

Features for Learning Local Patterns in Time-Stamped Data

Katharina Morik and Hanna Köpcke

Univ. Dortmund, Computer Science Department, LS VIII
morik@ls8.informatik.uni-dortmund.de
<http://www-ai.cs.uni-dortmund.de>

Abstract. Time-stamped data occur frequently in real-world databases. The goal of analysing time-stamped data is very often to find a small group of objects (customers, machine parts,...) which is important for the business at hand. In contrast, the majority of objects obey well-known rules and is not of interest for the analysis. In terms of a classification task, the small group means that there are very few positive examples and within them, there is some sort of a structure such that the small group differs significantly from the majority. We may consider such a learning task learning a local pattern.

Depending on the goal of the data analysis, different aspects of time are relevant, e.g., the particular date, the duration of a certain state, or the number of different states. From the given data, we may generate features that allow us to express the aspect of interest. Here, we investigate the aspect of state change and its representation for learning local patterns in time-stamped data. Besides a simple Boolean representation indicating a change, we use frequency features from information retrieval. We transfer Joachim's theory for text classification to our task and investigate its fit to local pattern learning. The approach has been implemented within the MiningMart system and was successfully applied to real-world insurance data.

1 Introduction

When designing a knowledge discovery application, the choice of the representation of examples and hypotheses is the most important issue. Choosing the right representation for hypotheses has been called "model selection". Learnability is a statement about a pair of example and hypotheses representation: we want to represent examples and hypotheses such that concepts of interest can be learned in at least polynomial time. This constrains the search for an appropriate representation on one side. The majority of known solutions to model selection deals with global models. What, if we are looking for local patterns? On the other side, the search is constrained by the given data. Transforming them into an appropriate representation is an effort, which we want to minimize. Moreover, not every example representation which is well suited for a learning algorithm can be constructed from given raw data. Figure 1 shows the chain of processes that

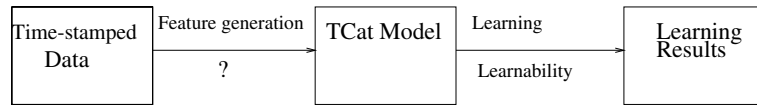


Fig. 1. Representation – from raw data to hypotheses

lead to the learning results. What are the characteristics of raw data that estimate whether we can construct an example representation for which, in turn, we know learnability results for a certain learning algorithm? Concerning the example representation, our standard approach to feature selection requires a subset of given features to separate the data according to the target concepts. What, if the learning task has to cope with an internal structure where attributes occurring in the target concept do occur in the remaining examples as well? It is our goal to develop guidelines for the choice of an appropriate example representation. We want to estimate *before* the transformation, whether it enlarges or shrinks the representation space and whether it favours a certain learning algorithm, or is appropriate for many of them. In terms of Figure 1, we are concerned with the question mark between raw data and the example language.

In this paper, we generalize the applicability of Thorsten Joachim’s TCat model which gives clear learnability bounds for text classification using the support vector machine (SVM) [1]. The TCat model shares characteristics with local patterns (cf. Section 2). It refers to the bag-of-words representation of texts. It is straight forward to construct this example representation from texts. Now, we found that frequency features as used in the bag-of-words representation can effectively be constructed from time-stamped data, too. Moreover, we developed a heuristic that efficiently estimates for raw data, whether the transformation to frequency features will shrink or enlarge the data space, and whether it fits the distribution to which TCat models fit. For learning local patterns from time-stamped data we found that the representation enhanced the learning results of several algorithms, when compared to using the raw data.

The paper is organised as follows. We first state a working hypothesis on the notion of local patterns (Section 2). Then, we clarify the time aspects in time-stamped data (Section 3). For the aspect of state changes and their frequency, we show how to transform a relational database into one with frequency features for state change. We illustrate the procedure by real-world insurance data (Section 4). Then we can make use of the transformed data for learning a local pattern. We conducted several experiments with insurance data (Section 5.1). The good learning results are – for the SVM – explained by Joachim’s theory. We describe the (transformed) insurance data by a TCat model (Section 6). We then focus on the the step from raw data to the TCat model. Section 6.1 characterizes the raw data with respect to the transformation and presents a heuristic which estimates the effect of the transformation. A program calculates a TCat model for given data set. Experiments with artificial data sets investigate the impact of the local patterns characteristics to TCat models and learning results (Section 7). This

completes the chain from the time-stamped raw data to SVM learning of local patterns.

2 Local Patterns

Local patterns are not yet clearly defined. Common to all definitions is that we want to learn about rare events in a large collection. In other words, the distribution is skewed, offering few instances of the local pattern and very many instances of a global model. Local patterns occur with a low frequency (cf. the papers in [2]). Several authors have characterised local patterns as small regions with high deviations from a global model, e.g., [3], [4], [5]. Paul Cohen’s definition of local patterns as “low frequency and low entropy” also relates the local pattern to the regular (global) frequency and entropy values. Moreover, it points at the internal structure of local patterns or their examples. His learning task is to detect boundaries of episodes in a sequence. The comparison of frequency and entropy of one n-gram with those of all other n-grams of the same length delivers the standardized frequency and entropy [6]. Then, a local pattern is detected between two maximally (standardized) frequent n-grams and directly following an n-gram with highest (standardized) entropy. Episodes are complex patterns with an internal structure, which is represented here by the moving n-grams. Multimedia data (e.g., texts), time series, and DNA data are other instances of complex data types. Arno Siebes models the structure within complex data types using wavelets [4]. Transforming data into wavelets allows him to compute the similarity between examples which can then be used for further analysis. The wavelet transformation can be considered feature generation for structured data. We may find an internal structure also within a simple attribute-value representation. Interesting are the characteristics which Thorsten Joachims has found for texts in the bag of words representation [1]: The characteristic that several attribute values indicate the class membership has been called a *high level of redundancy*. The characteristic that instances do not (necessarily) share an attribute value being valid has been called *heterogenous use of terms*. In addition, we find that attribute values occurring frequently in one pattern do so in the remaining other observations, as well. We may call this an *overlap*. Redundancy, heterogeneity, and overlap are characteristics of the internal structure of (text) instances.

As a working hypothesis, we end up with three characteristics of local patterns:

- Local patterns describe rare events. In other words, the distribution is skewed, offering few instances of the local pattern and very many instances of a global model.
- Given a dataset for which a global model can be determined, local patterns deviate significantly from the global model.
- Local patterns describe data with an internal structure. Redundancy, heterogeneity, and overlap are aspects of internal structure.

3 Time-Stamped Data

Time-related data include time series (i.e. equidistant measurements of one process), episodes made of events from one or several processes, and time intervals which are related (e.g., an interval overlaps, precedes, or covers another interval). Time-stamped data refer to a calendar with its regularities. They can easily be transformed into a collection of events, can most often be transformed into time intervals, and sometimes into time series.

Time series are most often analysed with respect to a prediction task, but also trend and cycle recognition belong to the statistical standard (see for an overview [7,8]). Following the interest in very large databases, indexing of time series according to similarity has come into focus [9,10]. Clustering of time series is a related topic (cf. e.g., [11]) as is time series classification (cf. e.g., [12,13]). The abstraction of time series into sequences of events or time intervals approximates the time series piecewise by functions (so do [14,15,16]). Other segmentation methods are presented in [17].

Event sequences are investigated in order to predict events or to determine correlations of events [18,19], [20,21,22]. The approach of Frank Höppner abstracts time series to time intervals and uses the time relations of James Allen in order to learn episodes [23,24]. The underlying algorithm is one of learning frequent sets as is Apriori [25]. The resulting episodes are written as association rules. Other basic algorithms (e.g., regression trees) can be chosen as well [26], delivering logic rules with time annotations. Also inductive logic programming can be applied. Episodes are then written as a chain logic program, which expresses direct precedence by chaining unified variables and other time relations by additional predicates [27,28].

Time-stamped data have been investigated in-depth in [29]. They offer a framework for time granularities and specialised databases for temporal data. However, from a practical point of view, building up a temporal database before analysing the data is too demanding for a knowledge discovery application. Hence, we prefer to transform the given data into an appropriate representation for data analysis. The given data are usually stored in a multirelational database where some attributes offer a time stamp of the same granularity (i.e., minute, hour, day, week, year). If time stamps in the stored data are equidistant, data actually are time series. We exclude this case, here. As was already stated above, time-stamped data can most often be transformed into events and time intervals. It is hard to select the appropriate representation [30]. In general, we have the following options:

Snapshot: We ignore the time information and reduce the data to the most current state. This state can be written as one or several events. It may well happen that such a snapshot already suffices for learning.

Events with time intervals: We aggregate time points to time intervals where attribute values are similar enough (segmentation). For nominal attributes, it is straight forward to construct time intervals from the start and end time of each attribute value. In addition, we might want to represent relations

between the intervals. Learning algorithms which make good use of time information (episode learning) can then be applied.

Feature generation: Time aspects are encoded as regular attributes of the examples such that any learning algorithm can be applied. Simple encodings are seasons simply stated by flags and vectors, where the attributes summarize the history preceding a target event.

In this paper, we investigate feature generation for learning from time-stamped data. For each attribute, we represent whether the attribute changed at all over the recorded time span (Boolean), or how often it has changed (frequency feature). The latter allows us to link our results to the TCat model for text classification [1].

4 Using TF/IDF Features

Time-stamped data often describe the same object (customer, contract, engine...) by several rows in a database table, each for one of the object's states. Figure 2 illustrates this by an excerpt of an insurance contract database table, where each contract (VVID) is described by several attributes. Whenever an attribute's value has changed, a new row is added. The snapshot approach would

	VVID	VVAENDNR	VVWIVON	VVWIBIS	VVAENDDAT	VVAENDART	...
history of a contract	16423	1	1946	1998	1946	1000	
	16423	2	1998	1998	1998	27	
	16423	3	1998	1998	1998	4	
	16423	4	1998	1998	1998	54	
	16423	5	1998	1998	1998	4	
	16423	6	1998	9999	1998	61	
history of another contract	5016	1	1997	1999	1997	33	
	5016	2	1999	2001	1999	33	
	5016	3	2001	2001	2001	33	
	5016	4	2001	2001	2001	33	
	5016	5	2001	2002	2001	81	
	5016	6	2002	9999	2002	94	
...							

Fig. 2. Excerpt from the contract table

just extract the most current row for a contract. The time interval approach would use the “begin” attribute (VVWIVon) and the “end” attribute (VVWIBis) and indicate the other attributes from a row as an event. A Boolean representation would just state whether an attribute had been changed over the lifetime of a contract, or not. This reduces the data space a lot. If we transform the raw

data (about contracts) into a frequency representation, we possibly condense the data space in an appropriate way. We simply order the rows according to the data and count, how often an attribute's value changes, giving us the frequency count of that attribute. Figure 3 illustrates the procedure. However, we must

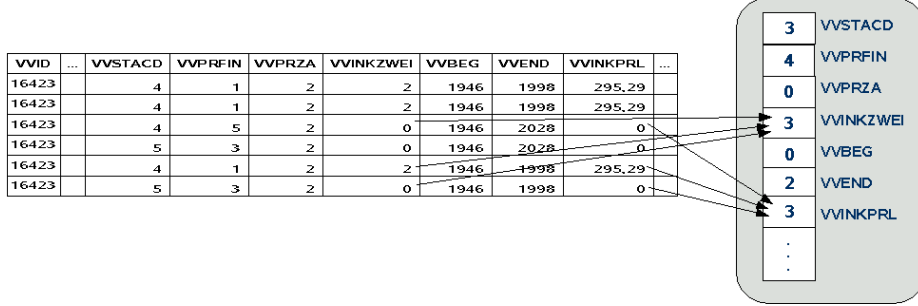


Fig. 3. Calculating the term frequency for the original attributes

exclude the frequencies of those changes that are common to all contracts, e.g. because of a change of law. The feature from statistical text representation formulates exactly this: term frequency and inverse document frequency (TFIDF) [31].

Term frequency here describes how often a particular attribute a_i of c_j , the contract or one of its components, has been changed.

$$tf(a_i, c_j) = \|\{x \in \text{time points} \mid a_i \text{ of } c_j \text{ changed}\}\|$$

The document frequency here corresponds to the number of contracts in which a_i has been changed. The set of all contracts is written C . The document frequency is just the number of contracts with a term frequency greater than 0.

$$df(a_i) = \|\{c_j \in C \mid a_i \text{ of } c_j \text{ changed}\}\|$$

Hence the adaptation of the TF/IDF text feature to contract data becomes for each contract c_j :

$$tfidf(a_i) = tf(a_i, c_j) \log \frac{\|C\|}{df(a_i)}$$

5 Local Pattern Learning

Now that we have introduced frequency features for the aspect of change in time-stamped databases, we can bring together the local pattern learning and the time-stamped data. We first report on local pattern learning from a real-world database (Section 5.1). Then we present experiments with artificial data sets (Section 7).

5.1 Local Patterns in Insurance Data

In the course of enhanced customer relationship management, the Swiss Life insurance company investigated opportunities for direct marketing [32]. A more difficult task was to predict surrender in terms of a customer buying back his life insurance. We worked on knowledge discovery for the classification into early termination or continuation of policies. The task was clearly one of local pattern learning:

- Only 7.7% of the contracts end before their end date. Hence, the event to be predicted is rare.
- Internal studies at the insurance company found that for some attributes the likelihood of surrender differed significantly from the overall likelihood. The TCat model of the data (6) also clearly indicates this.
- Contract data have an internal structure.
 - First studies showed that frequent sets in the group of continued contract were frequent sets in the group of terminated contracts, as well [33] (overlap).
 - In each contract, there are several attributes indicating surrender or continuation (redundancy).
 - We also found that within the group of terminated contracts, there were those which do not share attributes (heterogenous use of terms).

Hence, the internal structure of contracts shares characteristics with text data.

The given anonymous database consists of 12 tables with 15 relations between them. The tables contain information about 217,586 policies and 163,745 customers. If all records referring to the same policy and component (but at a different status at different times) are counted as one, there are 533,175 components described in the database. We selected 14 attributes from the original database. 13 of them were transformed as described above (Section 4). One of them is the reason for a change of a contract. There are 121 different reasons. We transformed these attribute values into binary attributes a . Thus we obtained $13+121=134$ features describing changes of a contract. To calculate the TF/IDF values for these binary features we considered the history of each contract. For the 121 newly created features we counted how often they occurred within the mutations. Figure 4 shows how the calculation was done. We compared the learning results on this generated representation to those on the selected original data for different learning algorithms. We used 10-fold cross validation on a sample of 10,000 examples. In order to balance precision and recall, we used the F -measure:

$$F_{\beta} = \frac{(\beta^2 + 1)Prec(h)Rec(h)}{\beta^2 Prec(h) + Rec(h)} \quad (1)$$

where β indicates the relative weight between precision and recall. We have set $\beta = 1$, weighting precision and recall equally. Table 1 shows the results. For all algorithms, the frequency features are better suited than the original attributes.

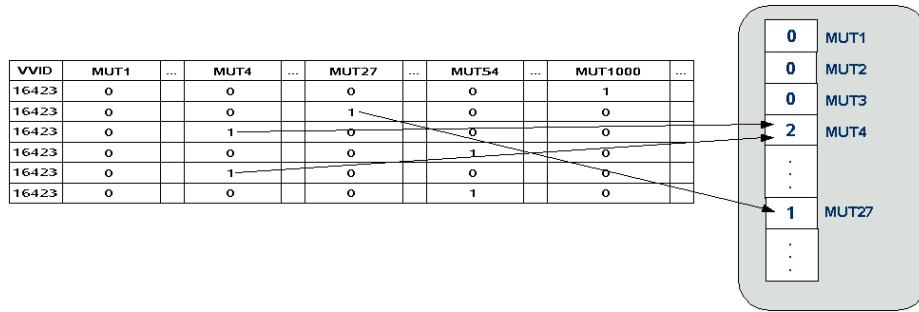


Fig. 4. Calculating the term frequency for the newly created features

6 Characterizing the Data by the TCat Model

The transformation into a frequency representation allows to model the data as TCat-concepts. TCat-concepts model text classification tasks such that their learnability can be proven [1].

Definition of TCat-concepts: "The TCat-concept

$$TCat([p_1 : n_1 : f_1], \dots, [p_s : n_s : f_s])$$

describes a binary classification task with s sets of disjoint features. The i -th set includes f_i features. Each positive example contains p_i occurrences of features from the respective set, and each negative example contains n_i occurrences. The same feature can occur multiple times in one document." [1]

In order to describe the newly constructed data set in terms of TCat-concepts, we need to partition the feature space into disjoint sets of positive indicators, negative indicators and irrelevant features. For the insurance application, we selected features by their odds ratio. There are 2 high-frequency features that indicate positive contracts (surrender) and 3 high-frequency features indicating negative contracts (no surrender). Similarly, there are 3 (4) medium-frequency features that indicate positive (negative) contracts. In the low-frequency spectrum there are 19 positive indicators and 64 negative indicators. All other features are assumed to carry no information. Since the same feature can occur in p_i as well as in n_i , listing the features would not show the difference. The internal structure prohibits this. However, the number of occurrences clearly shows the significant difference between continued and early terminated contracts.

To abstract from the details of particular contracts, it is useful to define what a typical contract for this task looks like. An average contract has 8 features. For positive examples, on average 25% of the 8 features come from the set of the 2 high-frequency positive indicators while none of these features appear in an average negative contract. The relative occurrence frequencies for the other

Apriori		
	TF/IDF attributes	Original attributes
Accuracy	93,48%	94,3%
Precision	56,07%	84,97%
Recall	72,8%	18,39%
F-Measure	63,35%	30,24%
J4.8		
	TF/IDF attributes	Original attributes
Accuracy	99,88%	97,82%
Precision	98,64%	96,53%
Recall	99,8%	70,08%
F-Measure	99,22%	81,21%
mySVM		
	TF/IDF attributes	Original attributes
Accuracy	99,71%	26,65%
Precision	97,06%	8,73%
Recall	98,86%	100%
F-Measure	97,95%	16,06%
Naive Bayes		
	TF/IDF attributes	Original attributes
Accuracy	88,62%	87,44%
Precision	38,55%	32,08%
Recall	78,92%	77,72%
F-Measure	51,8%	45,41%

Table 1. Results comparing different learning algorithms and feature spaces

features are given in Table 2. Applying these percentages to the average number of features, this table can be directly translated into the following TCat-concept. Note, that p_i and n_i indicate frequencies. Hence, the second high-frequency set of features consists of three attributes, which occur one time in positive and three times in negative examples.

TCat ([2 : 0 : 2], [1 : 4 : 3], # high frequency
[3 : 1 : 3], [0 : 1 : 4], # medium frequency
[1 : 0 : 19], [0 : 1 : 64], # low frequency
[1 : 1 : 39] # rest
)

The learnability theorem of TCat-concepts [1] bounds the expected generalization error of an unbiased support vector machine after training on n examples by

$$\frac{R^2}{n+1} \frac{a+2b+c}{ac-b^2} \quad (2)$$

	high frequency		medium frequency		low frequency		
	2 pos.	3 neg.	3 pos.	4 neg.	19 pos.	64 neg.	39 rest
pos. contract	25%	12.5%	37.5%	0%	12.5%	0%	12.5%
neg. contract	0%	50%	12.5%	12.5%	0%	12.5%	12.5%

Table 2. Composition of an average positive and an average negative contract

where R^2 is the maximum Euclidian length of any feature vector in the training data, and a, b, c are calculated from the TCat-concept description as follows:

$$a = \sum_{i=1}^s \frac{p_i^2}{f_i} \quad b = \sum_{i=1}^s \frac{p_i n_i}{f_i} \quad c = \sum_{i=1}^s \frac{n_i^2}{f_i}$$

$a = 5.41, b = 2.326, c = 5.952$ can be calculated directly from the data. The Euclidian length of the vectors remains to be determined. We want to see whether the data transformation condenses the data properly. The data space with the original 15 attributes could be such that each attribute is changed m times giving us $m\sqrt{15}$ – the largest case. The smallest case is that only one attribute is changed m times giving us the small data size of m . For texts, Zipf’s law gives the approximation [34]: if one ranks the words by their frequency, the r -th most frequent words occur $\frac{1}{r}$ times the frequency of the most frequent words. We can apply this law for natural language to collections c of natural language texts. Experimental data suggests that Mandelbrot distributions [35]

$$TF_i = \frac{c}{(k+r)^\phi}$$

with parameters c, k and ϕ provide a better fit. For the contract data Figure 5 plots term frequency versus frequency rank. The line is an approximation with $k = -0.6687435$ and $\phi = 1.8$. We see that (as is true for text data) also the contract data can be shrunk by the frequency transformation.

$$R^2 = \sum_{r=1}^d \left(\frac{c}{(r+k)^\phi} \right)^2 \quad (3)$$

We bound $R^2 \leq 37$ according to the Mandelbrot distribution and come up with the bound of the expected error according to equation 3 of $\frac{37 \cdot 0.5978}{n+1}$, consequently after training on 1000 examples the model predicts an expected generalization error of less than 2.2%. It turns out that the transformed data sets can easily be separated by a support vector machine. Hence, the good learning results (0.6% error) are explained.

6.1 Characterizing the Raw Data

In order to ease the design process of knowledge discovery applications we should know before the transformation whether the data space will be condensed, or

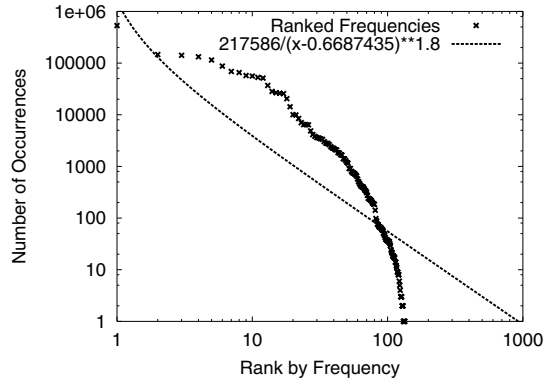


Fig. 5. Distribution of term frequencies in the contract data on a log log scale. The line is an approximation of the observed curve using a Mandelbrot distribution.

not. In other words, we want to measure the sparseness of data which can be estimated by the maximum Euclidian length of a vector. We order the original table with time stamps such that the states of the same individual (e.g., contract) are in succeeding rows. We consider each individual c a vector and calculate the frequency of changes for each of its n attributes $a_1 \dots a_n$ in parallel “on the fly” We can then determine in one database scan the maximum value of the Euclidian length of a vector:

$$\hat{R} = \max \left(\sqrt{\sum_{i=1}^n tf(a_i, c_j)^2} \right) \quad (4)$$

If $\hat{R} \leq \frac{m\sqrt{n}}{2}$ where m is the maximum frequency, the data will be condensed and learning will be fast. In the insurance case $n = 14$ and $m = 15$ so that $\hat{R} = 22,913$ which is in fact less than $\frac{15\sqrt{14}}{2} = 28.6$.

Of course, the heuristic does not tell anything about the learnability within this representation. What we control with this heuristic is the transformation into the example representation.

7 Experiments with Artificial Data

In order to abstract away from the real-world application, we conducted experiments with artificial datasets. We created 10,000 examples each with 100 attributes in both, binary and TF/IDF representation. The MYSVM was run with a 10-fold cross validation.

7.1 Frequency of Changes Vs. Particular Changes

Datasets were generated according to two target concepts:

1. The change of particular attributes determined the classification.
2. The number of changes determined the target concept.

Of course, what should happen, is that the TF/IDF representation is best suited for the second target concept.

We wanted to check, whether the TF/IDF representation makes the dataset robust with respect to skewedness and local structure. In other words, we tested the characteristic of local patterns.

We systematically varied the skewedness of the data, positive examples being 50% (not skewed), 25%, 12.5%, or 6.25% (skewed) of the data.

The first concept could perfectly be learned from the binary representation, being robust with respect to skewedness (100% F-measure). Clearly, no internal structure is preventing here the selection of a feature. The TF/IDF representation is little less perfect (95.03% F-measure for 6.25% positive examples). See Figure 6 for details. The heuristic states that the data size becomes little. As is shown in Figure 6, the second concept could not be learned from the binary representation, when the distribution becomes skewed. The default is too dominant, hence the recall comes to zero. For the TF/IDF representation, the learning results degraded gracefully when the distribution became more skewed from 93.17% to 88.98% F-measure. The heuristic dissuades from the transformation in all cases, although the binary representation is only superior learning the first concept. Indeed, as already stated, the larger data space is necessary for the skewed second concept. The heuristic does not inform us about learnability. Hence, when learning the second concept from skewed data fails using the binary representation, the transformation is tried, anyway.

Varying the sparseness of the data from 50 attributes being changed over 25 to 5 attributes being changed, we found again that using the binary representation the first concept could be learned perfectly. In contrast, the second concept could only be learned successfully using the TF/IDF representation. See Figure 6 for details.

7.2 Internal Structures

We also varied the local structure within the artificial data. We used the TCat model to generate datasets, varying p_i and n_i within f_i . Note, that for a group of f_i (high frequency, medium frequency, or low frequency) p_i of them can be arbitrarily chosen. Hence, it could happen that $\frac{f_i}{p_i}$ individuals do not share any attribute. This fraction indicates how heterogenous the use of terms is. We varied the heterogenous use of terms from 4 individuals which could be completely different but be in the same class (little heterogeneity) to 20 individuals being disjoint but in the same class. If we keep the sparseness throughout all experiments being 20 from 100 attributes given, we automatically vary the redundancy from 0.5 in the little heterogenous case to 0.1 in the extremely heterogenous case. The

redundancy can be expressed by $\frac{p_i}{f_i}$ or $\frac{n_i}{f_i}$ for indicative attributes (here: medium and low frequency attributes). Tabelle 3 gives an overview of the TCat models used for generating datasets. Figure 6 shows the achieved learning results.

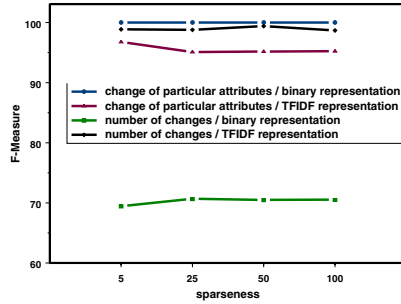
	Not heterogenous, redundant, no overlap	Little heterogeneous, redundant, little overlap	Medium heterogeneous, little redundant, no overlap
heterogeneity	$\frac{20}{5} = 4$	$\frac{20}{4} = 5$	$\frac{20}{2} = 10$
redundancy	$\frac{10}{20} = 0.5$	$\frac{10}{20} = 0.5$	$\frac{2}{20} = 0.2$
TCat model	[10 : 10 : 20], [5 : 0 : 20], [0 : 5 : 20], [5 : 0 : 20], [0 : 5 : 20],	[10 : 10 : 20], [4 : 1 : 20], [1 : 4 : 20], [4 : 1 : 20], [1 : 4 : 20],	[16 : 16 : 20] [2 : 0 : 20], [0 : 2 : 20], [2 : 0 : 20], [0 : 2 : 20],
error bound	1.33%	3.3%	3%
	High heterogenous, little redundant, no overlap	Medium heterogeneous, redundant, high overlap	
heterogeneity	$\frac{20}{1} = 20$	$\frac{20}{3} = 6.6$	
redundancy	$\frac{2}{20} = 0.1$	$\frac{10}{20} = 0.5$	
TCat model	[18 : 18 : 20] [1 : 0 : 20], [0 : 1 : 20], [1 : 0 : 20], [0 : 1 : 20],	[10 : 10 : 20] [3 : 2 : 20], [2 : 3 : 20], [3 : 2 : 20], [2 : 3 : 20],	
error bound	72.4%	28%	

Table 3. TCat models used for generating datasets

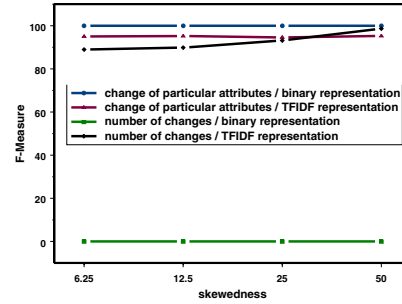
The TCat model tells that the learning results should decrease gracefully when increasing heterogenous use of terms or overlap. The actual learning results show that only the overlap decreases the learning result and only for the binary representation. The TF/IDF representation is robust to the variation of both, heterogenous use of terms and overlap. This means that the TF/IDF representation is particularly appropriate for the internal structure within local patterns.

8 Conclusion

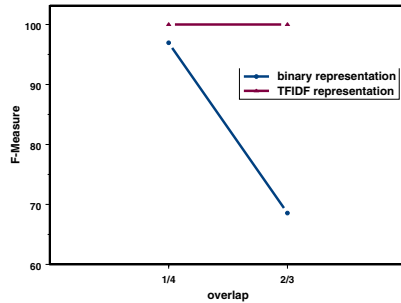
The design of a knowledge discovery application is supported by learnability results as soon as the appropriate example representation has been found. The design support missing was a principled approach to when to generate which features for a given dataset. The transformation of given (raw) data to the example representation is a matter of feature generation and selection [36,37,38]. The automatic selection of features becomes difficult whenever no proper subset of features distinguishes positive from negative examples. This is particularly the case, if the target concept has an internal structure in terms of the heterogenous use, redundancy, and overlap of attributes. If, in addition, the distribution offers a majority of negative and very few positive examples, we are confronted



(a) varying sparseness



(b) varying skewedness



(c) varying overlap

Fig. 6. Learning results

with local pattern learning. The TCat model covers such an internal structure, is robust with respect to skewed distributions, and offers learnability results for the SVM. Hence, if we form a TCat model for our data, we can estimate how well learning will succeed. However, the TCat model is based on frequency features. Calculating frequency features for texts is a standard approach. Now, we have shown how time-stamped data can be transformed into frequency features. Moreover, experiments with artificial datasets indicate that

- the TF/IDF representation is superior to the binary one if attributes indicative within the positive and negative example sets overlap.
- The TF/IDF representation outperforms the binary one in case of sparse data.
- the TF/IDF representation is robust with respect to skewed data
- the TF/IDF representation is robust with respect to redundancy or heterogeneous use of terms.

The true difference between the binary and the frequency representation we found in the experiments is given by the skewedness and the sparseness of the data. TF/IDF outperforms the binary representation clearly, if the data are sparse or the distribution is skewed and the concept to be learned is about the number of changes.

The heuristic and the generation of frequency features for time-stamped data with respect to the aspect of state change has been implemented within the MiningMart system [39]. The calculation of a TCat model for given data has been implemented as a JAVA program to be integrated into the YALE system [38], which has run the cross validation and MY SVM runs. This supports the design of knowledge discovery applications. The method can easily be applied to other time-stamped datasets. If a general model can be found, the binary representation will be appropriate. For a local pattern, the TF/IDF representation is better suited. Given a time-stamped database and the task of classification (not that of episode learning), one can now proceed systematically from the smallest to successively larger data spaces. First, try learning with the snapshot representation, then try the binary representation, and finally generate the frequency features and calculate the error bound using the TCat model. Our approach only covers a certain type of raw data, namely those describing non-equidistant state changes. It focuses on finding local patterns, characterised by a skewed distribution, separable patterns, and an internal structure which does not allow to select a proposer subset of features. More research on the transformation from raw data to example representations is needed.

Acknowledgment We thank Jörg-Uwe Kietz and Regina Zücker for their information about the insurance practice and the anonymised database. For stimulating discussions on the support vector machine, mathematics, and representation languages we thank Stefan Rüping wholeheartedly.

References

1. Joachims, T.: Learning to Classify Text using Support Vector Machines. Volume 668 of Kluwer International Series in Engineering and Computer Science. Kluwer (2002)
2. Hand, D., Bolton, R., Adams, N.: Determining hit rate in pattern search. In Hand, D., Adams, N., Bolton, R., eds.: Pattern Detection and Discovery. Springer (2002)
3. Hand, D.: Pattern detection and discovery. In Hand, D., Adams, N., Bolton, R., eds.: Pattern Detection and Discovery. Springer (2002)
4. Siebes, A., Struzik, Z.: Complex data mining using patterns. In Hand, D., Adams, N., Bolton, R., eds.: Pattern Detection and Discovery. Springer (2002)
5. Morik, K.: Detecting interesting instances. In Hand, D.J., Adams, N.M., Bolton, R.J., eds.: Proceedings of the ESF Exploratory Workshop on Pattern Detection and Discovery. Volume 2447 of LNAI., Berlin, Springer Verlag (2002) 13–23
6. Paul Cohen, Brent Heeringa, Niall M. Adams: An unsupervised algorithm for segmenting categorical timeseries into episodes. In Hand, D.J., Adams, N.M., Bolton, R.J., eds.: Pattern Detection and Discovery. Volume 2447 of Lecture notes in computer science., London, UK, ESF Exploratory Workshop, Springer (2002) 1–12

7. Box, G.E.P., Jenkins, G.M., Reinsel, G.C.: Time Series Analysis. Forecasting and Control. Third edn. Prentice Hall, Englewood Cliffs (1994)
8. Schlittgen, R., Streitberg, B.H.J.: Zeitreihenanalyse. 9. edn. Oldenburg (2001)
9. Keogh, E., Pazzani, M.: Scaling up dynamic time warping for datamining applications. In: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press (2000) 285–289
10. Agrawal, R., Faloutsos, C., Swami, A.: Efficient similarity search in sequence databases. In: Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms. Volume 730., Springer (1993) 69–84
11. Oates, T., Firoiu, L., Cohen, P.R.: Using dynamic time warping to bootstrap hmm-based clustering of time series. In: Sequence Learning ? Paradigms, Algorithms, and Applications. Volume 1828 of Lecture Notes in Computer Science. Springer Verlag (2001) 35?–52
12. Geurts, P.: Pattern extraction for time series classification. In: Proceedings of the 5th European Conference on the Principles of Data Mining and Knowledge Discovery. Volume 2168 of Lecture Notes in Computer Science., Springer (2001) 115–127
13. Lausen, G., Savnik, I., Dougarjapov, A.: Msts: A system for mining sets of time series. In: Proceedings of the 4th European Conference on the Principles of Data Mining and Knowledge Discovery. Volume 1910 of Lecture Notes in Computer Science., Springer Verlag (2000) 289–298
14. Das, G., Lin, K.I., Mannila, H., Renganathan, G., Smyth, P.: Rule Discovery from Time Series. In Agrawal, R., Stolorz, P.E., Piatetsky-Shapiro, G., eds.: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98), New York City, AAAI Press (1998) 16 – 22
15. Guralnik, V., Srivastava, J.: Event detection from time series data. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, San Diego, USA (1999) 33 – 42
16. Morik, K., Wessel, S.: Incremental signal to symbol processing. In Morik, K., Kaiser, M., Klingspor, V., eds.: Making Robots Smarter – Combining Sensing and Action through Robot Learning. Kluwer Academic Publ. (1999) 185 –198
17. Salatian, A., Hunter, J.: Deriving trends in historical and real-time continuously sampled medical data. *Journal of Intelligent Information Systems* **13** (1999) 47–71
18. Agrawal, R., Psaila, G., Wimmers, E.L., Zaït, M.: Querying shapes of histories. In: Proceedings of 21st International Conference on Very Large Data Bases, Morgan Kaufmann (1995) 502–514
19. Domeniconi, C., shing Perng, C., Vilalta, R., Ma, S.: A classification approach for prediction of target events in temporal sequences. In Elomaa, T., Mannola, H., Toivonen, H., eds.: Principles of Data Mining and Knowledge Discovery. Lecture Notes in Artificial Intelligence, Springer (2002)
20. Blockeel, H., Fürnkranz, J., Prskawetz, A., Billari, F.: Detecting temporal change in event sequences: An application to demographic data. In De Raedt, L., Siebes, A., eds.: Proceedings of the 5th European Conference on the Principles of Data Mining and Knowledge Discovery. Volume 2168 of Lecture Notes in Computer Science., Springer (2001) 29–41
21. Mannila, H., Toivonen, H., Verkamo, A.: Discovering frequent episode in sequences. In: Procs. of the 1st Int. Conf. on Knowledge Discovery in Databases and Data Mining, AAAI Press (1995)
22. Mannila, H., Toivonen, H., Verkamo, A.: Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery* **1** (1997) 259–290

23. Höppner, F.: Discovery of Core Episodes from Sequences. In Hand, D.J., Adams, N.M., Bolton, R.J., eds.: Pattern Detection and Discovery. Volume 2447 of Lecture notes in computer science., London, UK, ESF Exploratory Workshop, Springer (2002) 1–12
24. Allen, J.F.: Towards a general theory of action and time. *Artificial Intelligence* **23** (1984) 123–154
25. Agrawal, R., Imielinski, T., Swami, A.: Database mining: A performance perspective. *IEEE Transactions on Knowledge and Data Engineering* **5** (1993) 914–925
26. Nunez, M.: Learning patterns of behavior by observing system events. In: Proc. of the European Conference on Machine Learning. Lecture notes in Artificial Intelligence, Springer (2000)
27. Klingspor, V., Morik, K.: Learning understandable concepts for robot navigation. In Morik, K., Klingspor, V., Kaiser, M., eds.: Making Robots Smarter – Combining Sensing and Action through Robot Learning. Kluwer (1999)
28. Rieger, A.D.: Program Optimization for Temporal Reasoning within a Logic Programming Framework. PhD thesis, Universität Dortmund, Dortmund, Germany (1998)
29. Bettini, C., Jajodia, S., Wang, S.: Time Granularities in Databases, Data Mining, and Temporal Reasoning. Springer (2000)
30. Morik, K.: The representation race - preprocessing for handling time phenomena. In de Mántaras, R.L., Plaza, E., eds.: Proceedings of the European Conference on Machine Learning 2000 (ECML 2000). Volume 1810 of Lecture Notes in Artificial Intelligence., Berlin, Heidelberg, New York, Springer Verlag Berlin (2000)
31. Salton, G., Buckley, C.: Term weighting approaches in automatic text retrieval. *Information Processing and Management* **24** (1988) 513–523
32. Kietz, J.U., Vaduva, A., Zücker, R.: Mining Mart: Combining Case-Based-Reasoning and Multi-Strategy Learning into a Framework to reuse KDD-Application. In Michalki, R., Brazdil, P., eds.: Proceedings of the fifth International Workshop on Multistrategy Learning (MSL2000), Guimares, Portugal (2000)
33. Fisseler, J.: Anwendung eines Data Mining-Verfahrens auf Versicherungsdaten. Master's thesis, Fachbereich Informatik, Universität Dortmund (2003)
34. G.K.Zipf: Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology. Addison-Wesley (1949)
35. Mandelbrot, B.: A note on a class of skew distribution functions: Analysis and critique of a paper by H.A.Simon. *Information and Control* **2** (1959) 90 – 99
36. Kohavi, R., John, G.H.: Wrappers for feature subset selection. *Artificial Intelligence* **97** (1997) 273–324
37. Liu, H., Motoda, H.: Feature Extraction, Construction, and Selection: A Data Mining Perspective. Kluwer (1998)
38. Ritthoff, O., Klinkenberg, R., Fischer, S., Mierswa, I.: A hybrid approach to feature selection and generation using an evolutionary algorithm. Technical Report CI-127/02, Collaborative Research Center 531, University of Dortmund, Dortmund, Germany (2002) ISSN 1433-3325.
39. Morik, K., Scholz, M.: The MiningMart Approach to Knowledge Discovery in Databases. In Zhong, N., Liu, J., eds.: Intelligent Technologies for Information Analysis. Springer (2004)