Modeling and Reasoning with ProbLog: An Application in Recognizing Complex Activities

Timo Sztyler, Gabriele Civitarese, Heiner Stuckenschmidt

CoMoRea´18

19.03.2018
Content

1. Motivation
2. Model & System
3. Preliminary Results
4. Discussion
5. Ongoing Work
Scenario

We focus on recognizing complex activities like “watering plants” or “taking medicine”.

- So far we relied on...

  ... ontological and probabilistic reasoning
  (as it overcomes well-known limitations)

- What we are targeting right now ...

  ... online recognition and active learning
Scenario

Activities timeline

Preparing dinner
Eating
Taking medicines

Sensor events timeline

e_1, e_2, ..., e_k

ai_1, ai_2, ai_3

time
Motivation

Until now we used Markov Logic Networks (RockIt) to model our scenario, but ...

- ... weights are not this intuitive as probabilities
- ... much computational effort
- ... did not support time-constraints and marginal inferences (RockIt).

We decided to investigate ProbLog in respect of Activity Recognition
**Idea**

*ProbLog is a probabilistic extension of Prolog which allows to explicitly define probabilistic facts and rules.*

- Segment sensor stream into windows

  Build for each window a ProbLog program

  - Allows to query the user in almost real-time

  - “Marginal Inferences” allows to apply rules but also interpret the results across windows
Contribution

- We clarify the benefits of \textit{ProbLog} for activity recognition

- We introduce a guidance on how to write \textit{ProbLog} programs.

- We present a \textit{ProbLog}-based method to recognize activities of daily living in an online fashion

- We show potential advantages of ProbLog’s marginal inference
MODEL & SYSTEM
System

1. Extracting probabilities (sensor type ↔ activity)

extracted from the dataset (supervised)

2. Segmentation

Standard windowing approach with a fixed size

3. ProbLog Program Generation

As soon as a window is finalized we generate the ProbLog program

4. Online Activity Recognition

We choose the most likely activity
Building a ProbLog Program (1/4)

Event and Instance Clauses (Example)

1.0::event(e₁, water, 5).
1.0::event(e₂, absent, 6).

This implies that these two clauses have to be part of each possible world (ground truth).

0.5::instance(ai₁, ac₁, 0, 7); 0.5::instance(ai₁, ac₂, 0, 7).
0.5::instance(ai₂, ac₁, 4, 10); 0.5::instance(ai₂, ac₂, 4, 10).

e.g. cleaning XOR predicate
Building a ProbLog Program (1/4)

Probabilistic Facts (Example)

\[
0.9::\text{related}(X, \text{watering}) :- \text{event}(X, \text{water}, T)
\]

allows to incorporate mined probabilities

Temporal Constraints (Example)

\[
\text{closeAfter}(T_1,T_2) :- T_1>T_2, T_3 \text{ is } T_1-T_2, T_3<3.
\]

\[
0.6::\text{producedBy}(X_2,I) :- \text{event}(X_1,Y_1,T1), \text{event}(X_2,Y_2,T_2), \text{closeAfter}(T_2,T_1), \text{producedBy}(X_1,I)
\]

Intuitively, temporally close events more likely belong to the same activity instance.
Building a ProbLog Program (1/4)

Knowledge-based Facts (Example)

```
0.9:: bond(Y, waterplants) :- event(X1, water, T1), event(X2, can, T2), closeAfter(T1, T2), producedBy(X1, Y).
```

We assume that “can” is only used for “waterplants”

Domain Constraints (Example)

```
r1 :- related(e_1, ac_1), +related(e_1, ac_2);
      related(e_2, ac_1), +related(e_2, ac_2);
      evidence(r1, true).
```

We want to be sure that each sensor event is assigned to exactly one activity.
An instance of our program

\[ \text{instance}(ai_{1}, ac_{1}, 0, 7) \]

\[ \text{producedBy}(e_{1}, ac_{1}) \]

\[ \text{related}(e_{1}, \text{cleaning}) \]

Probability of a certain program instance

We have to formulate a query

Each of them has a probability

The probabilities of all instances answer our query

Timo Sztyler, Gabriele Civitarese, Heiner Stuckenschmidt
PRELIMINARY RESULTS
Dataset: CASAS

CASAS: A smart home in a box, D.J. Cook et al., 2013

Timo Sztyler, Gabriele Civitarese, Heiner Stuckenschmidt
The activities are well recognized but there is a gap between the worst and best result (ac1 vs. ac5)
Results: Marginal Inference

This militates for the quality of the ranking but also for the reliability of the approach.
Runtime

Cumulative probability

Computation time of a segment (seconds)
DISCUSSION
Discussion

The results look promising but ...

- ... difficult to compare to existing works due to different setups
- ... only a standard windowing approach
- ... probabilities were extracted from the dataset
- ... Interleaved activities are not explicitly modeled
ONGOING WORK
Ongoing Work

*Our next steps are ...*

- ... fully exploit marginal inferences
- Refine the classification of the windows
- ... investigating active learning for personalization
  (e.g. correcting/adapting probabilities)
- Currently, in another work, we focus on suitable online and active learning techniques
Thank you for your attention :)