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 database 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 significantly improve the process, compared to single runs of discriminative graph 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 the subgraph associations being statistically validated, the subgraph sets created by the proposed methods are also small, compared to ordinary graph mining, and may thus be beneficial for statistical models, as well as for 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 graph mining on class-labeled graph databases, there is no way of inferring the exact correlations between graph fragments (subgraphs) and the effect symbolized by the target classes, which could serve to identify the set of patterns that primarily lead to the occurrence of the effect, 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 (toxicology), for example in computational models, such as structure activity relationships (QSARs) (citation needed).

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 [4, 2]. Such a workflow is easily embedded when building a bagged predictor based on subgraph descriptors, where it may be obtained as a valuable by-product, as demonstrated by Breiman [2].

This work employs out-of-bag instances with the aim to more precisely estimating the frequencies (support values) of the subgraphs on the target classes that govern the training data than is possible with ordinary discriminative graph mining. 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, which may serve for diagnostic or predictive reasoning. 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 graph mining (section 3), our proposed methods for estimating support values and p-values (section 4), as well as algorithmic implementation (section 4.4). Experiments include validation of support values and p-values on publicly available databases of various sizes and numbers of target classes (section 5), before we draw conclusions

(section 6). The contributions of this work are summarized as follows:

- Repeated discriminative graph mining on bootstrap samples of a class-labeled graph database is proposed as a means to stabilize estimates for subgraph properties, such as support values per class, compared to single runs of discriminative graph mining. An important area of application is the identification of chemical fragments related to chemical reactivity from toxicology.
- Two methods for the estimation of support values are described, where both handle multiple classes. Significance tests are applied to these estimated values 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 ordinary discriminative graph mining, where the effect is larger the smaller the datasets are. We conclude that 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 [2] as well as the generalization error and accuracy of bagged predictors [4]. In the work by Bylander [4], generalization error was well modeled by out-of-bag estimation, and its already small bias could be further reduced 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 [3, 14]. To this end, there is a variety of well-known statistical learning algorithms available "off the shelf" [6, 12]. 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 [7]. However, graph 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 [14, 5, 8].

Subgraph boosting [13] 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 proves to be much more stable, and 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 consists of at least two values. We consider labeled, undirected graphs, i.e. 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 graph $g' = (V', E', \Sigma', \lambda')$ subgraph-isomorphic to g if $V' \subseteq V$ and $E' \subseteq E$ with $V' \neq \emptyset$ and $E' \neq \emptyset$, $\lambda'(v_1) = \lambda(v_2)$

	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.

whenever $v_1 = v_2$, and $\lambda'(e_1) = \lambda(e_2)$ whenever $e_1 = e_2$, for all nodes and edges in g'. Then, graph g' is also referred to as 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.* [5].

4.2 Significance Test

For a given subgraph g, we seek a $|I| \times 2$ contingency table that lists the support values per class in the first column and the overall distribution of target classes in the second column, as in 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 discuss methods to obtain the k_i entries in the table from the recorded support values (section 4.3). 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 4.3). Additionally, $bias(g) := argmax_i(\frac{k_i}{k} / \frac{G^i}{G})$, is determined as the dominant class for g.

4.3 Formulation of Out-Of-Bag Discriminative Graph Mining

We refer to the subset that contains all the graphs $g \in G$ 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 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 instances). Subgraph mining mines the subgraphs on the drawn instances, but looks up the support values per class on the out-of-bag instances, by performing subgraph isomorphism tests (so-called "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 up vectors $\mathbf{k_i}$ across target

classes: $\mathbf{k} = \sum_{i=1}^{|I|} \mathbf{k_i}$. The vectors $\mathbf{k_i}$ are generally sparse, due to the instability of discriminative graph 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, after the end of bootstrapping.

The estimation of support values is performed seperately for each remaining subgraph. Two methods, described in the next sections, 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, the significance test from section 4.2 is run on the estimated support values.

4.3.1 Sample Mean Method

We set the value of 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 of k is the sum of the k_i.

4.3.2 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 the same class bias as g (see section 4.2). 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. In a second step, the subgraphs with the same class bias as g are used to correct g's local frequencies by weighting. This approach has some similarity to the work by Bylander [4], 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 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:

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

Again, we set the value of k_i in Table 1 to $\overline{\mathbf{k_i}}$.

4.4 Algorithm

According to section 4.3, graph 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 graph mining algorithm of choice. In this work, an in-house development, called backbone refinement class mining (BBRC) [10], is used. It has high compression potential, which has been shown theoretically and

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

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

```
1: if numBoots = 1 then
2:     [subgraphs, values] ← BBRC(dataBase, minFrequencyPerSample, alpha)
3: else
4:     hash ← {}
5:     for i := 1 → numBoots do                                   ▷ Parallel Processing
6:         sample, OOB ← drawBsSample(dataBase)
7:         [subgraphs, values] ← BBRC(sample, minFrequencyPerSample, alpha)
8:         insert(hash, match(subgraphs, OOB))
9:     [subgraphs, values] ← []
10:    for subgraph ∈ keys(hash) do
11:        if length(hash[subgraph]) ≥ minSamplingSupport then
12:            [candidateSupportValues, candidateBias] ← method(hash[subgraph])
13:            if candidatePValue ← SignificanceTest(candidateSupportVals) < alpha then
14:                subgraphs ← subgraphs ∪ subgraph
15:                values ← values ∪ [candidateSupportValues, candidatePValue, candidateBias]
```

Output: $[subgraphs, values]$

empirically, while retaining good database coverage [11]. It is characterized by low running times in practice and small result set sizes. In the output, two isomorphic subgraphs are always represented by the same string identifier. We employ SMARTS as identifier, a kind of regular expression to encode molecular fragments as strings. This approach 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 $numBoots > 1$ first. Line 4 creates an initially empty hash table to gather results from BBRC mining in line 7. The resulting subgraphs are matched on the out-of-bag instances in line 8, and the results stored in the hash. On termination of the loop, 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 13) do not significantly deviate from the overall distribution of classes, as assessed by the χ^2 test (section 4.2), with contingency table calculated according to mean (section 4.3.1) or maximum likelihood estimation (section 4.3.2) method. If $numBoots = 1$, support values per class are obtained directly from a single BBRC run, without any matching (see section 5.1). Note that BBRC and matching compute support values, p-values and biases directly from their respective results.

5 Experiments

5.1 Experimental Setup

Three methods were compared by their ability to estimate the discriminative potential of subgraphs, by assessing the deviations between the class specific support values, p-values, and biases of subgraphs, as a) estimated by the respective method, and b) obtained by matching the subgraphs onto an independent test set. The methods compared are

1. Out-of-bag estimation with Algorithm 1, according to section 4.3.2. Denote this method by MLE.
2. Out-of-bag estimation with Algorithm 1, according to section 4.3.1. Denote this method by MEAN.
3. Single runs of the BBRC algorithm. Denote this method by BBRC.

The process was repeated 100 times for methods 1, 2, and 3, with 100 bootstrap samples for methods 1 and 2. Five different error measures to compare the methods were assessed:

Algorithm 2 Error Measures

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

1: $\mathbf{E_1} = \mathbf{E_2} = \mathbf{E_3} = \mathbf{E_4} = \mathbf{E_5} = []$
2: **for** $i := 1 \rightarrow 100$ **do**
3: $[trainSet, testSet] \leftarrow splitStratified(graphDatabase, 0.5)$ ▷ Split 50:50
4: $numBoots \leftarrow 100$; **if** $method = BBRC$ $numBoots \leftarrow 1$
5: $\left[subgraphs, values^B\right] \leftarrow Algorithm\,1(trainSet, numBoots, alpha = 0.05)$ ▷ $numBoots = 1$ for BBRC
6: $\left[subgraphs, values^T\right] \leftarrow match(subgraphs, testSet)$
7: **for** $j := 1 \rightarrow 5$ **do**
8: $\mathbf{E_j} \leftarrow \left[\mathbf{E_j}, errorJ(values^T, values^B)\right]$

Output: $\mathbf{E_1}, \mathbf{E_2}, \mathbf{E_3}, \mathbf{E_4}, \mathbf{E_5}$

1. E_1, the mean of $p_i^B - p_i^T$ over subgraphs, i.e. the bias of p-value errors.
2. E_2, the mean of $\left| p_i^B - p_i^T \right|$ over subgraphs, i.e. the absolute p-values errors.
3. E_3, the mean of $\left(\frac{1}{|I|} \sum_{i=1}^{|I|} \left| \frac{k_i^B}{k^B} - \frac{k_i^T}{k^T} \right| \right)$ over subgraphs, i.e. the relative support value errors.
4. E_4, 1 - the mean of $\delta(bias_i^B, bias_i^T x)$ over subgraphs, i.e. the fraction with wrongly recognized class bias.
5. E_5, 1 - the mean of $\delta(p_i^B \leq \alpha, p_i^T \leq \alpha)$ over subgraphs, i.e. the fraction wrongly recognized as significant.

The whole procedure is described in Algorithm 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 equal size. The training set is treated by the selected method, which returns a vector of subgraphs and a vector $\mathbf{values^B}$ of values, including p, class support, and bias (line 5). The subgraphs are matched on the test set, yielding analogous values $\mathbf{values^T}$ (line 6). Finally, the residual vectors capture the differences between $\mathbf{values^B}$ and $\mathbf{values^T}$ by the error measures E1 - E5.

Six molecular, class labeled datasets were used in the experiments. The first five were drawn from the carcinogenic potency database (CPDB)[a], namely "Combined Carcinogenicity and Mutagenicity" (MUL, 4 classes, 677 compounds), "*Salmonella* Mutagenicity" (SAL, 2 classes, 800 compounds), "Mouse Carcinogenicity" (MOU, 2 classes, 914 compounds), "Multi-Cell Call" (MCC, 2 classes, 1050 compounds), and "Rat Carcinogenicity" (RAT, 2 classes, 1128 compounds). The sixth is the Kazius/Bursi mutagenicity dataset [9] (KAZ, 2 classes, 4069 compounds). The MUL dataset labels consist of all four cross-combinations of the binary MCC and SAL labels, for all chemicals with both values for both labels present.

5.2 Results

Table 2 details the results (mean values across bootstraps) Figure 1 and Figure 2 plot the results.

We first consider E1, E2, and E3. For four of the six datasets (apart from SAL and KAZ), there is a clear improvement when out-of-bag estimation is applied (MLE or MEAN), compared to single runs of discriminative subgraph mining (BBRC). In all these cases, the differences are statistically significant at the $\alpha = 0.025$ level. This means that here, the proposed out-of-bag estimation of support values improves significantly on mining subgraphs directly (BBRC), regardless of the specific method (MEAN or MLE). However, E3 is not significantly different for SAL and KAZ between BBRC and MEAN or MLE. Instead, since SAL's E1 and E2 are significantly lower for BBRC, compared to MLE and MEAN, BBRC works better for this dataset. For KAZ, E1 and E2 are practically zero, and E3 is practically the same for all three methods.

Comparing the sampling methods, for the four "well-behaved" datasets, there is a draw concerning p-value bias (E1), which is, however, low (in absolute values < 0.02). Concerning absolute p-value error (E2), MOU has a quite

[a] `http://potency.berkeley.edu/cpdb.html`

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

high value between 0.035 - 0.04 for both MLE and MEAN, which is only 3 - 4 times lower than that of BBRC. The others remain low, with no clear winner.

However, for E5, the error on significance estimation, MLE performs significantly better than MEAN in all four cases. Interestingly, for E5, MLE and MEAN are both significantly better than BBRC. This means that, despite getting closer to the test values of support and p, it still judges a lot of subgraphs as significant at the 0.05 significance level, that are actually insignificant, or vice versa. E4, the estimation of class bias, is the easiest exercise. However, MLE and MEAN perform significantly better than BBRC for all datasets except KAZ. The effect is most drastic for the multi-class dataset MUL, but also remarkable for MOU and RAT.

Table 2 also gives the mean number of subgraphs generated in the last column. There is a general trend for the smaller numbers of subgraphs generated with the sampling methods with shrinking dataset sizes. This reflects the higher uncertainty due to the lack of data. However, as the line plots in Figures 1 and 2 show, there is a trend towards a higher gap in errors between the sampling methods on the one hand, and BBRC on the other hand with shrinking dataset sizes. It becomes also clear from these plots, that SAL behaves different than the other datasets (as discussed before).

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

[3] 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.

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

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

[6] 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.

[7] 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.

[8] 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.

[9] Jeroen Kazius, Ross McGuire, and Roberta Bursi. Derivation and Validation of Toxicophores for Mutagenicity Prediction. *Journal of Medicinal Chemistry*, 48(1):312–320, 2005.

[10] 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.

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

[12] 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.

[13] 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.

[14] 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.

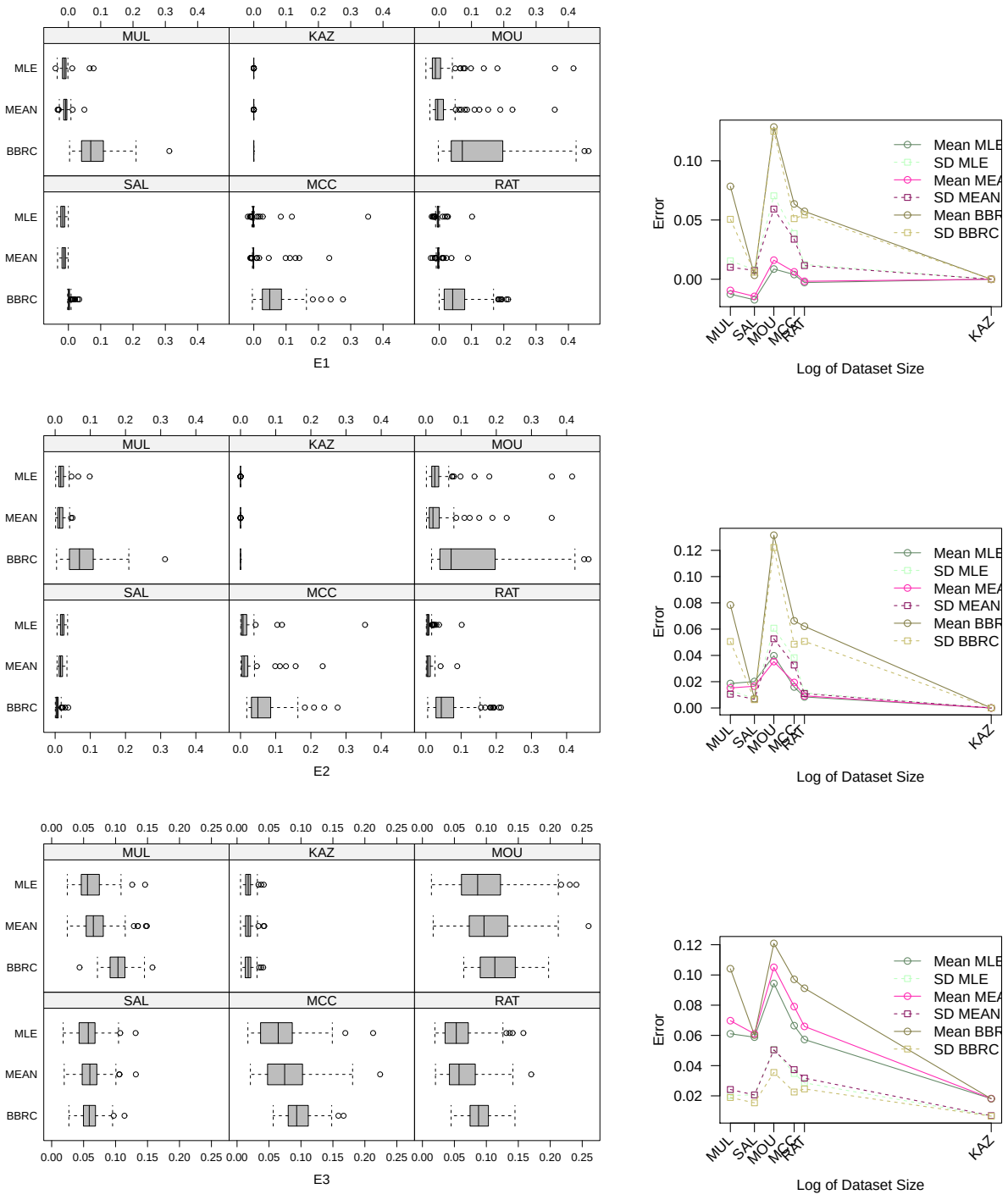

Figure 1: Results

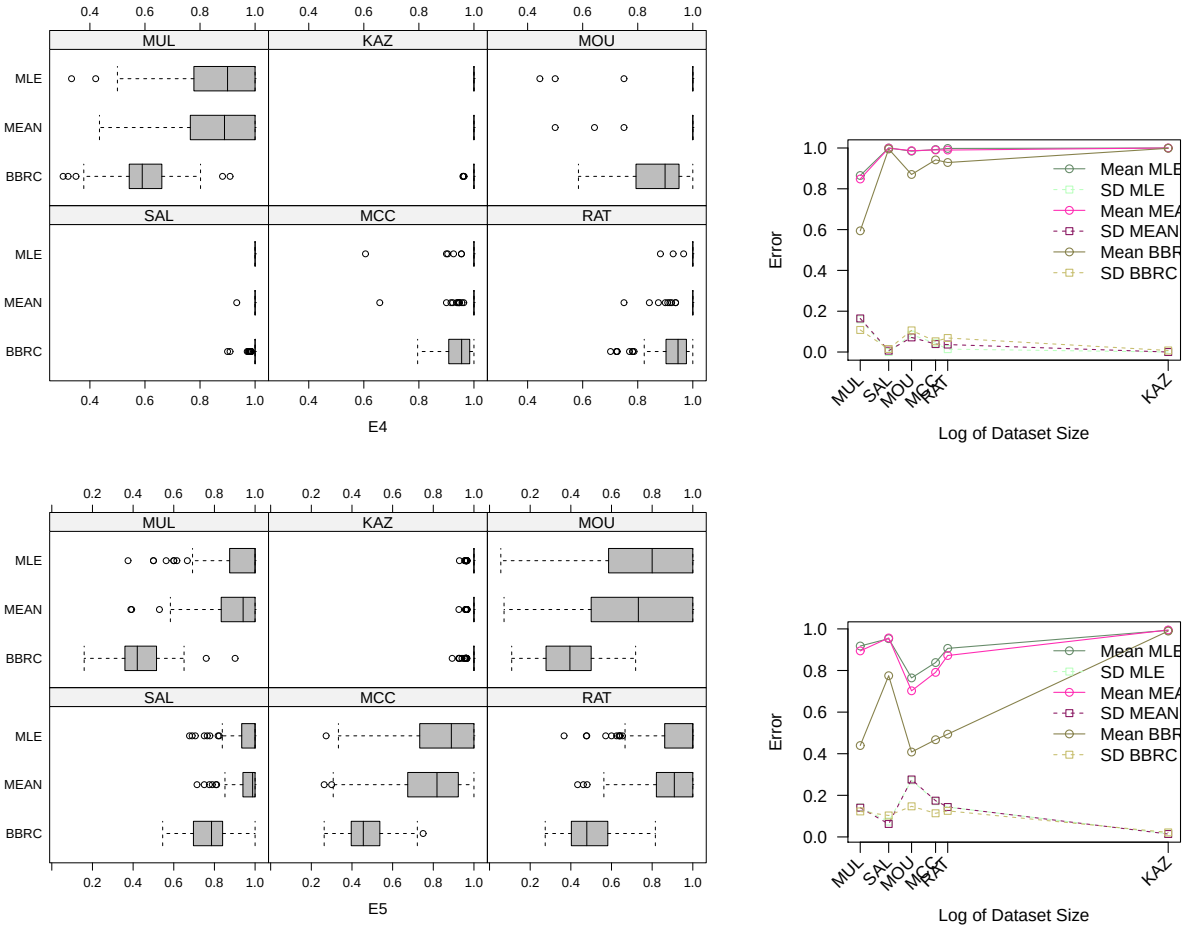

Figure 2: Results