Clustering Relational Data based on Randomized Propositionalization

Grant Anderson and Bernhard Pfahringer

Department of Computer Science, University of Waikato, Hamilton, New Zealand

Abstract. Clustering of relational data has so far received a lot less attention than classification of such data. In this paper we investigate a simple approach based on randomized propositionalization, which allows for applying standard clustering algorithms like KMeans to multi-relational data. We describe how random rules are generated and then turned into boolean-valued features. Clustering generally is not straightforward to evaluate, but preliminary experimental results on a number of standard ILP datasets show promising results. Clusters generated without class information usually agree well with the true class labels of cluster members, i.e. class distributions inside clusters generally differ significantly from the global class distributions. The two-tiered algorithm described shows good scalability due to the randomized nature of the first step and the availability of efficient propositional clustering algorithms for the second step.
Keywords: clustering, propositionalization, randomization

1 Introduction

Clustering is a process by which instances are divided into groups, appropriate groupings being determined by some distance measure. Relational clustering applies this process to relational data. Such distance measures are more complex to determine for relational data than for propositional data, as relational data cannot easily be fitted to a Euclidean framework, and therefore use inter-instance similarity to determine clusters. RDBC [8], for example, uses the distance measure of RIBL [3, 5], which recursively compares the relational elements of the data until features can be propositionally compared. [6] describes a metric for terms and clauses, and relational distance measures can also be derived from relational kernels [4, 13].

Clustering of relational data has so far received a lot less attention than classification of such data. One approach based on a relational clustering tree as a variant of the relational tree learner Tilde is described in [1]. In this paper, we present a two-tiered approach to relational clustering that obviates the need for a relational distance measure, allowing us to apply standard propositional clustering algorithms to multi-relational data. In the first step we propositionalize[9] the relational data using randomly generated first-order rules (similar to the relational association rules generated by Warmr [7]), which are then converted into boolean features, based on their coverage. The generation process restricts

the rules to be within certain coverage minima and maxima to avoid overly specific or general rules, respectively. The rules are also generated in a manner that encourages even coverage across the data. In the second step, the resulting propositional dataset is clustered using a standard propositional clusterer such as KMeans [10]. The next section will detail the algorithm, Section 3 reports on experiments, and the final section summarizes and outlines future work.

2 Method

The RRC (Randomized Relational Clustering) algorithm comprises two tiers: a first level which generates random rules aiming to cover all examples as uniformly as possible, and a second level which turns these rules into boolean features for a propositional representation which acts as input for any propositional clustering algorithm. The experiments reported below employed standard KMeans using standard Euclidean distance.

Random rules are generated in the following way: at each stage a literal or test is chosen uniformly at random with the following restriction: for a literal exactly one variable/argument must be an already present variable, all others will be new variables. Tests on the other hand may not add any new variables. Tests include the usual equal and not-equal comparisons to other variables or theory constants, as well as range comparisons for numeric arguments.

To ensure that the randomly generated rules actually allow for clustering, some constraints are imposed on the generation process: rules must cover a user-defined minimum number of examples and may not cover more than a user-defined maximum, either. This prevents against both very specific and also against very general rules; worst cases would be universally true rules or rules covering to just a single example. These constraints operate on individual rules. Furthermore examples should be covered by roughly equal numbers of rules. This "uniformity of coverage" is a ruleset-level constraint. To obtain such uniform coverage, random rules are generated in small batches, then the most uniformity-preserving non-zero subset of such a batch of rules is added to the current ruleset. The basic algorithm for RRC is given in Algorithm 1.

The complexity of RRC is the sum of the complexity of both stages. Usually, when using propositionalization in ILP, the propositionalization stage dominates the total complexity, and this is true for RRC as well. Even though generating a random rule is extremely fast, its coverage still has to be determined both for checking the coverage constraints and uniformity of coverage, as well as to generate the propositional data-set. In the worst case this coverage computation can be exponential, even for a single rule. The complexity of propositional clustering algorithms on the contrary is often linear or quadratic at worst. Still, in practice we find that RRC enjoys very acceptable runtimes, and at times, especially for larger rulesets, the propositional clustering can actually dominate over the propositionalization stage.

Algorithm 1 Pseudocode for the RRC algorithm

```
while Number of rules in ruleset is less than the minimum do
  while Number of rules in batch is less than the minimum do
    Generate a Rule
    if Rule is within coverage constraints then
      Add Rule to rule batch
    end if
  end while
  Calculate the most uniform subset of rules in the current rule batch
  Add those rules to the ruleset
end while
use ruleset to generate boolean-valued propositional dataset
apply any propositional clustering algorithm
```

3 Experiments

An evaluation of RRC on several datasets has been conducted, always generating random rules for the full dataset, and then clustering the resulting propositional data using KMeans [10] using Euclidean distance. The following standard ILP datasets were used: Mutagenesis (with and without regression-unfriendly instances), Musk1, Cancer (using only the Atom and Bond tables) and Diterpenes. For Mutagenesis and Cancer we only use low-level structural information as represented by atoms and bonds, we do not include any global properties (e.g. lumo or logP) nor predefined functional groups. In the case of the Diterpenes, three versions were generated: all pairwise combinations of the three largest classes called 3, 52, and 54.

To study the influence of the number of clusters that number was varied from 2 up to about 50. As there exists no single universally agreed upon measure for clustering quality and as we actually have true class labels available for all datasets, which have not been used in any way inside RRC, one measure of cluster quality is the agreement of clusters with classes. Clearly one would expect better accuracies with more clusters, as it should be easier to find smaller class-pure clusters than larger ones. One caveat here is that when taking the majority class of each cluster as its "label", clusters with only one example will automatically be correct (at least for clustering algorithms like KMeans, which do not allow for fuzzy assignment of examples to multiple clusters). Given the sizes of all datasets except Musk1, even when allowing for 50 clusters, usually no single example clusters are generated. If there are some such single example clusters present, they can safely be ignored for computing accuracies. The results of this evaluation for all seven datasets are depicted in Figure 1. The trends visible follow our expectations, higher number of clusters lead to smaller error rates. Furthermore these error rates seem to level out at a reasonable number of clusters. It should be noted that the final error rates are actually quite competitive to what has been reported in the literature for relational classifiers on these datasets.

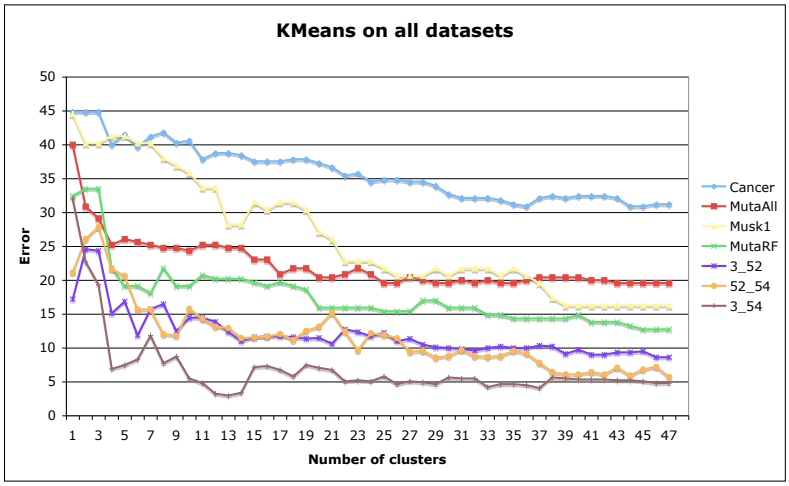


Fig. 1. *Error rates for all data-sets and different number of clusters.*

To study the sensitivity of RRC to its user-settable parameters, we have performed a series of experiments varying both the number of rules used as well as the minimum and maximum coverage values for single rules. Here we present only results for one problem: class 3 versus class 52 of the Diterpenes. Qualitatively all other sets show similar behavior, but with different optimal parameter values. Figures 2 and 3 show the behavior of RRC for larger and larger number of rules. Numbers up to 10 are clearly not sufficient, as the error-levels quickly flatten out, not improving for larger number of clusters. Larger sizes show better performance. For this specific dataset 100 rules seem to perform slightly better than 200 rules, indicating that overfitting could be an issue for RRC.

Figure 4 shows the effects of varying coverage limits. Again for this dataset having too low a minimum, which allows for very specialized rules to be created, seems to hurt performance. Rules that have to cover at least 10% and not more than half of all the data perform best. But the default setting of 25% to 75% does well, too.

To get a better insight into the quality of clustering, Figure 5 depicts the class distribution for a particular 20 cluster partition of the 188 regression-friendly compounds from the mutagenicity dataset using only 10 random rules. Still, 8 of the 20 clusters are class-pure, all for the active class, though. Two of the random features generated are:

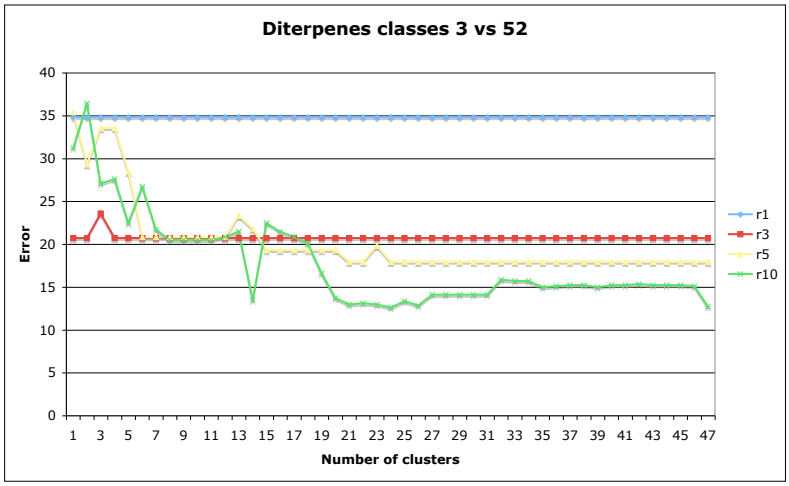


Fig. 2. *Error rates for very small random rule sets.*

```
active(MolId) :-
       bond(MolId,_,_,2),
       atom(MolId,_,_,27,_).

active(MolId) :-
       atom(MolId,AtomId1,_,QuantaType1,Charge),
       Charge >= 0.172,
       Atom(MolId,AtomId2,_,QuantaType2,_),
       AtomId2 != AtomId1,
       QuantaType1 == QuantaType2.
```

Respectively, they represent compounds with at least one double bond plus an atom of Quanta type 27, as well as compounds with two distinct atoms of the same Quanta type, where one must have a charge of at least 0.172. Picking e.g. cluster number 4, which comprises four examples of the same class, their boolean feature values are:

```
1  2  3  4  5  6  7  8  9 10
f, t, f, f, t, t, t, t, t, t  example1
f, t, f, f, t, t, t, t, t, t  example2
f, t, f, f, t, t, f, t, t, t  example3
```

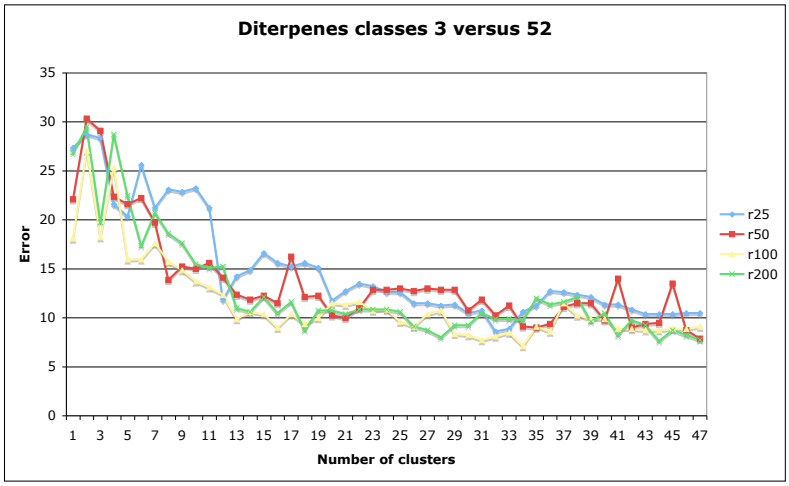


Fig. 3. *Error rates for larger random rule sets.*

```
f, t, f, f, t, t, t, t, t, t  example4
```

Notice that these four examples are almost identical under this propositionalization, with only one exception for attribute 7 for example3. Table 1 shows the structure formulas for these four compounds, and indeed three of the four are almost identical, only one nitro-group is positioned differently for each of them, and the fourth compound (example3, third from the left) is also very similar in structure to the others.

4 Summary and Future Work

Relational clustering has not received much attention in ILP so far. This paper has described a two-tiered approach based on randomized propositionalization and an arbitrary propositional clustering algorithm. The experimental results reported above look promising. This research will be extended into various directions. First and foremost comparisons to more standard clustering approaches are needed. Both relational distances [5, 13] and kernels for relational data [4] could be used together with clustering algorithms like KernelKMeans [2]. The rule generation process could be replaced by either a relational association rule finder like WarmR [7], systems like RSD[14], or class-blind variants of relational rule learners like Foil [12] or Progol [11]. Lastly, this approach to class-blind

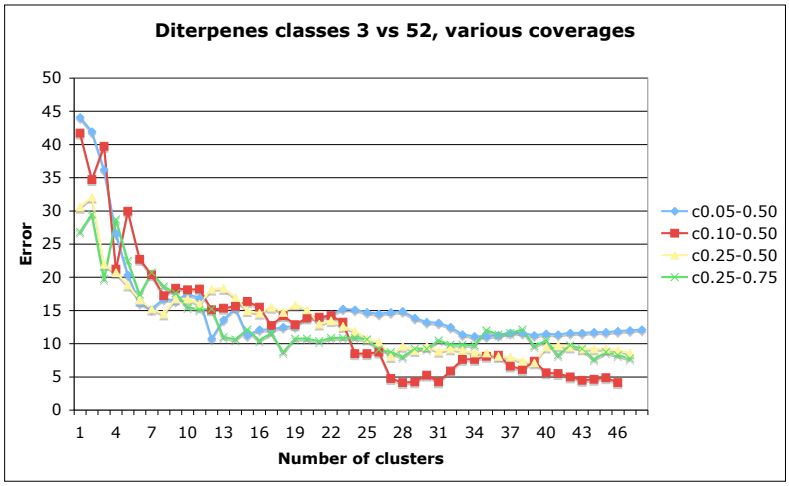
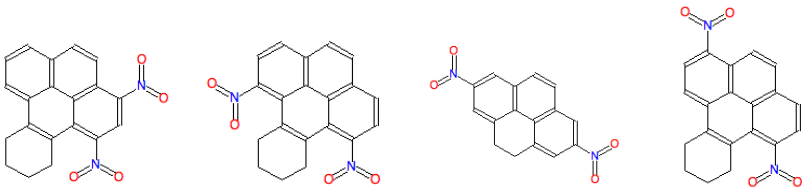


Fig. 4. *Error rates for different minimum and maximum coverage rates.*

propositionalization might also be useful for classification problems, especially in lazy or semi-supervised settings, as the generation process guarantees good coverage of all data including the unlabeled portion of it. More work is also needed to determine the suitability of this approach for different types of data. All experiments reported here used data-sets comprising distinct examples with no linkage between single examples. There are applications where links between examples can carry essential information. RRC will have to be evaluated for such data as well.

Table 1. The four, all active compounds of cluster 4



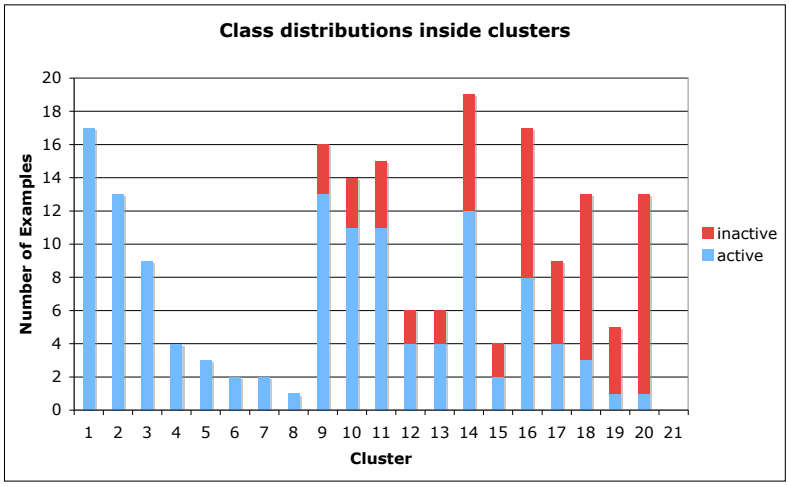


Fig. 5. *Regression-friendly mutagenesis: class distributions across 20 clusters.*

References

[1] H. Blockeel, L. De Raedt, and J. Ramon. Top-down induction of clustering trees. Proceedings of the 15th International Conference on Machine Learning, pages 55-63, 1998.

[2] Francesco Camastra and Alessandro Verri . A Novel Kernel Method for Clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence Vol. 27 No. 5 (2005), 801–804

[3] W. Emde and D. Wettschereck. Relational instance-based learning. Proceedings of the 13th International Conference on Machine Learning, pages 122-130, 1996.

[4] T. Gärtner, J. W. Lloyd, P. A. Flach. Kernels and Distances for Structured Data. Machine Learning 57 (2004).

[5] Tamás Horváth, Stefan Wrobel and Uta Bohnebeck. Relational Instance-Based Learning with Lists and Terms. Machine Learning 43 (2001), 53–80.

[6] A. Hutchinson. Metrics on terms and clauses. Proceedings of the 9th European Conference on Machine Learning, pages 138-145, 1997.

[7] Ross D. King, Ashwin Srinivasan and Luc Dehaspe Warmr: A Data Mining Tool for Chemical Data Journal of Computer Aided Molecular Design, 15:173–181, 2001.

[8] Mathias Kirsten and Stefan Wrobel. Relational Distance-Based Clustering. Proceedings of the 8th International Workshop on Inductive Logic Programming (1998), 261–270.

[9] Kramer S., Lavrac N., Flach P. Propositionalization Approaches to Relational Data Mining. Relational Data Mining, Springer, 2001.

[10] J. B. MacQueen. Some Methods for classification and Analysis of Multivariate Observations. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability (1967), 1:281-297
[11] S. Muggleton. Inverse entailment and Progol. New Generation Computing, Special issue on Inductive Logic Programming, 13(3-4):245–286, 1995.
[12] Quinlan, J.R. Learning logical definitions from relations. Machine Learning 5 (1990), 239-266.
[13] Adam Woźnica, Alexandros Kalousis and Melanie Hilario. Kernels over Relational Algebra Structures. PAKDD 2005, 588-598.
[14] Zelezny F., Lavrac N. Propositionalization-Based Relational Subgroup Discovery with RSD. Machine Learning 62(1-2):33-63, 2006.