Relational Text Mining and Visualization

Boris KOVALERCHUK
Dept. of Computer Science, Central Washington University, Ellensburg, WA 9892, USA
borisk@cwu.edu

Abstract. Discovering hidden patterns in distributed heterogeneous textual databases and unstructured data is a new challenge in data mining. Traditional data mining often assumes that preprocessing is already done -- homogeneous data are available on the needed level. For distributed heterogeneous textual data this is not the case. Complex relations between items/entities (e.g., relations between people in a fraud detection task) should be discovered and generalized. This paper offers a new hierarchical relational clustering method (Φ-method) based on the Φ-equivalence concept. The method permits to process: (i) incomplete relations: (ii) relations without converting them to attributes of individual entities, and (iii) relations presented in distributed heterogeneous databases using XML tags. Clustering produced by the method is invariant, has a clear meaning and natural visualization.

1. Introduction

A new challenge in Data Mining and Visualization (DMV) is developing techniques for discovering hidden patterns in ***distributed heterogeneous textual databases and unstructured data*** [1]. This problem includes two stages:

- *Preprocessing*: Knowledge Representation (KR) and Evidence Extraction (EE) and
- *Actual Discovery*: Link Discovery (LD) and Pattern Learning (PL).

Traditional data mining operates with a much simpler and smaller preprocessing stage. Often it starts with an assumption that preprocessing is already done -- homogeneous (numeric and nominal) data are available. That is data are uniformly organized in a single data table with rows representing cases (items) and columns representing their attributes. Also often, it is assumed that the data table already contains necessary domain knowledge implicitly. For distributed heterogeneous textual data, preprocessing is extremely complex and its result may not be a single homogeneous data table of attributes of individual items/entities. In many cases these data are relations between items/entities. For instance, in fraud detection it can be relations between people with the goal of discovering suspicious relations.

Intensive study on Evidence Extraction and Link Discovery (EELD) is currently underway with DARPA support [1]. This program reflects recommendations of the DARPA-sponsored Workshop on Knowledge Discovery, Data Mining, and Machine Learning [2] at Carnegie Mellon University: "Whereas conventional data mining and knowledge discovery is sometimes described as 'finding a needle in a haystack,' the system discussed at the Workshop must help analysts to 'reassemble needles from pieces that have been deliberately hidden in many haystacks.'" The workshop and EELD program identified parameters of this type of data mining. Below in table 1 we organized them into three categories: (1) general environment, (2) data and prior knowledge, and (3) patterns.

The current research challenge is developing a new class of data mining models and techniques that satisfy parameters described in Table 1 (based on [1]). In [1] such data mining models are called ***Models of Relational Data***. Another similar term for these models is ***Rela-***

tional Data Mining [3]. It was emphasized in [1,3] that these models are very different from traditional "Relational" Models associated with relational databases**.**

Table 1. Data Mining Parameters

(1) ***General environment***: *Uncircumscribed domain, Multiple simultaneous problems, High costs of failure* (e.g., a missed terrorist event).	
(2) ***Data and prior knowledge:*** • *A fraction* of what could be known; • Vast volume of *potentially relevant* data items; • *Few actually relevant* data items; • Large volume of explicit and implicit *general and domain-specific knowledge;* • Most of data items are examples of *normal (unsuspicious) patterns* • *Heterogeneous* data sources and data items. • The available data only represent ***low-level objects*** and events.	(3) ***Patterns to be discovered****:* • Highly structured (relational), dynamic and not exact; • Can be deliberately obscured; • Built on relations that include temporal and spatial relationships; • Built on the low-level objects, which should be generalized from lower-level recorded data items;

2. Approach to learn structural (relational) patterns

The following categories and potential approaches for relational pattern learning are identified in [1]:

- ***Relational classification***/clustering;
- Probabilistic relational models -- Extend Bayes nets to *richer relational settings*
- Mutual bootstrapping/*co-training;*
- *Stochastic relational learning techniques* (as enhancement of techniques such as Inductive Logic Programming, ILP);
- *Sequence* Discovery and Classification.

This paper focuses on relational classification/clustering. In fact, relational classification tasks have a long history in theoretical considerations, e.g., [4]. New data mining tasks in distributed heterogeneous textual databases and unstructured data foster a renewed and very practical interest in this subject.

The mentioned paper [4] uses model-theoretical knowledge representation [5] to define meaningful classification (clustering) in relational terms using the concept of Φ- equivalence (physical equivalence) [6]. One of the theorems proved in [4] states that every meaningful relational clustering should keep Φ-equivalent objects in the same class. It is proved that if a clustering algorithm assigns two Φ-equivalent objects a and b to different clusters then this algorithm will not be invariant even for renaming objects.

Informally two objects a and b are ***Φ-equivalent*** in a model <A,P> with objects A and predicate P(x,y) if for every x not equal to a and b (x≠a and x≠b) P(x,a)=P(x,b), P(a,x)=P(b,x), P(a,a)=P(b,b) and P(a,b)= P(b,a). However, Φ-equivalence *does not require* all properties of the standard equivalence. It is possible that P(a,b)≠P(b,b) and P(b,a)≠P(a,a). The standard equivalence of a and b requires P(a,b)=P(b,b) and P(b,a)=P(a,a). In the standard equivalence a and b are indistinguishable by predicate P. This difference shows that Φ-equivalence of a and b requires that a and b behave identically in their relation with all other elements of A. But a and b *may not be interchangeable in some of their relations to each other,* that is P(b,a)≠P(a,a). In contrast the standard equivalence requires that P(b,a)=P(a,a). For instance, it is possible that P(a,a) and P(b,b) are true, that is "a likes himself ", "b likes himself ", but P(a,a) and P(b,b) are false, that is "a hates b" and "b hates a". For Φ-equivalent objects it may not be possible to say which one we deal with by analyzing their behavior in predicate P. Below we provide an example of Φ-equivalence for a model with a predicate with two arguments. This example permits compare Φ-equivalence approach with *common directed-graph analysis approach* for clustering. Note that graph-based methods are designed to analyze a single predicate with two arguments. In contrast Φ-equivalence in defined for *any number predicates of arguments of predicates*.

3. Example of use of Φ-equivalence

Table 2 presents an extract for textual database with records such as “Mr. Smith hates Mr. Brown”, “Ms. Smith likes Mr. Smith and hates every other person that hates Mr. Smith”, and so on. Table 2 uses notation a,b,x,y,z,w for persons to store these relations. Table 2 encodes these sentences in a predicate form: P(s,q)=1 if s hates q and P(s,q)=0 if s likes q. For instance the bold cell can be read that “a hates b”.

Table 2

	Entity a	Entity b	Entity x	Entity y	Entity z	Entity w
Entity a	0	**1**	0	0	0	0
Entity b	1	0	0	0	0	0
Entity x	0	0	0	0	0	0
Entity y	1	1	1	0	1	1
Entity z	1	1	1	1	0	1
Entity w	0	0	0	0	0	0

The first picture on Figure 1 presents predicate P in a directed graph form. Picture (2) in Figure 1 shows typical traditional generalization of objects to create larger objects. The most connected nodes are grouped together. It is hard to interpret this grouping.

Use of Φ-equivalence produces better interpretable clustering. This process has several steps.

Step1: Select a first pair of elements from table 2.
Result: Pair a and b is selected.
Step 2. Test pair a and b to be a Φ-equivalent pair, by applying the definition of Φ-equivalent pair.
Result: The pair (a,b) is a Φ-equivalent pair, because
P(a,a)=P(b,b)=0; P(a,b)=P(b,a)=1;
P(a,x)=P(b,x)=0; P(a,y)=P(b,y)=0; P(a,z)=P(b,z)=0; P(a,w)=P(b,w)=0;
P(x,a)=P(x,b)=0; P(y,a)=P(y,b)=1; P(z,a)=P(z,b)=1; P(w,a)=P(w,b)=0.
See informal and formal definitions in this paper.
Step 3. Combine a Φ-equivalent pair into a single node (cluster).
Result: Pair a and b is combined to a single node (see picture (3), Figure1).
Step 4: Interpret the combined node (cluster)
Result: This node/cluster has a clear meaning – a and b hate each other, but like all others and are liked by all others. We can call this node an “internally hating” but “externally loving” but node (HH-LL-node).
Comment: Elements a and b behave identically with respect to all others. This means that we really can combine them into a single node.
Step 5. Select another pair to test for Φ-equivalence.
Result: Pair a and w is selected.
Step 6. Test a and w to be Φ-equivalent.
Result: Test fails. There is an element b such that a and w behave differently with respect to b (a hates b, but w likes b, P(a,b)=1, P(w,b)=0).
Step 7. Looping steps 6 and 7 thorough all pairs of elements in table 2.
Result: Pair z and y is Φ-equivalent.
P(z,y)=P(y,z)=1; P(z,w)=P(y,w)=1;P(w,z)=P(w,y)=0; P(z,x)=P(y,x)=1;P(x,z)=P(x,y)=0

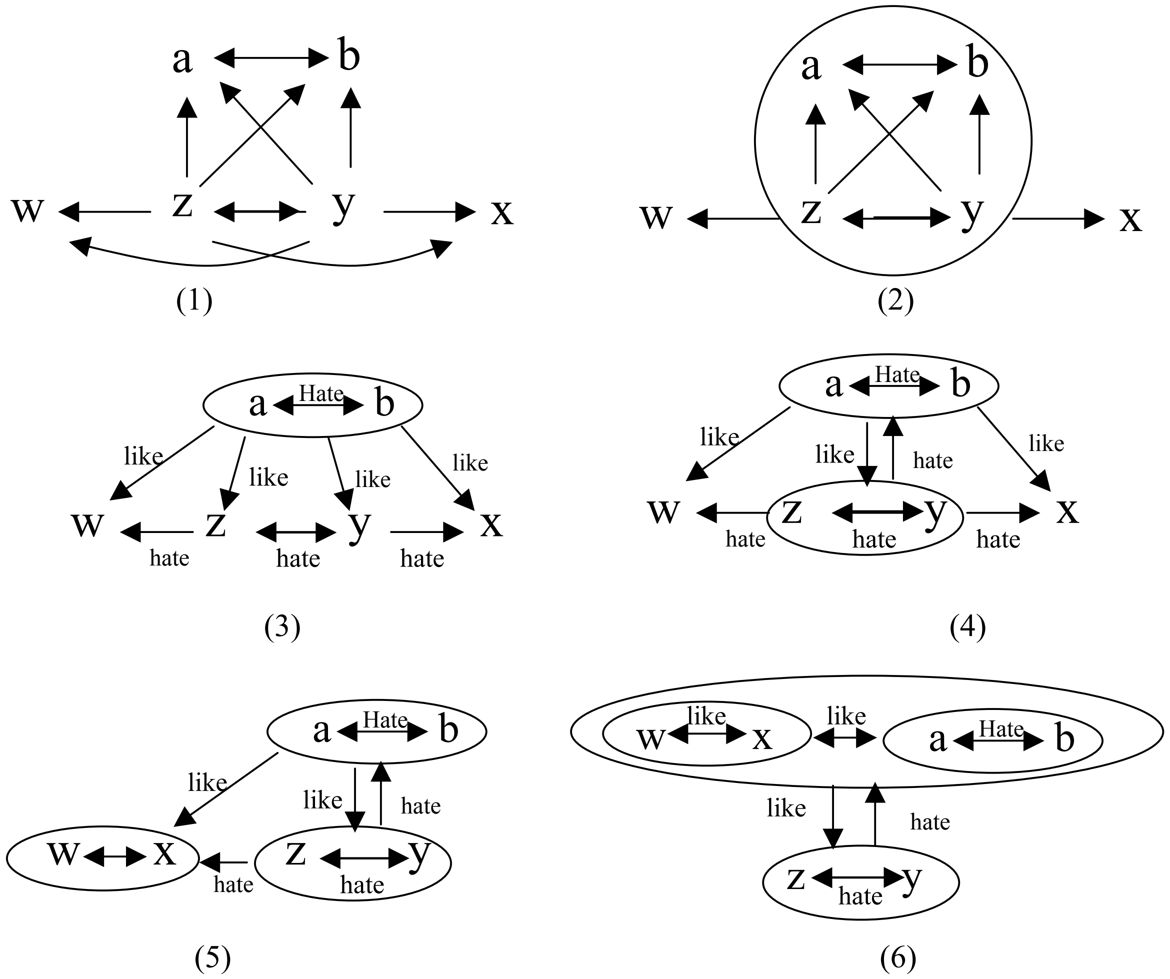

Figure 1. Steps of discovery of structural relations using physical equivalence

Step 8. Combine a Φ-equivalent pair to a single node (cluster).
Result: Pair z and y is combined to a single node (see picture (4), Figure1).
Step 9: Interpret the combined node (cluster)
Result: This node/cluster has a clear meaning – y and z hate each other, both hate all others, and all others like them, e.g., P(a,z)=0, P(b,y)=0. We can call this node an "internally hating" and "externally mixed" node (HH-HL-node).
Comment: Elements y and z behave identically with respect to all others. This means that we really can combine them to a single node.
Step 10. Eliminate redundant links between combined nodes.
Result: Only two links between two combined nodes are shown (see picture (4)) instead of showing four links between (a,y), (a,z), (b,y) and (b,z) (see picture (1)).
Comment: The structure has been simplified in a meaningful way.
Step11. Discover another Φ-equivalent pair.
Result: Pair w and x is discovered.
Step12. Combine a Φ-equivalent pair (w,x) into a single node (cluster).
Result: Pair w and x is combined into a single node (see pictures (5) and (6), Figure1).
Step 13: Interpret the combined node (cluster)
Result: This node/cluster has a clear meaning – both like each other and both like all others. We can call this node an "externally and internally loving " node.
Step 14. Eliminate redundant links between combined nodes.
Result: See (pictures (5) and (6), Figure 1).
Comment: The total structure has been simplified in a significant way. We have two types of nodes with their internal structure: "hating" and "loving". There is one node (z,y) that

hates two other nodes and that two other nodes like all other nodes. Thus, we can say that there are: *Two externally friendly nodes, (a,b) and (w,x); one hostile node (z,y) with mutual internal hatred inside; and friendly node (a,b) has a mutual internal hatred.*

Now we can compare this result with clustering based on finding most connected subgraph (see picture (2), Figure 1). Picture (5) provides meaningful information on data structure. Picture (2) missed discovery that there are two externally friendly nodes and only one externally hostile.

4. Definitions, Generalization, and Related Work

Definition. Elements a and b are ***Φ-equivalent*** (physically equivalent) if

$$\forall\ <x>_{vi}\ \ i\in J\ \ P_i(<x(a,b)>_{vi})=P_i(<x>_{vi}).$$

Here $\{P_{i,}\}$ is a set of predicates, $i\in J$, $<x>_{vi}$ is a sequence of elements of set A of length v_i and $<x(a,b)>_{vi}$ is a modified sequence $<x>_{vi}$ with every occurrence of a is substituted by b and every occurrence of b is substituted by a.

We also introduce a weak Φ-equivalence (***VΦ-equivalence***) that indicates how close a pair to be Φ-equivalent. We count the number of violations V(a,b) of Φ-equivalence for pair (a,b). In general terms, nodes a, b are Φ-equivalent if and only if V(a,b)=0. Thus t and s are 3Φ-equivalent if there are three violations, V(t,s)=3. We interpret VΦ-equivalence as a kind of **distance** between nodes that can be processed by standard clustering algorithms.

After all Φ-equivalent items are clustered we may want to go to the next level of clustering (generalization) and discover patterns on the next level. For instance, we can join combined nodes (w,x) and (a,b). Both nodes have the same type of relations with node (z,y), that is "like-hate" (see picture (5), Figure 1). Formally, this procedure means that we applied Φ-equivalence to combined nodes and found that nodes (a,b) and (w,x) are Φ-equivalent.

The discovered structure can be written as follows: *one complex externally friendly node, ((a,b), (w,x)); one externally hostile node (z,y) with a mutual hatred internal, and externally friendly node ((a,b), (w,x)) has a mutual internal hatred in (a,b) subnode.*
Note that picture (6) shows only five links, which is less than the number of links on any other pictures in Figure 1. On the hand, picture (6) still captures all information presented in other pictures. Often generalization means loosing some lower-level information. We made the generalization ***without missing any lower level information***.

Now we can show the use of *Φ-clustering in distributed data mining* (DDM). Every predicate P(x,y) or even individual statement "Jon likes Mary", Like(Jon,Mary) can be stored separately (in different tables, databases and sites). The fact that data are stored at different locations does not change the method. For instance, let right and left halves of the table 2 are stored in different locations. We can test Φ-equivalence partially in each of these locations and then combine results. If Φ-equivalence of a and b is refuted in one part then there is no need to test it in another part. If a and b appear to be Φ-equivalent in both parts then a and b are Φ-equivalent. This is true for any decomposition of the table 2 for three, four or more parts. This freedom of decomposition may not be the case for other data mining methods. Other methods may require specific decomposition methods. Several decomposition methods have been developed in DDM (e.g., BODHI and JAM systems [7]). These methods also can be applied for Φ-equivalence.

There is also a benefit to use *Φ-clustering for text mining tasks* outlined in [1]. Let us illustrate it by contrasting typical distributed data mining [7] and new text mining tasks [1]. In typical distributed data mining one may want to learn dependency between hepatitis-C (stored in a large DB A) and weather in the US (stored in a large DB B in another location) [7]. In distributed text mining tasks outlined in [1] one may want uncover money laundering scheme or suspicious terrorist activities. Relevant information could be deliberately spread over dozens, hundreds or thousands of "haystacks". There are two large well-organized DBs in the first case, and hundreds of unstructured sources in the second case. Bringing all of these sources to a single location does not help much – information is not structured. Φ-clustering does not requires a rigid structuring of the sources in a single flat file with multi-dimensional records. Note that this is a typical requirement for many data mining methods. An individual 1 Mb text source can contain only a single relevant statement "John hates Brian". Φ-clustering will work with this text without need to bring it to a single location. With proliferation of semantic web it will be more and more common that the statement will be tagged:

<name> John </name> <relation> hates </relation><name>Brian</name>.

In this case converting to the table similar to table 2 is not necessary.

In general the problem of combining data into a single flat file from tables in relational databases is well known in both conventional and distributes data mining. For instance, it is noticed in [7, p.19] that 100 Mb DB results in 2.5 Gb flat file for a fraud detection mining. What is one of the major sources of the problem? The traditional answer is -- data mining methods require a flat file to run. Current works in DDM (BODHI and JAM systems [7]) focus on removing this requirement by redesigning and hybridizing conventional methods to work with a set of *fragments of a flat file.* A recent workshop [10] on multi-relational data mining (MRDM) addresses this issue too. It considers MRDM as the multi-disciplinary field dealing with knowledge discovery from relational databases consisting of multiple tables. "The field aims at integrating results from existing fields such as ILP, KDD, ML and Relational Databases, as well as at producing new techniques" [10].

5. Conclusion

This paper offered a new hierarchical relational clustering method (Φ-method) applicable to discovering hidden patterns in distributed heterogeneous textual databases and unstructured data. The method can process incomplete relations. That is relations between some objects can be unknown. The method does not require converting relations to attributes of individual entities as required by many other methods. Such conversion often is accompanied by explosion of the size of data to be processed. Φ-method generalizes without missing lower level information, clusters produced have a clear meaning and natural graphical visualization as it is shown in Figure 1.

References

[1] T. Senator, EELD Program, http://www.darpa.mil/ito/research/eeld/EELD_BAA.ppt, 2001

[2] Workshop on Knowledge Discovery, Data Mining, and Machine Learning (KDD-ML), 1998 http://www.darpa.mil/ito/research/eeld/KDD-ML_Report.doc

[3] B. Kovalerchuk, Vityaev, E. Data Mining in Finance: Advances in Relational and Hybrid Methods, Kluwer Acad. Publ., Boston, 2000.

[4] B. Kovalerchuk, Classification structures invariant to renaming of objects, *Computational Systems*, **55** (1973) 90-97, Institute of Mathematics, Novosibirsk (In Russian).

[5] A.I. Malcev, Algebraic Systems, Springer, 1973.

[6] J. Czajsner , Equivalence relations determining useful properties, *Studia Logika*, XXIV (1969)

[7] Advances in Distributed and Parallel Knowledge Discovery, Eds. H. Kargupta, P. Chan, MIT, 2000.

[8] S. Dzeroski, Inductive Logic Programming and knowledge discovery and data mining, In: Advances in Knowledge Discovery in Databases. Eds. U. Fayad, G. Piatetsky-Shapiro, P.Smyth, R. Uthrusamy, AAAI/MIT press, 1996, pp. 117-152.

[9] Relational data mining, S. Dzeroski, N. Lavrac, editors, Springer, Berlin, 2001

[10] Workshop Multi-relational Data Mining, MRDM 2001, September 6, 2001, Freiburg, Germany, http://mrdm.dantec.nl/