Lazy associative graph classification

Yury Kashnitsky, and Sergei O. Kuznetsov

National Research University Higher School of Economics
Moscow, Russia
{ykashnitsky, skuznetsov}@hse.ru

Abstract. In this paper, we introduce a modification of the lazy associative classification which addresses the graph classification problem. To deal with intersections of large graphs, graph intersections are approximated with all common subgraphs up to a fixed size similarly to what is done with graphlet kernels. We illustrate the algorithm with a toy example and describe our experiments with a predictive toxicology dataset.

Keywords: graph classification, graphlets, formal concept analysis, pattern structures, lazy associative classification

1 Introduction

Classification methods for data given by graphs usually reduce initial graphs to numeric representation and then use standard classification approaches, like SVM [1] and Nearest neighbors with graph kernels [2], graph boosting [3], etc. By doing so, one usually constructs numeric attributes corresponding to subgraphs of initial graphs or computes graph kernels, which usually are also based on the number of common subgraphs of special type. In this paper, we suggest an approach based on weak classifiers in the form of association rules [4] applied in a "lazy" way: not all of the association rules are computed to avoid exponential explosion, but only those that are relevant to objects to be classified. Lazy classification is well studied experimentally [5], here we extend the approach to graphs and propose a uniform theoretical framework (based on pattern structures [6]) which can be applied to arbitrary kinds of descriptions. We show in a series of experiments with data from the Predictive Toxicology Challenge (PTC [7]) that our approach outperforms learning models based on SVM with graphlet kernel [8] and kNN with graphlet-based distance.

The rest of the paper is organized as follows. In Section 2, we give main definitions on labeled graphs, pattern structures, and lazy associative classification. In Section 3, we consider an example. In Section 4, we discuss the results of computational experiments on PTC dataset. In Section 5, we give the conclusion and discuss directions of further research.

2 Main definitions

In this section, we give the definitions of the main concepts used in the paper.

2.1 Labeled graphs and isomorphism

First, we recall some standard definitions related to labeled graphs, see e.g. [9,10,11].

Undirected graph is a pair $G = (V, E)$. Set V is referred to as a set of *nodes* of a graph. Set $E = \{\{v, u\} \mid v, u \in V\} \cup E_0$, a set of unordered elements of V, is called a set of *edges*, and $E_0 \subseteq V$ — is a set of *loops*. If $E_0 = \emptyset$, then G is called a *graph without loops*.

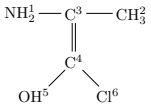
Graph $H = (V_H, E_H)$ is called a *subgraph* of graph $G = (V_G, E_G)$, if all nodes and edges of H are at the same time nodes and edges of G correspondingly, i.e. $V_H \subseteq V_G$ and $E_H \subseteq E_G$.

Graph $H = (V_H, E_H)$ is called an *induced subgraph* of graph $G = (V_G, E_G)$, if H is a subgraph of G, and edges of H are comprised of all edges of G with both nodes belonging to H.

Given sets of nodes V, node labels L_V, edges E, and edge labels L_E, a *labeled* graph is defined by a quadruple $G = ((V, lv), (E, le))$ such that

- $lv \subseteq V \times L_V$ is the relation that associates nodes with labels, i.e., lv is a set of pairs (v_i, l_i) such that node v_i has label l_i,
- $le \subseteq V \times V \times L_E$ is the relation that associates edges with labels, i.e., le is a set of triples (v_i, v_j, l_{ij}) such that edge (v_i, v_j) has label l_{ij}.

Example 1. A molecule structure can be represented by a labeled graph.

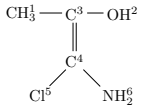
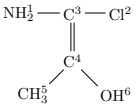


Here $V = \{1, 2, 3, 4, 5, 6\}$, $E = \{(1, 3), (2, 3), (3, 4), (4, 5), (4, 6)\}$,
$lv = \{(1, NH_2), (2, CH_3), (3, C), (4, C), (5, OH), (6, Cl)\}$,
$le = \{(1, 3, 1), (2, 3, 1), (3, 4, 2), (4, 5, 1), (4, 6, 1)\}$, and edge type 1 corresponds to a single bond (ex. HN_2—C) while edge type 2 – to a double bond (ex. $C = C$).

A labeled graph $G_1 = ((V_1, lv_1), (E_1, le_1))$ *dominates* a labeled graph $G_2 = ((V_2, lv_2), (E_2, le_2))$ with given order $\leq$ (e.g. natural, lexicographic) on vertex and edge labels, or $G_2 \leq G_1$ (or G_2 is a *subgraph* of G_1), if there exists an injection $\varphi\colon V_2 \to V_1$ such that it:

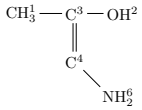
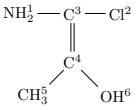
- respects edges: $(v, w) \in E_2 \Rightarrow (\varphi(v), \varphi(w)) \in E_1$,
- fits under labels: $lv_2(v) \leq lv_1(\varphi(v)), (v, w) \in E_2 \Rightarrow le_2(v, w) \leq le_1(\varphi(v), \varphi(w))$.

Two labeled graphs G_1 and G_2 are called *isomorphic* ($G_1 \simeq G_2$) if $G_1 \leq G_2$ and $G_2 \leq G_1$.

Example 2. G_1 :  G_2 : 

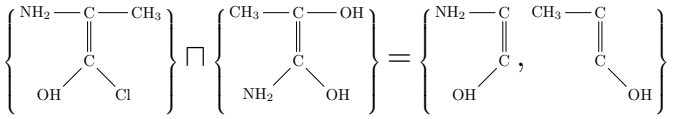
$G_1 \simeq G_2$ as $\exists\varphi : V_2 = \{1,2,3,4,5,6\} \to V_1 = \{1,2,3,4,5,6\} = (6,5,4,3,1,2)$, satisfying the definitions of graph dominance and isomorphism.

An injective function $f : V \to V'$ is called a *subgraph isomorphism* from G to G', if there exists a subgraph of G': $S \leq G'$, such that f is a graph isomorphism from G to S, or $G \simeq S$.

Example 3. G_1 :  G_2 : 

G_1 is subgraph-isomorphic to G_2.

Given labeled graphs G_1 and G_2, a set $G_1 \sqcap G_2 =$
$\{G \mid G \leq G_1, G_2, \quad \forall G_* \leq G_1, G_2 \quad G_* \not\geq G\}$ is called a set of *maximal common subgraphs* of graphs G_1 and G_2. We also refer to $G_1 \sqcap G_2$ as to *intersection* of graphs G_1 and G_2, and to $\sqcap$ – as to *similarity operator* defined on graphs.

Example 4. 

For sets of graphs $\mathcal{G} = \{G_1, \ldots, G_k\}$ and $\mathcal{H} = \{H_1, \ldots, H_n\}$ the similarity operator is defined in the following way:

$\mathcal{G} \sqcap \mathcal{H} = \mathrm{MAX}_{\leq}\{G_i \sqcap H_i \mid G_i \in \mathcal{G}, H_j \in \mathcal{H}\}$

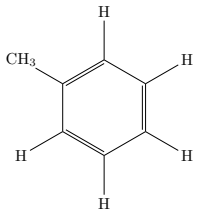
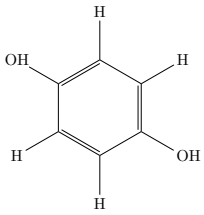
Given sets of labeled graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, we say that a set of graphs $\mathcal{G}_1$ is *subsumed* by a set of graphs $\mathcal{G}_2$, or $\mathcal{G}_1 \sqsubseteq \mathcal{G}_2$, if $\mathcal{G}_1 \sqcap \mathcal{G}_2 = \mathcal{G}_1$.

2.2 Graphlets

Definition 1. *A labeled graph g is called a k-**graphlet** of a labeled graph G if g is a connected induced subgraph of graph G with k nodes [12].*

Definition 2. *A set of labeled graphs $\mathcal{G}^k$ is called a k-**graphlet representation** of a labeled graph G if any $g \in \mathcal{G}$ is a unique (up to subgraph isomorphism) k-graphlet of graph G, i.e*
$\forall g \in \mathcal{G}^k$ graph g is a k-graphlet of G, $\forall g_1, g_2 \in \mathcal{G}$ one does not have $g_1 \leq g_2$.

Definition 3. *k-**graphlet distribution** of a labeled graph G is the set $\{(g_i, n_i)\}$, where g_i is a k-graphlet of G and n_i is the number of k-graphlets in G isomorphic to g_i.*

Example 5. G_1 :  G_2 : 

$\mathcal{G}_1 = \{C-C=C,\ C-C-H,\ C=C-H, C-C-C\}$,
$\mathcal{G}_2 = \{C-C=C,\ C-C-H, C=C-H,\ C-C-O,\ C=C-O,\ C-O-H\}$ – are 3-graphlet representations of graphs G_1 and G_2 correspondingly (with benzene rings comprised of carbon molecules C). 3-graphlet distributions of graphs G_1 and G_2 are given in Table 1.

Table 1. 3-graphlet distributions of graphs G_1 and G_2 (benzene rings are comprised of carbon molecules C).

	CC=C	CCH	C=CH	CCO	C=CO	COH	CCC
G_1	7	8	5	0	0	0	1
G_2	6	4	4	2	2	2	0

Graphlets were introduced in biomedicine and are used to compare real cellular networks with their models. It is easy to demonstrate that two networks are different by simply showing a short list of properties in which they differ. It is much harder to show that two networks are similar, as it requires demonstrating their similarity in all of their exponentially many properties [12].

Graphlet distribution serves as a measure of network local structure agreement and was shown to express more structural information than other metrics such as centrality, local clustering coefficient, degree distribution etc. In [12], they considered all 30 combinations[1] of graphlets with 2, 3, 4 and 5 nodes.

2.3 Pattern structures

Pattern structures are natural extension of ideas proposed in Formal Concept Analysis [13], [6].

Definition 4. *Let G be a set (of objects), let $(D,\ \sqcap)$ be a meet-semi-lattice (of all possible object descriptions) and let $\delta : G \to D$ be a mapping between objects and descriptions. Set $\delta(G) := \{\delta(g)|g \in G\}$ generates a complete subsemilattice $(D_\delta,\ \sqcap)$ of $(D,\ \sqcap)$, if every subset X of $\delta(G)$ has infimum $\sqcap X$ in $(D,\ \sqcap)$.* ***Pattern structure*** *is a triple $(G,\ \underline{D},\ \delta)$, where $\underline{D} = (D,\ \sqcap)$, provided that the set $\delta(G) := \{\delta(g) \mid g \in G\}$ generates a complete subsemilattice $(D_\delta,\ \sqcap)$ [6,11].*

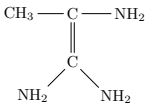
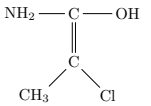
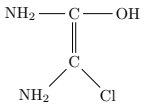
[1] `https://parasol.tamu.edu/dreu2013/OLeary`

Definition 5. ***Patterns*** *are elements of* D*. Patterns are naturally ordered by subsumption relation* $\sqsubseteq$*: given* $c,\ d \in D$ *one has* $c \sqsubseteq d \Leftrightarrow c \sqcap d = c$*. Operation* $\sqcap$ *is also called a* ***similarity*** *operation. A pattern structure* $(G,\ \underline{D},\ \delta)$ *gives rise to the following* ***derivation operators*** $(\cdot)^\diamond$*:*

$$A^\diamond = \prod_{g \in A} \delta(g) \qquad \text{for } A \in G,$$
$$d^\diamond = \{g \in G \mid d \sqsubseteq \delta(g)\} \qquad \text{for } d \in (D,\ \sqcap).$$

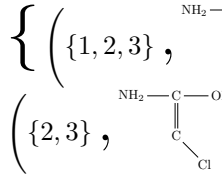
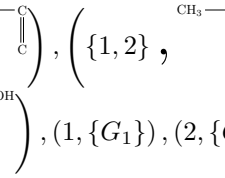
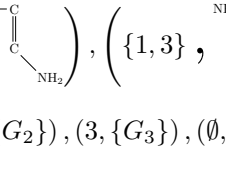
Pairs $(A,\ d)$ *satisfying* $A \subseteq G,\ d \in \underline{D},\ A^\diamond = d$*, and* $A = d^\diamond$ *are called* ***pattern concepts*** *of* $(G,\ \underline{D},\ \delta)$*.*

Example 6. Let $\{1, 2, 3\}$ be a set of objects, $\{G_1, G_2, G_3\}$ – be a set of their descriptions (i.e., graph representations):

G_1 :  G_2 :  G_3 : 

D is the set of all sets of labeled graphs, $\sqcap$ is a graph intersection operator, $\underline{D} = (D,\ \sqcap)$. A set of objects (graphs) $\{1, 2, 3\}$, their "descriptions" (i.e. graphs themselves) $D = \{G_1, G_2, G_3\}$ $(\delta(i) = G_i, i = 1, \ldots, 3)$, and similarity operator $\sqcap$ comprises a pattern structure $(\{1, 2, 3\}, \underline{D}, \delta)$.
$\{1, 2, 3\}^\diamond = \{NH_2 - C = C\}$, because $\{NH_2 - C = C\}$ is the only graph, subgraph-isomorphic to all three graphs $1, 2$, and 3. Likewise,
$\{NH_2 - C = C\}^\diamond = \{1, 2, 3\}$, because graphs $1, 2$, and 3 subsume graph $\{NH_2 - C = C\}$.
$\{1, 2\}^\diamond = \{CH_3 - C = C - NH_2\}$, because $\{CH_3 - C = C - NH_2\}$ is a graph, subgraph-isomorphic to 1, and 2, but not to graph 3. Likewise,
$\{CH_3 - C = C - NH_2\}^\diamond = \{1, 2\}$, because only graphs 1, and 2 subsume graph $\{CH_3 - C = C - NH_2\}$, but graph 3 does not.

Here is the set of all pattern concepts for this pattern structure:

$\Big\{$  ,  ,  , $, (1, \{G_1\}), (2, \{G_2\}), (3, \{G_3\}), (\emptyset, \{G_1, G_2, G_3\}) \Big\}$.

For some pattern structures (e.g., for the pattern structures on sets of graphs with labeled nodes) even computing subsumption of patterns may be NP-hard. Hence, for practical situations one needs approximation tools, which would replace the patterns with simpler ones, even if that results in some loss of information. To this end, we use a contractive monotone and idempotent mapping $\psi : D \to D$ that replaces each pattern $d \in D$ by $\psi(d)$ such that the pattern

structure $(G,\ \underline{D},\ \delta)$ is replaced by $(G,\ \underline{D},\ \psi \circ \delta)$. Under some natural algebraic requirements that hold for all natural projections in particular pattern structures we studied in applications, see [11], the meet operation $\sqcap$ is preserved: $\psi(X \sqcap Y) = \psi(X) \sqcap \psi(Y)$. This property of a projection allows one to relate premises in the original representation with those approximated by a projection. In this paper, we utilize projections to introduce graphlet-based classification rules.

2.4 Lazy associative classification

Consider a binary classification problem with a set of positive examples G_+, negative examples G_-, test examples G_{test}, and a pattern structure $(G_+ \cup\ G_-, \underline{D}, \delta)$ defined on the training set.

Definition 6. *A pattern $h \in D$ is a **positive premise** iff [11]*

$$h^\diamond \cap G_- = \emptyset \text{ and } h^\diamond \cap G_+ \neq \emptyset$$

A positive premise is a subset of the *least general generalization* of descriptions of positive examples, which is not contained in (does not cover) any negative example. A *negative premise* is defined similarly. Various classification schemes using premises are possible, as an example consider the following simplest scheme from [6]: if the description $\delta(g)$ of an undetermined example g contains a positive premise h, i.e., $h \sqsubseteq \delta(g)$, then g is *classified positively*. Negative classifications are defined similarly. If $\delta(g)$ contains premises of both signs, or if $\delta(g)$ contains no premise at all, then the classification is contradictory or undetermined, respectively, and some probabilistic techniques allowing for a certain tolerance should be applied.

Definition 7. Class association rule *(CAR) [5] for a binary classification problem is an association rule in a form $h \to \{+, -\}$, where h is a positive or negative premise, respectively.*

The definition means that for a binary graph classification problem, for instance, we can mine classification association rules in a form $\{g_i\} \to \{+, -\}$, i.e. if a test graph subsumes a subgraph g_i, that is common only to positive (negative) training examples, it is therefore classified as positive (negative). We elaborate this idea in the next subsection. As there might be lots of such CARs, we might come up with a single classification rule taking into account these CARs. For instance, we can count all positive and negative CARs for each test object and classify it with a majority voting procedure. Of course, the idea is easily generalized to multi-label classification problem. The described classification schemes are explored in [5].

Another advantage of the lazy classification framework is its obvious parallelization. Suppose there are K processors. If we consider classification of an unlabeled object we can divide the training set into K separate subsets. Then, for each subset we perform intersections between the labeled objects with the unlabeled one to be classified. After all unfalsified intersections are found we can go on to the classification phase which involves voting based on those intersections.

2.5 Graphlet-based lazy associative classification

In this subsection, we combine the ideas of pattern structures and their projections, graphlets, and lazy associative classification, and introduce our algorithm. First, we recall the definition of k-projection producing all graphs with less than or equal to k nodes.

Definition 8. *Given a graph pattern structure $(G, \underline{D}, \delta)$, we call $\psi_k(G) = \{H_i = ((V_i, lv_i), (E_i, le_i)) \mid H_i \leq G, H_i$ is connected, $|V_i| \leq k\}$ a k-**projection**, defined for graph descriptions G.*

Obviously, this operator is a projection, i.e. contractive, monotone, and idempotent function.

Definition 9. *Given a graph pattern structure $(G, \underline{D}, \delta)$, k-**graphlet derivation operator** $\delta_k = \bigcup_{1 \leq l \leq k} \psi_l \circ \delta$ takes an object g described by graph G and produces all l-graphlets of G for $l = 1, \ldots k$.*

Example 7. For object 1 with "graph description" G_1 from example 5 $\delta_3(1)$ is the set of all 1-,2-, and 3-graphlets of graph 1:
$\delta_3(1) = \{C,\ H,\ C-C,\ C=C,\ C-H,\ C-C=C,\ C-C-H,\ C=C-H,\ C-C-C\}$. To clarify, here $\delta(1) = \{G_1\}, \delta_3(1) = \psi_3(\delta(1)) = \psi_3(G_1) = \{H_i = ((V_i, lv_i), (E_i, le_i)) \mid H_i \leq G_1, |V_i| \leq 3\}$.

Definition 10. *Given k-graphlet representations $\mathcal{G}_1^k$ and $\mathcal{G}_2^k$ of labeled graphs G_1 and G_2, the intersection $\mathcal{G}_1^k \sqcap_k \mathcal{G}_2^k$ is called k-**graphlet intersection** of G_1 and G_2. The $\sqcap_k$ operator is further called k-**graphlet similarity operator**.*

Example 8. For graphs 1 and 2 with "graph descriptions" G_1 and G_2 from example 5 $G_1 \sqcap_3 G_2 = \{C,\ H, C-C, C=C, C-H, C-C=C, C-C-H, C=C-H\}$ is the set of all common 1-, 2-, and 3-graphlets of graphs 1 and 2.

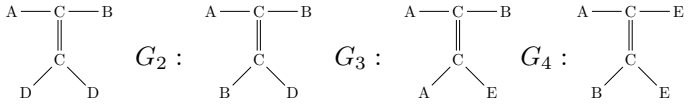
Here are the main steps of our algorithm:

1. All k-graphlet intersections of test examples and positive training examples are computed: $h_+ = G_{tr} \sqcap_k G_+$;
2. Each intersection h_+ is tested on subsumption by negative training examples. If some of them subsumes h_+, then this intersection is *falsified*. Otherwise, h_+ gives a vote for positive classification of the test example G_{tr};
3. The same procedure is done for each intersection of G_{tr} with negative examples;
4. Test example G_{tr} is classified according to the weighted majority rule where each unfalsified intersection is given a weight equal to its cardinality (the cardinality of the corresponding set of graphs).

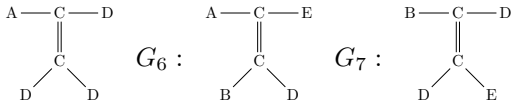
3 A toy example

We illustrate the principle of our method with a toy example. Let us consider the following training and test sets comprised of molecular descriptions of toxic (G_1 – G_4) and non-toxic (G_5 – G_7) chemical compounds. The task is to build a discriminative classifier able to determine whether the objects from the test set ($G_8 - G_{11}$) are toxic or not. The main steps of the algorithm, described in the previous section, are briefly illustrated with Tables 2 and 3. First, we build 3-graphlet intersections of test and training examples (we use only graphlets with 3 nodes for the purpose of illustration). Then, a "+" or "—" sign with cardinality of intersection is put in Table 3 if this intersection is not subsumed by any example of the opposite class. Otherwise, the counter-example subsuming this intersection is given.

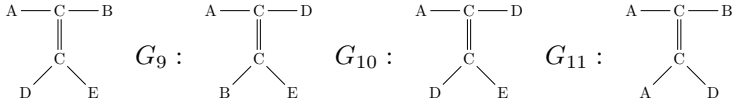
Positive examples:



Negative examples:



Test examples:



3-graphlet intersections of training and test examples are given in Table 2. For instance, graphs G_1 and G_8 have 4 common 3-graphlets: A–C–B, A–C=C, B–C=C, and C=C–D. In this simple case, we do not differentiate between a single and a double bond (e.g., ACC here stands for A–C=C without ambiguity).

Further, Table 3 summarizes the procedure. For instance, a '+4' sign for graphs G_1 and G_8 means that all common 3-graphlets of G_1 and G_8 (i.e., A–C–B, A–C=C, B–C=C, and C=C–D) are not subgraph-isomorphic to any of the negative examples G_5 – G_7 altogether at the same time. Thus, this intersection "gives a vote" of weight 4 (the cardinality of the mentioned set of graphlets) for positive classification of G_8. On the contrary, all common 3-graphlets of G_4 and G_8 (A–C=C, B–C=C, and C=C–E) are altogether subgraph-isomorphic to negative example G_6, therefore, the intersection of G_4 and G_8 doesn't "give a vote" for positive classification of G_8.

Thus, molecules G_8 and G_{11} are classified as toxic, G_9, G_{10} are classified as non-toxic.

4 Experiments

The proposed algorithm was tested with the 2001 Predictive Toxicology Challenge dataset in comparison with SVM with graphlet kernel and k-Nearest-

Table 2. All common 3-graphlets of test ($G_8 - G_{11}$) and training examples.

	G_8	G_9	G_{10}	G_{11}
G_1	ACB, ACC, BCC, CCD	ACC, BCC, CCD	ACC, CCD	ACB, ACC, BCC, CCD
G_2	ACB, ACC, BCC, CCD	ACC, BCC, CCD	ACC, CCD	ACB, ACC, BCC, CCD
G_3	ACB, ACC, BCC, CCE	ACC, BCC, CCE	ACC, CCE	ACB, ACC, BCC
G_4	ACC, BCC, CCE	ACC, BCC, BCE, CCE	ACC, CCE	ACC, BCC
G_5	ACC, CCD	ACC, ACD, CCD	ACC, ACD, CCD	ACC, ACD, CCD
G_6	ACC, BCC, CCD, CCE	ACC, BCC, CCD, CCE	ACC, CCD, CCE	ACC, BCC, CCD
G_7	BCC, CCD, CCE, DCE	BCC, CCD, CCE	CCD, CCE, CDE	BCC, CCD

Table 3. Lazy classification table

	G_1	G_2	G_3	G_4	G_5	G_6	G_7	Score	Class
G_8	+4	+4	+4	G_6	G_1	–4	–4	4:0	+
G_9	G_6	G_6	G_6	+4	–3	–4	–3	0:6	—
G_{10}	G_5	G_5	G_6	G_6	–3	–3	–3	0:9	—
G_{11}	+4	+4	+3	G_6	–3	G_1	G_1	8:0	+

Neighbor with graphlet-based Hamming distance. SVM classifiers are considered to be good benchmarks for graph classification problem [8]. We implemented a Scikit-learn [14] version of Support Vector Classifier with graphlet kernel and graphlets having up to 5 nodes. We also adopted a k-Nearest-Neighbor for graph classification problem by defining a Hamming distance between two graphs (0 if two objects have a certain graphlet in common, 1 otherwise). For instance, for two graphs from example 5 in case of graphlets with up to 3 nodes this distance is equal to 7 (G_1 subsumes graphlet $C-C-C$ not subsumed by G_2, while G_2 subsumes graphlets $\{O,\ C-O,\ O-H,\ C-C-O,\ C=C-O,\ C-O-H\}$ not subsumed by G_1).

The training set is comprised of 417 molecular graphs of chemical compounds with indication of whether a compound is toxic or not for a particular sex and species group out of four possible groups: {mice, rats} × {male, female}. Thus, 4 separate sets were built for male rats (MR, 274 examples, 117 are toxic for male rats, 157 are non-toxic), male mice (MM, 266 examples, 94 are positive, 172 are negative), female rats (FR, 281 examples, 86 are positive, 195 are negative) and female mice (FM, 279 examples, 108 are positive, 171 are negative).

We run 5-fold cross-validation for each group (MR, MM, FR, FM) and compared average classification metrics for each fold. The results for male rats are presented in Table 4 (we got similar results for other groups).

The parameters for SVM and kNN classifiers were tuned through the process of GridSearch cross-validation[2]. The 'K nodes' parameter determines the maximum number of nodes in graphlet representation of graphs, i.e. when it is equal to 4, all graph are approximated with their 4-graphlet representation, or all unique (in the sense of isomorphism) graphlets with up to 4 nodes.

As we can observe, graphlet-based lazy associative classification is reasonable with at least 3-graphlet descriptions. In case of 2-graphlet descriptions the

[2] `http://scikit-learn.org/stable/modules/grid\_search.html`

Table 4. Experimental results for the male rats group. "GLAC" stands for "Graphlet-based lazy associative classification", "SVM" here denotes "Support Vector Machine with graphlet kernel" "kNN" here stands for a k-Nearest-Neighbor classifier with Hamming distance.

	K nodes	Accuracy	Precision	Recall	F-score	Time (sec.)
GLAC	2	0.36	0.32	0.33	0.32	5.78
	3	0.68	0.83	0.68	0.75	17.40
	4	0.59	0.57	0.62	0.59	65.72
	5	0.55	0.7	0.62	0.66	196.03
SVM	2	0.45	0.15	0.33	0.21	1.54
	3	0.52	0.35	0.35	0.35	9.03
	4	0.41	0.27	0.28	0.28	61.31
	5	0.36	0.24	0.25	0.24	295.89
kNN	2	0.45	0.15	0.33	0.21	3.35
	3	0.34	0.21	0.23	0.22	15.75
	4	0.48	0.31	0.32	0.31	73.38
	5	0.45	0.30	0.31	0.30	211.58

algorithm often refuses to classify test objects, because 2-graphlet intersections of positive and test objects are falsified by negative objects and vice versa. But 3-graphlet descriptions are optimal for this method as the model is probably overfitted in case of 4- and 5-graphlet descriptions.

5 Conclusion

In this paper, we have proposed an approach to graph classification based on the combination of graphlets, pattern structures and lazy classification. The key principle of lazy classification is that one does not have to produce the whole set of classification rules whatever they are. Instead, one generates those rules that allow one to classify the current test object. The framework favors the complex structure of objects as soon as the algorithm does not require a training phase.

We have carried out a number of experiments in molecule classification within the proposed lazy classification framework. We compared classification performance of our method and SVM with graphlet kernel and KNN with graphlet-based distance. The reason for such a choice is that SVM classifiers are considered to be good benchmarks for graph classification problem, while kNN is a famous lazy classification method.

In our experiments graphlet-based lazy classification - following the same learning curve as the other methods - shows better classification performance compared to the classical methods in case of molecule toxicology prediction problem. Further, we plan to investigate the overfitting problem for our algorithm, in particular, the dependency of classification metrics on the number of considered nodes in graphlets. Other types of descriptions and a parallel version of our algorithm are also promising directions of study.

References

1. Corinna Cortes and Vladimir Vapnik, "Support-Vector Networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sept. 1995.
2. S. V. N. Vishwanathan, Nicol N. Schraudolph, Risi Kondor, and Karsten M. Borgwardt, "Graph Kernels," *J. Mach. Learn. Res.*, vol. 11, pp. 1201–1242, Aug. 2010.
3. Hiroto Saigo, Sebastian Nowozin, Tadashi Kadowaki, Taku Kudo, and Koji Tsuda, "GBoost: a mathematical programming approach to graph classification and regression," *Machine Learning*, vol. 75, no. 1, pp. 69–89, 2009.
4. Rakesh Agrawal and Ramakrishnan Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," in *Proceedings of the 20th International Conference on Very Large Data Bases*, San Francisco, CA, USA, 1994, VLDB '94, pp. 487–499, Morgan Kaufmann Publishers Inc.
5. Adriano Veloso, Wagner Meira Jr., and Mohammed J. Zaki, "Lazy Associative Classification," in *Proceedings of the Sixth International Conference on Data Mining*, Washington, DC, USA, 2006, ICDM '06, pp. 645–654, IEEE Computer Society.
6. Bernhard Ganter and Sergei Kuznetsov, "Pattern Structures and Their Projections," in *Conceptual Structures: Broadening the Base*, Harry Delugach and Gerd Stumme, Eds., vol. 2120 of *Lecture Notes in Computer Science*, pp. 129–142. Springer, Berlin/Heidelberg, 2001.
7. Christoph Helma and Stefan Kramer, "A Survey of the Predictive Toxicology Challenge 2000-2001," *Bioinformatics*, vol. 19, no. 10, pp. 1179–1182, 2003.
8. Nino Shervashidze, S. V. N. Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten M. Borgwardt, "Efficient graphlet kernels for large graph comparison," *Journal of Machine Learning Research - Proceedings Track*, vol. 5, pp. 488–495, 2009.
9. Reinhard Diestel, *Graph Theory (Graduate Texts in Mathematics)*, Springer, August 2005.
10. Horst Bunke and Kim Shearer, "A Graph Distance Metric Based on the Maximal Common Subgraph," *Pattern Recogn. Lett.*, vol. 19, no. 3-4, pp. 255–259, Mar. 1998.
11. Sergei O. Kuznetsov, "Scalable Knowledge Discovery in Complex Data with Pattern Structures," in *PReMI*, Pradipta Maji, Ashish Ghosh, M. Narasimha Murty, Kuntal Ghosh, and Sankar K. Pal, Eds. 2013, vol. 8251 of *Lecture Notes in Computer Science*, pp. 30–39, Springer.
12. Natasa Przulj, "Biological network comparison using graphlet degree distribution," *Bioinformatics*, vol. 23, 2003.
13. Bernhard Ganter and Rudolf Wille, *Formal Concept Analysis: Mathematical Foundations*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition, 1997.
14. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.