Direct Mining of Discriminative and Essential Frequent Patterns via Model-based Search Tree

How to find good features from semi-structured data for classification

Wei Fan, Kun Zhang, Hong Cheng, Jing Gao, Xifeng Yan, Jiawei Han, Philip S. Yu, Olivier Verscheure (KDD '08)

Presented by Marina Danilevsky

March 4th, 2010

Acknowledgment:
Wei Fan et al (KDD'08)

Pattern-Based Graph Classification

- For each graph G, extract graph substructures

$$F = \{g_1, g_2 \ldots g_n\}$$

- Represent graph in the dataset into a feature vector

$$X = \{x_1, x_2 \ldots x_n\}$$

where x_i is the frequency of the *i*-pattern in *G*. Each vector is associated with a class label.

- Build a classification model.

Pattern-Based Graph Classification

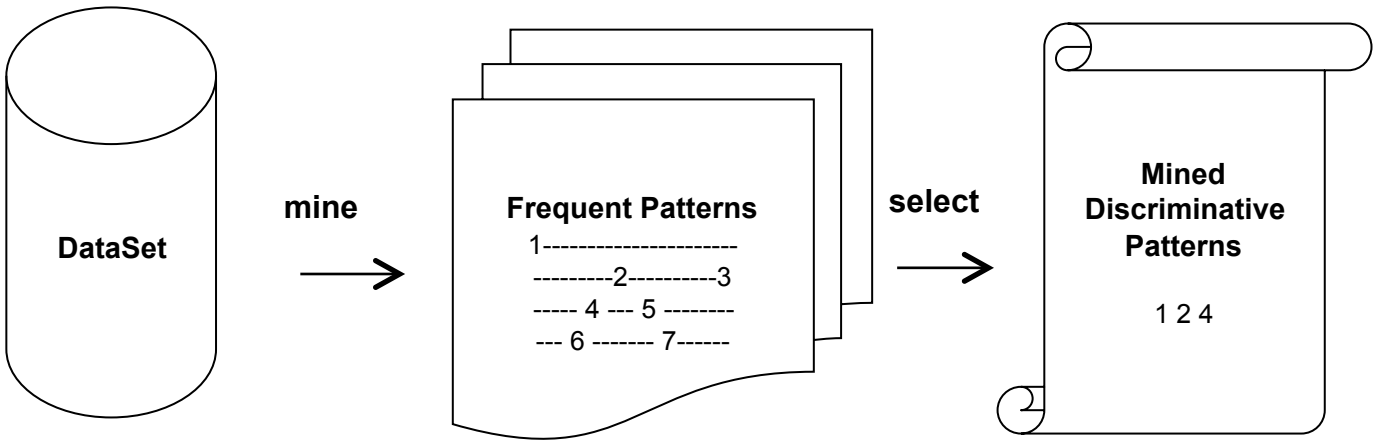
- What are good features?
  - Frequent pattern is a good candidate for discriminative features (Hong etc. ICDE'07, 08)
- Frequent-pattern enumeration is NP-complete.
- However, most discovered patterns either do not carry much information gain or are correlated in their predictability.
- Major research interest for frequent pattern mining: how to discover those discriminative and essential patterns efficiently.

Discriminative Frequent Patterns as Features

- Conventional procedure: Two-Step Batch Method
  1. Mine frequent patterns (> *minsup*)
  2. Select most discriminative patterns
  3. Represent data in feature space using the patterns
  4. Build classification models (SVM, NN, etc.)

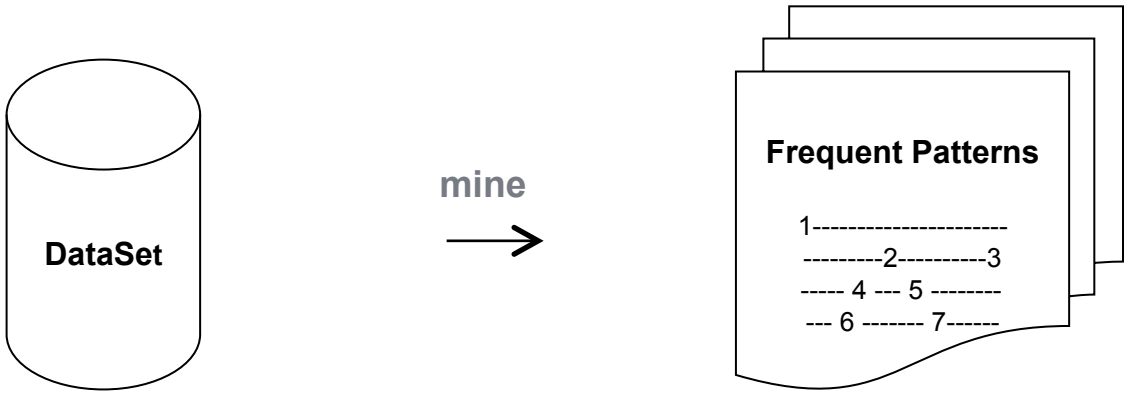
1. mine

2. select



Problems: Feature Mining Step

- Number of candidate frequent patterns can be too large for effective feature selection – exponential explosion
- If the frequency of discriminative features is below minsup, those features won't even be considered . And many algorithms can't handle a very low minsup, thus possibly sacrificing good candidate frequent patterns.

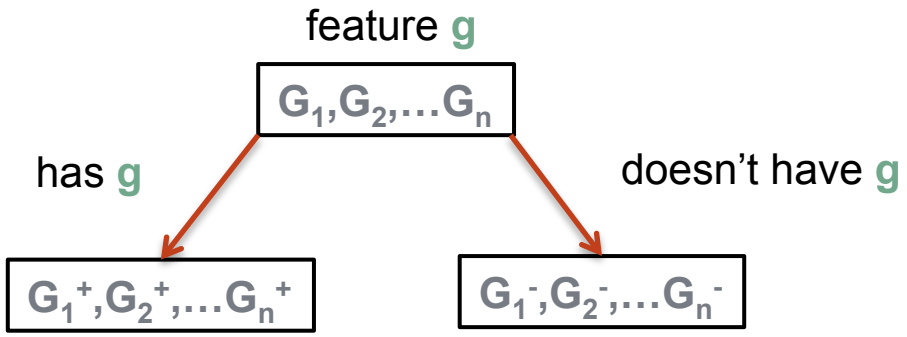


Problems: Feature Evaluation Step

- The discriminative power of patterns is evaluated against the complete dataset, but not on the subset that other patterns do not predict well. However, a feature not quite predictive on the complete dataset, can actually be quite informative on a subspace.
- The correlation among multiple features are not directly evaluated. The batch approach prefers features that are uncorrelated, either using covariance-based or coverage-based criteria. However, when two features are uncorrelated, they may not necessarily help each other to cover examples that each of them does not predict well by itself.

Model-Based Search Tree: Divide and Conquer Approach

- Work with the data in a top-down manner to construct the tree
- At each node, partition the data set into two subsets, one containing the chosen feature, the other not.
- Pick the best splitting feature at each step according to some metric(e.g., information gain)

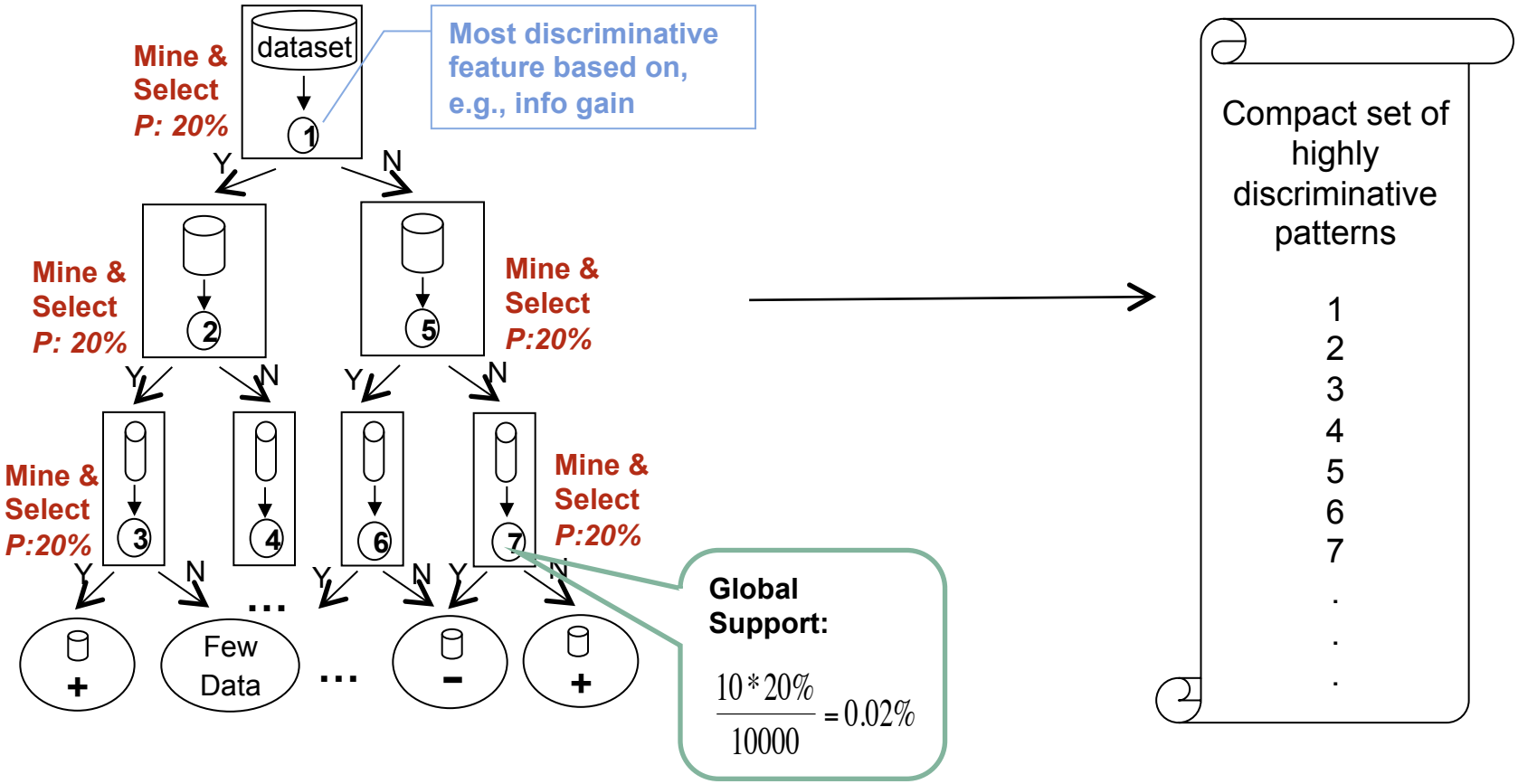


Model-Based Search Tree: Divide and Conquer Approach

- Each node maintains a discovered pattern as the discriminative feature.
- As each node being expanded, a frequent-pattern algorithm is invoked only on the examples that the node is responsible for.
  - the motivation: given that one feature or a number of features is already chosen, the next ideal feature should be the one that can cover the subset of examples where currently mined patterns do not cover well. The algorithm should purposefully look for features that can correctly cover subspaces where currently discovered features cannot.
- The search and tree construction terminates when either:
  - every example in the note belongs to the same class; or
  - the number of examples is less than a given threshold
- Result: a predictive decision tree, and a set of discriminative features kept in the non-leaf nodes of the tree

Model-Based Search Tree

Assume a dataset with n=10000, node support p=20%, minimum node size = 10 (if node size < 10, we stop)



Experimental Studies: Graph Mining

	Proposed Method ($\geq 10\%$)		SVM Bchmk 5% + fs		C4.5 Bchmk 5% + fs	
Data Set	M^bT	DT M^bT	org	rBlcd	org	rBlcd
NCI1	0.685	**0.74**	0.583	0.736	0.589	0.65
NCI33	**0.743**	**0.745**	0.512	0.737	0.536	0.648
NCI41	**0.765**	**0.763**	0.679	0.72	0.603	0.606
NCI47	0.708	0.727	0.501	**0.75**	0.63	0.64
NCI81	0.696	0.723	0.541	**0.739**	0.589	0.652
NCI83	**0.734**	**0.722**	0.633	0.692	0.594	0.608
NCI109	0.699	**0.746**	0.508	0.727	0.555	0.64
NCI123	**0.667**	**0.679**	0.517	0.619	0.606	0.608
NCI145	0.747	0.752	0.55	**0.755**	0.595	0.654
H1	**0.675**	**0.667**	0.632	0.661	0.399	0.556
H2	0.707	0.695	0.519	**0.845**	0.427	0.682

7 Wins 4 losses

Model-Based Search Tree Properties

- Integrates feature mining and classifying
- Handles the explosive growth of frequent patterns
- Succefully identifies discriminative frequent patterns, even with a very small minsup (<0.03%)
  - many other algorithms may not even finish running with a low minsup since the virtual memory can be exhausted.
- Bound on the number of discriminative features returned
  - O(n) (where n is the number of of examples)
- Will not overfit the data, due to specifying a minimum threshold requirement for node size
- Optimality under exhaustive search
- Not limited to only one type of frequent pattern

Thank you