Association Discovery in Two-View Data

Matthijs van Leeuwen and Esther Galbrun

Abstract—*Two-view datasets* are datasets whose attributes are naturally split into two sets, each providing a different view on the same set of objects. We introduce the task of finding small and non-redundant sets of associations that describe how the two views are related. To achieve this, we propose a novel approach in which sets of rules are used to *translate* one view to the other and vice versa. Our models, dubbed *translation tables*, contain both unidirectional and bidirectional rules that span both views and provide lossless translation from either of the views to the opposite view.

To be able to evaluate different translation tables and perform model selection, we present a score based on the Minimum Description Length (MDL) principle. Next, we introduce three TRANSLATOR algorithms to find good models according to this score. The first algorithm is parameter-free and iteratively adds the rule that improves compression most. The other two algorithms use heuristics to achieve better trade-offs between runtime and compression. The empirical evaluation on real-world data demonstrates that only modest numbers of associations are needed to characterize the two-view structure present in the data, while the obtained translation rules are easily interpretable and provide insight into the data.

Index Terms—Association discovery, Two-view data, Minimum description length, Association rule mining, Redescription mining

1 INTRODUCTION

TWO-VIEW datasets are datasets whose attributes are split into two sets, providing two alternative views on the same set of objects. Two-view data is a form of multi-view data, in which an arbitrary number of views can occur. In practice, a data analyst is often given different sets of descriptors on the same set of objects, and asked to analyze associations across these views.

In the medical domain, for example, persons could be the objects of interest, and one could have both demographic and medical data. The two views represent clearly different *types* of information. Alternatively, products could be the objects, and one could have both product information and aggregated customer data (e.g., sales, churn, sentiment). Or consider movies, for which we could have properties like genres and actors on one hand and collectively obtained tags on the other hand.

In each of these examples, there are two views that convey different information concerning the same objects. An obvious question to a data analyst would be: *what associations are present in these views?* This is a typical *exploratory data mining* [3] question: the task is to discover patterns that together describe the structure of the data. In particular, we are interested in associations that span both views. For instance, certain demographic properties might imply a certain medical condition with high probability. Sometimes, such an association might hold in both directions, implying that the two observations occur mostly together.

- *M. van Leeuwen is with the Machine Learning group, Department of Computer Science, KU Leuven, Leuven, Belgium*
  E-mail: matthijs.vanleeuwen@cs.kuleuven.be
- *E. Galbrun is with the Department of Computer Science, Boston University, Boston, MA, United States*
  E-mail: galbrun@cs.bu.edu

It is important to note that we explicitly aim to find a *compact* and *non-redundant* set of such associations, to avoid overwhelming the analyst with a plethora of discoveries. On the other hand, the set should also be *complete* with respect to the structure in the data it describes. Furthermore, we are primarily interested in scenarios where the two views are expressed over *different, typically disjoint, sets of attributes*, rather than two sets of tuples over the same attributes.

As another example, which we will revisit during the empirical evaluation, consider a set of music tracks for which we have both music features, such as genres and instruments, and manually collected information on the evoked emotions. In this case it would be of interest to investigate which emotions are evoked by which types of music: how are the music features associated to emotions? Example patterns our method finds are, e.g., that R&B songs are typically catchy and associated with positive feelings, that alternative rock music is often listened to while driving, and that aggressive vocals are associated with high energy songs.

Existing association discovery and pattern mining techniques were not designed to be used with multi-view data. As a consequence, these methods cannot be directly applied on two-view data, while merging the two views would result in the loss of the distinction between the views. Association rule mining [1] algorithms can be modified to return only rules that span two views of a dataset, but these methods suffer from the infamous *pattern explosion*: the number of rules found is enormous and it is therefore impracticable for a data analyst to manually inspect and interpret them. Acknowledging this problem, methods have been proposed to discover smaller sets of rules, for example via closed itemsets [25] or statistical testing [21]. We will empirically compare to the latter approach, as it results in small sets of high-confidence rules. Other pattern set mining methods, such

as KRIMP [19], also address the pattern explosion, but no existing techniques target the (symmetric) two-view setting that we consider.

1.1 Approach and contributions

The problem we address in this paper is to discover a *small* and *non-redundant* set of rules that together provide an *accurate* and *complete* description of the associative structure across a Boolean two-view dataset. Solving this problem will enable data analysts to perform exploratory mining on two-view data and discover new knowledge.

For this, we consider sets of objects characterized by two Boolean datasets over two disjoint item vocabularies. Without loss of generality, we refer to these as left-hand side and right-hand side datasets and denote them by $\mathcal{D}_{\mathbf{L}}$ (over $\mathcal{I}_{\mathbf{L}}$) and $\mathcal{D}_{\mathbf{R}}$ (over $\mathcal{I}_{\mathbf{R}}$) respectively.

In this context, consider a *rule* $r = X \rightarrow Y$, where X is an itemset over $\mathcal{I}_{\mathbf{L}}$ and Y is an itemset over $\mathcal{I}_{\mathbf{R}}$. Such a rule can be interpreted as indicating that if X occurs in a transaction of $\mathcal{D}_{\mathbf{L}}$, then Y is likely to occur in the corresponding transaction of $\mathcal{D}_{\mathbf{R}}$. In other words, given the left-hand side of the data, rules provide information about occurrences of items in the right-hand side. Thus, they can be used to *translate* $\mathcal{D}_{\mathbf{L}}$ to $\mathcal{D}_{\mathbf{R}}$ and are therefore dubbed *translation rules*. Similarly, we define rules in the other direction, and symmetric rules for which both directions hold.

After discussing related work in Section 2, Section 3 presents the first main contribution of the paper: *we introduce pattern-based models for Boolean two-view data*. A model, called *translation table*, consists of translation rules and can be used to reconstruct one side of the data given the other, and vice versa. We introduce a translation scheme that takes a Boolean view and translation table as input, and returns a reconstructed opposite view as output. Each individual rule spans both views of the data and hence provides insight in how the two sides are related. In addition, we use both bidirectional and unidirectional rules, which allows us to construct succinct models that allow for easy interpretation.

Given a dataset, different translations tables will clearly result in different translations and an important question is *how good* a specific translation table is. In general, some items might be missing from the reconstructed view while some might be introduced erroneously. To make translation completely lossless, we add a so-called *correction table* that corrects both of these types of errors; the larger the reconstruction error, the larger the number of corrections. Given this, we could try to find the model that minimizes the size of the correction table, but this would result in overly complex translation tables.

For that reason, Section 4 presents our second main contribution: *model selection for translation tables based on the Minimum Description Length (MDL) principle* [7]. The MDL principle takes both the complexity of the model and the complexity of the data given the model into account, and is therefore very useful for model selection when a balance between these complexities is desirable. In the current context, we use it to select small sets of rules that provide accurate translations.

Having defined our models and a way to score them, we need to search for the optimal translation table with respect to this score. Unfortunately, exhaustive search for the globally optimal translation table is practically unfeasible. Still, it is possible to find the single rule that gives the largest gain in compression given a dataset and current translation table, allowing us to construct a good translation table in a greedy manner. Our third main contribution, described in Section 5, consists of *three* TRANSLATOR *algorithms*, each of which takes a two-view dataset as input and induces a good translation table by starting from an empty table and iteratively adding rules. By introducing an exact method for finding the best rule in each iteration, we have the best possible baseline to which we can compare the heuristic approaches (on modestly sized problem instances).

Then, the proposed model and algorithms are empirically evaluated in Section 6. The obtained compression ratios indicate that two-view structure in datasets can be discovered. Comparisons demonstrate that TRANSLATOR discovers more compact and complete models than existing methods. Finally, we show by means of examples that the translation rules found are expressive and intelligible. Section 7 concludes the paper.

2 RELATED WORK

Two-view data, an instance of multi-view data, is strongly related to the concept of *parallel universes* [22], which also concerns multiple descriptor spaces over the same set of objects. However, learning in parallel universes usually has the goal to also identify structure within each of the individual views, whereas multi-view learning focuses on structure across the different views, as we do in this paper.

Multi-view data and parallel universes have both been extensively studied in the context of traditional learning and clustering tasks [2], [11], [14], [22], but have received little attention in the context of (rule-based) association discovery and pattern mining. *Subspace clustering* [8] aims to find all (low-dimensional) clusters in all subspaces of high-dimensional data, but does not distinguish between different views. The relation between subspace clustering and pattern mining was recently surveyed [20].

In the remainder of this section, we focus on work most closely related to ours, divided into three parts: pattern mining for two-view data, association rule mining, and compression-based model selection.

2.1 Pattern mining for two-view data

Both Exceptional Model Mining (EMM) [9] and Redescription Mining (RM) [6], [13] are concerned with finding patterns in two-view data. EMM aims at finding subsets of the data that stand out with respect to a

designated 'target'. As such, EMM is highly asymmetric, with one side used for descriptions and the other purely as target, as is the case with multilabel classification [17]. Redescription Mining, on the other hand, aims at finding pairs of queries, one for each view, that are satisfied by almost the same set of objects. Such query pairs are called *redescriptions*, and quality is usually measured with the Jaccard coefficient of the queried object sets. Similar to the approach discussed here and unlike EMM, RM treats both sides equally. However, there are two important differences with our work. First, associations are required to hold in both directions, i.e., a redescription can be interpreted as a bidirectional high confidence association rule. Second, redescriptions are judged individually and the complete set of redescriptions is therefore often redundant in practice. Hence, redescription mining discovers individual high-confidence, bidirectional rules, whereas our approach induces non-redundant, global models consisting of both unidirectional and bidirectional rules. We empirically compare our proposed approach to redescription mining in Section 6.

2.2 Association rule mining

At first sight, mining association rules across the two views might seem an obvious alternative to our proposal. Association rules have been widely studied since their introduction in [1]. Unfortunately, association rules are unidirectional and have other disadvantages [21], the most important being the so-called *pattern explosion*: humongous amounts of highly similar rules are found and, consequently, support and confidence thresholds are hard to tune. Acknowledging this problem, methods have been proposed to find smaller sets of the rules [25]. One recent and well-known such method employs statistical testing [21]. In particular, a Bonferroni correction is applied to correct for multiple testing, and the discovered patterns are assessed on holdout data. This results in relatively strict rule selection and we will therefore empirically compare our method to this statistical approach in Section 6.

Supervised pattern set mining methods [4] approach the problem from a classification perspective, which assumes the existence of a single property of interest, i.e., the class label or target. We do not assume any such target and instead of inducing predictive models consisting only of high-confidence rules, we aim at discovering descriptive, non-redundant models that include bidirectional rules.

2.3 MDL based model selection

A recent trend that addresses the pattern explosion in local pattern mining, is the development of pattern-based *models* using the Minimum Description Length (MDL) principle [7]. Examples include methods for Boolean data [19] and for sequences [16]. Advantages of this approach over exploratory data mining are twofold. First, it results in small, pattern-based models, which are interpretable and may hence provide the data analyst with valuable insight in the data. Second, using compression allows the models to be used for other tasks [5], such as clustering [18].

Our high-level approach is related to existing methods, but our work differs in two main aspects. First, we explicitly consider two-view datasets and their particular structure to discover sets of rules. Concatenating the two sides of the data and applying KRIMP, for example, yields very different results, as we will demonstrate in the experiments. Particularly, our framework compresses the mapping across two views rather than the data itself, to ensure that we (only) find associations *across* the two sides of the data. Second, in contrast to existing approaches, we present an *exact* method for finding the best rule given a translation table. Within the context of greedy search, which is unavoidable, this gives us the best possible baseline to compare our heuristics to.

3 Translation Models for Boolean Two-View Data

We consider Boolean data over a set of objects denoted by $\mathcal{O}$. Each object is characterized by a transaction over two sets of items, $\mathcal{I}_{\mathbf{L}}$ and $\mathcal{I}_{\mathbf{R}}$ ($\mathbf{L}$ for left, $\mathbf{R}$ for right). That is, each transaction t can be regarded as a pair of itemsets $t = (t_{\mathbf{L}}, t_{\mathbf{R}})$ concerning the same object $o \in \mathcal{O}$, such that $t_{\mathbf{L}} \subseteq \mathcal{I}_{\mathbf{L}}$ and $t_{\mathbf{R}} \subseteq \mathcal{I}_{\mathbf{R}}$. A *two-view dataset* $\mathcal{D}$ is a bag of transactions. Let $|\mathcal{D}|$ denote its size, i.e., $|\{t \in \mathcal{D}\}|$. We use $\mathcal{D}_{\mathbf{L}}$ (resp. $\mathcal{D}_{\mathbf{R}}$) to denote the dataset $\mathcal{D}$ projected onto $\mathcal{I}_{\mathbf{L}}$ (resp. $\mathcal{I}_{\mathbf{R}}$). An itemset Z is said to *occur* in a transaction t iff $Z \subseteq t_{\mathbf{L}} \cup t_{\mathbf{R}}$. The *support* of an itemset Z in dataset $\mathcal{D}$ is the bag of transactions in which Z occurs, i.e., $\text{supp}_{\mathcal{D}}(Z) = \{t \in \mathcal{D} \mid Z \subseteq t_{\mathbf{L}} \cup t_{\mathbf{R}}\}$. We typically omit the index when $\mathcal{D}$ is unambiguous from the context.

Given this notation, we now introduce and formally define the patterns and pattern-based models that we consider in this paper, i.e., translation rules and tables. In the following, we assume a given dataset $\mathcal{D}$ with corresponding item vocabularies $\mathcal{I}_{\mathbf{L}}$ and $\mathcal{I}_{\mathbf{R}}$ over which all itemsets are defined.

Definition 1 (Translation Rule): A *translation rule*, denoted $X \diamond Y$, consists of a left-hand side itemset $X \subseteq \mathcal{I}_{\mathbf{L}}$ ($X \neq \emptyset$), a direction $\diamond \in \{\rightarrow, \leftarrow, \leftrightarrow\}$, and a right-hand side itemset $Y \subseteq \mathcal{I}_{\mathbf{R}}$ ($Y \neq \emptyset$).

Definition 2 (Translation Table): A *translation table* T is a three-column table in which each row contains a translation rule $X \diamond Y$, where the three columns correspond to X, $\diamond$, and Y respectively. $\mathcal{T}$ denotes the set of all possible translation tables for a given dataset.

A translation table can be used to translate one side of the data to the other side. Next we present the mechanism that performs this translation. For ease of presentation, we introduce the definitions and methods only for translating $\mathcal{D}_{\mathbf{L}}$ to $\mathcal{D}_{\mathbf{R}}$ given a translation table T. However, the translation scheme is symmetric and we assume the reverse direction to be defined analogously.

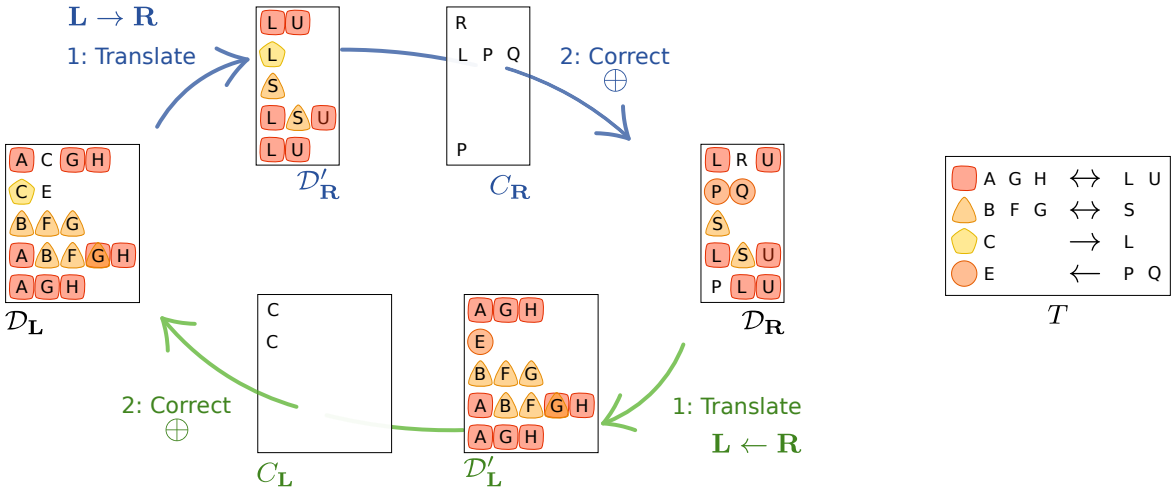

Fig. 1. Translating a toy dataset, consisting of the two views $\mathcal{D}_\mathbf{L}$ and $\mathcal{D}_\mathbf{R}$, with translation table T (on the right). The blue and green arrows indicate left to right and right to left translations respectively. For each translation, the first step is to obtain the translated dataset $\mathcal{D}'_\mathbf{R}$ (resp. $\mathcal{D}'_\mathbf{L}$) by applying the rules in T in the appropriate direction. To complete lossless translation, the second step is to flip the values for all items in correction table $C_\mathbf{R}$ (resp. $C_\mathbf{L}$).

A *translation* is an exact mapping from one view of a multi-view dataset to another view. In two-view data, we have two such mappings: one from left to right and one from right to left, which we denote by $\mathcal{D}_{\mathbf{L}\to\mathbf{R}}$ and $\mathcal{D}_{\mathbf{L}\leftarrow\mathbf{R}}$ respectively. In other words, $\mathcal{D}_{\mathbf{L}\to\mathbf{R}}$ can be regarded as a function that translates $t_\mathbf{L}$ to $t_\mathbf{R}$ for each $t \in \mathcal{D}$.

Translation can be done on a per transaction basis, because transactions are assumed to be independent from one another. The translation scheme is presented as Algorithm 1. It takes $t_\mathbf{L}$ and T as input and returns a translated transaction $t'_\mathbf{R}$, i.e., $t'_\mathbf{R} = \textsc{Translate}_{\mathbf{L}\to\mathbf{R}}(t_\mathbf{L}, T)$. The algorithm first initializes $t'_\mathbf{R} = \emptyset$ and then considers each translation rule $X \diamond Y \in T$ in turn. For each rule of the form $X \to Y$ or $X \leftrightarrow Y$, it checks whether the antecedent occurs in the left-hand side, i.e., whether $X \subseteq t_\mathbf{L}$. If this is the case, Y is added to $t'_\mathbf{R}$.

Note that with this scheme, the order of the rules in T does not influence translation. Also, a translation table may contain both unidirectional and bidirectional rules and thus allows both symmetric and asymmetric associations to be used.

Ideally, we would have $t'_\mathbf{R} = t_\mathbf{R}$ for each transaction. However, for any realistic dataset $\mathcal{D}$ it will be impossible to find a translation table T that achieves this for all transactions. Therefore, we introduce a *correction table* $C_\mathbf{R}$ that represents the errors between the original and translated datasets. For each transaction t, $c^t_\mathbf{R} \in C_\mathbf{R}$ is the *difference* between $t_\mathbf{R}$ and the translated itemset $t'_\mathbf{R}$, i.e., $c^t_\mathbf{R} = t_\mathbf{R} \oplus t'_\mathbf{R}$, where $\oplus$ denotes exclusive or.

Putting everything together, translation $\mathcal{D}_{\mathbf{L}\to\mathbf{R}}$ can be performed losslessly using T and correction table $C_\mathbf{R}$: for each $t_\mathbf{R} \in \mathcal{D}_\mathbf{R}$ we have $t_\mathbf{R} = \textsc{Translate}_{\mathbf{L}\to\mathbf{R}}(t_\mathbf{L}, T) \oplus c^t_\mathbf{R}$.

To illustrate the translation scheme, Fig. 1 shows translations in both directions on a toy dataset. Translation $\mathcal{D}_{\mathbf{L}\to\mathbf{R}}$, for example, is indicated by the blue arrows. The antecedent of the first rule in T occurs in the first, fourth and fifth rows of $\mathcal{D}_\mathbf{L}$, which results in the addition of items L and U in the corresponding transactions in $\mathcal{D}'_\mathbf{R}$. Similarly, the second rule is matched and applied to the second and third transactions, resulting in the item S in the translated transactions. After all rules in T have been applied using the TRANSLATE algorithm, correction table $C_\mathbf{R}$ is applied using exclusive or. This both adds and removes items from $\mathcal{D}'_\mathbf{R}$, e.g., L is removed from the second transaction while both P and Q are added, which results exactly in $\mathcal{D}_\mathbf{R}$. Translation $\mathcal{D}_{\mathbf{L}\leftarrow\mathbf{R}}$ goes in the other direction and is indicated with green arrows.

Algorithm 1 The $\textsc{Translate}_{\mathbf{L}\to\mathbf{R}}$ algorithm

Input: Transaction $t_\mathbf{L}$, translation table T
Output: Translated transaction $t'_\mathbf{R}$

```
t'_R ← ∅
for all X ◇ Y ∈ T do
    if ◇ ∈ {→, ↔} ∧ X ⊆ t_L then
        t'_R ← t'_R ∪ Y
return t'_R
```

4 Scoring Translation Tables

Having defined our models, i.e., translation tables, a natural question that arises is *how good* a given model is. Given a dataset and a set of candidate models, we need to be able to score them so that we can choose the best one. Since it is our goal to find compact yet descriptive translation tables, we use the Minimum Description Length principle [7]. The MDL principle embraces the slogan *Induction by Compression* and is *the* induction principle for descriptions.

The MDL principle states that given a set of models $\mathcal{M}$ and a dataset $\mathcal{D}$, the best model is the model $M \in \mathcal{M}$ that minimizes

$$L(\mathcal{D} \mid M) + L(M),$$

where $L(\mathcal{D} \mid M)$ is the length, in bits, of the data encoded with M and $L(M)$ is the length, in bits, of the model. Simply put, the best model is the one that gives the best compression of data and model combined.

Our model class $\mathcal{M}$ is defined as the set of possible *translation tables* $\mathcal{T}$. In the standard situation, such as with KRIMP, encoding the data is straightforward: each transaction is encoded by the model. However, the current problem is different and we are not interested in encoding the data *directly*. Instead, to capture any cross-view associations we are interested in encoding the *translations* $\mathcal{D}_{\mathbf{L}\to\mathbf{R}}$ and $\mathcal{D}_{\mathbf{L}\leftarrow\mathbf{R}}$. Translation tables do not directly capture the underlying data distributions, instead they capture these translations.

Hence, it is these translations that should be considered as 'data' and compressed accordingly. Combining the left-to-right and right-to-left translations to make the problem symmetric, the total encoded length of a bidirectional translation given a model, denoted by $L(\mathcal{D}_{\mathbf{L}\leftrightarrow\mathbf{R}} \mid T)$, is defined as

$$L(\mathcal{D}_{\mathbf{L}\leftrightarrow\mathbf{R}} \mid T) = L(\mathcal{D}_{\mathbf{L}\to\mathbf{R}} \mid T) + L(\mathcal{D}_{\mathbf{L}\leftarrow\mathbf{R}} \mid T).$$

In Section 3 we defined the space of possible models $\mathcal{T}$ and presented the translation mechanism. In particular, we showed how $\mathcal{D}_{\mathbf{L}}$ can be perfectly translated into $\mathcal{D}_{\mathbf{R}}$ using T and the correction table $C_{\mathbf{R}}$. The translation table is our model and therefore encoded on itself, i.e., $L(M)$ is replaced by $L(T)$. To encode a translation $\mathcal{D}_{\mathbf{L}\to\mathbf{R}}$ given T, we only need to encode $C_{\mathbf{R}}$: given the translation and correction tables, $\mathcal{D}_{\mathbf{R}}$ can be losslessly reconstructed from $\mathcal{D}_{\mathbf{L}}$.

Hence, the encoded length of the left-to-right translation given T becomes

$$L(\mathcal{D}_{\mathbf{L}\to\mathbf{R}} \mid T) = L(C_{\mathbf{R}} \mid T),$$

and vice versa for the other direction

$$L(\mathcal{D}_{\mathbf{L}\leftarrow\mathbf{R}} \mid T) = L(C_{\mathbf{L}} \mid T).$$

Given this, the task becomes that of finding the translation table that best compresses the translations between the two sides of a given two-view dataset.

Problem 1: Given a two-view dataset $\mathcal{D} = (\mathcal{D}_{\mathbf{L}}, \mathcal{D}_{\mathbf{R}})$ with corresponding translation $\mathcal{D}_{\mathbf{L}\leftrightarrow\mathbf{R}}$, find

$$\arg\min_{T\in\mathcal{T}} L(\mathcal{D}_{\mathbf{L}\leftrightarrow\mathbf{R}}, T) = L(T) + L(C_{\mathbf{L}} \mid T) + L(C_{\mathbf{R}} \mid T),$$

where $\mathcal{T}$ is the set of possible translation tables for $\mathcal{D}$, and $C_{\mathbf{R}}$ and $C_{\mathbf{L}}$ are the correction tables for $\mathcal{D}_{\mathbf{L}\to\mathbf{R}}$ given T and $\mathcal{D}_{\mathbf{L}\leftarrow\mathbf{R}}$ given T, respectively.

To complete the definition of our problem, we need to specify how to compute these encoded lengths.

4.1 Computing encoded lengths

To encode a translation table, we need to specify how to encode the itemsets it contains. The solution is to encode each item independently, assigning a code with length based on its empirical probability of occurring in the data. For each $I \in \mathcal{I}_{\mathbf{L}}$ this probability is given by

$$P(I \mid \mathcal{D}_{\mathbf{L}}) = \frac{|\{t \in \mathcal{D}_{\mathbf{L}} \mid I \in t\}|}{|\mathcal{D}|}.$$

From information theory, we have that the optimal code length corresponding to probability distribution P is $L(I \mid \mathcal{D}_{\mathbf{L}}) = -\log_2 P(I \mid \mathcal{D}_{\mathbf{L}})$. The encoded length of an itemset X is now given by

$$L(X \mid \mathcal{D}_{\mathbf{L}}) = \sum_{I\in X} L(I \mid \mathcal{D}_{\mathbf{L}}) = -\sum_{I\in X} \log_2 P(I \mid \mathcal{D}_{\mathbf{L}}).$$

We use this encoding for the itemsets over $\mathcal{I}_{\mathbf{L}}$ in the first column of a translation table, and similarly for itemsets over $\mathcal{I}_{\mathbf{R}}$ in the third column. For the directions, i.e., the second column of the table, a first bit indicates whether a rule is unidirectional or bidirectional, and a second bit represents the direction in case of a unidirectional rule. The length of a direction $\diamond$ is thus

$$L(\diamond) = \begin{cases} 1 & \text{if } \diamond = \leftrightarrow \\ 2 & \text{otherwise} \end{cases}$$

Summing up, the encoded length of a translation table T is given by

$$\begin{aligned} L(T) &= \sum_{X\diamond Y\in T} L(X \diamond Y), \text{ with} \\ L(X \diamond Y) &= L(X \mid \mathcal{D}_{\mathbf{L}}) + L(\diamond) + L(Y \mid \mathcal{D}_{\mathbf{R}}). \end{aligned}$$

For the encoding of the correction tables, note that we are only interested in the discovery of *cross-view* associations. This implies that we should not exploit any structure *within* one of the two views for compression, because that would prevent us from finding all cross-view structure. That is, we assume that we can capture all *relevant* structure in the translation table, and the contents of the correction table should be regarded as residue. Under this assumption, we can use the same 'independent' encoding for the itemsets in the correction tables as for the translation table, giving

$$L(C_{\mathbf{R}} \mid T) = \sum_{c\in C_{\mathbf{R}}} L(c \mid \mathcal{D}_{\mathbf{R}}).$$

Note that using the empirical data distribution of the complete dataset for the encoding of both the translation and correction tables may lead to an encoding that is not completely optimal: their distributions may deviate from the overall distribution. However, we accept and proceed with this choice for three reasons. First, as we will show later, translation tables are relatively small, hence using the optimal encoding would hardly change the results in practice. Second, we want compression to be the result only of structure captured by the rules, not of structure within the correction table. Third, this choice makes it possible to devise an exact algorithm for finding the best rule, which would otherwise be practically infeasible.

Encoding details For ease of presentation we did not mention three design choices so far, but they are important to make our encoding lossless. Requirements for

this are that the model can be transmitted in $L(T)$ bits, independent of the data, and that the translation can be fully constructed given T and the encoded data. We will now briefly discuss these details, and explain why we can safely ignore them in the remainder of the paper.

First, we need a code table that assigns a code to each item $I \in \mathcal{I}$. Since the lengths of these codes are based on their empirical probabilities in the data, $P(I \mid \mathcal{D})$, such a code table adds the same additive constant to $L(\mathcal{D}_{\mathbf{L}\leftrightarrow\mathbf{R}})$ for any M over $\mathcal{I}$. Therefore it can be disregarded when minimizing the total encoded size; for a fixed dataset $\mathcal{D}$ it is always the same.

Second, we do not mark the end of the rows in either of the correction tables, i.e., we do not use stop-characters. Instead, we assume given two sufficiently large frameworks that need to be filled out with the correct items upon decoding. Since such frameworks are the same for all correction tables for $\mathcal{D}$, this is again an additive constant we can disregard.

Last, each row of the translation table can be encoded and decoded by first encoding the direction and then the union of its two itemsets. Since we are only interested in the complexity of the content of the translation table, we disregard the complexity of its structure. That is, as for the correction tables, we assume a static framework that fits any possible translation table. The complexity of this framework is equal for any translation table T and dataset $\mathcal{D}$ over $\mathcal{I}$, and therefore we can also disregard this additive constant when calculating $L(\mathcal{D}, T)$.

5 The Translator algorithms

Given a dataset $\mathcal{D}$, there are $2^{|\mathcal{I}_\mathbf{L}|}-1$ (resp. $2^{|\mathcal{I}_\mathbf{R}|}-1$) non-empty itemsets for the left-hand side (resp. right-hand side). Since each pair of non-empty itemsets, one over $\mathcal{I}_\mathbf{L}$ and over $\mathcal{I}_\mathbf{R}$, can form three different rules $(\rightarrow, \leftarrow, \leftrightarrow)$, there are $|\mathcal{R}| = 3 \times (2^{|\mathcal{I}_\mathbf{L}|}-1) \times (2^{|\mathcal{I}_\mathbf{R}|}-1)$ possible rules. Without further assumptions on the number of rules in a translation table, each possible subset of $\mathcal{R}$ needs to be considered.

Since there is no structure that can be used to prune the search space, we resort to a greedy method, as is usual when the MDL principle is used for model selection [15]. Specifically, we start with an empty model and iteratively add the best rule until no rule that improves compression can be found. This parameter-free algorithm, dubbed Translator-Exact, allows to find good translation tables on datasets with a moderate number of attributes. We also introduce two variants that select rules from a fixed candidate set, making the approach applicable on larger datasets.

5.1 Computing the gain of a single rule

Before presenting our algorithms, we investigate how to efficiently compute the gain in compression that can be attained by adding a single rule to a translation table.

Each item in a correction table C occurs for one of two reasons: either the item is missing after translation and needs to be added, or it is introduced erroneously and needs to be removed. Hence, we can split C into two separate tables U and E, as follows. Let $U_\mathbf{R}$, for *Uncovered*, be a table such that $U^t_\mathbf{R} = t_\mathbf{R} \setminus t'_\mathbf{R}$ for each $t \in \mathcal{D}$, where $t'_\mathbf{R} = \textsc{Translate}(t_\mathbf{L}, T)$ as before. Similarly, let $E_\mathbf{R}$, for *Errors*, be a table such that $E^t_\mathbf{R} = t'_\mathbf{R} \setminus t_\mathbf{R}$ for each $t \in \mathcal{D}$. From this it follows that $U \cap E = \emptyset$ and $C = U \cup E$.

In practice, U initially equals $\mathcal{D}$; T is empty, and all items are uncovered. By adding rules to T, more items become covered, U becomes smaller, and thus the encoded length of C decreases. On the other hand, E is empty when we start and can only become larger (but to a lesser extent than the decrease of C, or rules would not be added). Once an error is inserted into E it cannot be removed by adding rules.

Now, let $\Delta_{\mathcal{D},T}(X \diamond Y)$ denote the decrease in total compressed size obtained by adding a rule $r = X \diamond Y$ to a translation table T, i.e. $\Delta_{\mathcal{D},T}(X \diamond Y) = L(\mathcal{D}_{\mathbf{L}\leftrightarrow\mathbf{R}}, T) - L(\mathcal{D}_{\mathbf{L}\leftrightarrow\mathbf{R}}, T \cup \{r\})$. Given the previous, this can be defined as the reduction in length of the correction table minus the length of the rule itself, as follows:

$$\Delta_{\mathcal{D},T}(X \diamond Y) = \Delta_{\mathcal{D}|T}(X \diamond Y) - L(X \diamond Y), \quad (1)$$

$$\Delta_{\mathcal{D}|T}(X \rightarrow Y) = \sum_{t \in \mathcal{D} \wedge X \subseteq t_\mathbf{L}} L(Y \cap U^t_\mathbf{R} \mid \mathcal{D}_\mathbf{R}) - L(Y \setminus (t_\mathbf{R} \cup E^t_\mathbf{R}) \mid \mathcal{D}_\mathbf{R}). \quad (2)$$

These equations follow directly from the definitions given so far. $\Delta_{\mathcal{D}|T}(X \leftarrow Y)$ is defined analogously with $\mathbf{L}$ and $\mathbf{R}$ reversed, and $\Delta_{\mathcal{D}|T}(X \leftrightarrow Y)$ is simply the sum of the two unidirectional variants. Given this, the best candidate rule is the one that maximizes $\Delta_{\mathcal{D},T}(X \diamond Y)$.

Algorithm 2 The Translator-Exact algorithm

Input: Two-view dataset $\mathcal{D}$
Output: Translation table T

```
T ← ∅
repeat
    r* ← arg max_{r∈R} Δ_{D,T}(r)
    if L(D, T ∪ {r*}) < L(D, T) then
        T ← T ∪ {r*}
until no rule added to T
return T
```

5.2 Iteratively finding the best rule

The idea of the Translator-Exact algorithm, presented in Algorithm 2, is to iteratively add the optimal rule to the current translation table. The greedy scheme starts from an empty translation table, and iteratively adds the rule that improves compression most, until no further improvement can be achieved. Note that the order of the rules in the table does not matter, and that provisional results can be inspected at any time.

To find the optimal rule r^* that maximizes the gain in compression, we use a search based on the Eclat

algorithm [24], traversing the pattern space depth-first while maintaining transaction sets for both X and Y and pruning where possible. Without additional pruning, all non-empty itemset pairs X and Y that occur in the data would be enumerated. For each such pair, all three possible rules are evaluated, i.e., one for each direction. To find r^* we only need to keep track of the best solution found so far.

To make search efficient, it is essential to find good solutions as early as possible, and to prune the search space based on the best solution so far. Unfortunately, $\Delta_{\mathcal{D},T}(X \diamond Y)$ is not (anti)monotonic. However, each XY should occur in the data and therefore all XY that do not occur in $\mathcal{D}$ are pruned (we do not consider rules for which either $X = \emptyset$ or $Y = \emptyset$, as these are not cross-view associations). Furthermore, from the definition of the gain of a rule in Equation 2, we observe that any positive gain must come from covering items that are currently uncovered. We can exploit this with a pruning technique similar to those used in high-utility itemset mining [23]. We trivially have that $L(Y \cap U^t_{\mathbf{R}} \mid \mathcal{D}_{\mathbf{R}}) \leq L(U^t_{\mathbf{R}} \mid \mathcal{D}_{\mathbf{R}})$ for any Y and $U^t_{\mathbf{R}}$, and will use it to derive an upper-bound.

That is, for each $t_{\mathbf{R}} \in \mathcal{D}$ the gain for that transaction is upper-bounded by the encoded size of its *uncovered items*. Let $tub(t_{\mathbf{R}})$ denote this transaction-based upper-bound, defined as $tub(t_{\mathbf{R}}) = L(U^t_{\mathbf{R}} \mid \mathcal{D}_{\mathbf{R}})$. Since for any transaction $tub(t_{\mathbf{R}})$ is constant during search for a single rule, these values are computed once prior to search. We can now check in which rows of the database a rule would be applied and sum the transaction-based bounds. For any rule $X \to Y$, this gives the following:

$$\Delta_{\mathcal{D},T}(X \to Y) \leq \sum_{t \in \mathcal{D} \text{ s.t. } t_{\mathbf{L}} \supseteq X} tub(t_{\mathbf{R}}).$$

For a given $X \diamond Y$, the bidirectional instantiation always has the highest potential gain, so we should sum the bounds for the two directions. We therefore have:

$$\Delta_{\mathcal{D},T}(X \diamond Y) \leq \sum_{t \in \mathcal{D} \text{ s.t. } t_{\mathbf{L}} \supseteq X} tub(t_{\mathbf{R}}) + \sum_{t \in \mathcal{D} \text{ s.t. } t_{\mathbf{R}} \supseteq Y} tub(t_{\mathbf{L}}).$$

Finally, we should take the size of the rule into account: extensions of the current rule will be at least as large as the current rule. We thus define the rule-based upper-bound, denoted rub, as

$$rub(X \diamond Y) = \sum_{t \in \mathcal{D} \text{ s.t. } t_{\mathbf{L}} \supseteq X} tub(t_{\mathbf{R}}) + \sum_{t \in \mathcal{D} \text{ s.t. } t_{\mathbf{R}} \supseteq Y} tub(t_{\mathbf{L}}) - L(X \leftrightarrow Y).$$

This bound is based on the supports of itemsets X and Y and decreases monotonically with either support cardinality. Therefore, $X \diamond Y$ and all its possible extensions can be safely pruned when the potential gain given by this bound is lower than the gain of the current best rule. That is, the pruning condition is $rub(X \diamond Y) \leq \Delta_{\mathcal{D},T}(r^*)$.

Prior to search, all $I \in \mathcal{I}$ are ordered descending by $tub(\{I\})$, which determines the order of the depth-first search. This helps find rules with high compression gain as quickly as possible and thus increases the amount of pruning that can be performed.

Finally, the gain for any rule $X \diamond Y$ can be quickly bounded by an upper-bound on the bidirectional rule:

$$qub(X \diamond Y) = |\text{supp}(X)|\, L(Y \mid \mathcal{D}_{\mathbf{R}}) + |\text{supp}(Y)|\, L(X \mid \mathcal{D}_{\mathbf{L}}) - L(X \leftrightarrow Y).$$

Although this gives no guarantee for rule extensions and thus cannot be used to prune the search space, it is useful to quickly determine whether computing $\Delta_{\mathcal{D},T}(X \to Y)$ is needed; this computation can be skipped when $qub(X \diamond Y) \leq \Delta_{\mathcal{D},T}(r^*)$.

Depending on the dataset and current translation table, exhaustive search for the best rule may still be computationally too intensive. Therefore, we also propose two faster, approximate methods.

Algorithm 3 The TRANSLATOR-SELECT algorithm

Input: Two-view dataset $\mathcal{D}$, integer k, candidates C
Output: Translation table T

```
T ← ∅
repeat
    R ← select k rules with highest Δ_{D,T}(r) from C
    used ← ∅
    for i = 1 ... k do
        consider R_i as X ◇ Y
        if X ∩ used = ∅ ∧ Y ∩ used = ∅ then
            if L(D, T ∪ {X ◇ Y}) < L(D, T) then
                T ← T ∪ {X ◇ Y}
                used ← used ∪ X ∪ Y
until no rule added to T
return T
```

5.3 Iteratively finding good rules

The second algorithm, dubbed TRANSLATOR-SELECT, strongly resembles its exact counterpart: it also greedily adds rules to the table, but does not guarantee to find the best possible rule in each iteration. Instead of generating candidate rules on-the-fly, it *selects* them from a fixed set of candidates. This set consists of two-view frequent itemsets, i.e., all itemsets Z for which $|\text{supp}(Z)| > minsup$, $Z \cap \mathcal{I}_{\mathbf{L}} \neq \emptyset$, and $Z \cap \mathcal{I}_{\mathbf{R}} \neq \emptyset$. These candidates are given as input, and can be mined using any frequent itemset mining algorithm that is modified such that each itemset contains items from both views.

TRANSLATOR-SELECT(k), presented in Algorithm 3, selects the top-k rules with regard to compression gain $\Delta_{\mathcal{D},T}$ among all possible rules that can be constructed from the candidate itemsets. Three rules can be constructed for each candidate itemset: one for each direction. When k is set to 1, this implies that the single best rule among the candidates is chosen in each iteration, similar to Algorithm 2. To further speed-up the process, it is possible to choose a larger k, so that multiple rules are selected in each iteration. The selected rules are

TABLE 1
Dataset properties. The densities of $\mathcal{D}_\mathbf{L}$ and $\mathcal{D}_\mathbf{R}$ are denoted by $d_\mathbf{L}$ and $d_\mathbf{R}$, respectively. $L(\mathcal{D}, \emptyset)$ denotes the uncompressed size (empty translation table).

Dataset	$\lvert\mathcal{D}\rvert$	$\lvert\mathcal{I}_\mathbf{L}\rvert$	$\lvert\mathcal{I}_\mathbf{R}\rvert$	$d_\mathbf{L}$	$d_\mathbf{R}$	$L(\mathcal{D}, \emptyset)$
Abalone[2]	4 177	27	31	0.185	0.129	170 748
Adult[1]	48 842	44	53	0.179	0.132	2 845 491
CAL500[3]	502	78	97	0.241	0.074	76 862
Car[1]	1 728	15	10	0.267	0.300	42 708
ChessKRvK[1]	28 056	24	34	0.167	0.088	889 555
Crime[2]	2 215	244	294	0.201	0.194	1 865 057
Elections	1 846	82	867	0.061	0.034	451 823
Emotions[3]	593	430	12	0.167	0.501	375 288
House[2]	435	26	24	0.347	0.334	31 625
Mammals	2 575	95	94	0.172	0.169	468 742
Nursery[1]	12 960	19	13	0.263	0.308	453 443
Tictactoe[1]	958	15	14	0.333	0.357	36 396
Wine[1]	178	35	33	0.200	0.212	11 608
Yeast[2]	1 484	24	26	0.167	0.192	52 697

added to the translation table one by one, but rules that contain an itemset that overlaps with an itemset of a rule previously added in the current iteration are discarded (to this aim, the set of *used* items is maintained). The reason for this is that the compression gain of such a rule has decreased, and it can therefore no longer be assumed to be part of the top-k for this round.

5.4 Greedily finding good rules

Our third method, called TRANSLATOR-GREEDY, employs single-pass filtering: given a dataset and a candidate set of frequent itemsets (ordered descendingly first by length, then by support in case of equality), it iteratively considers all itemsets one by one. For each itemset that is considered, compression gain is computed for each of the three possible rules, one for each direction. The corresponding rule with the largest gain is added if that gain is strictly positive. If there is no such rule for an itemset, it is discarded and never considered again. This very greedy procedure resembles the selection mechanism of KRIMP.

6 EXPERIMENTS

In this section we empirically evaluate the performance of the three TRANSLATOR methods, compare to existing methods, and present examples of obtained rules.

Data pre-processing. Except for Mammals and Elections, all datasets were obtained from the LUCS/KDD,[1] UCI,[2] and MULAN[3] repositories. Statistics of the datasets are presented in Table 1.

The LUCS/KDD repository provides Boolean datasets, the datasets from the other two repositories were pre-processed to make them Boolean: numerical attributes were discretized using five equal-height bins and each categorical attribute-value was converted into an item. For CAL500, the *genre*, *instruments* and *vocals* attributes are used as right-hand side, the rest as left-hand side. In Emotions, all audio features form the left-hand side, while the right-hand side consists of the different emotion labels. For the other repository datasets, the attributes were split such that the items were evenly distributed over two views having similar densities.

The Mammals dataset contains presence records of mammal species in Europe and is a natively Boolean real-world dataset [10]. We split the dataset into two views of similar sizes and densities.

Elections contains information about the candidates that participated in the 2011 Finnish parliamentary elections.[4] This dataset was collected from www.vaalikone.fi, the "election engine" of the Finnish newspaper *Helsingin Sanomat*. The left-hand side contains candidate properties such as party, age, and education, while the answers provided to 30 multiple-choice questions and the assigned importances form the right-hand side. We created an item for each attribute-value. Items that occurred in more than half of the transactions were discarded because they would result in many rules of little interest. Like CAL500, it is a good example of a natural two-view dataset, where one looks for associations between candidate profiles and political views.

Evaluation criteria. To compare the different methods, we primarily focus on three criteria: the number of rules found (denoted by $|T|$), the ratio between the compressed and uncompressed size of the translation (L% $= L(\mathcal{D}, T)/L(\mathcal{D}, \emptyset)$), and the runtime needed to mine the pattern set ($runtime$).

In addition, to facilitate a more extensive comparison to existing methods, we consider the relative size of the correction table and we introduce *maximum confidence*. According to our problem statement, the aim is to find a small set of patterns that accurately describes a translation. Hence, the number of rules should be low and the number of ones in the correction table should be small. We therefore define $|C|\%$ as the fraction of items in C to the total size of $\mathcal{D}$, i.e.,

$$|C|\% = \frac{|C|}{(|\mathcal{I}_\mathbf{L}| + |\mathcal{I}_\mathbf{R}|)|\mathcal{D}|}.$$

Note that $|C| = |U| + |E|$.

The *confidence* of a rule $X \to Y$ is normally defined as

$$\mathrm{c}(X \to Y) = \frac{|\mathrm{supp}(X \cup Y)|}{|\mathrm{supp}(X)|}.$$

However, in the current context we have both unidirectional and bidirectional rules. To avoid penalizing methods that induce bidirectional rules, we take the maximum confidence in either direction of a rule, and define c^+ as

$$\mathrm{c}^+(X \diamond Y) = max\{\mathrm{c}(X \to Y), \mathrm{c}(X \leftarrow Y)\}.$$

1. http://cgi.csc.liv.ac.uk/~frans/KDD/Software/LUCS-KDD-DN/DataSets/dataSets.html
2. http://archive.ics.uci.edu/ml/
3. http://mulan.sourceforge.net/
4. http://blogit.hs.fi/hsnext/hsn-vaalikone-on-nyt-avointa-tietoa

c^+ slightly resembles all-confidence [12], which also combines confidences for different "rule instantiations".

In the following, we will report average c^+ values computed over result sets. Note, however, that it is *not* our intention to discover rule sets that maximize average confidence. This could be easily achieved by, e.g., mining the top-k rules with respect to confidence. Unfortunately, due to redundancy in the pattern space, the top-k rules are usually very similar and therefore not of interest to a data analyst. Our aim is therefore to discover a *non-redundant* set of rules that accurately describe the *complete* translation; confidence should be reasonably high, but our aims are better captured by the combination of the other evaluation criteria.

Implementation. We implemented TRANSLATOR in C++. The source code, datasets, and the splits required to be able to reproduce the results are publicly available[5].

6.1 Comparison of search strategies

We first compare the three different variants of the TRANSLATOR algorithm. As candidate sets for both TRANSLATOR-SELECT and TRANSLATOR-GREEDY we use closed frequent two-view itemsets up to a given minimum support threshold. Furthermore, TRANSLATOR-SELECT(k) is evaluated for $k = 1$ and $k = 25$.

For the first batch of experiments we set the lowest possible minimum support threshold, i.e., $minsup = 1$ (threshold not needed for TRANSLATOR-EXACT). Consequently, for these experiments we use only datasets with a moderate numbers of items. The results, presented in the top half of Table 2, show large variations in both compression ratio and runtime, which both heavily depend on the characteristics of the dataset. We observe that using compression as stopping criterion results in relatively few rules: in all cases, there are much fewer rules than there are transactions in the dataset. Together with the observation that compression ratios up to 54% are attained, this implies that rules that generalize well are found. On the other hand, some datasets can hardly be compressed, indicating that there are only few cross-view associations and/or that they do not cover large areas of the data. This is an advantage of the compression-based translation approach that we advocate: if there is little or no structure connecting the two views, this will be reflected in the attained compression ratios. Note, however, that also other properties of the data influence compression. For example, dense data generally results in better compression than sparse data (see Table 1).

The four method instances all yield similar compression ratios and numbers of rules. However, TRANSLATOR-EXACT needs to dynamically construct and explore large parts of the search space in each iteration, and this results in relatively long runtimes. This is caused by the fact that the pruning strategies are only effective in the first few iterations. After that, the gain in compression that a single rule can achieve decreases significantly, so that a much larger part of the search space needs to be explored. This is demonstrated by closer inspection of the construction of translation tables (see Section 6.2).

TRANSLATOR-SELECT and TRANSLATOR-GREEDY do not suffer from the aforementioned problem, as they generate a candidate set once and only perform candidate testing. TRANSLATOR-SELECT tests all candidates in each iteration, TRANSLATOR-GREEDY tests each candidate exactly once. The compression ratios obtained by TRANSLATOR-SELECT are slightly worse than those obtained by the exact method, because it only considers *closed* itemsets as candidates. This could be addressed by using all itemsets, but this would lead to much larger candidate sets and hence longer runtimes. TRANSLATOR-GREEDY is clearly the fastest, and often approximates the best solution quite well. However, there are exceptions to this. For `Wine`, for example, the compression ratios obtained by TRANSLATOR-EXACT and TRANSLATOR-SELECT are 10% lower (= better) than those obtained by TRANSLATOR-GREEDY.

We now shift our focus to the lower half of Table 2, which presents results obtained on the larger datasets. We do not have results for the exact method because it takes too long to finish on these datasets. We fix $minsup$ such that the number of candidates remains manageable (between 10K and 200K). We again observe varying results dependent on the data. Unsurprisingly, the TRANSLATOR-GREEDY method is much faster than the TRANSLATOR-SELECT alternatives, but in some cases this also results in poor compression. For example, on `House` it only achieves a compression ratio of 71.45%, compared to 49.26% obtained by TRANSLATOR-SELECT(1).

Discussion. As expected, the three proposed TRANSLATOR variants offer different trade-offs between runtime and solution quality. TRANSLATOR-EXACT is parameter-free, iteratively adds the optimal rule to the table, and attains the best compression ratios, but can only be used on small datasets. By iteratively selecting rules from a set of candidates, TRANSLATOR-SELECT is substantially faster and in practice approximates the best possible compression ratio very well. As such, it provides a very good trade-off between compression and runtime. Depending on the dataset, choosing a larger k can be useful to speed-up the search. For example, on `Crime` compression remains practically the same while runtime decreases from 5h 15m to 1h 27m. The third variant, TRANSLATOR-GREEDY, greedily selects rules in a single pass over a set of candidates and is the fastest of the three, but does not always find a good solution. This may be the best choice when the dataset is very large.

6.2 Construction of a translation table

Here we zoom in on TRANSLATOR-SELECT(1), the search strategy that provides the best trade-off in terms of compression and runtime, and the `House` dataset. For

5. http://patternsthatmatter.org/software.php

TABLE 2
Comparison of TRANSLATOR-EXACT, TRANSLATOR-SELECT, and TRANSLATOR-GREEDY. For each experiment, we report the number of obtained rules $|T|$, the compression ratio L%=$L(\mathcal{D}, T)/L(\mathcal{D}, \emptyset)$, and the runtime.

		T-EXACT			T-SELECT(1)			T-SELECT(25)			T-GREEDY		
Dataset	$msup$	$\|T\|$	L%	runtime	$\|T\|$	L%	runtime	$\|T\|$	L%	runtime	$\|T\|$	L%	runtime
Abalone	1	88	**54.81**	3h 22m	86	54.86	27m 58s	86	54.95	10m 51s	114	57.75	19s
Car	1	12	**94.18**	1m 14s	9	94.67	28s	9	94.67	20s	12	95.27	3s
ChessKRvK	1	320	**94.89**	2d 47m	311	94.94	17h 19m	315	94.95	6h 22m	314	95.60	3m 21s
Nursery	1	28	**98.36**	3h 19m	27	**98.36**	1h 47m	27	**98.36**	1h 15m	19	98.83	3m 46s
Tictactoe	1	61	85.18	35m 8s	64	85.20	8m 16s	66	**84.86**	3m 31s	73	90.97	7s
Wine	1	38	**67.99**	1h 22m	27	69.15	15s	30	69.10	8s	48	79.98	$<$ 1s
Yeast	1	49	**81.99**	45m 52s	32	82.73	2m 16s	32	82.73	2m 15s	38	83.00	4s
Adult	4885		—		8	**54.29**	49m 48s	8	**54.29**	49m 14s	19	55.50	7m 8s
CAL500	20		—		59	**86.45**	36m 6s	60	86.48	13m 5s	92	88.88	40s
Crime	200		—		144	**87.45**	5h 15m	146	87.47	1h 27m	183	88.51	2m 7s
Elections	47		—		80	93.28	35m 46s	83	**93.27**	12m 19s	132	94.49	28s
Emotions	40		—		22	97.35	20m 24s	24	**97.34**	14m 8s	37	97.54	54s
House	8		—		37	**49.26**	14m 31s	37	49.27	7m 49s	50	71.45	23s
Mammals	773		—		55	**68.23**	58m 21s	56	68.31	29m 33s	39	85.85	1m 4s

this combination we examine the changes in encoded lengths and coverage while rules are iteratively added to the translation table. Fig. 2 (top) shows how the numbers of uncovered ones ($|U|$) and errors ($|E|$) evolve, for both sides. Fig. 2 (bottom) shows how the encoded lengths evolve, i.e., the encoded length of the left-to-right translation $L(\mathcal{D}_{\mathbf{L}\to\mathbf{R}} \mid T)$, the encoded length of the right-to-left translation $L(\mathcal{D}_{\mathbf{L}\leftarrow\mathbf{R}} \mid T)$, the length of the translation table $L(T)$, and the total encoded length of the bidirectional translation $L(\mathcal{D}_{\mathbf{L}\leftrightarrow\mathbf{R}}, T)$, which is the sum of the three parts.

As expected, the number of uncovered items quickly drops as rules are added to the translation table, while the number of errors slowly rises. As new rules are added to the translation table, the encoded lengths of both sides decrease accordingly. We note as a general trend, that compression gain per rule decreases quite quickly. This is also what we observed with the exact search strategy, and what limited the power of the pruning scheme. As a consequence, exact search is most attractive when one is only interested in few rules.

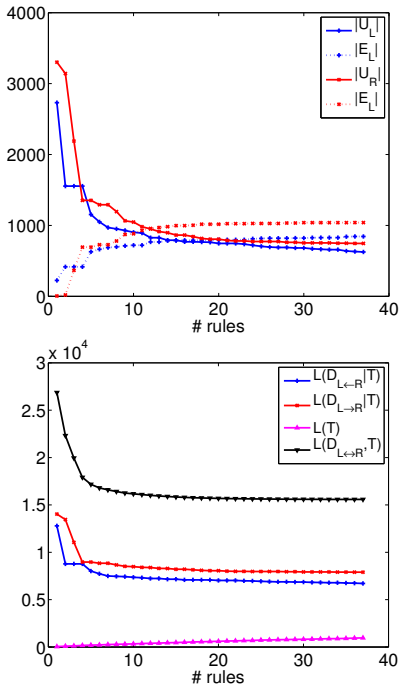

Fig. 2. Evolution of the number of uncovered and erroneous items (top), and encoded lengths (bottom) during the construction of a translation table for House with TRANSLATOR-SELECT(1).

6.3 Comparison with other approaches

Association rule mining, redescription mining, and KRIMP have each been designed to tackle different a problem from the one we consider in this paper. Here we empirically demonstrate that TRANSLATOR provides more compact and complete descriptions of the structure in two-view data than these three methods.

Association rule mining. We first consider the traditional association rule mining task [1], for which we need to choose minimum confidence and support thresholds before mining. To ensure that we can find similar cross-view associations as with our methods, we use the lowest c^+ and $|supp|$ values for any rules found in our translation tables as respective thresholds (per dataset). Using these tuned thresholds, we mine all cross-view association rules of either direction using an adapted miner that only mines rules spanning the two views. This results in several thousands of association rules per dataset (up to 153 609 for House), i.e., up to several orders of magnitude more than are selected by our methods (up to 311 rules, for ChessKRvK). It is evident that it is impossible for a data analyst to manually inspect and interpret thousands and thousands of rules.

Hence, to address the pattern explosion, we resort to a technique designed to strongly reduce the number of association rules. In particular, we consider significant

TABLE 3
Comparing TRANSLATOR to MAGNUM OPUS, REREMI, and KRIMP. We report the number of rules $|T|$, their average length (l), the relative sizes of the correction tables ($|C|\%$), the maximum confidences c^+ averaged over the pattern set, and compression ratio L%.

	T-SELECT(1)					MAGNUM OPUS					REREMI					KRIMP				
Dataset	$\|T\|$	l	$\|C\|\%$	$\overline{\mathrm{c}^+}$	$L\%$	$\|T\|$	l	$\|C\|\%$	$\overline{\mathrm{c}^+}$	$L\%$	$\|T\|$	l	$\|C\|\%$	$\overline{\mathrm{c}^+}$	$L\%$	$\|T\|$	l	$\|C\|\%$	$\overline{\mathrm{c}^+}$	$L\%$
Abalone	86	5.22	0.08	0.60	54.86	143	2.59	0.12	0.83	76.91	35	3.66	0.11	0.73	76.16	352	5.22	0.45	0.37	330.46
Adult	8	6.63	0.06	0.79	54.29	174	2.99	0.18	0.75	136.83	12	3.00	0.14	0.76	102.58	312	9.05	0.10	0.69	86.59
CAL500	59	4.19	0.12	0.63	86.45	101	2.05	0.20	0.75	141.61	25	3.12	0.15	0.61	104.40	204	3.39	0.37	0.84	272.73
Car	9	3.44	0.26	0.60	94.67	12	2.33	0.32	0.77	123.13	5	2.00	0.30	0.68	115.28	109	3.94	0.67	0.33	271.91
ChessKRvK	311	4.50	0.11	0.70	94.94	140	2.56	0.32	0.42	300.75	14	2.36	0.16	0.40	134.18	1619	4.34	0.83	0.27	816.34
Crime	144	3.84	0.17	0.74	87.45	115	2.26	0.19	0.75	95.88	41	2.93	0.19	0.75	93.66	742	3.02	0.60	0.89	307.33
Elections	80	3.53	0.04	0.58	93.28	151	2.35	0.04	0.58	110.14	19	2.68	0.04	0.62	101.51	792	2.86	0.16	0.88	445.39
Emotions	22	5.64	0.17	0.73	97.35	140	2.30	0.19	0.75	107.99	30	3.27	0.18	0.66	101.26	524	2.45	0.58	0.96	342.93
House	37	5.89	0.15	0.72	49.26	145	2.49	0.24	0.85	77.20	26	3.50	0.19	0.81	59.64	95	4.63	0.51	0.80	200.03
Mammals	55	4.56	0.11	0.86	68.23	189	3.16	0.14	0.87	84.03	43	4.53	0.12	0.83	72.95	157	4.47	0.16	0.88	97.25
Nursery	27	3.85	0.28	0.57	98.36	48	2.65	0.36	0.46	136.19	4	2.25	0.29	0.43	105.15	232	5.29	0.69	0.27	265.48
Tictactoe	64	4.80	0.28	0.49	85.20	28	2.43	0.36	0.62	105.37	14	2.00	0.34	0.57	99.08	165	3.90	0.64	0.46	212.74
Wine	27	6.44	0.12	0.79	69.15	51	2.12	0.19	0.76	99.09	20	3.00	0.16	0.76	81.28	57	3.98	0.29	0.68	165.24
Yeast	32	5.78	0.13	0.74	82.73	27	2.30	0.23	0.66	138.19	15	2.53	0.21	0.56	121.16	127	4.91	0.54	0.48	395.80
Average	68	4.88	0.15	0.68	79.73	104	2.47	0.22	0.70	123.81	21	2.92	0.18	0.66	97.74	391	4.39	0.47	0.63	300.73

rule discovery [21], which has been implemented in the MAGNUM OPUS[6] mining tool. MAGNUM OPUS is a flexible tool that provides a number of adjustable parameters. In particular, it allows to specify the items that can occur on either side of the rule. Thus, in order to obtain comparable output for the two-view setting, we apply MAGNUM OPUS twice on every dataset, once requiring the antecedent to consist only of items from the left-hand side and the consequent only items from the right-hand side, once with the reverse requirement. Finally, the two sets of rules are merged, with rules found in both sets resulting into a single bidirectional rule. Apart from that, default settings are used.

The results are shown in Table 3. Clearly, the rule sets obtained with MAGNUM OPUS are of more interest than the raw set of associations. Still, they are less compact than those obtained with TRANSLATOR, which typically contain fewer rules involving more items. MAGNUM OPUS achieves good average c^+, sometimes above that of TRANSLATOR. The price for this higher confidence, however, is a larger number of incorrectly translated items. This results in relatively large correction tables, indicated by $|C|\%$, and poor compression ratios, especially in the sparser datasets. This is strongly reflected in the average compression ratios given in the bottom row.

Redescription Mining. Next, we mined redescriptions with the REREMI algorithm [6], restricted to monotone conjunctions. This algorithm selects (bidirectional) redescriptions based on ad-hoc pruning, driven primarily by accuracy. Table 3 shows that REREMI finds rules with average c^+ values that are generally on par to those of TRANSLATOR. The result sets contain small numbers of rules over few items. However, they fail to explain all of the two-view structure in the data, as evidenced by the larger correction tables and the poor compression ratios, sometimes even inflating the data (compression ratios above 100% for eight datasets). To summarize, RM aims to find individual bidirectional rules of high accuracy, but these are likely to be redundant and do not explain all associations across the two views of the data (and certainly not unidirectional ones).

Visual comparison of rule sets. These differences between the results returned by TRANSLATOR-SELECT(1), MAGNUM OPUS, and REREMI are also apparent in the visualizations of the rule sets obtained for CAL500 and House, given in Fig. 3. In each of the six graphs, the nodes on the left-hand side and on the right-hand side represent all items from either sides, and nodes in the middle represent the rules. Each rule is connected to the items it contains, where the line is drawn in grey if the implication is only away from the item, and in black otherwise (bidirectional). MAGNUM OPUS returns more rules involving fewer items than TRANSLATOR-SELECT(1) and REREMI. The rules from the latter method involve a less diverse set of items and all rules are exclusively bidirectional. Our approach, on the other hand, returns bidirectional as well as unidirectional rules that all contain a mixture of items. In this way, translation tables offer a more complete yet succinct description of the translation, compared to MAGNUM OPUS and REREMI.

The KRIMP algorithm. Finally, we briefly compare to KRIMP. Although both aim to induce pattern based models using the MDL principle, KRIMP and our proposed approach reveal different aspects of the data. In particular, KRIMP uses itemsets and TRANSLATOR uses rules, which has as consequence that a direct comparison is impossible. Nevertheless, we can still show that the itemsets found by KRIMP do not capture the same associations as the rules discovered by TRANSLATOR.

For this, we transform a set of itemsets into a translation table. Note that this necessarily implies that we

6. http://www.giwebb.com/

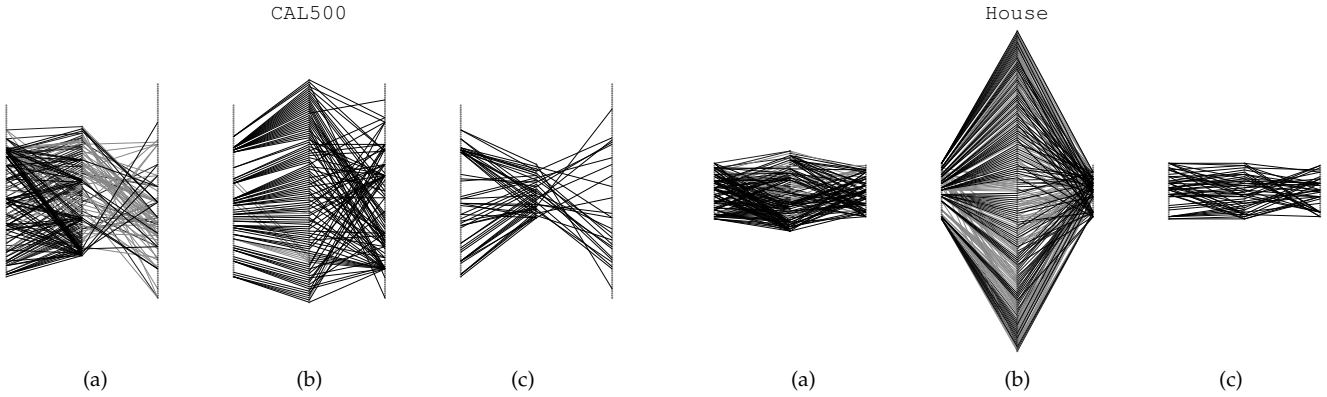

Fig. 3. Visualization of the rules found on CAL500 (left) and House (right) with TRANSLATOR-SELECT(1) (a), MAGNUM OPUS (b) and REREMI (c). In each visualization, the left- and rightmost nodes represent the left-hand and right-hand side items respectively, the nodes in the middle represents the rules. Each edge (line) indicates that a rule contains the corresponding item; gray indicates that the rule is unidirectional, black that it is bidirectional.

T-SELECT(1)	c^+
el-salvador-aid:Y ↔ crime:Y ∧ mx-missile:N ∧ synfuels-corporation-cutback:N ∧ education-spending:Y ∧ superfund-right-to-sue:Y ∧ duty-free-exports:N ∧ export-administration-act-south-africa:Y	1.00
democrat ∧ physician-fee-freeze:N ∧ el-salvador-aid:N ∧ aid-to-nicaraguan-contras:Y ↔ mx-missile:Y ∧ crime:N ∧ synfuels-corporation-cutback:N ∧ education-spending:N ∧ superfund-right-to-sue:N ∧ duty-free-exports:Y ∧ export-administration-act-south-africa:Y	1.00
democrat ∧ physician-fee-freeze:N ← mx-missile:? ∧ immigration:N	1.00
MAGNUM OPUS	
democrat ↔ crime:N	0.98
el-salvador-aid:Y ← mx-missile:N ∧ education-spending:Y	0.97
el-salvador-aid:N ← crime:N ∧ mx-missile:Y	0.97
REREMI	
democrat ↔ education-spending:N	0.91
democrat el-salvador-aid:N ↔ crime:N	0.89
el-salvador-aid:Y ↔ mx-missile:N	0.89

Fig. 4. Example rules mined from House.

T-SELECT(1)	c^+
Red Fox ↔ European Hedgehog ∧ Least Weasel	0.98
Bank Vole ↔ European Water Vole ∧ Common Shrew ∧ Eurasian Pygmy Shrew ∧ Red Squirrel ∧ Brown rat ∧ Least Weasel	0.97
Brown long-eared bat ∧ Field Vole ∧ European Badger → Eurasian Pygmy Shrew	0.97
MAGNUM OPUS	
European Polecat ∧ European Mole → European Hare	0.98
European Badger ∧ European Mole → European Hare	0.97
Eurasian Water Shrew ∧ European Mole → European Hare	0.97
REREMI	
European Mole ∧ Red Fox ↔ Harvest Mouse ∧ European Hare	0.92
Brown long-eared bat ∧ European Mole ∧ Red Fox ↔ Eurasian Pygmy Shrew ∧ European Hare ∧ Least Weasel	0.91
Bank Vole ∧ Red Fox ↔ European Pine Marten ∧ Red Squirrel	0.91

Fig. 5. Example rules mined from Mammals.

use the translation and compression schemes as defined in Sections 3 and 4 for this comparison, as we did for computing $|C|\%$ and $L\%$ in the previous comparisons. That is, KRIMP code tables mined from the joint two-view datasets are directly interpreted as bidirectional rules and put in a translation table. Then, compression is computed using the scheme introduced in this paper.

The results in Table 3 clearly demonstrate that KRIMP aims at finding associations that are very different from those that TRANSLATOR identifies. KRIMP finds many more associations, and when treated as translation table the complete set of associations results in extremely bad compression: compression ratios range up to 816.34%, implying that the translation is inflated to more than eight times its original encoded size. This demonstrates that the associations found by KRIMP are not a good solution to the task considered in this paper.

6.4 Example rules

To conclude this section, we turn to a qualitative assessment of the rules found by the different algorithms.

Figures 4 and 5 show the top three rules obtained with TRANSLATOR-SELECT(1), MAGNUM OPUS and REREMI for House and Mammals respectively. Note that we do not consider KRIMP here, because it does not produce rules and because of its bad quantitative performance.

The House dataset pertains to politics, with rules capturing associations between votes by U.S. House of Representatives Congressmen on key topics of the 2nd Congress session in 1984. 'N', 'Y' and '?' stand for *yea*, *nay*, and unknown disposition, respectively. For instance, the third rule from TRANSLATOR-SELECT(1) indicates that congressmen who opposed the immigration bill and did not take position on the vote about the MX-missiles program are democrats who also opposed the freeze on physician's fee, and this holds with confidence one.

The Mammals dataset, on the other hand, originates from biology and ecology. The obtained rules provide information about combinations of mammals species

T-SELECT(1)	c^+
¬ Emotion:Light-Playful ∧ Song:Quality ∧ Song:Texture-Electric ∧ Usage:Driving → Genre:Rock	0.72
¬ Emotion:Loving-Romantic ∧ ¬ Emotion:Tender-Soft ∧ ¬ Emotion:Touching-Loving ∧ Song:High-Energy ∧ Song:Texture-Electric ∧ ¬ Song:Very-Danceable ↔ Genre:Rock	0.68
¬ Emotion:Sad ← Genre:Rock ∧ Instrument:Backing-Vocals ∧ Instrument:Male-Lead-Vocals	0.64
Usage:Driving ← Genre:Alternative ∧ Genre:Rock ∧ Instrument:Male-Lead-Vocals	0.62
MAGNUM OPUS	
Song:Texture-Electric ↔ Genre:Rock	0.86
¬ Emotion:Loving-Romantic ↔ Genre:Rock	0.65
¬ Emotion:Touching-Loving ↔ Genre:Rock	0.64
¬ Emotion:Tender-Soft ↔ Genre:Rock	0.62
¬ Emotion:Calming-Soothing ↔ Genre:Rock	0.51
REREMI	
¬ Emotion:Touching-Loving ∧ Song:Texture-Electric ↔ Genre:Rock	0.57

Fig. 6. Example rules mined from CAL500.

that inhabit the same areas. According to the first rule returned by REREMI, the Harvest Mouse and the European Hare can commonly be found in areas where both the European Mole and the Red Fox live, and vice versa.

The characteristics of the algorithms observed in the quantitative results are also noticeable here: in both cases, the rules output by our algorithm tend to be *longer and less redundant* than those found by the other methods.

It is also interesting to look at the different rules involving a given specific item. In Fig. 6 we focus on rock music, that is, we present all rules from CAL500 containing the item 'Genre:Rock' obtained by each of the three methods. We observe that the second rule found by TRANSLATOR-SELECT(1) is a superset of the single rule obtained with REREMI. It combines all but the weakest rules returned by MAGNUM OPUS, with some additional items, yielding a relatively high maximum confidence of 0.64. The remaining rules provide further rich characterizations of rock music in different contexts, in the form of unidirectional associations.

Finally, Fig. 7 presents rules obtained for Elections. The four rules in this anecdotal example clearly conform to the common understanding of the Finnish political landscape. That is, the first rule highlights views on defense, finance, development aid and nuclear energy that are commonly ascribed to the Green party. The second rule conveys that candidates for Change 2011, a Finnish party known for being critical towards immigration, think that current immigration policy is too loose. Observe that the rule is not bidirectional, implying that there are also candidates for other parties that have this opinion. This shows that having both bidirectional and unidirectional rules is useful. Furthermore, the rules are generally easy to interpret by domain experts.

Overall, we conclude that translation tables have substantially different properties from the results of the related methods considered in this paper, and that TRANSLATOR provides better results to the problem considered: smaller sets of rules that provide a more complete characterization of the associations across the two data views.

T-SELECT(1)	c^+
party = 'Green League' ↔	0.81
Question: The new government might decide to cut the public expenditure. Below are some cost-cutting measures that have been proposed. Which one of these would you select first? **Answer:** Military spending should be reduced. ∧ **Q:** Finland, along with the other Eurozone countries, has helped to save other Eurozone countries with hundreds of billions of euros worth of support. In Spring 2010, Finland agreed to loan 1.6 billion euros to Greece. Furthermore, Finland agreed to guarantee European Financial Stability Facility's 750 billion euro loaned capital with over 8 billion euros. This might not be enough in the long run. Which of the following claims is closest to your opinions? **A:** Supporting the countries that were in trouble was in Finlands interests, as bankruptcy of any country in the Eurozone would endanger the economy in the whole zone. ∧ **Q:** The financial crisis has increased the demands to tax the financial sector and to force it to take part in paying the damages. The European Commission has proposed the Financial Transaction Tax (FTT) to tax bond, stock, currency, and derivative transactions. Which of the following claims is closest to your opinions? **A:** EU should collect transaction taxes even if the rest of the world does not. ∧ **Q:** Which of the following statements best describes your views regarding development aid? **A:** Finland must increase its commitment to the development to 0.7 percent during the next legislature. ∧ **Q:** In Spring 2009, the government grant permissions to two new nuclear power plants. The third applicant, Fortum, did not receive the permission, but is hopeful to get granted a permission to replace two of their reactors in Loviisa. Should this permission be granted? **Importance:** high	
party = 'Change 2011' →	0.95
Q: The 2007-2011 electoral term saw Finnish immigration policy becoming more strict. What is your opinion about the current immigration policy of Finland? **A:** It is too loose.	
gender = 'female' ←	0.64
Q: In Fall 2010, the permission to own firearms was made harder to obtain; for example, the minimum age for owning a handgun was raised to 20 years. What should be done to the guns legislation? **A:** Storing handguns at home should be illegal. ∧ **Q:** Child allowance is paid for each child living in Finland until they are 17 years old, irrespective of the parents' income. What should be done for child allowances? **Importance:** high	
party = 'Social Democratic Party' ∧ *Municipal Rep.* ←	0.34
Q: From the begin of the year, Russia has banned foreigns to own real estates from its border regions. In Finland, there is virtually no restrictions on foreign land owners and in recent years, Russians have bought thousands of real estates from Finland. What would be the proper course of action? **A:** Finland should restrict Russians right to buy land and real estates to achieve parity in the legislation. ∧ **Q:** Should Finland apply for NATO membership? **A:** Not at least during the next election term.	

Fig. 7. Example rules mined from Elections.

7 CONCLUSIONS

We introduced the exploratory data mining task of finding small and non-redundant sets of associations that provide insight in how the two sides of two-view datasets are related. To this end, we proposed a translation-based approach that uses rules to translate one view into the other and vice versa. These translation rules can be either unidirectional or bidirectional, and a set of rules together forms a translation table. Our

approach generalizes existing methods such as association rule mining and redescription mining, but also avoids redundancy by mining a *set of patterns* rather than individual patterns. For this purpose we introduced a model selection method based on the MDL principle.

We presented three TRANSLATOR algorithms for inducing translation tables. The exact variant is parameter-free and iteratively adds the optimal rule to the table, while the second variant iteratively selects the best rule from a fixed set of candidates and is therefore substantially faster. Nevertheless, in practice it approximates the best possible compression ratio very well. The third variant greedily selects rules in a single pass over a set of candidates and is the fastest of the three, but does not always find a good solution.

The experiments demonstrate that only modest numbers of rules are needed to characterize any cross-view associations in the two-view data. In general, having both bidirectional and unidirectional rules proves useful; the obtained rules are easy to inspect, non-redundant, and provide insight in the data.

Directions for future work include, for instance, extending this approach to other data types and to cases with more than two views. This requires designing a suitable pattern based encoding for the data, and a procedure to enumerate the corresponding search space.

Acknowledgments Matthijs van Leeuwen is supported by a Postdoctoral Fellowship of the Research Foundation Flanders (FWO). Most of the work was done while Esther Galbrun was a doctoral student at the University of Helsinki, with support from the Academy of Finland, grants 125637 and 255675. The authors would like to thank Antti Ukkonen for his help with the election data.

REFERENCES

[1] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *Proc. of the 1993 ACM SIGMOD International Conference on Management of Data (SIGMOD'93)*, pages 207–216. ACM Press, 1993.

[2] Steffen Bickel and Tobias Scheffer. Multi-view clustering. In *Proc. of the 4th IEEE International Conference on Data Mining (ICDM'04)*, pages 19–26. IEEE Computer Society, 2004.

[3] Tijl De Bie and Eirini Spyropoulou. A theoretical framework for exploratory data mining: Recent insights and challenges ahead. In *Proc. of the European Conference on Machine Learning and Knowledge Discovery in Databases, Part III, (ECML/PKDD'13)*, pages 612–616. Springer, 2013.

[4] Luc De Raedt and Albrecht Zimmermann. Constraint-based pattern set mining. In *Proc. of the 7th SIAM International Conference on Data Mining (SDM'07)*, pages 237–248. SIAM / Omnipress, 2007.

[5] Christos Faloutsos and Vasileios Megalooikonomou. On data mining, compression, and kolmogorov complexity. *Data Mining and Knowledge Discovery*, 15(1):3–20, August 2007.

[6] Esther Galbrun and Pauli Miettinen. From black and white to full color: extending redescription mining outside the boolean world. *Statistical Analysis and Data Mining*, 5(4):284–303, 2012.

[7] Peter D. Grünwald. *The Minimum Description Length Principle*. MIT Press, 2007.

[8] Hans-Peter Kriegel, Peer Kröger, and Arthur Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data*, 3(1), 2009.

[9] Dennis Leman, Ad Feelders, and Arno Knobbe. Exceptional model mining. In *Proc. of the European Conference on Machine Learning and Knowledge Discovery in Databases, Part II, (ECML/PKDD'08)*, pages 1–16. Springer, 2008.

[10] Anthony J. Mitchell-Jones et al. *The Atlas of European Mammals*. Academic Press, London, 1999.

[11] Emmanuel Müller, Thomas Seidl, Suresh Venkatasubramanian, and Arthur Zimek, editors. *Workshop at SDM 2012: 3rd MultiClust Workshop - Discovering, Summarizing and Using Multiple Clusterings*, 2012.

[12] Edward Omiecinski. Alternative interest measures for mining associations in databases. *IEEE Transactions on Knowledge and Data Engineering*, 15(1):57–69, 2003.

[13] Laxmi Parida and Naren Ramakrishnan. Redescription mining: Structure theory and algorithms. In *Proc. of the 20th National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference (AAAI'05)*, pages 837–844. AAAI Press / The MIT Press, 2005.

[14] Stefan Rüping and Tobias Scheffer, editors. *Workshop at ICML 2005: Learning with Multiple Views*, 2005.

[15] Arno Siebes, Jilles Vreeken, and Matthijs van Leeuwen. Item sets that compress. In *Proc. of the 6th SIAM International Conference on Data Mining (SDM'06)*, pages 395–406. SIAM / Omnipress, 2006.

[16] Nikolaj Tatti and Jilles Vreeken. The long and the short of it: summarising event sequences with serial episodes. In *Proc. of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '12)*, pages 462–470. ACM, 2012.

[17] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook*, pages 667–685. 2010.

[18] Matthijs van Leeuwen, Jilles Vreeken, and Arno Siebes. Identifying the components. *Data Mining and Knowledge Discovery*, 19(2):176–193, 2009.

[19] Jilles Vreeken, Matthijs van Leeuwen, and Arno Siebes. Krimp: mining itemsets that compress. *Data Mining and Knowledge Discovery*, 23(1):169–214, 2011.

[20] Jilles Vreeken and Arthur Zimek. When pattern met subspace cluster. In Emmanuel Müller, Stephan Günnemann, Ira Assent, and Thomas Seidl, editors, *MultiClust@ECML/PKDD*, volume 772 of *CEUR Workshop Proc.*, pages 7–18. CEUR-WS.org, 2011.

[21] Geoffrey I. Webb. Discovering significant patterns. *Machine Learning*, 68(1):1–33, 2007.

[22] Bernd Wiswedel, Frank Höppner, and Michael R. Berthold. Learning in parallel universes. *Data Mining and Knowledge Discovery*, 21(1):130–152, 2010.

[23] Cheng Wei Wu, Bai-En Shie, Vincent S. Tseng, and Philip S. Yu. Mining top-k high utility itemsets. In *Proc. of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (KDD'12)*, pages 78–86. ACM, 2012.

[24] Mohammed J Zaki, Srinivasan Parthasarathy, Mitsunori Ogihara, and Wei Li. New algorithms for fast discovery of association rules. In *Proc. of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD'97)*, pages 283–286. ACM, 1997.

[25] Mohammed Javeed Zaki. Mining non-redundant association rules. *Data Mining and Knowledge Discovery*, 9(3):223–248, 2004.

Matthijs van Leeuwen is a postdoctoral researcher in the Machine Learning group at KU Leuven, Belgium. He received his PhD in Computer Science in 2010 at Universiteit Utrecht, the Netherlands. He previously was a postdoctoral researcher in Utrecht for almost two years. His main research interest is exploratory data mining, often based on pattern (set) mining in combination with (algorithmic) information theory. www.patternsthatmatter.org

Esther Galbrun is a postdoctoral researcher at the Computer Science department of Boston University, US-MA. She was previously a doctoral student at the Helsinki Institute for Information Technology (HIIT) and the University of Helsinki, Finland, from which she received a PhD in Computer Science in 2014. Her research interests lie in algorithmic data analysis in general and redescription mining in particular.