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 dataset 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 models 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 incorporates 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 do not consider the additional information of the feature. We have tested our algorithm using 5 different real-world data sets and have demonstrated 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
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 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.


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 $\text{minsup} = \frac{2}{3}$. Using $F_1$ and $F_2$ as features, the object-feature matrix $X$, where each row is a graph and each column is a feature, is represented as:
should be applicable for any types of graph data. Algorithms primarily using data sets from chemical struc-

gression using subgraph features. Though we evaluate our

so is that they treat the feature as an atomic element and

The reason why current regularization methods fail to do

To solve the supervised graph learning problem,

algorithms that select simpler features to construct super-

complexity of subgraph features. Second, we investigate the

complexity in the model selection process, we investigate two

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 \cite{19} based regression

method will randomly pick up a feature and assign coeffi-

cient to it because $F_1$ and $F_2$ have the same correlation with

labels. Ridge \cite{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 sup-

ervised 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 re-

gression using subgraph features. Though we evaluate our

algorithms primarily using data sets from chemical struc-

ture 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 $L_2$ 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 evalua-

tion. 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 back-

ground 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 \cite{7} developed ridge regression based on

L2 norm regularization. Tibshirani\cite{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 coeffi-

cients. Efron & Hastie \cite{3} designed a novel algorithm, Least

Angel Regression (LARS), to solve the optimization problem

in Lasso efficiently. Zou & Hastie \cite{24} developed a regres-

sion 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 \cite{22} studied the case

when features have a natural group structure and designed

a technique to select grouped features called group Lasso.

Zhao & Yu \cite{23} integrated a hierarchical relation on features

to regression and proposed a method called iCAP. Quanz &

Huan \cite{13} assumed a general undirected graph relationships

of features and employed the feature graph Laplacian in lo-

gistic regression for graph classification.

Though regularized regression has been studied for a long

time, none of the existing method considers the special char-

acteristics 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 in-

formation 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 sub-

graph 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 applica-

tions, a graph is labeled, where labels are drawn from a label set

$\sigma$. A labeling function $\lambda : V \cup E \rightarrow \Sigma$ 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 $\Sigma$ 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 $V, E, \Sigma, \lambda$ are explained be-

fore. 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 $f : V \rightarrow V'$ such that
2.2 Frequent Subgraph Mining

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

\[ \text{sup}_{G} = \frac{|G' \in GD | G \subseteq G'|}{|GD|} \]

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. supG ≥ 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

\[ \sum_{j=1}^{n} c_i c_j K(x_i, x_j) \geq 0 \quad (1) \]

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

\[ k(x, x') = \langle \Phi(x), \Phi(x') \rangle \quad (2) \]

for all \( x, x' \in X \). \( (x, y) \) 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:

\[ Y = X \beta + \epsilon \quad (3) \]

where \( Y \) is a \( n \times 1 \) vector, \( X \) is a \( n \times p \) matrix and \( \beta \) is a coefficient vector with the size of \( p \times 1 \) and \( \epsilon \) is gaussian noise with mean 0 and standard deviation \( \delta \). Ordinary Least Square (OLS) minimizes the sum of squared errors \( ||Y - X\beta||^2 \), where \( ||.|| \) is L2 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 L2 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 \( \mathcal{G} = G_1, G_2, \ldots, G_m \), 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 \( F = F_1, F_2, \ldots, F_n \) 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:

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

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. $K_h(z, z')$ is the kernel between the sequences of vertex and edge labels traversed in the random walk.

To avoid singularity, we add a 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’ 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 $X = (x_1, \ldots, x_p)$, where $x_j = (x_{j1}, \ldots, x_{jn})^T$, $j = 1, \ldots, p$. We also assume that the predictors are standardized and the response is centered so that for all $j$, $\sum_{i=1}^{n} x_{ij} = 0$, $\sum_{i=1}^{n} x_{ij}^2 = 1$ and $\sum_{i=1}^{n} y_i = 0$. The regression function is linear with the following form: $Y = X\beta$, where $\beta$ is a $n \times 1$ coefficient vector.

The Lagrange form of the objective function is:

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

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

Our goal is to find $\beta$ such that equation 5 is minimized. It is nontrivial to solve this optimization problem because the objective function is in quadratic form. Compute the first derivative of equation 5 with respect to $\beta$, we have:

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

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

$$0 = -2X^T(Y - X\beta) + 2\lambda L\beta$$

$$(X^T X + \lambda L)\beta = X^T Y$$

$$\hat{\beta} = (X^T X + \lambda L)^{-1}X^T Y$$

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 $K$, 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 $\hat{\beta} = (X^T X + \lambda I)^{-1}X^T Y$. 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 $r = \text{rank}(X^T, L_r^T) \le p$.

- $\Sigma_1$ is $n \times r$, $\Sigma_2$ is $p \times r$, both are real, nonnegative and diagonal, and $\Sigma_1^2 \Sigma_1 + \Sigma_2^2 \Sigma_2 = I$. Write $\Sigma_1^2 \Sigma_1 = \text{diag}(\alpha_1^2, \ldots, \alpha_r^2)$ and $\Sigma_2^2 \Sigma_2 = \text{diag}(\gamma_1^2, \ldots, \gamma_r^2)$, where $\alpha_i$ and $\beta_i$ 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 $X L_r^{-1}$, where the singular values of $X L_r^{-1}$ are equal to the generalized singular values of $X$ and $K_r$. That is: $X L_r^{-1} = (U \Sigma_1 R Q^T)(V \Sigma_2 R Q^T)^{-1} = U(\Sigma_1 \Sigma_2^2)^{-\frac{1}{2}} V^T$.

Since $\text{rank}(L_r) = \text{rank}(L_r^T + \lambda L) = \text{rank}(L)$ and $L$ is non-singular in our framework, $[0, R] = R$. Our estimation of $Y$ is:

$$\hat{Y} = X \hat{\beta}$$

$$= X (X^T X + \lambda L_r^T L_r)^{-1} X^T Y$$

$$= U \Sigma_1 R Q^T (Q R^T \Sigma_1 U^T U \Sigma_1 R Q^T + \lambda Q R^T \Sigma_1 V^T V \Sigma_1 R Q^T)^{-1} Q R^T \Sigma_1 U^T Y$$

$$= U \Sigma_1 R Q^T (Q R^T \Sigma_1^2 R Q^T + \lambda Q R^T \Sigma_2^2 R Q^T)^{-1} Q R^T \Sigma_1 U^T Y$$

$$= U \Sigma_1 (\Sigma_1^2 + \lambda \Sigma_2^2)^{-\frac{1}{2}} \Sigma_1 U^T Y$$

$$= \sum_{i=1}^{n} \frac{u_i}{\alpha_i^2 + \lambda \gamma_i^2} u_i^T Y$$

$$= \sum_{i=1}^{n} \frac{u_i}{1 + \lambda / (\alpha_i / \gamma_i)^2} u_i^T Y$$

$$= \sum_{i=1}^{n} \frac{u_i}{1 + \lambda / d_i} u_i^T Y$$
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:

\[
D = \text{diag}(\alpha_1^2 / (\alpha_1^2 + \lambda_1^2), \ldots, \alpha_p^2 / (\alpha_p^2 + \lambda_p^2))
\]

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 \( x_3 \) 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.

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>
<thead>
<tr>
<th>Data set</th>
<th>S</th>
<th>V</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDKB</td>
<td>59</td>
<td>18.5</td>
<td>20.1</td>
</tr>
<tr>
<td>CarboxinC</td>
<td>327</td>
<td>23.8</td>
<td>24.8</td>
</tr>
<tr>
<td>Carboxylesterase</td>
<td>143</td>
<td>16.4</td>
<td>17.5</td>
</tr>
<tr>
<td>CathepsinK</td>
<td>257</td>
<td>32.8</td>
<td>34.7</td>
</tr>
<tr>
<td>CathepsinD</td>
<td>103</td>
<td>45.7</td>
<td>48.4</td>
</tr>
</tbody>
</table>

### 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.

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

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.

### 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...
Acknowledgments

Our method significantly outperforms the Lasso and ridge on majority of the tested data sets. In this framework, we penalize on $L^2$ 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.

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 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


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

<table>
<thead>
<tr>
<th>Data set</th>
<th>LassoE</th>
<th>RidgeE</th>
<th>FKE</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDKB</td>
<td>0.436 ± 0.145</td>
<td>0.413 ± 0.171</td>
<td>0.483* ± 0.092</td>
</tr>
<tr>
<td>CarbonicI</td>
<td>0.521 ± 0.052</td>
<td>0.538 ± 0.061</td>
<td>0.574* ± 0.052</td>
</tr>
<tr>
<td>CarboxyesteraseI</td>
<td>0.150 ± 0.134</td>
<td>0.151 ± 0.186</td>
<td>0.55* ± 0.133</td>
</tr>
<tr>
<td>CathepsinD</td>
<td>0.379* ± 0.140</td>
<td>0.299 ± 0.135</td>
<td>0.330 ± 0.132</td>
</tr>
<tr>
<td>CathepsinK</td>
<td>0.479 ± 0.100</td>
<td>0.511 ± 0.087</td>
<td>0.513* ± 0.051</td>
</tr>
</tbody>
</table>

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


