Searching for Temporal Patterns in AmI Sensor Data

Romain Tavenard[1], Albert A. Salah[2], and Eric J. Pauwels[2]

[1] IRISA/ENS de Cachan, Av. du Gén. Leclerc, 35042 Rennes Cedex
[2] CWI, Kruislaan 413, 1098 SJ Amsterdam, NL

Abstract. Data streams emanating from sensors in AmI scenarios often exhibit interesting temporal patterns that point to semantically meaningful events. In this paper we explore two methodologies to uncover this hidden temporal structure: compression-based pattern extractors utilize temporal ordering whereas T-patterns attempt to include estimates for the inter-event times. We suggest improvements to simplify the statistical underpinnings of T-patterns and discuss the relative merits of both methodologies in the light of a number of simulation experiments.

1 Introduction

The success of Ambient Intelligence (AmI) depends on observing the activities of humans and responding to their behaviour patterns intelligently. In ubiquitous environments, where a wealth of sensory data is produced, mining the data for temporal patterns serves this need by discovering associations and structure, either in an offline manner to pave the way for new designs and applications, or in an online manner to ensure adaptation to the user of the AmI environment.

In Section 2, a brief survey of the relevant literature is presented. Two prominent algorithms that have been put forward recently (compression-based methods and T-patterns) are inspected in more detail in Section 3 and Section 4, respectively. In the latter section we also present a modification to the T-pattern approach that increases its accuracy significantly.

We test these methods in a scenario where movement patterns of persons in a building are inferred from activity detected by interruption sensors. These sensors are very cheap, so it is possible to install a very dense networks of sensors at minimal cost. In addition, they are seen to be far less intrusive and privacy-critical than high-resolution ones (such as surveillance cameras). Finally, for simple applications (e.g. monitoring movements of people inside a building), low-resolution sensors achieve results comparable to the ones obtained from high-resolution ones [8].

2 Related Work

The most straightforward way to detect temporal events is by representing them spatially, where portions of the input feature are associated with increasing time

indices, and spatial pattern recognition techniques are applied. This approach has several problems. One obvious problem is that the size of the feature vector scales linearly with the represented discretized time slots. This usually also means significant redundancy in the input vector. The second problem is that of the resolution; it is very difficult to learn events of different temporal scales with a single model with this sort of representation. A third problem is that the absolute position in the feature vector is not relevant, but the relative position is. Dealing with this issue may mean exhaustive computations for all possible relative positions.

A more appropriate way of representing time is to make it a part of the model. In recurrent neural networks, a group of *context units* is added to the input that receive their input from the hidden layer of a multi-layer perceptron, and feed back into the hidden layer [4]. At each iteration, the context units receive input from the hidden units of the previous iteration. This structure allows the network to develop an internal representation of the temporal nature of the data, and can be used for prediction. However, the learning assumes that there is structure in all data, and the network tries to make sense of all the input patterns. Another problem is that of scales; the recurrent networks are used to predict the next item in a sequence, and cope poorly with various temporal scales. Other neural network approaches to temporal pattern recognition face similar problems.

A large body of work in multimodal signal processing deals with applications of hidden Markov models (HMM). Building on their success in modelling the temporal patterns encountered in speech recognition [6], HMMs have become the mainstay of spatio-temporal segmentation [1]. However, the classical HMM approach has two main disadvantages for clustering of time series emanating from sensors. First of all, the estimation algorithms assume that the topology of the HMM-structure (in terms of states and transitions) is known, which is not the case. Secondly, Markov models have difficulty incorporating temporal patterns across different timescales. Rao and Cook [7] tackle the first issue by defining high-level inhabitant activity (states of the model) as clusters of elementary actions. Other spatio-temporal learning paradigms that rely on state transitions include dynamic time warping and finite state machines.

A recent approach involves PCA-based methods to uncover daily human behaviour routines [3]. The data for each subject are stored in an activity matrix, whose most prominent eigenvectors (dubbed *eigenbehaviors*) are then interpreted. One obvious drawback with this method is that it requires a fixed sized activity vector. Additionally, there is no hierarchical decomposition of activities.

Two additional methodologies that overcome some of the mentioned problems, viz. *compression-based modelling* and *T-patterns*, which will be discussed in greater detail in the following sections.

3 Compression-based Pattern Extraction

3.1 Overview Compression Algorithms

The underlying idea is to use the Lempel-Ziv compression algorithm (which is known to achieve Markov entropic compression) as pattern extractors. Recall that the basic Lempel-Ziv algorithm (**LZ78**) uses an automatically updated dictionary to extract recurring "words" (patterns) in a string. This dictionary lists the patterns occurring in the stream. The Lempel-Ziv-Welch (**LZW**) variant starts off with a pre-defined basic dictionary (which is available in the case of sensor networks: each symbol corresponds to a single sensor being triggered) to face the issue of ill-detected patterns at the beginning of the stream and also introduces some continuity by not completely erasing w whenever a new pattern is found, but keeping its last symbol. **Active LeZi** [2] builds on the fact that Lempel-Ziv-like algorithms for prediction are equivalent to finding the order-k Markov model that fits the training sequence best. For that reason, the authors adapt LZ78 algorithm in order to avoid its "boundary effect": they use a sliding window of length l (length of the longest phrase in LZ table) on the stream to extract all possible sequences of size l.

LZW and Active LeZi both aim at adding continuity to LZ pattern extraction, yet they still have linear complexity, which is a beneficial feature for a real-time event detection system. On the other hand, none of the compression based methods take into account the temporal structure of the patterns, as the time delays are not modeled, and subsequently overlapping events may escape detection. For a dense, low-cost sensor network without the identification of event source, this is a major drawback as is clearly borne out by the experimental results reported below. This is the main reason why we turn our attention to T-patterns as discussed in the next section.

3.2 Experimental setup

To test these algorithms, we coded a MATLAB package[3] that simulates outputs of low resolution sensors positioned in a building, in accordance with the ground plan of the building and the location of the sensors. We tested two simple layouts: in the first (see Fig. 3.2, left) four doors open into a circular corridor. People enter through one door and exit through another, both doors are chosen at random. Six simple infrared interruption sensors are positioned along the length of the corridor. People using this corridor as a transit zone between two doors will generate various interruption patterns. In the second layout (Fig. 3.2, right) an entrance door (on the left) opens up into a hallway where an RFID reader is located. Persons entering through this door are supposed to proceed to their offices located at the end of the three corridors which again are lined with simple interruption sensors. In this case the system has to learn to associate individual

[3] This package, as all our MATLAB code used for these tests, can be downloaded at `http://homepages.cwi.nl/tavenard/LeSeNe/`.

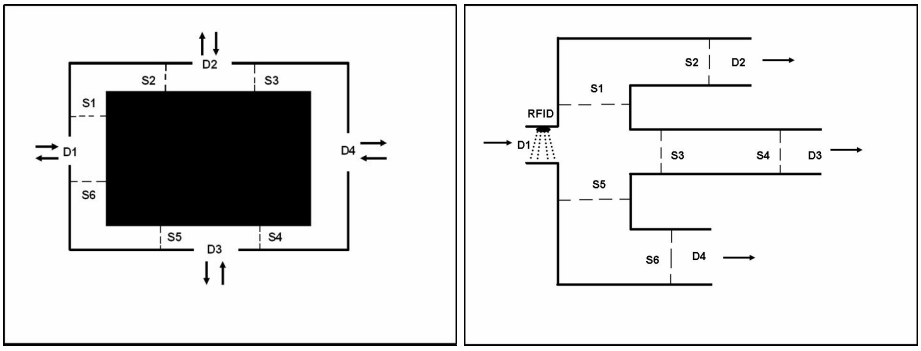

Fig. 1. Ground plan for two corridor layouts used in experiments. Left: Layout 1 shows 4 doors and 6 interruption sensors. Right: Layout 2 shows an entrance door and 3 exit doors, as well as an RFID reader and 6 binary interruption sensors.

RFID tags with movement patterns as signalled by the interruption sensors. In each of the above layouts we have simulated two cases: in the first we assume that at any given time at most one person is present. The second scenario relaxes this and allows two persons to be present simultaneously. The latter complicates the analysis considerably as the interruption sequences are now intermingled.

For each configuration, we generated two sequences of $1,000$ symbols each. These sequences represent a simulation of persons walking in the corridors, each sensor triggering leading to the emittance of a symbol identifying the triggered sensor. In a training phase, the first sequence was fed into the above-described compression-based pattern extractors in order to detect consistently recurring patterns. The second sequence was then used for testing purposes. More precisely, we investigated to what degree we could use the patterns discovered in the training phase as predictors for the next symbol in the stream.

Clearly, the two spatial layouts detailed above are quite simple. Given that temporal patterns will be relatively short (a person entering a corridor through one door and exiting through another one will trigger 2 to 4 sensors) it follows that the prediction error rate for our algorithms is relatively hight. Indeed, the first symbol emitted by each new pattern is random and therefore completely unpredictable, and as individual patterns are short, this phenomenon will affect the prediction 25-50% of all symbols in the streams. Nevertheless, as we aim at benchmarking algorithms and as all algorithms will suffer from the same issue, this will not impair our conclusions.

3.3 Experimental Results

Figs. 2 and 3 summarize the experimental results for the different compression algorithms. In each case the x-axis represents the minimal confidence in the prediction. Confidence is high (in fact 100%) whenever the current pattern unambiguously predicts a unique symbol. If there more potential outcomes, con-

fidence drops accordingly. The dotted line indicates the percentage of cases for which prediction is possible with the confidence specified on the x-axis. The left column show results for scenarios in which only one person is present, the right column shows results for the case when two person intermingle. The two horizontal lines indicate the upper bound for the achievable accuracy (recall that the first symbol in each pattern is unpredictable) and the accuracy of a random prediction.

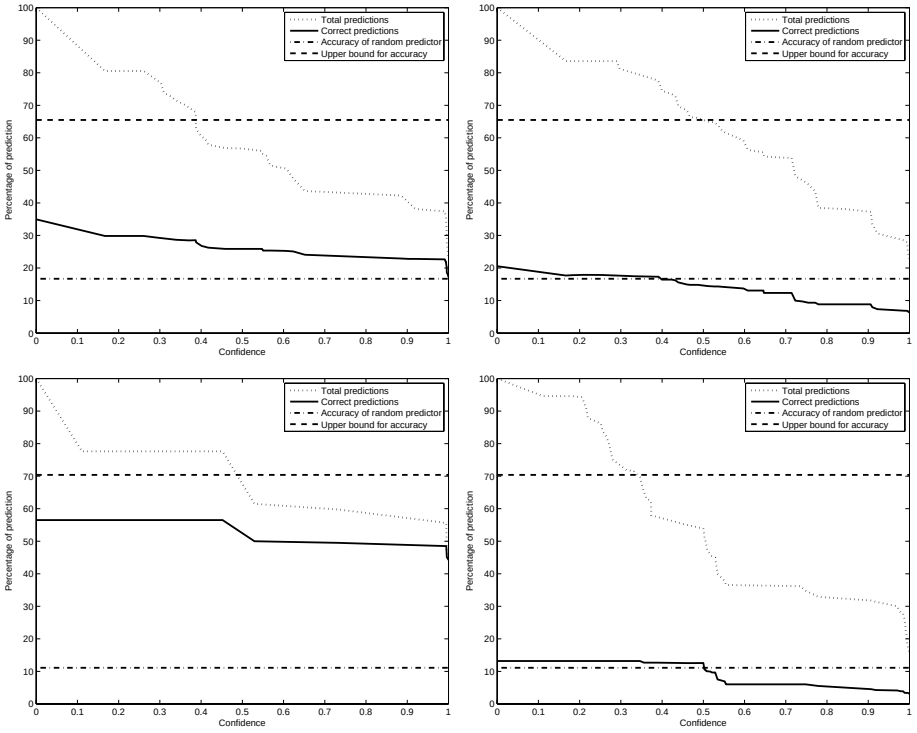

Fig. 2. Predition results for the LZ compression algorithm for layout 1 (top row) and 2 (bottom row). The left column show results for scenarios in which only one person is present, the right column shows results for the case when two person intermingle. See main text for a more detailed discussion.

4 T-patterns

4.1 Introduction and Definition

In the preceding sections we intentionally ignored part of the available data by discarding the time information: data streams were collapsed into symbol

strings thereby erasing the information about the inter-event time-intervals. In this section we undo this simplification and attempt to tackle the problem in its full complexity.

Our methodology of choice are *T-patterns* as introduced and explored by Magnusson [5]. He investigated symbolic time series where each symbol represent the onset of particular event or activity. His principal goal was to elucidate possible relationships between pairs of symbols and then build trees of such temporal dependencies in a hierarchical fashion: as soon as a pair of strongly correlated events is found, they are labelled as a single new event, and the search is resumed.

To recast this problem in a mathematical framework, we first introduce some notation. We denote by $A = (A_1, A_2, \ldots, A_n, \ldots)$ the (ordered) sequence of times at which an A-event occurs. For the inter-event time-intervals we use the notation T_A; more precisely: $T_A(n) = A_n - A_{n-1}$. Similar notation is used for B-events. Since we need to find out whether A-events tend to induce B-events, we refer to the combination of an A-event and the *first* subsequent B-event as an AB-event. The time-interval separating these two events is denoted by T_{AB}. More formally:

$$T_{AB}(k) = B_{k^*} - A_k \quad \text{where} \quad k^* = \arg\min\{j \mid B_j > A_k\}$$

Finally, we will denote by $\tilde{T}_B$ the time-interval between two successive B-events between which at least one A-event occurred. This definition means that set of $\tilde{T}_B$'s constitute a subset of the T_B-times, with a bias towards longer T_B-values as short B-intervals are less likely to contain an A-event.

Magnusson introduced the notion of a *critical interval* (CI): $[d_1, d_2]$ is considered to be a CI for the pair of symbols (events) (A, B) if the occurrence of A at time t entails that B is more likely to occur in the time interval $[t + d_1, t + d_2]$ than in a random interval of the same size. He then suggests to use the standard p-value to gauge how exceptional the observed frequency of the combination under scrutiny is. More precisely, suppose the total data stream has length T with N_A and N_B occurrences of A and B, respectively. If we assume (following Magnusson [5]) as null-hypothesis that A and B are independent and uniformly distributed in this interval, this makes A and B Poisson processes with intensity (i.e. the average number of events per unit time interval) $\lambda_B = N_B/T$, and $\lambda_A = N_A/T$, respectively. Now, assume in addition that there are N_{AB} occurrences of B in a predefined CI (of length d) trailing each A-event. Notice that under the null-hypothesis the expected number of B-events in a time interval of length d equals $\mu_B = \lambda_B d$. In particular, the probability of not observing a B-event in this CI is therefore equal to $\pi_0 = e^{-\mu_B} = e^{-\lambda_B d}$. The above-mentioned p-value is then computed as the probability of observing at least N_{AB} B-events in the CI, if we assume that A and B are independent. Hence,

$$\begin{aligned} p &= P(N_{AB} \text{ B-events or more} \mid \text{ A, B are independent}) \\ &= 1 - P(\text{strictly less than } N_{AB} \text{ B-events} \mid \text{ A, B are independent}) \\ &= 1 - \sum_{k=0}^{N_{AB}-1} P(\text{exactly } k \text{ B-events} \mid \text{ A, B are independent}) \end{aligned}$$

$$= 1 - \sum_{k=0}^{N_{AB}-1} \binom{N_A}{k} (1-\pi_0)^k \pi_0^{(N_A-k)}.$$

Magnusson suggests, as a T-pattern detection scheme, to test, for every possible pair of symbols of the form (A, B), every possible CI, from the largest to the smallest one, until the p-value is sufficiently small indicating significance (.05 is a typical upper bound). Note that p will be high for high values of d, which means that short intervals will be favored. In the remainder of this section we point out two problems with this approach and suggest ways to address them.

4.2 Testing independence between two temporal point processes

It should be pointed out that the repeated significance testing expounded in the preceding paragraph substantially increases the risk of false positives (suggesting spurious dependencies). Applying a Bonferroni correction would be one way to mitigate this adverse effect. In this paper, however, we put forward a more efficient way of testing this independence between A and B which is based on the following proposition.

Proposition 1 *If A and B are independent temporal point process, then*

$$T_{AB} \sim U(0, \tilde{T}_B).$$

In plain language this proposition asserts that if the A and B processes are independent, then whenever an A-event occurs between two successive B-events, it will be uniformly distributed in that interval. Due to lack of space we will not attempt to give a rigorous proof, but it is intuitively clear that non-uniformity of A within the B-interval, would allow a keen observer to improve his or her prediction of the next B-event, thus contradicting independence (for a graphical illustration of this proposition we refer to Fig.4. This therefore allows us to formulate a statistical procedure to test whether A and B are dependent: using the notation established above we compare for each event A_k the time till the next B-event to the current B-interval length:

$$U(k) = \frac{T_{AB}(k)}{\tilde{T}_B(k)} = \frac{B_{k^*} - A_k}{B_{k^*} - B_{k^*-1}}$$

which, under the assumption of independence, should be uniformly distributed between 0 and 1: $U \sim U(0, 1)$. This can be easily checked by any number of standard statistical test (e.g. Kolmogorov-Smirnov). If the null hypothesis (independence) is rejected then it makes sense to start looking for inter-event time intervals (i.e. CI's). This is taken up in the next section.

4.3 Modelling T_{AB} times

The CI detection scheme as proposed in [5] has the drawback that only the first occurrence of B following A is considered. However, if the average occurrence rate of A is relatively high, or if the inter-event time for B is long, this

could lead to fallacious associations. For this reason, we propose to proceed differently. If the above-discussed uniformity test has rejected independence, then we look for the characteristic period by modelling the conditional probability $P(B \text{ at } t + \Delta t \,|\, A \text{ at } t)$ using Gaussian Mixture Models (GMM). More precisely, all the A-events are aligned at time zero, whereupon all subsequent B-events are plotted. If an A-event tends to induce a B-event after a delay of t time-units, this will show up in this plot as a significant peak. All the non-related B-events will contribute to a very diffuse background. For that reason, we model the B-events as a 2-component GMM. One sharp and localized peak sits on top of the critical interval, while all the other B-events give rise to a flat and broad second component. The standard variation of the sharp peak immediately suggest a value for the width of the CI.

4.4 Experimental Results

In order to be able to compare results we use the same experimental layout (see Fig. 1).

The top two rows displayed in Fig. 5 show the results for the original CI-extraction (as detailed in [5]) and should be contrasted with the lower two rows that use the GMM modelling expounded above. It transpires that Magnusson's original scheme produces too many (spurious) T-patterns making high-confidence prediction impossible as is clear from the way the curves quickly drop to zero. This is most apparent in the 2-person scenario where the intermingling of 1-person patterns generates a large number of new combinations, a fair bit of which are erroneously identified as T-patterns. The GMM approach fares much better, even in the more difficult 2-person scenario.

5 Conclusion

In this paper we have reviewed two methodologies for the discovery of temporal patterns. The first one collapses the sequence into a string and then uses compression-based techniques to extract repetitive "words". The second one (so-called T-patterns) takes advantage of the time dimension to find the typical delay between related events. We have proposed some improvements to the basic T-pattern methodology (referred to in this text as GMM T-patterns) that significantly improve the performance. Experiments show that T-patterns outperform the compression-based techniques, which is not really surprising as the compression discards most of the temporal information. The experiments also show that the proposed T-pattern improvements (independence testing and GMM-modelling of correlation times) yield more reliable results.

To conclude we summarize the experimental results in Table 1. It was obtained by computing for each experiment the correct prediction rate for a confidence level of 20% (this amounts to constructing a vertical line at the x-value 0.20 in each of the figures and reading of the intersection with the solid curve). The significance of the proposed improvements is obvious.

	Layout 1		Layout 2	
	1 person	2 persons	1 person	2 persons
LZ	29.8	17.7	56.5	13.2
ALZ	21.1	18.8	66.4	19.6
LZW	28.9	22.0	60.5	15.1
T-patterns	28.8	17.1	61.5	24.2
GMM T-patterns	**34.8**	**29.3**	**61.9**	**48.3**

Table 1. Percentage correct predictions at the 20% confidence level.

Acknowledgment

This work was done while R.T. was at CWI. The authors gratefully acknowledge partial support by EC's NOE MUSCLE (FP6-507752), and the Dutch BSIK/BRICKS project. We are grateful to M.S. Magnusson for providing us with additional information on T-patterns.

References

1. Manuele Bicego, Marco Cristani, and Vittorio Murino. Unsupervised scene analysis: a hidden markov model approach. *Comput. Vis. Image Underst.*, 102(1):22–41, 2006.
2. Diane J. Cook. Prediction algorithms for smart environments. In Diane J. Cook and Sajal K. Das, editors, *Smart Environments: Technologies, Protocols, and Applications*. Wiley Series on Parallel and Distributed Computing, 2005.
3. N. Eagle and A. Pentland. Eigenbehaviors: Identifying structure in routine. In *Proc. of Ubicomp'06*, 2006.
4. J.L. Elman. Finding structure in time. *Cognitive Science*, 14:179–211, 1990.
5. M. S. Magnusson. Discovering hidden time patterns in behavior: T-patterns and their detection. *Behav Res Methods Instrum Comput*, 32(1):93–110, February 2000.
6. L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of IEEE*, 77(2):257–285, 1989.
7. S.P. Rao and D.J. Cook. Predicting inhabitant action using action and task models with application to smart homes. *International Journal on Artificial Intelligence Tools*, 13(1):81–99, 2004.
8. C.R. Wren, D.C. Minnen, and S.G. Rao. Similarity-based analysis for large networks of ultra-low resolution sensors. *Pattern Recognition*, 39:1918–1931, 2006.

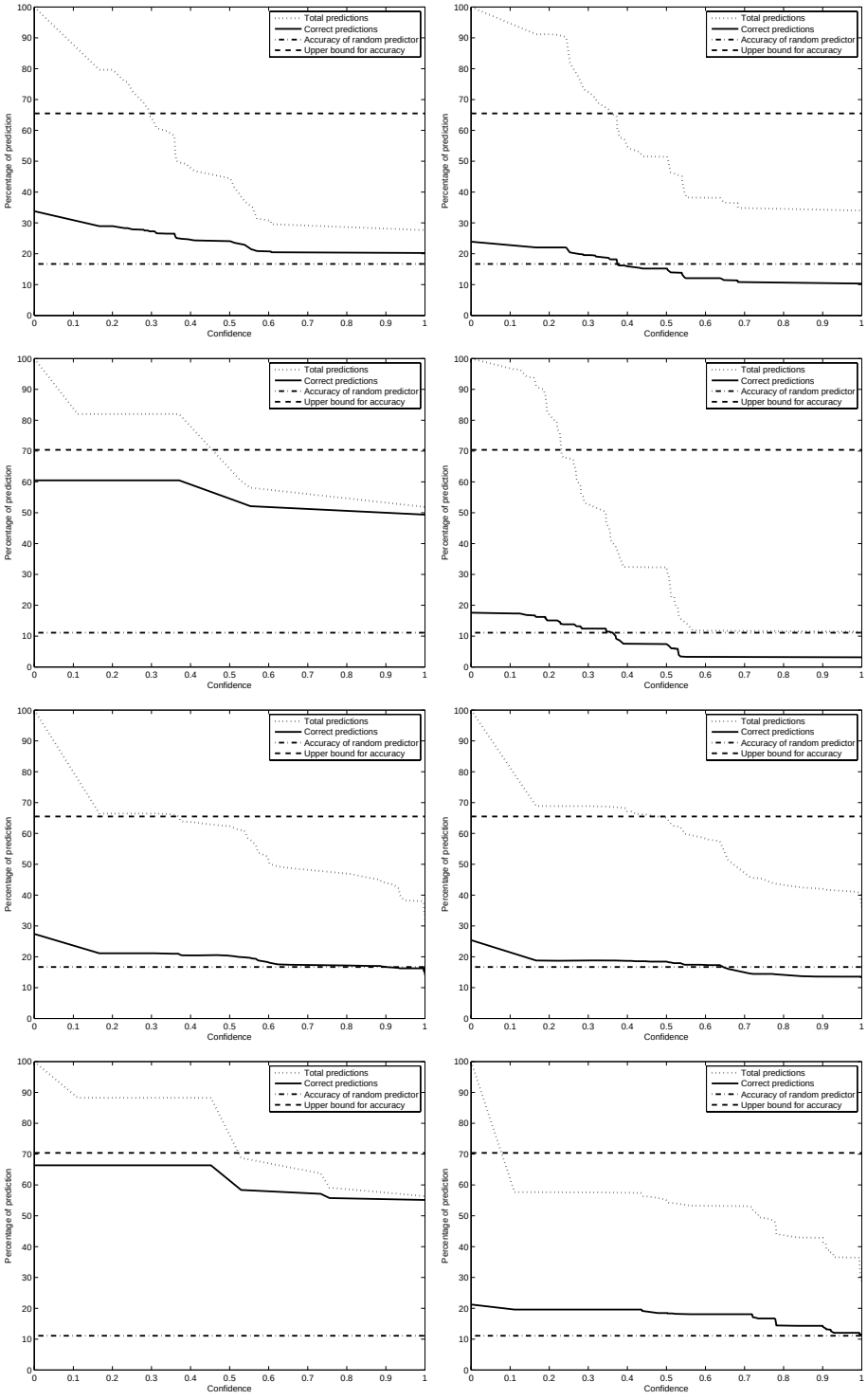

Fig. 3. Predition results for (i) the LZW compression algorithm for layout 1 (top row) and 2 (row 2) and (ii) the ALZ algorithm, row 3 corresponds to layout 1 and row 4 to layout 2. The left column show results for scenarios in which only one person is present, the right column shows results for the case when two person intermingle. See main text for a more detailed discussion.

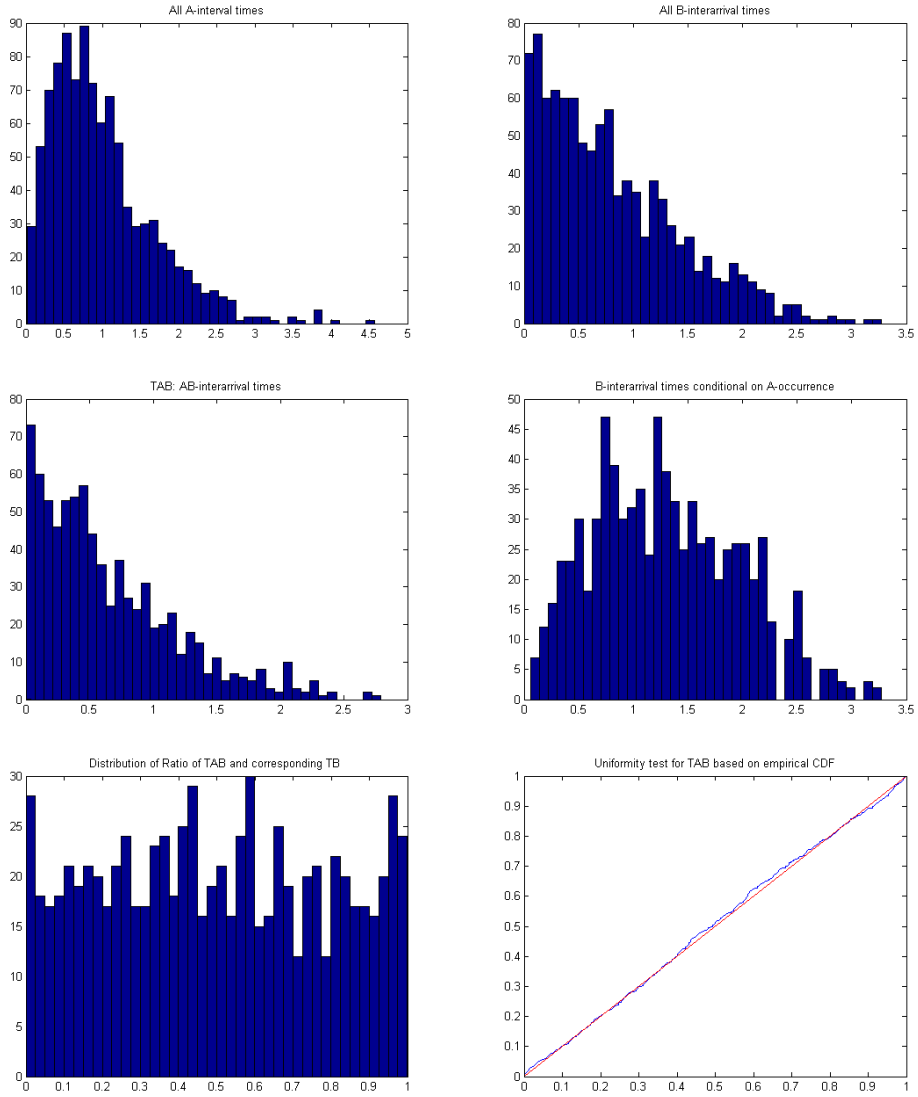

Fig. 4. *Top row:* Histogram for interevent times for the A (*left*) and B (*right*) process; *Middle row: left* T_{AB} distribution: time intervals between the occurrence of A and *next* B-event; *right:* Lengths of B-intervals in which a A-event occurred; notice the bias towards longer intervals (compared to histogram of all B interevent times above). *Bottom row: left:* Histogram of ratio $T_{AB}/\tilde{T}_B$, if A and B are independent, this ratio should be uniformly distributed between 0 and 1, a fact which is even more clearly borne out by its cumulative density function to the theoretically predicted one. (The p-value in this case was 0.61 which means that the null-hypothesis of independence is accepted.)

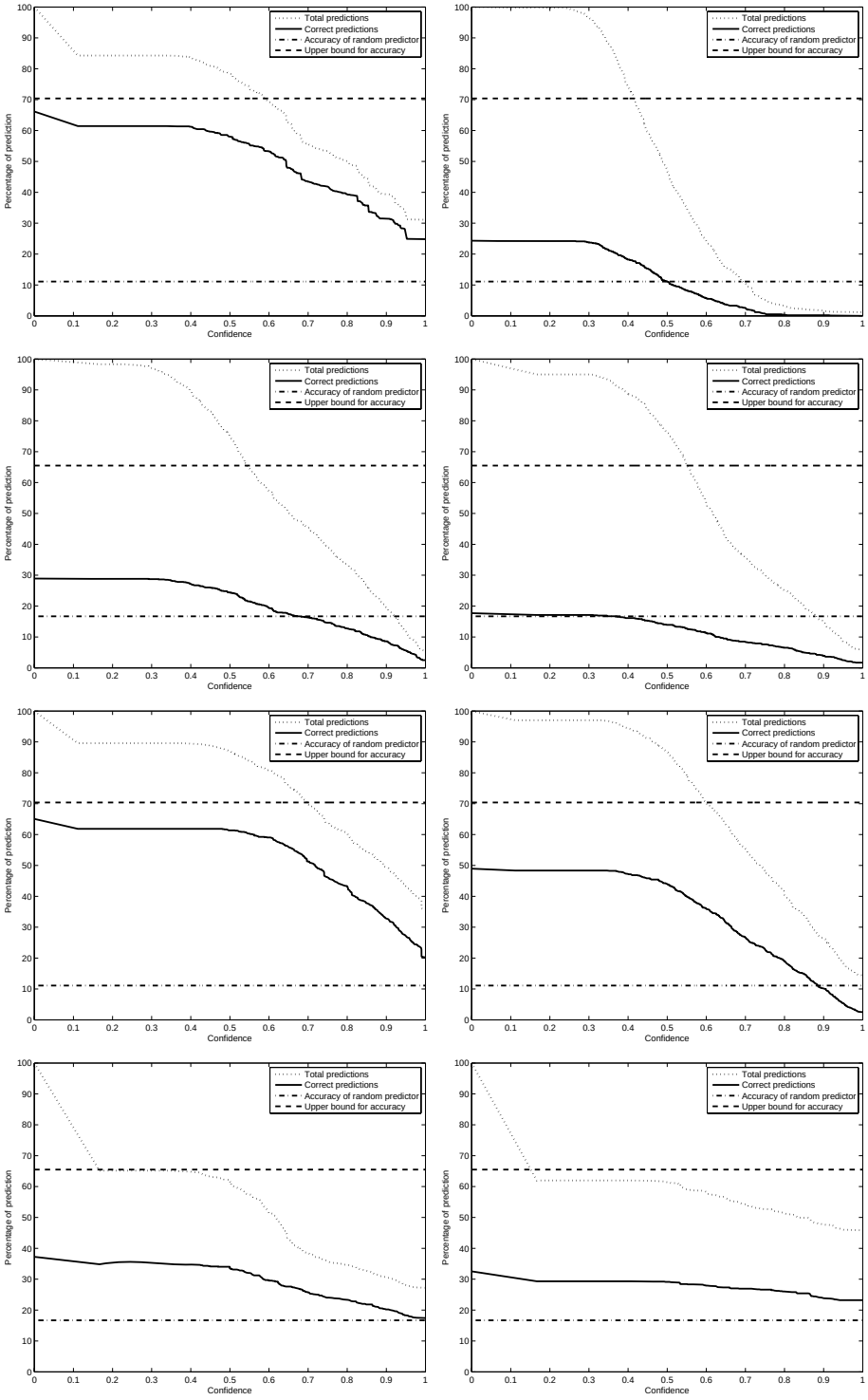

Fig. 5. Predition results for two implementation of T-patterns. The top two rows show the results for the original CI-extraction (as detailed in [5]) and should be contrasted with the lower two rows that use the GMM modelling explained in Section 4.3