Improving Activity Recognition by Segmental Pattern Mining

Umut Avci, Andrea Passerini
Dipartimento di Ingegneria e Scienza dell'Informazione
Università degli Studi di Trento
Trento, Italy
{*avci, passerini*}*@disi.unitn.it*

***Abstract*—Activity recognition is a key task for the development of advanced and effective ubiquitous applications in fields like Ambient Assisted Living. Most automated approaches for the task fail to incorporate dependencies between non-close time instants. In this paper we present a simple approach for introducing longer-range interactions based on sequential pattern mining. The algorithm searches for patterns characterizing time segments during which the same activity is performed. Novel sequences are tagged according to matches of the extracted patterns. An experimental evaluation shows that enriching sensor-based representations with the mined patterns allows improving results of sequential and segmental labeling algorithms on most of the cases.**

***Keywords*-Activity recognition; Pattern Mining; Segmental Labeling.**

I. Introduction

The automatic recognition of activities from sensor data is crucial for developing advanced applications in areas like Ambient Assisted Living and Assisted Cognition. Services like reminders and automatic reporting to clinicians and medical staff can help in improving healthcare and elders' independent living. From a machine learning viewpoint, activity recognition can be formalized as a sequence labeling task: given a sequence of sensor readings covering a timespan of interest (e.g. a day), predict the sequence of activities being performed. The timespan is typically divided into small time intervals, to be labeled with the activity or the activities taking place. We will refer to these time intervals as instants in this paper. A number of machine learning algorithms have been applied to this task, ranging from simple Naive Bayes [1] to sequential approaches like Hidden Markov Models [2], Conditional Random Fields [3] and their variants [4].

Local techniques like Naive Bayes or standard Support Vector Machines label each time instant independently, possibly extending its input representation over neighboring instants. On the other hand, sequential approaches collectively assign labels to all instants within the period of interest. This allows exploiting the relationship between activities performed at different time and usually results in performance improvements, other things being equal [5]. In modeling temporal interactions, however, these models are limited to rather small spans. Sequential approaches rely on a Markovian assumption to limit the number of parameters to be learned and keep inference tractable. In order to account for longer-range dependencies, shortcut links should be added between arbitrary time instants along the sequence, drastically increasing the complexity of the model and the cost of inference.

In this paper we address the problem of introducing longer-range dependencies in sequential labeling algorithms relying on sequential pattern mining techniques. An activity usually spans a certain amount of time, its average duration depending on the specific activity being performed (e.g. taking a shower or watching TV). We define an activity segment as a sequence of consecutive time instants in which the same activity is performed. Segmental labeling can be accomplished by semi-Markov models [6], which explicitly account for duration information. However, incorporating long-range dependencies between observations within each segment is a non-trivial task. Our solution consists of mining segmental patterns, i.e. sequential patterns which cover segments corresponding to a certain activity. By allowing gaps between matches of individual pattern elements, distant observations can be related. The set of mined patterns is employed to enrich the sensor-based representation of novel sequences: all time instants falling within a certain pattern are tagged with the pattern identifier. This enriched representation can be fed to any learning algorithm. Our experimental evaluation shows that including patterns allows improving results for sequential and segmental learning algorithms on most of the scenarios.

II. Data Representation

An input example $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$ consists of a consecutive sequence of observations, each covering a certain time instant t. An observation $\mathbf{x}_t$ is represented by the set of sensors which are active at that time instant (i.e. within its time interval). Different choices can be made in deciding when a sensor is considered active, as will be discussed in the experimental section. When feeding input sequences to labeling algorithms (see Section V), observations will be represented as binary vectors rather than sets. Given N sensors, an observation $\mathbf{x}_t$ will thus be encoded as a binary feature vector $\mathbf{x}_t = (x_t^1, \ldots, x_t^N)$, each feature being 1 if the corresponding sensor is active and 0 otherwise. The labeling task consists of predicting a sequence of activity

labels $\mathbf{y} = \{y_1, \ldots, y_T\}$, one for each time instant. Each label $y_t \in [1, L]$ is one of L possible activities, with one indicating no activity. We assume here that activities are not simultaneous, i.e. a single (or no) activity is performed at each time. The segmental pattern mining algorithm can however be quite straightforwardly generalized to deal with multiple simultaneous activities. We define an activity segment as a sequence of consecutive time instants labeled with the same activity. A segment $s_u = (b_u, e_u, y_u)$ is represented by its starting and ending time instants $b_u, e_u \in [1, T]$, with $e_u \geq b_u$, and the segment label y_u. A label sequence $\mathbf{y}$ can be split into a sequence $\mathbf{s} = \{s_1, \ldots, s_U\}$ of activity segments such that $b_u = e_{u-1} + 1$ and $y_u \neq y_{u-1}$ for all u. We define as $\mathbf{x}(s_u)$ the segment of $\mathbf{x}$ ranging from b_u to e_u included. $\mathcal{D}$ represents a dataset of input-output sequences, $\mathcal{S}$ its segmented version. $\mathcal{S}_y$ is the set of segments for activity y, and $\mathcal{D}(\mathcal{S}_y)$ the corresponding set of input segments.

III. Segmental Pattern Mining

Our aim is mining patterns characterizing timespans during which a certain activity is performed. Algorithm 1 shows the pseudocode of our segmental pattern miner. Training sequences are first split into activity segments, each labeled with the corresponding activity. These segments are fed to a sequential pattern miner (procedure SEQUENTIALMINER). We employed `pboost` [7] which supports discriminative mining, i.e. mining of patterns distinguishing sequences of a certain class from the others. In the following we provide a brief description of the algorithm. Further details can be found in the original paper [7]. The algorithm takes as input sets of positive and negative examples, each example being a sequence of sets of integers (the sensor identifiers in our case). It returns patterns as subsequences matching positive and not negative examples. Let $\mathbf{p} = (\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_m)$ be a pattern of length m. Let $\mathbf{p}_i$ be a pattern element, corresponding to a non-empty set of active sensors. The pattern $\mathbf{p}$ matches sequence $\mathbf{x}$ if there is a match $(t_1, t_2, \ldots, t_m)$ such that: for all $i > j$, $t_i > t_j$; for all i, $\mathbf{x}_{t_i}$ contains active sensors $\mathbf{p}_i$ (i.e. $\mathbf{p}_i \subseteq \mathbf{x}_{t_i}$). By defining a canonical ordering for sequences, the pattern space is searched in a tree-based fashion starting from the empty pattern [8]. Pruning of the search space is conducted combining the standard notion of *support* (i.e. number of matching sequences) with that of *gain*: `pboost` considers each pattern as a feature and learns a linear classifier (LPBoost [9]) on top of them, discriminating between positive and negative examples. The gain provided by a feature can be compared with an upper bound on the maximal gain achievable by further extending the corresponding pattern. If the gain exceeds the upper bound there is no need to proceed in this search direction.

For each of the possible activities, `pboost` is run providing segments of the target activity as positive examples and all other segments as negative ones. Each of the returned patterns $\mathbf{p}$ is evaluated according to its discriminative power,

Algorithm 1 Procedure for segmental pattern mining

```
1: procedure SEGMENTALMINER(D, φ)
2:     Initialize P to the empty set
3:     Split training sequences into activity segments S
4:     for all activities y do
5:         S_y = segments for y
6:         S_ȳ = segments for y' ≠ y
7:         P_y = SEQUENTIALMINER(D(S_y), D(S_ȳ))
8:         for all p ∈ P_y do
9:             if SCORE(p, D(S_y), D(S_ȳ)) ≥ φ then
10:                g = MEDIANGAP(p, D(S_y))
11:                P = P ∪ {(p, g)}
12:            end if
13:        end for
14:    end for
15:    return P
16: end procedure
```

computed as its F1 score on the training segments (procedure SCORE). The F1 measure (see experiments) trades off precision, i.e. the fraction of segments covered by the pattern which do belong to the target activity, and recall, i.e. the fraction of segments of the target activity covered by the pattern. All patterns with a score lower than a certain threshold ϕ are discarded. The threshold ϕ is a parameter of the algorithm, which was optimized by a preliminary cross validation procedure as described in the experimental section.

Sequential patterns extracted by `pboost` allow for arbitrary gaps between the pattern elements. Our aim is that these patterns cover the largest possible portion of an activity segment. However, in the test phase when used to label a novel time sequence, the activity segmentation will be unknown and the patterns will be applied to the whole sequence (i.e. a day). We thus need to estimate the expected number of gaps in pattern matches for activity segments. This is crucial for a correct use of patterns: during test, allowing for arbitrary gaps within a pattern could produce matches involving very distant time instants (e.g. in the morning and afternoon), which likely belong to different activity segments. In order to provide an estimate of the correct number of gaps, the algorithm computes the median gap length of each pattern on the positive segments (procedure MEDIANGAP). The final outcome is a set of patterns characterizing all activities, together to their estimated gap lengths.

IV. Pattern-based Sequence Representation

During test, activity segments cannot be used as labeling information is not available and full sequences (e.g. a day) have to be considered. However, the fact that a pattern spans a certain subsequence is an indication that the corresponding activity could be performed during that period of time. We

Algorithm 2 Procedure for tagging sequence with pattern matches

procedure TAGSEQUENCE($\mathcal{P}$, $\mathbf{x}$)
  Let $\hat{\mathbf{x}}$ be a sequence of $|\mathbf{x}|$ binary vectors of size $|\mathcal{P}|$
  Initialize all vectors in $\hat{\mathbf{x}}$ to zero
  for all $(\mathbf{p}, g) \in \mathcal{P}$ **do**
    for all $t_1 \in [1, |\mathbf{x}|]$ **do**
      if HASMATCH($\mathbf{x}$,t_1,$\mathbf{p}$,g) **then**
        $(t_1, \ldots, t_m)$ = BESTMATCH($\mathbf{x}$,t_1,$\mathbf{p}$,g)
        for all $t \in [t_1, t_m]$ **do**
          $\hat{x}_t^p = 1$
        end for
      end if
    end for
  end for
  return $\hat{\mathbf{x}}$
end procedure

thus tag each element of the subsequence with the pattern identifier.

The sequence tagging algorithm is sketched in Algorithm 2. It takes as inputs a set of patterns with their gap lengths and the sequence to tag, and outputs a pattern-based representation of the sequence. Each time instant in the output sequence $\hat{\mathbf{x}}$ is represented as a binary vector with a length equal to the number of patterns. Each element $\hat{x}_t^p$ will be set to one if time instant t is contained in a match of pattern $\mathbf{p}$, and zero otherwise.

For each pattern, the algorithm scans the sequence looking for matches. Let $\mathbf{p} = (\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_m)$ be a pattern of length m and let g be its gap length. A gapped match $(t_1, t_2, \ldots, t_m)$ for the pattern is a sequence of instants such that: for all $i > j$, $t_i > t_j$; for all i, $\mathbf{p}_i \subseteq \mathbf{x}_{t_i}$; the sequence has at most g gaps, i.e. $t_m - t_1 + 1 \leq m + g$. The procedure BESTMATCH($\mathbf{x}$,t_1,$\mathbf{p}$,g) finds the longest gapped match of pattern $\mathbf{p}$ in sequence $\mathbf{x}$ starting at instant t_1. All instants from t_1 to t_m are then tagged with the identifier of pattern $\mathbf{p}$. The search is repeated for all possible starting instants t_1, from one to the end of the sequence.

The algorithm outputs a pattern-based representation of the sequence, where each instant is tagged with the set of patterns with matches containing it. This type of representation can complement sensor-based ones by adding information concerning long-range interactions, which would likely be lost otherwise.

V. ACTIVITY RECOGNITION ALGORITHMS

The pattern-enriched representation of sequences can be fed to any learning algorithm performing sequence labeling. Our aim is evaluating the impact of this enriched representation on learning algorithms with respect to the assumptions they make on the relationship between time instants. We thus focused on Hidden Markov Models and Hidden Semi-Markov Models, which perform sequential and segmental labeling respectively. These algorithms have been recently compared [4], [5] on a benchmark consisting of wireless sensor network data. Both algorithms model the joint probability distribution of input and output $p(\mathbf{x}, \mathbf{y})$ and return the output maximizing this probability, i.e. $\mathbf{y}^* = \text{argmax}_{\mathbf{y}} p(\mathbf{x}, \mathbf{y})$.

A Hidden Markov Model (HMM) is a sequential approach where: 1) the label at each time instant depends on the label at the previous time instant only; 2) the observation at each time instant depends on the label at that time instant only; 3) probabilities do not depend on the specific time instants but only on the values of labels/observations at those instants. The resulting joint probability is given by:

$$p(\mathbf{x}, \mathbf{y}) = \prod_{t=1}^{T} p(\mathbf{x}_t|y_t) p(y_t|y_{t-1})$$

where $p(y_1|y_0)$ stands for the probability of having y_1 as the initial label. In modeling the conditional probability of observations given label, a common simple approach (also followed in [4], [5]) consists of making a Naive assumption of independence between observation features given the label. The resulting probability is:

$$p(\mathbf{x}_t|y_t) = \prod_{i=1}^{N} p(x_t^i|y_t)$$

where for the binary case, probabilities for features $p(x_t^i|y_t)$ are represented as Bernoulli distributions.

HMMs imply an exponential distribution for state durations. The probability of seeing label l for d consecutive instants is d times the probability of a self transition $p(y_t = l|y_{t-1} = l)$. This assumption is often not appropriate when durations tend to have specific patterns, as happens in activity recognition tasks. Explicit duration distributions can be represented by Hidden Semi-Markov Models (HSMM), which consider probability of segmental labeling $(\mathbf{x}, \mathbf{s})$. Recall that $\mathbf{s}$ is a sequence of U consecutive segments $s_u = (b_u, e_u, y_u)$. The corresponding joint probability is represented as:

$$p(\mathbf{x}, \mathbf{s}) = \prod_{u=1}^{U} p(y_u|y_{u-1}) p(d_u|y_u) \prod_{t=b_u}^{e_u} p(\mathbf{x}_t|y_u)$$

where $d_u = e_u - b_u + 1$ is the duration of segment s_u, whose probability can be modeled by the desired distribution. For details on HSMM see [6]. Following [4], we employed a Gaussian distribution for label duration in HSMM and independent Bernoulli distributions for observation features (as HMM). We did not include Conditional Random Fields and their Semi-Markov extension in our comparison, as they require much longer training time and were shown to provide comparable and often worse results with respect

to their directed counterparts (HMM and HSMM) on this benchmark [4].

VI. Experiments

In this section we first present the experimental setup used for evaluating the proposed approach and then provide results of the experiments.

A. Setting

We performed our experiments on the freely available[1] benchmark described in [4], [5]. The datasets include information regarding three different houses comprising several wireless sensor networks. Each node of the network is attached to the ad hoc sensors, e.g., reed switches, passive infrared. Annotation of the activities was achieved by recording the start and end time of the corresponding activity either via handwritten diary or bluetooth headset. Table I presents a summary of the characteristics of the datasets.

Table I: Details of the datasets

	House A	House B	House C
Duration	25 days	14 days	19 days
Sensors	14	23	21
Activities	10	13	16
Annotation	Bluetooth	Diary	Bluetooth

Activities to be recognized were derived from the Katz ADL index which is a measure qualifying the ability of individuals to sustain their lives independently. Tables III and IV indicate those used in our experiments for House A and House C respectively. House B differs from House A by introducing new activities 'Getting dressed', 'Preparing brunch', 'Washing dishes', 'Eating dinner', and 'Eating Brunch' excluding 'Preparing breakfast' and 'Getting Snack'. 'Idle' represents the time spans at which either none of the annotated activities or no activity is observed.

Data acquired from the sensors were processed in different ways to create feature representations. We focused on the *Changepoint* (C) and the *Last-fired* (L) representations which were by far the most effective ones [5]. The former considers a sensor active (value 1) only in the time instants in which it alters its state. The latter keeps considering the last sensor which changed state active until another sensor changes its state. Figure 1 depicts the working mechanism of these representations as compared to the original raw one, where a sensor is active in all time instants in which it fires.

(a) Original (b) Changepoint (c) Last-fired

Figure 1: Feature representations

[1] https://sites.google.com/site/tim0306/codeFramework.zip

A 'Leave-one-day-out' (LOO) approach was used to split the datasets into training and test sets. Each day was in turn considered as a test set, while all other days made up the training set. Performance measures include precision (Pr=TP/(TP+FP)), recall (Re=TP/(TP+FN)), F1 (F1=$(2 \cdot \text{Pr} \cdot \text{Re})$/(Pr+Re)) and accuracy (Acc=(TP+TN)/(TP+FP+TN+FN)) for each class. Here TP and FP are the fraction of true and false positives respectively, while TN and FN are the fraction of true and false negatives respectively. F1 is the harmonic average of precision and recall trading off the two.

We compared HMM and HSMM with and without pattern-mined features, the latter corresponding to the experimental setting reported in [5]. Patterns were mined on the L sensor representation, as it provides a much larger fraction of time instants having active sensors. The C representation is extremely sparse preventing the mining of patterns complex enough to cover significant fractions of the activity segments. We employed a threshold ϕ on patterns F1 score in order to select the most discriminant patterns (see Section III). Determination of the threshold value was automated by performing internal cross validation within the training folds of the LOO approach. The training set of each fold was again split into training and testing sets by applying the same LOO approach. For each internal validation step, a range of possible threshold values was evaluated and the one maximizing the sequential learner F1 measure was recorded as a candidate ϕ. The final ϕ was chosen as the most frequently recorded one across the different internal cross validations. This was then employed in the outer cross validation procedure whose results are reported in the following section. Note that the same thresholds were obtained by this procedure for HMM and HSMM learners, i.e. $\phi = 0.95$ for House A and House B and $\phi = 0.6$ for House C.

B. Results

Results of the experiments are shown in the Table II for House A, House B, and House C respectively, averaged over days and activities. The C representation is usually superior to the L one, as previously observed [5]. Simply combining the two representations (rows C+L) does not allow improving over the best of the two. On the other hand, including the pattern-based features (C+LP) succeeds in improving results in four out of six cases. The two cases where patterns are not helpful are those for House B. By comparing results for C and L representations, it is apparent that the L one is much worse in this setting. Hence, patterns extracted from it inherit the same deficiency, which prevents the C+LP combination from boosting the results.

Table III and Table IV show the breakdown of the results by activity for the HSMM case, which performed better through all experiments, for the two houses where patterns provide improvements. Results indicate that the contribution

Table II: Results of the experiments averaged across activities

		House A			House B			House C		
Model	Feature	**Precision**	**Recall**	**F-1**	**Precision**	**Recall**	**F-1**	**Precision**	**Recall**	**F-1**
HMM	**C**	70±16	74±13	72±14	48±17	63±14	54±17	41±9	50±11	45±9
	L	55±17	70±13	61±15	39±16	47±20	42±17	41±10	54±16	46±11
	C+L	67±18	79±12	72±15	37±16	48±20	42±17	40±10	56±16	46±10
	C+LP	78±13	81±11	79±12	46±15	62±14	52±14	49±16	56±14	52±15
HSMM	**C**	71±16	75±12	72±14	50±16	65±13	56±15	44±10	52±13	47±11
	L	60±15	74±13	66±14	41±12	53±11	46±11	43±11	56±15	48±11
	C+L	67±17	80±13	73±15	42±13	59±10	49±11	41±10	57±16	47±11
	C+LP	77±13	82±11	79±12	47±15	66±13	54±13	50±16	58±14	53±15

Table III: Breakdown of the results by activity for House A

	C			C+L			C+LP		
	Precision	**Recall**	**F-1**	**Precision**	**Recall**	**F-1**	**Precision**	**Recall**	**F-1**
Idle	0.890	0.508	0.646	0.963	0.634	0.764	0.942	0.744	0.831
Leaving house	0.944	0.997	0.970	0.984	0.997	0.991	0.977	0.999	0.988
Using toilet	0.739	0.822	0.778	0.744	0.795	0.768	0.767	0.795	0.781
Taking shower	0.948	0.649	0.771	0.375	0.904	0.530	0.960	0.849	0.901
Brushing teeth	0.172	0.344	0.229	0.132	0.375	0.195	0.222	0.438	0.295
Going to bed	0.901	0.965	0.932	0.981	0.995	0.988	0.954	0.994	0.974
Preparing breakfast	0.566	0.690	0.622	0.550	0.575	0.562	0.591	0.747	0.660
Preparing dinner	0.670	0.516	0.583	0.291	0.770	0.423	0.710	0.596	0.648
Getting snack	0.426	0.548	0.479	0.162	0.548	0.250	0.512	0.524	0.518
Getting drink	0.674	0.674	0.674	0.593	0.653	0.621	0.674	0.592	0.630

of the L representation is rather unstable across activities, preventing an overall improvement for the combined C+L representation. Conversely, the C+LP representation is much more robust, managing to combine the advantages of the two representations. Consider, for instance, the activities 'Taking shower', 'Idle' and 'Leaving house'. These are closely related as they are typically performed in a row by the resident. A simple but commonly found pattern consists of a long repetition of the 'front door' sensor. This clearly indicates a 'Leaving house' activity is taking place. It also helps in disambiguating temporally close activities like the just mentioned 'Idle' and 'Taking shower' ones. Conversely, including L representation also introduces noisy features: a sensor activated while taking shower will continue to be considered active when the resident is actually idle, leaving to a drop in precision for the 'Taking shower' prediction. Concerning House C, a commonly found pattern consists of a sequence of sensor activations for the fridge, the herbs cupboard and the fridge again, characterizing the 'Preparing dinner' activity. Let us consider a possible scenario for this situation. A resident starts cooking his dinner by taking ingredients from the fridge. After a while, flavoring herbs and spices taken from the herbs cupboard are added into the blend. As soon as the meal is ready, the resident puts the remaining ingredients back to the fridge. Introducing such patterns allows relating observations which are not sufficient to discriminate among similar activities if taken alone. For instance, a similar scenario involving usage of fridge and other kitchen appliances can be observed for preparing breakfast. The patterns found for this last activity include activations for the cutlery drawer, the bowl cupboard, and the fridge. Both activities are actually better recognized using the C+LP representation.

The method is applicable for scenarios rich in features enabling distinctive activity patterns to be discovered. In the case of very sparse feature representations, matching patterns with wide range of possible gap lengths might be expensive. Cautious consideration could be required to constrain the set of valid patterns. As far as the system architecture is concerned, environmental sensors working in synergy with each other provide more complete information regarding activities by eliminating individual deficiencies. For instance, dependence mostly on motion detectors leads to produce similar patterns which is not enough to infer activities with high accuracy and sometimes incapable of capturing specific activities.

VII. Related Work

The problem of dealing with long-range dependencies is well-known in the machine learning community. A number of attempts at addressing it rely on hierarchical models like hierarchical HMMs [10] and their many recent variants [11], [12]. These try to model higher level dependencies between non-atomic activities which should account for the long-term dependencies. However, these models require a good amount of knowledge about the underlying structure of the problem in building the hierarchies. Indeed, a two-level hierarchical HMM, modeling activity segments at the lower lever and sequences of segments at the upper one, did not improve over plain HMM in our preliminary experiments. Another approach consists of explicitly adding links between distant time instants which are deemed to be in direct

Table IV: Breakdown of the results by activity for House C

	C			C+L			C+LP		
	Precision	**Recall**	**F-1**	**Precision**	**Recall**	**F-1**	**Precision**	**Recall**	**F-1**
Idle	0.588	0.390	0.469	0.655	0.702	0.678	0.697	0.870	0.774
Leaving house	0.765	0.951	0.848	0.982	0.860	0.917	0.979	0.863	0.917
Eating	0.375	0.372	0.373	0.423	0.344	0.380	0.452	0.479	0.465
Using toilet 1	0.380	0.582	0.460	0.290	0.684	0.408	0.497	0.532	0.514
Taking shower	0.644	0.390	0.485	0.365	0.584	0.449	0.529	0.574	0.551
Brushing teeth	0.306	0.406	0.349	0.273	0.267	0.270	0.242	0.228	0.235
Using toilet 2	0.259	0.475	0.335	0.198	0.488	0.282	0.278	0.438	0.340
Shaving	0.402	0.536	0.460	0.337	0.507	0.405	0.284	0.391	0.329
Going to bed	0.997	0.822	0.901	0.997	0.981	0.989	0.997	0.983	0.990
Getting dressed	0.581	0.670	0.622	0.602	0.688	0.642	0.658	0.688	0.673
Medication	0.308	0.267	0.286	0.129	0.267	0.174	0.150	0.200	0.171
Preparing breakfast	0.033	0.028	0.030	0.038	0.070	0.049	0.152	0.296	0.201
Preparing lunch	0.142	0.250	0.181	0.075	0.167	0.103	0.117	0.117	0.117
Preparing dinner	0.672	0.417	0.515	0.234	0.772	0.359	0.531	0.617	0.571
Getting snack	0.136	0.375	0.200	0.088	0.208	0.124	0.171	0.292	0.215

relationship, like in Skip-chain CRF [13]. Shortcut links considerably increase the complexity of the inference task and should thus be carefully selected. Skip-chain CRF [13] have been successfully applied to named-entity recognition, where shortcut links were added between pairs of identical capitalized words. Our segmental pattern miner extracts patterns which should approximately span activity segments. Their matches are thus natural candidates to add shortcuts, possibly connecting distant segments representing the same or closely related activities. We are pursuing this direction in our ongoing research.

VIII. Conclusions

We presented a segmental pattern mining approach for enriching sequential representations with patterns characterizing interactions within activity segments. Experimental results show the usefulness of the mined patterns in improving predictive power of learning algorithms. Our approach is not limited to activity recognition tasks and is readily applicable to sequential labeling problems characterized by segments of consecutive positions sharing the same label (e.g. intron-exon identification in DNA sequences). The segmental mining strategy can also be used for suggesting promising topologies for graphical models trying to directly incorporate long-range dependencies.

Acknowledgments

The research was funded by the Autonomous Province of Trento, Call for proposal Major Projects 2006 (project ACube).

References

[1] L. Bao and S. Intille, "Activity recognition in the home setting using simple and ubiquitous sensors," in *Proc. of PERVASIVE 2004*, 2004, pp. 273–297.

[2] M. Philipose, K. Fishkin, M. Perkowitz, D. Patterson, D. Fox, H. Kautz, and D. Hähnel, "Inferring activities from interactions with objects," *IEEE Pervasive Computing*, vol. 3, pp. 50–57, 2004.

[3] D. Vail, M. Veloso, and J. Lafferty, "Conditional random fields for activity recognition," in *Proc. AAMAS 2007*, 2007, pp. 1331–1338.

[4] T. van Kasteren, G. Englebienne, and B. Kröse, "Activity recognition using semi-markov models on real world smart home datasets," *Ambient Intelligence and Smart Environments thematic issue on Smart Homes*, vol. 2, no. 3, pp. 311–325, August 2010.

[5] ——, "Human activity recognition from wireless sensor network data:benchmark and software," in *Activity Recognition in Pervasive Intelligent Environment*. Atlantis Press Book, 2010, pp. 165–186.

[6] S.-Z. Yu, "Hidden semi-markov models," *Artificial Intelligence*, vol. 174, no. 2, pp. 215–243, 2010.

[7] S. Nowozin, G. Bakir, and K. Tsuda, "Discriminative subsequence mining for action classification," in *Proc. ICCV 2007*, 2007, pp. 1–8.

[8] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu, "Mining sequential patterns by pattern-growth: The prefixspan approach," *IEEE Trans. on Knowl. and Data Eng.*, vol. 16, pp. 1424–1440, 2004.

[9] A. Demiriz, K. P. Bennett, and J. Shawe-Taylor, "Linear programming boosting via column generation," *Mach. Learn.*, vol. 46, pp. 225–254, 2002.

[10] S. Fine, Y. Singer, and N. Tishby, "The hierarchical hidden markov model: Analysis and applications," *Mach. Learn.*, vol. 32, pp. 41–62, July 1998.

[11] T. Duong, D. Phung, H. Bui, and S. Venkatesh, "Efficient duration and hierarchical modeling for human activity recognition," *Artif. Intell.*, vol. 173, pp. 830–856, May 2009.

[12] P. Natarajan and R. Nevatia, "Hierarchical multi-channel hidden semi markov models," in *Proc. of IJCAI'07*, 2007, pp. 2562–2567.

[13] C. Sutton and A. McCallum, "An introduction to conditional random fields for relational learning," in *Introduction to Statistical Relational Learning*, L. Getoor and B. Taskar, Eds. MIT Press, 2007.