Out-Of-Bag Discriminative Graph Mining

Andreas Maunz
Institute for Physics, Hermann-Herder-Str. 3, 79104 Freiburg, Germany

1 Abstract

In class-labeled graph databases, each graph is associated with one from a finite set of target classes. This induces associations between subgraphs occurring in these graphs, and the target classes. The subgraphs with strong target class associations are called discriminative subgraphs. In this work, discriminative subgraphs are repeatedly mined on bootstrap samples of a graph databases in order to estimate the subgraph associations precisely. This is done by recording the subgraph frequencies (support values) per class over the out-of-bag instances of the bootstrap process. We investigate two different methods for the approximation of the true underlying support values from these empirical values, involving sample mean and maximum likelihood estimation. We show that both methods improve the estimation significantly, compared to single runs of discriminative subgraph mining, by applying the different methods to publicly available, chemical datasets. In computational models of toxicology, subgraphs (fragments of chemical structure) are routinely used to describe toxicological properties of molecules. Apart from being statistically validated, the sizes of subgraph sets created by the proposed methods is also much reduced compared to standard methods and may thus be useful for statistical models, as well as inspection by toxicological experts.

2 Introduction

It is a fundamental fact that no finite sample can be an exact representation of the unknown underlying distribution from which the sample was drawn. For example, in discriminative subgraph mining, there is no way of inferring the exact correlations between fragments occurring in the graphs and the target classes, which could serve to identify the set of patterns that primarily lead to the occurrence of a certain target class, e.g. the relations between molecular fragments and exhibited toxicity of molecules. Finding such relations is, however, a major goal in the research of chemical reactivity of compounds, namely in the area of quantitative structure activity relationships (QSARs). Graph mining is also used in other areas such as TODO: cite.

Moreover, discriminative graph mining is an unstable process, where slight changes in the sample lead to very different subgraph sets. Fortunately, this can be rectified by deliberately, repeatedly disturbing the sampling process and combining estimations over many samples. For example, in bootstrap sampling, estimates of generalization error obtained from the out-of-bag instances were shown to drastically improve on estimates obtained from the sample as a whole [3, 8]. This work employs the out-of-bag instances to precisely estimate the frequencies (support values) of the subgraphs on the target classes that govern the training data. Infrequently occurring subgraphs are filtered out after the bootstrapping. For each remaining subgraph, a significance test is run using the estimated support values to filter out insignificant subgraphs. This results in a statistically validated, highly significant, discriminative subgraph set.

In contrast to bagging, combined statistical models are not the subject of this work. However, class-correlated subgraph mining is supervised process that should be seen in close context to model building (see section 3). For example, a statistical model could be directly learned on the subgraph descriptors, incorporating their estimated support-values. Moreover, human experts should be able to draw conclusions from the greatly reduced set. Being subgraphs, they can be easily matched on the chemical compound structure.

The remainder of this work is structured as follows: We present related work with a focus on discriminative

subgraph mining (section 3), our proposed methods for estimating support values and p-values (section 4), as well as algorithmic implementation (section 5). Experiments include validation of support per class and p-values in several methods on publicly available databases of various sizes and numbers of target classes (section 6), before we draw conclusion in section 7.

The contributions of this work are summarized as follows:

- Repeated discriminative subgraph mining on bootstrap samples of a class-labeled graph database is proposed as a means to stabilize estimates of support values per class for the subgraphs, compared to single runs of discriminative subgraph mining. To this end, the subgraphs' distributions on the out-of-bag instances associated with different target classes are recorded along the bootstrapping process.
- Two methods for the estimation of support values from the recorded value are described. Both handle multiple classes. Significance tests are applied to these distributions and insignificant subgraphs are removed from the results.
- The estimated support values, predicted p-values, and predicted class associations are empirically validated on molecular databases of various sizes. The results indicate significant improvements over estimates ordinary discriminative subgraph mining, where the effect is larger the smaller the datasets are. We conclude that, for discriminative graph mining, out-of-bag estimation has the potential to generate concise, highly discriminative descriptor sets for use in statistical models and/or expert inspection of the molecular fragments.

3 Related Work

Out-of-bag methods have been used to robustly estimate node probabilities and node error rates in decision trees [8] as well as the generalization error and accuracy of bagged predictors [3]. In the work by Bylander [3], generalization error was well modeled by out-of-bag estimation, and its small bias could be further improved by a correction method, where, in order to correct the prediction of a given instance, similar out-of-bag instances with the same target class were employed. However, the method of out-of-bag estimation is not confined to these examples, and may be used to estimate general statistical properties.

Subgraph mining is often employed as a preprocessing step to statistical learning, because subgraphs may be useful as descriptors [2, 13]. To this end, there is a variety of well-known statistical learning algorithms available "off the shelf" [5, 11]. This makes a workflow attractive, where regularities and patterns are extracted from the data, represented in a unified format and fed into a machine learning algorithm [6]. However, subgraph mining often produces huge descriptor sets and experts would not be able to draw conclusions from the vast amount of very similar subgraphs, since the few useful patterns would be lost in the flood. Similarly, the high-dimensional pattern space would prevent machine learning methods from obtaining models in acceptable time [1]. Thus, post-processing would be required to lower redundancy and eliminate the vast majority of subgraphs [13, 4, 7].

Subgraph boosting [12] is an integrated method that employs subsampling internally, alternating between graph mining and model building. The method presented here is clearly different from subgraph boosting (and from boosting in general), because it calculates descriptors that can be used in wide variety of models afterwards. It is similar in that it outputs a small collection of most discriminative subgraphs, however, the collection computed here is much more stable, because it is robust against perturbations of the dataset.

4 Methods

4.1 Basic Graph Theory

A graph database is a tuple (G, Σ, a), where G is a set of graphs, $\Sigma \neq \emptyset$ is a totally ordered set of labels and $a : G \to C$, $C \subset \mathbb{N}$, is a function that assigns one from a finite set of class values to every graph in the database. The set of target classes C may consist of more than two values. We consider labeled, undirected graphs, i.e. a tuples $g = (V, E, \Sigma, \lambda)$, where $V \neq \emptyset$ is a finite set of nodes and $E \subseteq V = \{\{v_1, v_2\} \in \{V \times V\}, v_1 \neq v_2\}$ is a set of edges and $\lambda : V \cup E \to \Sigma$ is a label function. We only consider connected graphs here, i.e. there is a path between each two nodes in the graph.

A subgraph $g' = (V', E', \Sigma', \lambda')$ of g is a graph such that $V' \subseteq V$ and $E' \subseteq E$ with $V' \neq \emptyset$ and $E' \neq \emptyset$, $\lambda'(v_1) = \lambda(v_2)$ whenever $v_1 = v_2$, and $\lambda'(e_1) = \lambda(e_2)$ whenever $e_1 = e_2$. If g' is a subgraph of g, we also say that g' covers g. The subset of the database instances G that g' covers is referred to as the occurrences of g', and its size as support of g'. As a special case, the size of the subset of occurrences with $a(g) = i$, for any g in the occurrences, is referred to as the support of g' for class i. Thus, any subgraph has associated support values per class, ranging each between 0 and the support of g'.

The subgraphs considered in this work are free subtrees. Here, we define a tree as a graph with exactly $n-1$ edges that connects its n nodes. A free tree is a tree without a designated root node. For an introduction to tree mining, see the overview by Chi *et al.* [4].

4.2 Formulation of Out-Of-Bag Discriminative Graph Mining

We refer to the subset of a graph database G that contains all the graphs g with $a(g) = i$ as G^i. The procedure first splits the graph database randomly into equal-sized training and test databases G_{Train} and G_{Test}. Subsequently, stratified bootstrapping is performed on the training data, such that each sample comprises $|G^i_{Train}|$ graphs associated with class i, drawn with replacement and uniform probability $1/h_i$ inside each class i. On average, about 37% $(1/e)$ of training instances (molecules) will not be drawn in any bootstrap sample (out-of-bag). Subgraph mining mines the subgraphs on the in-bag instances, but looks up the support values per class on the out-of-bag instances, by performing subgraph isomorphism tests (matching).

After the inital split, the bootstrapping is repeated N times, where in each iteration, pairs of subgraphs and support values per class $(g, k_1, \ldots, k_{|I|})$ are produced, meaning that subgraph g occurs in k_i out-of-bag graphs associated with class i. The results are recorded over the N bootstrap samples, such that for each g, the list of support values is a tuple $(\mathbf{k_1} \ldots \mathbf{k_{|I|}})$, where $\mathbf{k_i}$ is a vector $(k_i^1 \ldots k_i^N)^T$, containing all the support values.

The total support is determined from the class specific support values by summing across target classes: $\mathbf{k} = \sum_{i=1}^{|I|} \mathbf{k_i}$. The vector $\mathbf{k}$ contains the (total) support values for each bootstrap sample, obtained by summing up vectors $\mathbf{k_i}$. The vectors $\mathbf{k_i}$ are generally sparse, due to the instability of discriminative subgraph mining, i.e. perturbations to the dataset (such as bootstrap sampling) yield almost always a different (but overlapping) selection of subgraphs. To cope with the variety of rare subgraphs, a fixed threshold removes all subgraphs with less than $\lfloor 0.3 * N \rfloor$ entries in the list of support values.

The estimation of support values is performed seperately for each remaining subgraph, as described in section 4.3. Two methods are employed, based either on the sample mean support per class, and using data of the current subgraph only, or on a maximum likelihood estimate, involving some of the other subgraphs. Then, a significance test is run on the estimated support values.

	g	*all*
class 1	k_1	$\lvert G^1\rvert$
class 2	k_2	$\lvert G^2\rvert$
...	...	...
class $\lvert I\rvert$	$k_{\lvert I\rvert}$	$\lvert G^{\lvert I\rvert}\rvert$
Σ	k	$\lvert G\rvert$

Table 1: Contingency table for subgraph g.

4.3 Significance and Support Values Estimation

The significance test scenario for each subgraph is described first, followed by the two methods to compute the input values to the test.

4.3.1 Significance Test

For a given subgraph g, we seek a $|I| \times 2$ contingency table that lists the support per class in the first column and the overall distribution of target classes in the second column, see Table 1. This data allows to check whether g's support values differ significantly from the overall class distribution. The χ^2_d function for distribution testing, defined as

$$\chi^2_d(x, y) = \sum_{i \in \{1,\ldots,|I|\}} \frac{(k_i - E(k_i))^2}{E(k_i)}, \tag{1}$$

where $E(k_i) = \frac{G^i k}{|G|}$ is the expected value of k_i, calculates the sum of squares of deviations from the expected support for all target classes. The function values are then compared against the χ^2 distribution function to obtain p-values and conduct a significance test with $|I| - 1$ degrees of freedom. The next sections discusses the methods to obtain the k_i entries in the table from the recorded support values (section 4.2). The $|G^i|$ values are constants and take the values of the overall $|G^i|$. This is possible due to the stratified bootstrapping, which maintains the overall class proportions in each sample (see section 4.2).

4.3.2 Sample Mean Method

We set k_i in Table 1 to $\overline{\mathbf{k_i}}$ for all $i \in \{1, \ldots, |I|\}$, the sample mean across the entries of vector $\mathbf{k_i}$ (ignoring missing values). The value k is determined by the k_i.

4.3.3 Maximum Likelihood Estimation Method

Here, we employ some of the other subgraphs to form estimates for the k_i. The first step in this process is to extract the subgraphs with same class bias as g, i.e the class is identified in which g appears more frequently than expected by comparing g's relative (local) frequencies on the target classes to the overall (global) relative frequencies of target classes, i.e. on G. Local ties are broken in favor of the dominant global class. In case of a further tie on the global level, one of the globally dominant classes is chosen with uniform probability. Given g's class bias, the other subgraphs with the same class bias (biases determined in the same manner) are set aside.

In a second step, the subgraphs with same class bias are used to correct g's local frequencies by weighting. This approach has some similarity to the work by Bylander [3], however, his aim is to correct instance predictions, and his correction employs similar out-of-bag instances, whereas our correction happens across bootstraps, and on the subgraphs (not instances) obtained collectively from all the bootstrap samples. For each class, we model the event that each $k_i^j \in \mathbf{k_i}$ would occur for each of the subgraphs with the same class bias as g as a multinomial selection

Algorithm 1 Estimate subgraph significance on out-of-bag instances

Input: $dataBase, numBoots, minSamplingSupport, method, minFrequencyPerSample, \alpha$

```
if numBoots = 1 then
    ans ← BBRC(dataBase, minFrequencyPerSample, α)
else
    hash ← {}
    for i := 1 → numBoots do                                   ▷ Parallel Processing
        res ← BBRC(drawBsSample(dataBase), minFrequencyPerSample, α)
        insert(hash, matchOOB(res))
    end for
    ans ← ∅
    for subgraph ∈ keys(hash) do
        if length(hash[subgraph]) ≥ minSamplingSupport then
            if (p = SignificanceTest(hash[subgraph]), method) < α then
                ans ← ans ∪ (subgraph, p)
            end if
        end if
    end for
end if
```

Output: ans

process. More specifically, we determine the class probabilities for each subgraph g' with the same class bias as g with a maximum likelihood estimator. It is the smoothed vector of relative class specific support values, defined as:

$$\alpha_{\mathbf{g}'} = \left(\frac{1 + |\mathbf{k_1}|_1}{|I| + |\mathbf{k}|_1}, \ldots, \frac{1 + |\mathbf{k_{|I|}}|_1}{|I| + |\mathbf{k}|_1} \right) \tag{2}$$

where the $\mathbf{k_i}$ and $\mathbf{k}$ pertain to g', and $|\cdot|_1$ is the one-norm (the sum of the vector elements). Following that, for each tuple $(k_1^j, \ldots, k_{|I|}^j)$ pertaining to g, a probability distribution is determined from this collection of multinomials:

$$p((k_1^j, \ldots, k_{|I|}^j)) = \frac{\sum_{g'} p((k_1^j, \ldots, k_{|I|}^j); \alpha_{\mathbf{g}'})}{\sum_{g'} 1} \tag{3}$$

Finally, the k_i^j values pertaining to g are corrected in a weighted average based on this probability distribution:

$$\overline{\mathbf{k_i}} = \frac{\sum_j k_i^j p((k_1^j, \ldots, k_{|I|}^j))}{\sum_j p((k_1^j, \ldots, k_{|I|}^j))} \tag{4}$$

Then, equation 1 is applied to determine the χ^2 value of g.

5 Algorithm

According to section 4.2, subgraph mining proceeds in two steps: mining the bootstrap sample and – for each subgraph found – looking up support values per class on the out-of-bag instances. The mining step is implemented using a discriminative subgraph mining algorithm of choice. In this work, our in-house algorithm backbone refinement class mining (BBRC) [9] is used. It has high compression potential, which has been shown theoretically and empirically, while retaining good database coverage [10]. This ensures low running times and small result set sizes, if the minimum frequency parameter is set appropriately. In the output, two isomorphic subgraphs will be always represented by the same string identifier. We employ SMARTS, a kind of regular expression to encode molecular fragments as strings. This allows to store results in a hash structure using SMARTS as keys.

The algorithm using $numBoots = N$ bootstrap samples is shown in Algorithm 1. We consider the case where

$numBoots > 1$. Line 4 creates an initially empty hash table to gather results from BBRC mining in line 6. The resulting subgraphs res are matched on the out-of-bag instances in line 7, where the values stored in the hash are the support values per class. It should be stressed that these values correspond to the out-of-bag instances, not the in-bag instances (bootstrap samples). The necessary step of matching the subgraphs on the out-of-bag instances happens inside the BBRC step. On termination of the loop in line 8, each hash entry has at most N support values per class. Post-processing the results is very fast and incurs negligible overhead compared to the graph mining step. It consists of removing subgraphs that (line 11) were not generated often enough by the sampling process (have not enough entries in the hash table, as determined by $minSamplingSupport$), or which (line 12) do not significantly deviate from the overall distribution of classes, as assessed by the χ^2 test (section 4.3.1), with contingency table calculated according to mean (section 4.3.2) or maximum likelihood estimation (section 4.3.3) method.
If $numBoots = 1$, support values per class are obtained directly from a single BBRC run, without any matching.

6 Experiments

Experiments were conducted to assess the quality of the out-of-bag estimation.

6.1 Comparison of p-Values

Three methods were compared by their ability to estimate the discriminative potential of subgraphs, by assessing the deviation between the p-values of subgraphs, as a) estimated by the respective method, and b) obtained by matching the subgraphs to an independent test set

The methods compared are

1. Out-of-bag estimation of p-values by computing chi-square values with Algorithm 1, according to section 4.3.3. Denote this method by MLE.
2. Out-of-bag estimation of p-values by computing chi-square values with Algorithm 1, according to section . Denote this method by MEAN.
3. Single runs of the BBRC algorithm. Denote this method by BBRC.

The process was repeated 100 times, with 50 bootstrap samples for methods 1 and 2. The whole procedure is described in Algorithm 2.

Algorithm 2 Estimation of p-values

Input: $graphDatabase, method$ ▷ method is MLE, MEAN, or BBRC

1: $\mathbf{E_1} = \mathbf{E_2} = []$
2: **for** $i := 1 \rightarrow 50$ **do**
3: $[trainSet, testSet] = splitStratified(graphDatabase, 0.5)$ ▷ Split 50:50
4: $\left[\mathbf{subgraphs}, \mathbf{p^B}\right] = Algorithm1(trainSet, numBoots = 100, \dots)$ ▷ $numBoots = 1$ for BBRC
5: $\mathbf{p'} = match(\mathbf{subgraphs}, test)$
6: $\mathbf{E_1} = \left[\mathbf{E_1}, E_1(\mathbf{p'}, \mathbf{p^B})\right]$
7: $\mathbf{E_2} = \left[\mathbf{E_2}, E_2(\mathbf{p'}, \mathbf{p^B})\right]$
8: **end for**

Output: $\mathbf{E_1}, \mathbf{E_2}$

Line 1 initializes empty vectors that capture the residuals in estimation. Inside the main loop, a stratified split (i.e. proportions of target classes inside each split equal overall proportions) generates a training and a test set of

Assay	Method	E1	E2	E3	E4	E5	Subgraphs
MUL	MLE	-0.0126 (0.0155)	0.0186 (0.0134)	0.0611 (0.0202)	0.8651 (0.1602)	0.9178 (0.1422)	9.66
MUL	MEAN	-0.0094 (0.0101)	0.0152 (0.0105)	0.0697 (0.0242)	0.8481 (0.1646)	0.8942 (0.1405)	10.93
MUL	BBRC	0.0784 (0.0506)	0.0784 (0.0506)	0.1041 (0.0189)	0.5939 (0.1081)	0.4392 (0.1226)	94.95
SAL	MLE	-0.0174 (0.0082)	0.0202 (0.0071)	0.0588 (0.0207)	1.0000 (0.0000)	0.9531 (0.0737)	24.85
SAL	MEAN	-0.0145 (0.0075)	0.0166 (0.0066)	0.0610 (0.0206)	0.9993 (0.0067)	0.9566 (0.0624)	25.82
SAL	BBRC	0.0033 (0.0059)	0.0070 (0.0063)	0.0602 (0.0154)	0.9952 (0.0147)	0.7750 (0.1031)	58.61
MOU	MLE	0.0086 (0.0704)	0.0397 (0.0607)	0.0944 (0.0508)	0.9845 (0.0851)	0.7643 (0.2719)	4.51
MOU	MEAN	0.0161 (0.0593)	0.0353 (0.0526)	0.1050 (0.0504)	0.9868 (0.0716)	0.7024 (0.2761)	5.52
MOU	BBRC	0.1285 (0.1249)	0.1315 (0.1222)	0.1209 (0.0355)	0.8702 (0.1059)	0.4081 (0.1471)	48.17
MCC	MLE	0.0040 (0.0386)	0.0159 (0.0382)	0.0664 (0.0348)	0.9925 (0.0424)	0.8382 (0.1716)	12.71
MCC	MEAN	0.0062 (0.0337)	0.0194 (0.0327)	0.0791 (0.0373)	0.9903 (0.0389)	0.7911 (0.1743)	14.75
MCC	BBRC	0.0636 (0.0511)	0.0663 (0.0485)	0.0971 (0.0225)	0.9417 (0.0524)	0.4673 (0.1137)	79.47
RAT	MLE	-0.0028 (0.0125)	0.0084 (0.0116)	0.0573 (0.0286)	0.9978 (0.0141)	0.9064 (0.1435)	13.00
RAT	MEAN	-0.0017 (0.0115)	0.0092 (0.0109)	0.0659 (0.0317)	0.9899 (0.0365)	0.8720 (0.1439)	14.68
RAT	BBRC	0.0573 (0.0544)	0.0621 (0.0507)	0.0912 (0.0246)	0.9288 (0.0687)	0.4941 (0.1253)	70.01
KAZ	MLE	-0.0000 (0.0000)	0.0000 (0.0000)	0.0181 (0.0067)	1.0000 (0.0000)	0.9932 (0.0155)	26.09
KAZ	MEAN	-0.0000 (0.0000)	0.0000 (0.0000)	0.0182 (0.0069)	1.0000 (0.0000)	0.9948 (0.0140)	25.57
KAZ	BBRC	0.0000 (0.0000)	0.0000 (0.0000)	0.0181 (0.0066)	0.9985 (0.0075)	0.9894 (0.0217)	25.24

Table 2: Bias and Accuracy

	SAL	MCC	RAT	MUL	KAZ	MOU
MLE vs. BBRC E1	tw	tw	tw	tw	tw	tw
MEAN vs. BBRC E1	tw	tw	tw	tw	tw	tw
MLE vs. MEAN E1	tw	w	tw	tw		tw
MLE vs. BBRC E2	tw	tw	tw	tw	tw	tw
MEAN vs. BBRC E2	tw	tw	tw	tw	tw	tw
MLE vs. MEAN E2	tw	w		tw	tw	w
MLE vs. BBRC E3		tw	tw	tw		tw
MEAN vs. BBRC E3		tw	tw	tw		tw
MLE vs. MEAN E3	tw	tw	tw	tw		tw
MLE vs. BBRC E5	tw	tw	tw	tw	tw	tw
MEAN vs. BBRC E5	tw	tw	tw	tw	tw	tw
MLE vs. MEAN E5		tw	tw	tw		tw

Table 3: Significance (p=0.025)

equal size. The training set is treated by the selected method, which returns a vector of subgraphs and a vector of p-values, called $\mathbf{p^B}$ (line 4). The subgraphs are matched on the test set, yielding p-values $\mathbf{p'}$ (line 5). Finally, the residual vectors capture the differences between $\mathbf{p^B}$ and $\mathbf{p'}$ by two different metrics:

1. $E_1 := \frac{1}{n}\sum_{i=1}^{n} p_i^B - p_i'$, to identify bias.
2. $E_2 := \frac{1}{n}\sum_{i=1}^{n} \left| p_i^B - p_i' \right|$, to assess accuracy

Table 2 details the results (mean values across bootstraps)

Table 3 details the results (n=100).

Figure 1 and Figure 2 plot the results.

7 Conclusion

References

[1] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, Jeremy Besson, and Mohammed J. Zaki. ORIGAMI: Mining Representative Orthogonal Graph Patterns. *ICDM 2007. Seventh IEEE International Conference on Data Mining*, pages 153–162, Oct. 2007.

[2] Björn Bringmann, Siegfried Nijssen, and Albrecht Zimmermann. From Local Patterns to Classification Models. In Saso Džeroski, Bart Goethals, and Pance Panov, editors, *Inductive Databases and Constraint-Based Data Mining*, pages 127–154. Springer New York, 2010.

[3] Tom Bylander. Estimating Generalization Error on Two-Class Datasets Using Out-of-Bag Estimates. *Machine Learning*, 48(1-3):287–297, 2002.

[4] Yun Chi, Richard R. Muntz, Siegfried Nijssen, and Joost N. Kok. Frequent Subtree Mining - An Overview. *Fundamenta Informaticae*, 66(1-2):161–198, 2004.

[5] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[6] Christoph Helma, Stefan Kramer, and Luc De Raedt. The Molecular Feature Miner MOLFEA. In *Proceedings of the Beilstein Workshop 2002: Molecular Informatics: Confronting Complexity*, 2003.

[7] Jun Huan, Wei Wang, Jan Prins, and Jiong Yang. SPIN: Mining Maximal Frequent Subgraphs From Graph Databases. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 581–586, New York, NY, USA, 2004. ACM.

[8] Breiman Leo. Out-Of-Bag Estimation. *Technical Report, Statistics Department, University of California*, 1996.

[9] Andreas Maunz, Christoph Helma, and Stefan Kramer. Large-Scale Graph Mining Using Backbone Refinement Classes. In *KDD '09: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 617–626, New York, NY, USA, 2009. ACM.

[10] Andreas Maunz, Christoph Helma, and Stefan Kramer. Efficient Mining for Structurally Diverse Subgraph Patterns in Large Molecular Databases. *Machine Learning*, 83:193–218, 2011.

[11] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[12] Hiroto Saigo, Sebastian Nowozin, Tadashi Kadowaki, Taku Kudo, and Koji Tsuda. gBoost: A Mathematical Programming Approach to Graph Classification and Regression. *Machine Learning*, 75(1):69–89, 2009.

[13] Leander Schietgat, Fabrizio Costa, Jan Ramon, and Luc De Raedt. Effective Feature Construction by Maximum Common Subgraph Sampling. *Machine Learning*, 83(2):137–161, 2011.

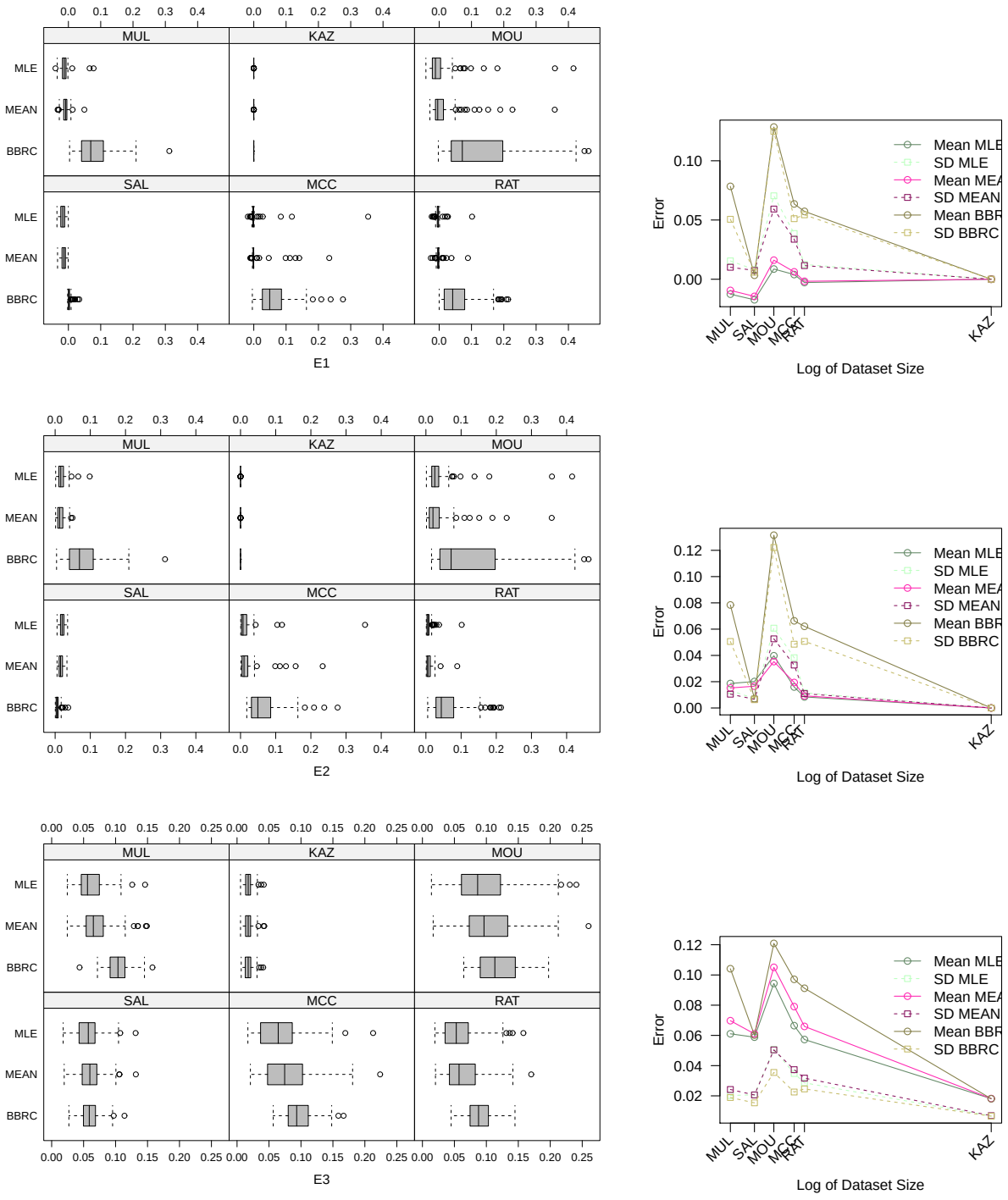

Figure 1: Results

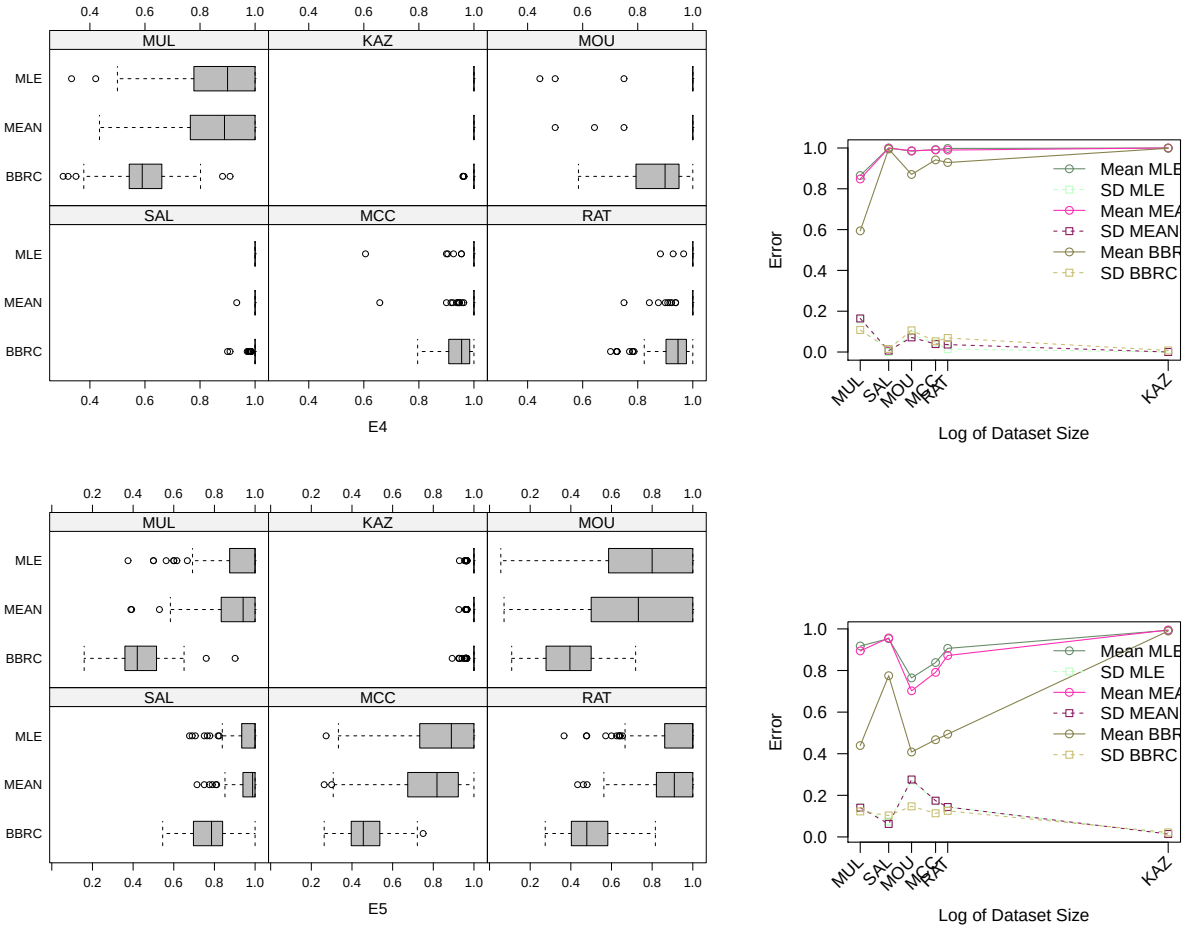

Figure 2: Results