

Relational Transformation-based Tagging for Human Activity Recognition

Niels Landwehr¹, Bernd Gutmann¹, Ingo Thon¹, Matthai Philipose², and Luc De Raedt¹

¹ Department of Computer Science
Katholieke Universiteit Leuven
Celestijnenlaan 200 A, B-3001 Heverlee, Belgium
firstname.lastname@cs.kuleuven.be

² Intel Research Seattle
1100 NE 45th Street
Seattle, WA 98105, USA
matthai.philipose@intel.com

Abstract. The ability to recognize human activities from sensory information is essential for developing the next generation of smart devices. Many human activity recognition tasks are — from a machine learning perspective — quite similar to tagging tasks in natural language processing. Motivated by this similarity, we develop a relational transformation-based tagging system based on inductive logic programming principles, which is able to cope with expressive relational representations as well as a background theory. The approach is experimentally evaluated on two activity recognition tasks and compared to Hidden Markov Models, one of the most popular and successful approaches for tagging.

1 Introduction

Smart systems that assist humans must be able to recognize the current context of the user and the activity she is performing in order to suggest or take actions in an intelligent manner. To recognize the context and activity, such systems can rely on streams of past activities, context, and sensory information (visual, object-interaction, ...). Recognizing the current activity or context then corresponds to inferring the activity or context from such sequential information. From a machine learning perspective, this task is akin to many tagging tasks pursued in natural language processing. For instance, in part-of-speech tagging, a form of "shallow parsing", the words in a sentence are to be labeled with the corresponding parts-of-speech (word categories). Many techniques have been developed and employed for this purpose. Two popular techniques for part-of-speech tagging are Hidden Markov Models and transformation-based learning [1]. However, whereas Hidden Markov models have been applied in many different areas, ranging from speech-recognition to activity recognition and bio-informatics, to the best of the authors' knowledge, transformation based learning has only seldomly been applied outside the field of natural language processing.

Because the structure of natural language is quite rigid as compared to that of typical activity recognition tasks, the existing transformation-based learners cannot directly

be applied for activity recognition. Therefore, we develop a more flexible *relational* transformation-based tagger within the inductive logic programming paradigm. This does not only provide an *expressive* representation but also allows one to easily incorporate background theory during the learning process. Thus the key contribution of this paper is a relational extension of transformation-based tagging based upon inductive logic programming principles. It also extends earlier work on relational transformation-based learning by [2] in that it focuses on *tagging* rather than *classification*. More specifically, from inductive logic programming (and the work by [2]) our technique inherits its search and refinement techniques (including a branch-and-bound algorithm) and from transformation-based learning the error driven stacking of rules.

The proposed method is evaluated in two activity recognition domains: “Activities of Daily Living” (ADL) recognition from a stream of “object interaction” data [6], and mobile phone profile prediction based on data collected by [8]. Experiments show that obtained tagging accuracies are competitive with those of HMM-based approaches, and it is easy to incorporate human-supplied background knowledge into the learning process. Furthermore, and that is perhaps the key advantage of the relational transformation-based tagger, the method can easily be extended to deal with variants of the tagging problem, for instance the prediction of structured output tags (as in Logical Hidden Markov Models [4]), and to cope with rich background knowledge.

2 Sequence Tagging

Sequence tagging is the task of assigning to each element in a given sequence an appropriate label or *tag*. Let $W = \{w^1, \dots, w^k\}$ denote the vocabulary of sequence elements, and $T = \{t^1, \dots, t^m\}$ the vocabulary of tags. The most prominent instance of the tagging problem is part-of-speech-tagging in natural language processing, where the task is to assign lexical categories $t \in T$ to words $w \in W$ in a given natural language sentence. Many other interesting sequence analysis problems can be cast in this framework, such as activity recognition in user modeling or gene finding and protein secondary structure prediction in bioinformatics.

In NLP, the two most common tagging approaches are transformation-based taggers (rule-based) and probabilistic methods (hidden Markov models or related techniques). Both of these approaches yield competitive results, and have received much attention. Before discussing our extension to transformation-based learning, we briefly review these two approaches in the next two sections.

2.1 Transformation-based Tagging

Transformation-based learning is a rule-based learning approach which iteratively stacks rules on top of each other to improve performance [1]. The basic transformation-based learning algorithm for the tagging problem is summarized in Algorithm 1. The algorithm takes as input a set S of sequences with known true tags L . During learning, it maintains a set of current tags \hat{L} for all $s \in S$. \hat{L} is initialized with some simple scheme, such as assigning to every element $w \in W$ its most common tag $t \in T$ in the training data (procedure *initial-tags*). The algorithm then tries to improve the current

Algorithm 1 Basic transformation-based tagging algorithm.

tb-tagging(input: sequences S ; true sequence tags L)

```

1  $\hat{L} := \text{initial-tags}(S, L)$ 
2 initialize  $R = []$ 
3 repeat
4      $r := \text{find-best-rule}(S, \hat{L}, L)$ 
5     update  $\hat{L} := \text{apply-rule}(\hat{L}, r)$ 
6     update  $R := \text{append}(R, r)$ 
7 until (no improvement)
8 return  $R$ 

```

tagging \hat{L} with respect to the true tagging L by learning a list of *transformation rules* R . Transformation rules can re-tag sequence elements based on the context they appear in. A transformation rule has the form $t' \leftarrow t : \text{context}$ and simultaneously replaces all occurrences of tag t in all sequences with t' whenever the constraint *context* is satisfied.

Example 1. As an example from NLP, the word “move” could be initially tagged as “verb”, but would be re-tagged as “noun” if the preceding word was tagged as “article”. This can be encoded by the following transformation rule:

$$\text{noun} \leftarrow \text{verb} : \text{word} = \text{move}, \text{preceding tag} = \text{article}$$

The transformation rule languages employed in traditional transformation-based tagging are mostly simple instantiations of some template—for instance, querying in *context* the word and tag at the current position and the next or preceding position(s). We will replace this constraint by a first-order logical expression in Section 3.

In every iteration, the transformation rule which yields the greatest reduction in error between \hat{L} and L is greedily selected (*find-best-rule*), applied to the current tagging \hat{L} and appended to the rule list R . As conditions of rules in R match not only sequence elements but also currently predicted tags \hat{L} , rules can effectively bootstrap the current predictions. This makes transformation-based learning strictly more powerful than standard rule learning [1].

2.2 Hidden Markov Model Tagging

Tagging with hidden Markov models is typically performed with a model in which there is a hidden state q_t for every possible tag t , and state emission symbols correspond to symbols $w \in W$. That is, the observed sequence of symbols is seen as being generated by the hidden sequence of tags. Formally, the joint probability of an observation sequence $s = w_1 \dots w_n$ with hidden tag sequence $t_1 \dots t_n$ is given by

$$P(w_1 \dots w_n, t_1 \dots t_n) = P(t_1) \prod_{i=1}^{n-1} P(t_{i+1} | t_i) P(w_i | t_i)$$

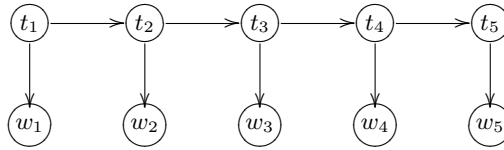


Fig. 1. Example lattice generated by unrolling a tagging HMM to a sequence w_1, \dots, w_5 . Inference in this model is carried out with the Viterbi algorithm, which yields the most likely joint state of the hidden variables t_1, \dots, t_5 given the observations on w_1, \dots, w_5 .

where $P(t_1)$ is an initial probability for tag t_1 and $P(w_i | t_i)$, $P(t_i | t_{i-1})$ are conditional probabilities for the emitted word w_i and next tag t_{i+1} given the current tag t_i . When such a model is applied to a sequence $w_1 \dots w_n$, it is unrolled into a lattice as depicted in Figure 1, and the Viterbi algorithm [7] is employed to efficiently compute

$$\begin{aligned} \hat{t}_1 \dots \hat{t}_n &= \arg \max_{t_1 \dots t_n} P(t_1 \dots t_n, w_1 \dots w_n) \\ &= \arg \max_{t_1 \dots t_n} P(t_1 \dots t_n | w_1 \dots w_n), \end{aligned}$$

the most likely sequence of tags for the given sequence.

This technique has been used successfully for tagging problems in many domains. For instance, HMM-based approaches are a popular technique for inferring hidden user activities from a stream of object-interaction data in the so-called ADL (“Activities of Daily Living”) problem [6, 10], which will be described in more detail below.

3 Relational Transformation-based Tagging

The general motivation for our work on relational transformation-based tagging is to apply the transformation-based tagging methodology to complex datastreams, which are generated for instance by sensors or sensor networks in ubiquitous computing environments. For such complex domains it is not always possible to represent all available information as flat (or *propositional*) symbols from a fixed alphabet. This problem can be overcome by using a more expressive *relational* representation for sequence elements. We will therefore extend the template-based rule language traditionally used in transformation-based learning to a more flexible *relational* rule language, which can take advantage of such richer representations for sequence elements. Furthermore, it is easy in this case to incorporate domain-specific background knowledge into the learning process. Analyzing such relational sequences has received considerable attention recently, for instance with relational extensions of Hidden Markov Models [4] or n-gram models [5].

Example 2. As an example, consider the ADL (“Activities of Daily Living”) recognition problem, which is visualized in Figure 2. In ADL recognition, objects which are used in activities of daily living such as making breakfast are equipped with small RFID tags that can be picked up by a wearable reader while a person performs an activity [6].

	$tag(w_1, toastBread)$ $tag(w_2, toastBread)$ $tag(w_3, toastBread)$... $tag(w_4, flavorToast)$ $tag(w_5, flavorToast)$ $tag(w_6, flavorToast)$...																																																																		
Relational Representation	$sensor(w_1, toast)$ $sensor(w_2, toaster)$ $sensor(w_3, toast)$... $sensor(w_4, knife)$ $sensor(w_5, butter)$ $sensor(w_6, toast)$...																																																																		
Background Knowledge	$time(w_1, 1, 2)$ $time(w_2, 3, 6)$ $time(w_3, 7, 8)$... $time(w_4, 9, 11)$ $time(w_5, 12, 13)$ $time(w_6, 14, 15)$																																																																		
Activity Tag	<table style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;"></th> <th style="width: 15%;">ToastBread</th> <th style="width: 15%;">FlavorToast</th> <th style="width: 15%;">BoilWater</th> <th style="width: 15%;">FlavorTea</th> </tr> </thead> <tbody> <tr> <td>Sensor Reading</td> <td>toast</td><td>toast</td><td>toaster</td><td>toaster</td><td>toaster</td><td>toast</td><td>toast</td><td>knife</td><td>knife</td><td>knife</td><td>butter</td><td>butter</td><td>toast</td><td>toast</td><td>knife</td><td>knife</td><td>jam</td><td>jam</td><td>water</td><td>water</td><td>water</td><td>stove</td><td>stove</td><td>cup</td><td>spoon</td><td>spoon</td><td>sugar</td><td>sugar</td><td>cup</td> </tr> <tr> <td></td> <td>01</td><td>02</td><td>03</td><td>04</td><td>05</td><td>06</td><td>07</td><td>08</td><td>09</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td><td>28</td><td>29</td><td>30</td> </tr> </tbody> </table>		ToastBread	FlavorToast	BoilWater	FlavorTea	Sensor Reading	toast	toast	toaster	toaster	toaster	toast	toast	knife	knife	knife	butter	butter	toast	toast	knife	knife	jam	jam	water	water	water	stove	stove	cup	spoon	spoon	sugar	sugar	cup		01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
	ToastBread	FlavorToast	BoilWater	FlavorTea																																																															
Sensor Reading	toast	toast	toaster	toaster	toaster	toast	toast	knife	knife	knife	butter	butter	toast	toast	knife	knife	jam	jam	water	water	water	stove	stove	cup	spoon	spoon	sugar	sugar	cup																																						
	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30																																					

Fig. 2. Relational representation of the ADL recognition problem. While a person is performing activities of daily living (such as preparing breakfast), a stream of object interaction data is generated from a wearable RFID reader (“sensor reading”). This can be represented in a relational form by collapsing identical sensor readings to one sequence element w_i , and encoding the starting point and duration of the observation in another predicate. Furthermore, additional background knowledge can be used to encode prior knowledge about the domain.

The task is to recover the activity currently performed from the stream of sensor data, that is, to tag the sequence of object interactions with activities.

It is obvious that this kind of data is less rigidly structured than natural language data: there are no “grammatical rules” which determine the exact sequence of touching knife, toast, butter and jam when adding flavor to a toast. Nevertheless, context information can help determine the right tag. For instance, using a spoon can indicate activities FlavorTea or EatCereals. This ambiguity can be resolved by looking at the context: the observation of a spoon closely followed by sugar indicates activity FlavorTea, while observation of a spoon after milk and cereals indicates activity EatCereals.

Furthermore, the stream of object data obtained from the sensor has some internal structure, as an object observation has a starting point and duration in time. A representation in first-order logic allows to capture this structure, and to express flexible rule conditions such as *object x has (not) been observed less than t seconds before/after the current time-step* or *the most frequent (currently estimated) tag around the current time-step is t* using manually defined background knowledge.

At the same time, activity recognition can be seen as a *data stream mining* task—the analysis of a continuous, potentially infinite stream of data. In this context, issues such as online learning (with only one pass through the data necessary) are of considerable interest. However, we will not address these issues in the paper, and instead assume that a limited amount of training data is given a priori. Extending the proposed methods to an online-learning scenario is an interesting direction for future work.

The next section will discuss the formal learning setting for relational transformation-based tagging, before discussing learning algorithms and experimental results.

3.1 Learning Setting

The learning setting for relational transformation-based tagging can be formalized as follows:

Given

- a relational language \mathcal{W} for describing sequence elements, i.e., a set of typed first-order logical predicates
- a set of tags T ;
- a set of training sequences $S = \{s_1, \dots, s_m\}$ with sequence elements described in \mathcal{W} and corresponding true tags L over T ;
- a scheme for setting initial tags given by a function *init*;
- a language \mathcal{L} of transformation rules $t' \leftarrow t : q$ where $t, t' \in T$, $q = l_1, \dots, l_r$ and the l_i are atoms in \mathcal{W} .

Find an ordered lists of transformations $R = [R_1, \dots, R_l]$, $R_i \in \mathcal{L}$, such that applying the initial tagging scheme and afterwards transformation rules R_1, \dots, R_l minimizes

$$error(\hat{L}) = \sum_{s \in S} \sum_{i=1}^{n_s} \delta(l_{is}, \hat{l}_{is})$$

where n_s is the length of sequence s and l_{is}, \hat{l}_{is} denote the tag assigned to element i in sequence s according to L and \hat{L} .

In contrast to standard (propositional) transformation-based tagging approaches, the languages \mathcal{W} (sequence elements) and \mathcal{L} (rules) employed are relational; that is, rule conditions q are first-order queries of the form l_1, \dots, l_k where the l_i are first-order logical atoms. Applying a first-order transformation rule $t' \leftarrow t : q$ means simultaneously replacing all tags t in \hat{L} by t' wherever the first-order context constraint q matches the relational description of the corresponding sequence element.

Example 3. As an example for a relational transformation rule in the ADL recognition domain consider

$$\begin{aligned} & FlavorTea \leftarrow EatCereals : \\ & \quad sensor(X, spoon), near(X, sugar, 10), not(near(X, bowl, 5)) \end{aligned}$$

where the variable X is bound to the sequence element under consideration and the background predicate *near/3* is defined by

$$\begin{aligned} & near(X, O, T) \leftarrow \\ & \quad time(X, S, D), sensor(X', O), time(X', S', D'), dist(S, D, S', D', T'), T' \leq T \end{aligned}$$

and $dist(S, D, S', D', T)$ measures the distance between the intervals $[S, S + D]$ and $[S', S' + D']$. This rule re-tags objects of type *spoon* from *EatCereals* to *FlavorTea* if implied by the context.

3.2 A Branch-and-Bound Learning Algorithm

For learning the list R of relational transformation rules, a large space of possible rules has to be searched. However, structure on the search space can be exploited to make this search more efficient. More specifically, the algorithm we use combines ideas from transformation-based learning (branch-and-bound search based on upper bounds for the error reduction of a transformation rule) and inductive logic programming (refinement search in a generalization/specialization lattice). It is closely related to the algorithm presented in [2].

Recall that the goal of learning is to find a list R of transformation rules which minimize $error(\hat{L})$ on a set of training sequences S with known true labels L . As in propositional transformation-based learning [1], the rule list is learned greedily: starting with an empty list, the algorithm incrementally adds one rule after the other, at every step selecting the rule which yields the greatest reduction in $error(\hat{L})$ and updating the current tagging \hat{L} (cf. Algorithm 1).

When searching for an individual rule with maximum error reduction, a significant part of the search space can be pruned away by computing upper bounds for the error reduction a rule can achieve. One obvious bound for the reduction achievable by a transformation rule $t^i \leftarrow t^j : context$ is given by the number of sequence elements whose true tag (in L) is t^i and which are currently (in \hat{L}) assigned tag t^j . Let \mathcal{M} denote the current confusion matrix, i.e., $\mathcal{M}[i, j]$ denote the number of sequence elements with true tag t^i currently tagged as t^j . This can be exploited by considering rules $t^i \leftarrow t^j : context$ in (decreasing) order of their potential $\mathcal{M}[i, j]$ for error reduction and keeping track of the best error reduction Δ_{best} found so far. Now, all rules of the form $t^i \leftarrow t^j : context$ for which $\mathcal{M}[i, j] \leq \Delta_{best}$ can be removed from consideration (cf. [1]).

This idea can be taken one step further if it is combined with a general-to-specific search for the first-order constraint $context$ [2]. As a complete search in the space of first-order constraints is infeasible in most cases, we perform a greedy general-to-specific search. To generate the specializations of the current condition q , a so-called refinement operator ρ under θ -subsumption is employed. A condition q_1 θ -subsumes a condition q_2 if and only if there is a substitution θ such that $q_1\theta \subseteq q_2$. A substitution is a set $\{V_1/t_1, \dots, V_l/t_l\}$ where the V_i are different variables and the t_i are terms, and the application of the substitution replaces the variables V_1, \dots, V_l by the corresponding terms t_1, \dots, t_l . $\rho(q)$ typically returns all minimal specializations of q within \mathcal{L} . For our purposes, the refinement operator specializes a condition $q = l_1, \dots, l_n$ simply by adding a new literal l to the clause yielding $h \leftarrow l_1, \dots, l_n, l$. This operator is monotone in the sense that for $q' \in \rho(q)$ the number of matches in the data can only decrease. Consequently, the maximum gain achievable from specializations of a transformation rule $t^i \leftarrow t^j : q$ can be bounded in terms of the current matches. More specifically, assume that a constraint q matches on a number of sequence elements in the training data S , and that for p_q of these it has a positive effect (current tag is t^j , but true tag is t^i) and for n_q it has a negative effect (current and true tag are t^j). The error reduction of applying the transformation $t^i \leftarrow t^j : q$ is $\Delta_q = p_q - n_q$. It is now obvious that no specialization $t^i \leftarrow t^j : q'$ with $q' \in \rho^*(q)$ can achieve an error reduction greater than $\Gamma_q = p_q$.

Algorithm 2 Branch-and-bound algorithm for relational transformation-based tagging

```

rtb-tagging(input: sequences  $S$ ; true sequence tags  $L$ ; language bias  $\mathcal{L}$ )
1   $\hat{L} := \text{initial-tags}(S, L)$ 
2  initialize  $R := []$ 
3  repeat
4      initialize  $\Delta_{best} := 0$ 
5      compute  $\mathcal{M} := \text{confusion-matrix}(\hat{L}, L)$ 
6      for all  $i, j \in \{1, \dots, k\}, i \neq j$ , sorted by  $\mathcal{M}[i, j]$  descending do
7          initialize  $\Gamma := \mathcal{M}[i, j]$ 
8          initialize  $q := \text{true}$ 
9          while ( $\Gamma > \Delta_{best}$ ) do
10             for all  $q' \in \rho(q, \mathcal{L})$  do
11                 compute  $\Delta_{q'} := \text{error-reduction}(t^j \leftarrow t^i : q')$ 
12                 compute  $\Gamma_{q'} := \text{max-reduction}(t^j \leftarrow t^i : q')$ 
13             end for
14             let  $q := \text{argmax}_{q'} \Delta_{q'}$ 
15             let  $\Delta_{best} := \max(\Delta_{best}, \Delta_q)$ 
16             let  $\Gamma := \Gamma_q$ 
17         end while
18     end for
19     let  $r := t^i \leftarrow t^j : q$  be a rule with error reduction  $\Delta_{best}$ 
20     update  $\hat{L} := \text{apply-rule}(\hat{L}, r)$ 
21     update  $R := \text{append}(R, r)$ 
22 until (no improvement)
23 return  $R$ 

```

A greedy branch-and-bound algorithm exploiting these two bounds is outlined in Algorithm 2. It takes as input a set of training sequences S , true sequence tags L , and the language bias \mathcal{L} . The algorithm starts with an empty rule list R and initial tags assigned in \hat{L} . Transformation rules are then greedily added to R , and their effect applied to the current tagging \hat{L} (lines 3–21). Transformations are considered in order of decreasing $\mathcal{M}[i, j]$ (line 6). At every step of the search for a single transformation $t^i \leftarrow t^j : q$ (lines 6–18), the algorithm keeps track of the largest reduction Δ_{best} achieved by a rule so far. During refinements of the context constraint q (lines 9–17) a bound Γ_q for the maximum reduction that any specialization of a rule q can still achieve is computed (max-reduction), and only parts of the search space for which Γ is greater than Δ_{best} are explored.

4 Experiments

The proposed method was implemented in the RETRO (for RELational TRANSformation-based tagging) system and experimentally evaluated in two real-world domains: Activity of Daily Living recognition (**ADL**) and mobile phone profile prediction (**Phone**).

Table 1. Example relations used to describe the activity data. Some relations are directly derived from the data (e.g. *sensor*, *duration*, *close*), others include human-supplied prior knowledge (e.g. *close_used*).

Relation	Description
$sensor(Id, Object)$	The object observed at sequence element Id is $Object$
$duration(Id, T)$	The object observation at sequence element Id lasted T seconds
$close(Id, Obj, T)$	The object Obj has been observed within T seconds of sequence element Id
$time_bin(T, Bin)$	The time span T falls into the bin $Bin \in \{short, medium, long\}$
$closest_tag(Id, Act)$	The closest sequence position to Id for which an activity (i.e., a tag \neq “no activity”) is assigned in \hat{L} is tagged with Act
$close_used(Id, Act, T)$	Less than T seconds away from sequence element Id an object has been observed which is typically used in Act

	$cell(w_1, 6672) \ cell(w_2, 6671) \ cell(w_3, 6673) \ \dots$															
	$time(w_1, 1, 15) \ time(w_2, 16, 25) \ time(w_3, 26, 38) \ \dots$															
Relational Representation	$usr_activity(w_1, act) \ usr_activity(w_2, idle) \ usr_activity(w_3, act) \ \dots$															
	$active_app(w_1, 101) \ active_app(w_1, 102) \ active_app(w_3, 101) \ \dots$															
	$comm(125, sms, incoming) \ comm(390, call, outgoing) \ \dots \ \dots$															
Phone profile	<table border="1"> <thead> <tr> <th></th> <th>normal</th> <th colspan="2">silent</th> <th>normal</th> <th>meeting</th> </tr> </thead> <tbody> <tr> <td>Cell</td> <td>6672</td> <td>6671</td> <td>6673</td> <td>7409</td> <td>6671</td> <td>7409</td> <td>7410</td> <td>6739</td> </tr> </tbody> </table>		normal	silent		normal	meeting	Cell	6672	6671	6673	7409	6671	7409	7410	6739
	normal	silent		normal	meeting											
Cell	6672	6671	6673	7409	6671	7409	7410	6739								

Fig. 3. Illustration of the Phone data (predicates for cell location, duration, user activity, active applications, and communication events).

In the ADL recognition domain, object-interaction data for a user having breakfast at home has been gathered by a wearable RFID reader and RFID tags on objects such as milk, cereals, kettle, water tap, cutlery etc. (23 objects in total). The stream of tags picked up by the RFID reader indicates which object is close (approximately 10–15 centimeters) to the wrist of the user at a particular point in time. A single object observation is returned at every second—if several tags are within reach, one is returned randomly. Note that the data is relatively noisy: tags might sometimes be missed, or a tag not related to a particular activity can be reported by the reader because the corresponding object is accidentally close. The task is to predict the current activity performed, out of a set of 24 possible activities such as boiling water, toasting bread, reading a newspaper or “no activity”. The sequence data obtained from the RFID reader is represented in a relational form by collapsing identical observations into one observation with a starting point and duration in time (cf. Figure 2 for an illustration). Furthermore, additional background predicates have been defined, see Table 1 for examples.

In the Context Phone domain, data about user communication behavior has been gathered using a software running on Nokia Smartphones. The software automatically logs communication and context data, such as the current provider cell, incoming and outgoing calls and text messages, and other phone status information. The task is to

Table 2. Average F-measure on the ADL Recognition and Phone problems based on a leave-one-sequence-out cross-validation.

Algorithm	ADL	Phone
Majority tag	19.5 ± 22.3	56.7 ± 13.1
HMM Tagger	74.9 ± 12.5	56.7 ± 13.1
RETRO	75.4 ± 7.8	67.7 ± 10.3

Table 3. Examples for rules learned by RETRO on the ADL dataset.

Learned Rules
$ObtainNewspaper \leftarrow ReadNewspaper: close(Id, Obj, T), Obj = door, time_bin(T, medium)$
$FlavorTea \leftarrow EatCereals: closest_tag(A, FlavorTea)$
$SteepTeaBag \leftarrow DrinkTea: close(Id, Obj, T), Obj = stove$
$PourCereal \leftarrow ObtainNewspaper: close_used(Id, PourCereal, T), not(close_used(Id, ObtainNewspaper, T^l)), time_bin(T, short)$
$SteepTeaBag \leftarrow noActivity: duration(Id, T), time_bin(T, long), closest_tag(ID, SteepTeaBag)$

predict the active profile of the phone (silent, meeting, or normal) at every point in time. See Figure 3 for an illustration of the data and the predicates used.

For comparison, we have also conducted experiments with a (propositional) HMM tagger on the two datasets. As it is not possible to encode all relevant information propositionally, we have selected the most relevant information to be used as the propositional alphabet W . For the ADL recognition problem, this is the sequence of objects observed.

For the phone domain, it is the sequence of cells the phone was located in.

For initializing the tagging \hat{L} in the transformation-based tagger, RETRO simply assigns the most frequent tag given the propositional symbol $w \in W$:

$$\text{init}(w) = \underset{t \in T}{\operatorname{argmax}} C(w, t)$$

where $C(w, t)$ is the number of times symbol w was tagged with t in the training data. More elaborate initialization schemes (such as using the HMM tagging as an initialization for the transformation-based tagger) are an interesting direction for future work. Furthermore, instead of a simple greedy search as outlined in Algorithm 2, a beam search with beam size $K = 10$ is used. The main loop of the algorithm is terminated if no rule with a gain of at least $min_gain = 10$ is found.

Table 2 lists the average F-measure for RETRO and HMM tagging based on a leave-one-sequence-out cross-validation. For the ADL recognition problem, there is no significant difference between the two approaches. In the phone domain, the HMM tagger fails to improve upon the majority tag prediction, while RETRO yields a (borderline) significant increase in F-measure (paired sampled t-test, $p = 0.051$). This shows that transformation-based approaches can be competitive with probabilistic methods in complex tagging domains. However, the presented experiments are still preliminary, and more empirical evaluation is needed to assess the potential of the method in more

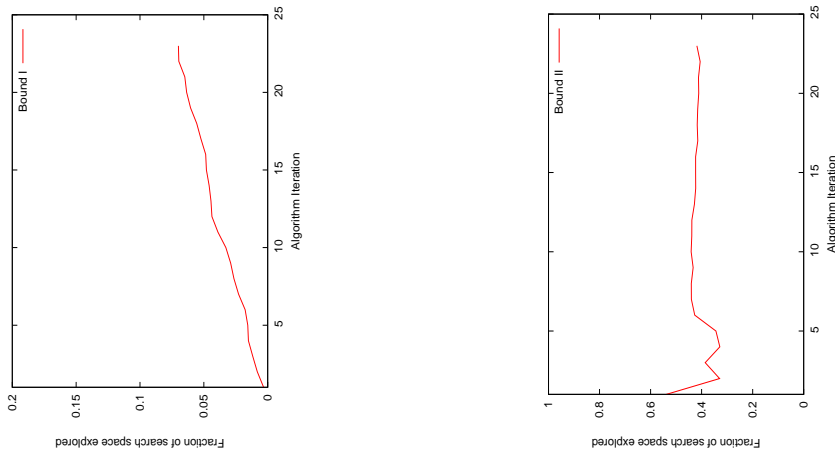


Fig. 4. Effectiveness of the two pruning schemes Bound I (maximum gain attainable from changing a certain tag into a certain other tag) and Bound II (maximum gain attainable from specializing a given rule). Results are averaged over a leave-one-sequence-out cross-validation.

detail. Note furthermore that although HMM tagging is a standard approach in activity recognition, more advanced probabilistic methods have recently been developed which would possibly yield slightly higher accuracy in this domain [9].

Examples for rules learned by RETRO on the ADL recognition task are shown in Table 3. For instance, consider the last rule: it encodes that if a sequence element corresponding to a long object observation is tagged with *noActivity* and the closest currently predicted activity is *SteepTeaBag*, this sequence element should also be tagged with *SteepTeaBag*. This rule is useful for “filling in gaps” as *SteepTeaBag* only causes characteristic object observations at the beginning and end of the activity.

Finally, Figure 4 visualizes the effectiveness of the pruning schemes based on the two upper bounds discussed above on the ADL recognition problem. More specifically, Figure 4 (left) shows the fraction of pairs (t^i, t^j) that have to be considered when searching for rules $t^i \leftarrow t^j$ in lines 6–18 of Algorithm 2 as a function of the algorithm iteration. This pruning scheme is very effective, reducing the search space by 93%–99%. It is more effective in earlier iterations as it is easier to find a rule with yields a large reduction in error. Figure 4 (right) shows which fraction of refinements is removed from the beam when rules are refined in lines 10–13 of Algorithm 2 because no further specialization can reach the performance of the best rule found so far. Note that this form of pruning does not affect the computational complexity of the algorithm but rather allows a more thorough search through the space of possible rules (given a limited beam size) by effectively reducing the branching factor of the search. On average, the branching factor is about halved, this is independent of the algorithm iteration.

5 Conclusions and Related Work

Motivated by the needs of activity recognition problems, we have introduced a relational transformation-based tagging system. It tightly integrates principles of inductive logic programming (especially search, representations, operators, background knowledge) with transformation-based tagging (error-driven search, branch-and-bound idea). The approach has been evaluated on two activity recognition data sets and the results are competitive with those of a Hidden Markov Model approach. Perhaps more important than the experimental results obtained so far is the ease with which one can extend the transformation-based tagging approach beyond the propositional HMM setting. Important directions in this regard include: the use of rich sources of background knowledge (that take not only into account the inputs but also the already available produced tags), the prediction of structured output sequences (predicting sequences of logical atoms, cf. [3], such as *call(anna,10)* denoting the prediction that *anna* will be called in 10 minutes), and relaxing the purely sequential nature of the output (which is important for the ADL dataset where different activities may overlap in time, and therefore ordering them is not always possible).

Acknowledgments We would like to acknowledge support for this work from the Research Foundation-Flanders (FWO-Vlaanderen).

References

1. E. Brill. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565, 1995.
2. L. Dehaspe and M. Forrier. Transformation-based learning meets frequent pattern discovery. In J. Cussens, editor, *Proceedings of the 1st Workshop on Learning Language in Logic*, pages 40–51, Bled, Slovenia, 1999.
3. K. Kersting, L. De Raedt, B. Gutmann, A. Karwath, and N. Landwehr. Relational sequence learning. In L. De Raedt, P. Frasconi, K. Kersting, and S. Muggleton, editors, *Application of Probabilistic ILP*. Springer, 2007. to appear.
4. K. Kersting, L. De Raedt, and T. Raiko. Logical hidden markov models. *Journal of Artificial Intelligence Research*, 25:425–456, 2006.
5. N. Landwehr and L. De Raedt. r-grams: Relational grams. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 907–912, Hyderabad, India, 2007.
6. D. Patterson, D. Fox, H. Kautz, and M. Philipose. Fine-grained activity recognition by aggregating abstract object usage. In *Proceedings of ISWC 2005*, Osaka, 2005.
7. L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
8. M. Raento, A. Oulasvirta, R. Petit, and H. Toivonen. ContextPhone - a Prototyping Platform for Context-aware Mobile Applications. *IEEE Pervasive Computing*, 4(2):51–59, 2006.
9. S. Wang, W. Pentney, A.-M. Popescu, T. Choudhury, and M. Philipose. Common sense based joint training of human activity recognizers. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2237–2242, 2007.
10. D. Wilson and M. Philipose. Maximum a posteriori path estimation with input trace perturbation: Algorithms and application to credible rating of human routines. In *Proceedings of IJCAI 2005*, Edinburgh, Scotland, August 2005.