad (8)$$

where $\hat{\beta}$ is our estimation and L is $p \times p$ laplacian matrix.

3.3 Relationship with Ridge Regression

Ridge regression is a classical L_2 norm regularization based linear regression. In our framework, ridge regression is a special case. Ridge regression minimizes $\|Y - X\beta\|^2 + \lambda\beta^T\beta$, which is exactly the same when we set the Laplacian matrix to the identical matrix.

Next, we will show that our feature kernel regression framework shrinks more on the directions where singular value of $X \cdot L^{-\frac{1}{2}}$ is smaller. Similarly Ridge regression penalizes more on the directions where singular value of X is smaller.

Applying eigen decomposition to the positive definite kernel matrix L, we can factor L into the product of a matrix and the transpose of the matrix L_r, represented as $L = L_r^T L_r$, where $L_r = D^{\frac{1}{2}} V^T$ and D is the diagonal matrix with eigen values and V is the matrix with columns as eigen vectors. The solution of our framework can be rewritten as $\hat{\beta} = (X^T X + \lambda L_r^T \cdot L_r)^{-1} X^T Y$. By employing Generalized Singular Value Decomposition for X $(n \times p)$ and K_r $(p \times p)$, we denote $X = U\Sigma_1[0, R]Q^T$ and $L_r = V\Sigma_2[0, R]Q^T$. The factorization satisfies following properties:

- U is $n \times n$, V is $p \times p$, Q is $p \times p$, and all three matrices are orthonormal.
- R is $r \times r$, upper triangular and nonsingular. [0,R] is $r \times p$ and $r = rank(X^T, L_r^T) \leq p$.
- Σ_1 is $n \times r$, Σ_2 is $p \times r$, both are real, nonnegative and diagonal, and $\Sigma_1^T\Sigma_1 + \Sigma_2^T\Sigma_2 = I$. Write $\Sigma_1^T\Sigma_1 = \text{diag}(\alpha_1^2, \ldots, \alpha_r^2)$ and $\Sigma_2^T\Sigma_2 = \text{diag}(\gamma_1^2, \ldots, \gamma_r^2)$, where α_i and β_i lie in the interval from 0 to 1. The ratios $\alpha_1/\gamma_1, \ldots, \alpha_r/\gamma_r$ are called the generalized singular values of the pair X, K_r.
- If L_r is fully ranked, then $r = p$. The generalized singular value decomposition of X and L_r is equivalent to the singular value decomposition of XL_r^{-1}, where the singular values of XL_r^{-1} are equal to the generalized singular values of X and K_r. That is: $XL_r^{-1} = (U\Sigma_1 RQ^T)(V\Sigma_2 RQ^T)^{-1} = U(\Sigma_1\Sigma_2^{-1})V^T$

Since $rank(L_r) = rank(L_r^T \cdot L_r) = rank(L)$ and L is nonsingular in our framework, $[0, R] = R$. Our estimation of Y is:

$$\begin{aligned} \hat{Y} &= X\hat{\beta} \\ &= X(X^T X + \lambda L_r^T \cdot L_r)^{-1} X^T Y \\ &= U\Sigma_1 RQ^T (QR^T\Sigma_1 U^T U\Sigma_1 RQ^T + \\ &\quad \lambda QR^T\Sigma_2 V^T V\Sigma_1 RQ^T)^{-1} QR^T\Sigma_1 U^T Y \\ &= U\Sigma_1 RQ^T (QR^T\Sigma_1^2 RQ^T + \lambda QR^T\Sigma_2^2 RQ^T)^{-1} QR^T\Sigma_1 U^T Y \\ &= U\Sigma_1(\Sigma_1^2 + \lambda\Sigma_2^2)^{-1}\Sigma_1 U^T Y \\ &= \sum_{i=1}^{n} u_i \frac{\alpha_i^2}{\alpha_i^2 + \lambda\gamma_i^2} u_i^T Y \\ &= \sum_{i=1}^{n} u_i \frac{1}{1 + \lambda/(\alpha_i/\gamma_i)^2} u_i^T Y \\ &= \sum_{i=1}^{n} u_i \frac{1}{1 + \lambda/d_i^2} u_i^T Y \end{aligned}$$

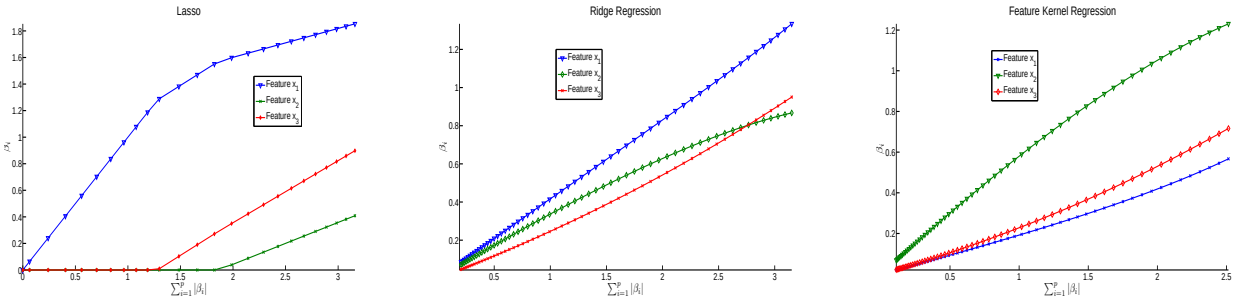

Figure 2: Left: Lasso estimates as a function of $\sum_{i=1}^{3}|\beta_i|$. Middle: Ridge estimates as a function of $\sum_{i=1}^{3}|\beta_i|$. Right: Feature Kernel Regression estimates as a function of $\sum_{i=1}^{3}|\beta_i|$.

where u_i is the column vector of U and $d_i = \alpha_i/\gamma_i$. From the result, we observe that L_2 norm regularized feature kernel regression first projects Y on the basis of U generated by singular value decomposition of $XL^{-\frac{1}{2}}$ where L is the Laplacian matrix, then the projected values are re-scaled according to the values encoded in the diagonal matrix:

$$D = diag(\alpha_1^2/(\alpha_1^2+\lambda\gamma_1^2), \ldots, \alpha_p^2/(\alpha_p^2+\lambda\gamma_p^2))$$

Finally the stretched values are re-described in the coordinate system by using the basis (columns) of U.

Compared with Ridge regression purely data driven, which just projects Y on the principle component of X and then shrinks coefficients along the direction lower singular value of X, we consider both data and features.

4. EXPERIMENTAL STUDY

4.1 A simulation study

The purpose of this simulation is to show that the L_2 norm feature kernel regression not only stabilizes the regression coefficients but assigns coefficient values based on the complexities of features. We generate multi-variate Gaussian data with n samples having zero mean and p features. For simplicity, we generate $n = 200$ samples and $p = 3$ features x_1, x_2, x_3, where x_1 and x_2 are correlated with correlation coefficient $\rho = 0.9$ and x_3 is independent from the rest two features. The response value Y is generated by

$$Y = X\beta + \epsilon, \epsilon \sim N(0,1)$$

Where $\beta = [1.1, 1.0, 0.5]^T$. Assume we have additional information about the features and the feature kernel matrix is given by:

$$K = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

We run Lasso, ridge and our method on this data set over a wide range of regularization parameters and show the regularization pathways of feature kernel regression, lasso and Ridge in Figure 2. From Figure 2, Lasso selects x_1 first regardless of the fact that x_1 is much more complicated than x_2, and the simple feature x_2 will not enter the active set until very small penalty; Ridge regression assigns almost equal coefficients to x_1 and x_2; for feature kernel regression, it is clear that the x_1 with high complexity obtains small coefficient and x_2 with low complexity is assigned large coefficient. This is desirable for building a regression model because the chance complex feature from training data occur in test data is very low and simple features will give better generalization performance.

4.2 Real-world data study

We have performed a comprehensive study of the performance of our regression framework using 5 chemical structure graph data sets. We have compared our method with 2 representative regularized regression methods: Lasso [19] and Ridge Regression [7].

For each data set, we used the FFSM algorithm [8] to extract frequent subgraph features from the data sets. We measured the regression performance of our regression method and compared ours with those from state-of-the-art methods using cross validation.

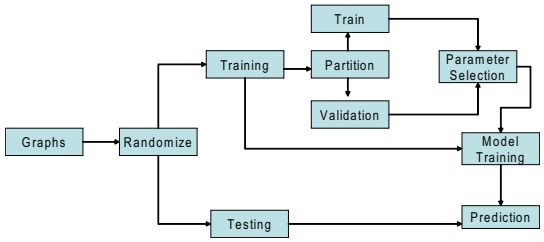

Figure 3: Experimental work flow for a single cross validation trial

4.3 Data Sets

We select 4 chemical data sets from Binding Database [12] and 1 data set EDKB from `http://edkb.fda.gov/databasedoor.html`. For each data set, the response values are chemical's binding affinity to a particular receprot. In this case, the affinity is measured by the concentration, which represents how much of this chemicals is needed to observe binding activity to a certain protein. See BindingDB [12] and ChemDB [1] for further details regarding the nature of the data sets.

We follow the same procedure [8] to use a graph to model a chemical structure: a vertex represents an atom and an edge represents a chemical bond. Hydrogen atoms are removed

in our graph representation of chemicals, as commonly done in the cheminformatics field. The characteristics of the data set is shown in Table 1.

Table 1: Data set: the symbol of the data set. S: total number of samples in the data set. $\overline{V}$: average number of nodes in the data set, $\overline{E}$: average number of edges in the data set

Data set	S	$\overline{V}$	$\overline{E}$
EDKB	59	18.5	20.1
CarbonicI	327	23.8	24.8
CarboxylesteraseI	143	16.4	17.5
CathepsinK	257	32.8	34.7
CathepsinD	103	45.7	48.4

4.4 Experimental Protocol

For each data set, we mined frequent subgraphs using the FFSM algorithm [8] with $min\_support = 25\%$ and with at least 2 nodes and no more than 10 nodes. Empirical study shows that there is no significant changes if we replace the fixed value 25 with a relatively wide range of values. We then treated each subgraph as a feature. We adopted two ways of extracting feature values: exact subgraph matching and approximate subgraph matching. For exact subgraph matching, We create a binary feature vector for each graph in the data set, indexed by the mined subgraphs, with values indicate the existence (1) or absence (0) of the related features. For approximate matching, the feature vector construction is exactly the same except that the feature value is a real number between 0 and 1 representing the ratio of the matching size and the feature size. To build feature kernel matrix, we use CHEMCPP, a public library available at `http://chemcpp.sourceforge.net/html/index.html`.

As indicated before, we compared our method with other 2 regression methods. To have a fair comparison, we use 5-fold cross validation to derive training and testing samples and run all the methods. Since we have regularization parameter λ in all three methods, we did internal 5 fold cross validation within the training data to obtain the best tuning parameter of each method, obtain regression model on the whole training data and apply the trained model to test data to make prediction. We repeat the whole process 10 times and report average performance. Figure 3 gives an overview of our experimental set up.

For one cross validation, the prediction accuracy is measured by R^2. R^2 is close to 1 when the regression function fits good, and is close to 0 when it does not.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y_i})^2}{\sum_{i=1}^{n}(y_i - \frac{1}{n}\sum_{i=1}^{n} y_i)^2}$$

where y_i is true value, $\hat{y_i}$ is the prediction and n is total number of samples. For each data set, we repeat 5 fold cross validation 10 times and report the average R^2 value and standard deviation. We perform all of our experiments on a desktop computer with a 3Ghz Pertium 4 processor and 4 GB of RAM.

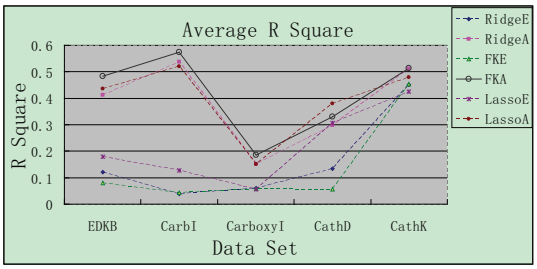

Figure 4: Average prediction accuracy comparison over 3 regression methods on 8 data sets.

4.5 Experimental Results

4.5.1 Performance Comparison

In this section, we present the performance of our method compared with two additional methods: Lasso and Ridge. Based on different feature extraction ways, we have six variations of methods to compare: LassoE (Lasso with exact subgraph matching), LassoA (Lasso with approximate subgraph matching), RidgeE (Ridge with exact subgraph matching), RidgeA (Ridge with approximate subgraph matching), FKE (feature kernel with exact subgraph matching) and FKA (feature kernel with approximate subgraph matching). The prediction performance is measured by average R^2 of validation results and is shown in Figure 4.

In Figure 4, the X-axis is labeled with data set name and Y-axis is average R^2 value. The performance of the three methods varies with the same trend. A clear trend is that prediction accuracy from approximate feature value extraction is better than that of exact feature value extraction for all the three methods. Among approximate feature extraction experiments, our method outperforms LassoA and RidgeA in 3 out of 5 data sets, and 1 comparable. For exact subgraph matching, LassoE outperforms RidgeE and FKE. A future work is to investigate why the performance of feature kernel regression is not consistent in different feature extraction ways.

Table 2 shows Average R^2 value and standard deviation for three methods under approximate subgraph matching. From Table 2, we observe that our method is relatively more stable than Lasso and Ridge with smaller standard deviation and higher prediction accuracy.

4.5.2 Method Robustness With min_sup

Since we have parameter min_sup in the feature extraction step, we changed the min_sup during the feature generation process to see the robustness of our method. In the following study, we have singled out a data set (the CathepsinK data set) and test the robustness of our method using this data set by varying the min_sup in frequent subgraph feature extraction process.

In Figure 5, we changed min_sup during feature generation process to see the robustness of our method. We change min_sup from 20% to 50%, and compute average R^2 based on the same experiment protocol. From Figure 5, we can see that our method remains stable with variant of min_sup.

Overall, our L_2 norm feature kernel method is effective and achieves good accuracy within a wide range minimum support. Our method is not constrained by marginal graph

Table 2: Average R^2 value and standard deviation of three methods with $*$ denoting the highest value.

Data set	LassoE	RidgeE	FKE
EDKB	0.436 ± 0.145	0.413 ± 0.171	0.483* ± 0.092
CarbonicI	0.521 ± 0.052	0.538 ± 0.061	0.574* ± 0.052
CarboxylesteraseI	0.150 ± 0.134	0.151 ± 0.186	0.55* ± 0.133
CathepsinD	0.379* ± 0.140	0.299 ± 0.135	0.330 ± 0.132
CathepsinK	0.479 ± 0.100	0.511 ± 0.087	0.513* ± 0.051

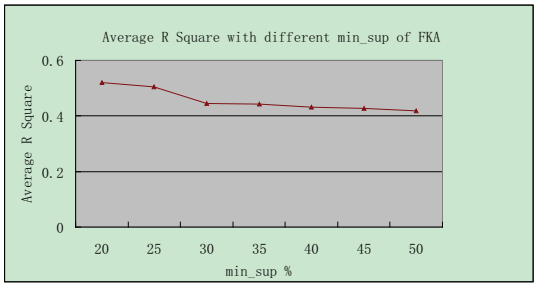

Figure 5: Average prediction accuracy for 5 fold cross validation with different min_sup for one data set

kernel or Dirac Kernel, any other graph kernel function can be combined with our framework.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we studied the regression problem in which the feature has intrinsic complexity and presented a novel L_2 norm feature kernel regression method for graph data. By incorporating laplacian induced from subgraph feature kernel matrix into penalize function, we solved this new optimization problem and revealed its connection with Ridge regression. Compared with current state-of-the-art methods as evaluated on 5 real world data sets, our method significantly outperforms the Lasso and ridge on majority of the tested data sets. In this framework, we penalize on L_2 norm of feature kernel only and that will not introduce sparseness to our model. In the future, we will design a new model that combine L_1 and L_2 norm penalization on features together to achieve both sparsity and stability of regression model. Also we will test more kernel function for this framework to see whether the performance is consistent.

Acknowledgments

This work has been partially supported by the Office of Naval Research (award number N00014-07-1-1042).

6. REFERENCES

[1] J. Chen, S. J. Swamidass, J. B. Y. Dou, and P. Baldi. Chemdb: A public database of small molecules and related chemoinformatics resources. *Bioinformatics*, 21(22):4133ÍC4139, 2005.

[2] M. Deshpande, M. Kuramochi, and G. Karypis. Frequent sub-structure-based approaches for classifying chemical compounds. *IEEE Transactions on Knowledge and Data Engineering*, 2005.

[3] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angel regression. *Ann. Statist.*, 32:407–499, 2004.

[4] H. Fei and J. Huan. Structure feature selection for graph classification. In *Proc. ACM 17th Conference on Information and Knowledge Management*, 2008.

[5] T. Gärtner, P. Flach, and S. Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Sixteenth Annual Conference on Computational Learning Theory and Seventh Kernel Workshop*, 2003.

[6] D. Haussler. Convolution kernels on discrete structures. *Technical Report UCSC-CRL099-10, Computer Science Department, UC Santa Cruz*, 1999.

[7] A. Hoerl and R. W. Kenard. Ridge regression: biased estimation for nonorghognal problems. *Technometrics*, 12:55–67, 1970.

[8] J. Huan, W. Wang, and J. Prins. Efficient mining of frequent subgraph in the presence of isomorphism. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM)*, pages 549–552, 2003.

[9] I. Jolliffe. *Principal Component Analysis*. Springer; 2nd ed. edition, 1986.

[10] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *Proc. of the Twentieth Int. Conf. on Machine Learning (ICML)*, 2003.

[11] B. K.M. and K. H.-P. Shortest-path kernels on graphs. In *in Proc. of International Conference on Data Mining*, 2005.

[12] T. Liu, Y. Lin, X. Wen, R. N. Jorissen, and M. K. Gilson. Bindingdb: a web-accessible database of experimentally determined proteinÍcligand binding affinities. *Nucleic Acids Research*, 35:198–201, 2007.

[13] B. Quanz and J. Huan. Aligned graph classification with regularized logistic regression. In *Proc. 2009 SIAM International Conference on Data Mining*, 2009.

[14] J. Ramon and T. Gärtner. Expressivity versus efficiency of graph kernels. In *Technical Report, First International Workshop on Mining Graphs, Trees and Sequences*, 2003.

[15] H. Saigo, N. Krämer, and K. Tsuda. Partial least squares regression for graph mining. In *Proc. SIGKDD08*, 2008.

[16] B. Schölkopf and A. J. Smola. *Learning with Kernels*. the MIT Press, 2002.

[17] A. Smalter, J. Huan, and G. Lushington. Structure-based pattern mining for chemical compound classification. *Proceedings of the 6th Asia Pacific Bioinformatics Conference*, 2008.

[18] S. W. Tamas Horvath, Thomas Gartner. Cyclic pattern kernels for predictive graph mining. *SIGKDD*, 2004.

[19] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society B*, 58:267–288, 1996.

[20] K. Tsuda. Entire regularization paths for graph data. In *ICML07*, 2007.

[21] X. Yan, H. Cheng, J. Han, and P. Yu. Mining significant graph patterns by leap search. pages 433–444, 2008.

[22] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.

[23] P. Zhao and B. Yu. Grouped and hierarchical model selection through composite absolute penalties. *Annals of Statistics*, 2006.

[24] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society B*, 67:301–320, 2005.