NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCE
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATION

MSc THESIS

A study on the Probabilistic Interval-based Event Calculus

Christos G. Vlassopoulos

Supervisors: **Dr. Alexander Artikis,** Assistant Professor (University of Piraeus), Senior Researcher (NCSR “Demokritos”)
Dr. George Paliouras, Research Director (NCSR “Demokritos”)
Dr. Panagiotis Stamatopoulos, Assistant Professor (National and Kapodistrian University of Athens)

ATHENS

February 2020

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Μελέτη του Πιθανοτικού Λογισμού Γεγονότων βάσει Χρονικών Διαστημάτων

Χρήστος Γ. Βλασόπουλος

Επιβλέποντες: **Δρ. Αλέξανδρος Αρτίκης,** Επίκουρος Καθηγητής (Πανεπιστήμιο Πειραιώς), Κύριος Ερευνητής (ΕΚΕΦΕ «Δημόκριτος»)
Δρ. Γεώργιος Παλιούρας, Διευθυντής Ερευνών (ΕΚΕΦΕ «Δημόκριτος»)
Δρ. Παναγιώτης Σταματόπουλος, Επίκουρος Καθηγητής (Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών)

ΑΘΗΝΑ

Φεβρουάριος 2020

MSc THESIS

A study on the Probabilistic
Interval-based Event Calculus

Christos G. Vlassopoulos
S.N.: M1479

Supervisors: **Dr. Alexander Artikis,** Assistant Professor (University of Piraeus), Senior Researcher (NCSR “Demokritos”)
Dr. George Paliouras, Research Director (NCSR “Demokritos”)
Dr. Panagiotis Stamatopoulos, Assistant Professor (National and Kapodistrian University of Athens)

Examination Committee: **Dr. Panagiotis Stamatopoulos,** Assistant Professor (National and Kapodistrian University of Athens)
Dr. Panagiotis Rondogiannis, Professor (National and Kapodistrian University of Athens)

February 2020

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Μελέτη του Πιθανοτικού Λογισμού Γεγονότων
βάσει Χρονικών Διαστημάτων

Χρήστος Γ. Βλασόπουλος
Α.Μ.: Μ1479

Επιβλέποντες: **Δρ. Αλέξανδρος Αρτίκης,** Επίκουρος Καθηγητής (Πανεπιστήμιο Πειραιώς), Κύριος Ερευνητής (ΕΚΕΦΕ «Δημόκριτος»)
Δρ. Γεώργιος Παλιούρας, Διευθυντής Ερευνών (ΕΚΕΦΕ «Δημόκριτος»)
Δρ. Παναγιώτης Σταματόπουλος, Επίκουρος Καθηγητής (Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών)

Εξεταστική Επιτροπή: **Δρ. Παναγιώτης Σταματόπουλος,** Επίκουρος Καθηγητής (Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών)
Δρ. Παναγιώτης Ροντογιάννης, Καθηγητής (Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών)

Φεβρουάριος 2020

ABSTRACT

Complex Event Recognition is the subdivision of Artificial Intelligence that aims to design and construct systems that quickly process large and often heterogeneous streams of data and timely deduce – based on definitions set by domain experts – the occurrence of non-trivial and interesting incidents. The purpose of such systems is to provide useful insights into involved and demanding situations that would otherwise be difficult to monitor, and to assist decision making. Uncertainty and noise are inherent in such data streams and therefore, Probability Theory becomes necessary in order to deal with them. The probabilistic recognition of Complex Events can be done in a timepoint-based or an interval-based manner.

This thesis focuses on PIEC, a state-of-the-art probabilistic, interval-based Complex Event Recognition algorithm. We present the algorithm and examine it in detail. We study its correctness through a series of mathematical proofs of its soundness and completeness. Afterwards, we provide thorough experimental evaluation and comparison to point-based probabilistic Event Recognition methods. Our evaluation shows that PIEC consistently displays better Recall measures, often at the expense of a generally worse Precision. We then focus on cases where PIEC performs significantly better and cases where it falls short, in an effort to detect and state its main strengths and weaknesses. We also set the general directions for further research on the topic, parts of which are already in progress.

SUBJECT AREA: Complex Event Recognition

KEYWORDS: Artificial Intelligence, Probabilistic Inference, Knowledge Representation and Reasoning, Probabilistic Logic Programming, Algorithms

ΠΕΡΙΛΗΨΗ

Η Αναγνώριση Σύνθετων Γεγονότων είναι το πεδίο εκείνο της Τεχνητής Νοημοσύνης το οποίο αποσκοπεί στο σχεδιασμό και την κατασκευή συστημάτων τα οποία επεξεργάζονται γρήγορα μεγάλες και πιθανώς ετερογενείς ροές δεδομένων και τα οποία είναι σε θέση να αναγνωρίζουν εγκαίρως μη τετριμμένα και ενδιαφέροντα συμβάντα, βάσει κατάλληλων ορισμών που προέρχονται από ειδικούς. Σκοπός ενός τέτοιου συστήματος είναι η αυτοματοποιημένη εποπτεία πολύπλοκων και απαιτητικών καταστάσεων και η υποβοήθηση της λήψης αποφάσεων από τον άνθρωπο. Η αβεβαιότητα και ο θόρυβος είναι έννοιες που υπεισέρχονται φυσικά σε τέτοιες ροές δεδομένων και συνεπώς, καθίσταται απαραίτητη η χρήση της Θεωρίας Πιθανοτήτων για την αντιμετώπισή τους. Η πιθανοτική Αναγνώριση Σύνθετων Γεγονότων μπορεί να πραγματοποιηθεί σε επίπεδο χρονικής στιγμής ή σε επίπεδο χρονικού διαστήματος.

Η παρούσα εργασία εστιάζει στον PIEC, έναν σύγχρονο αλγόριθμο για την Αναγνώριση Σύνθετων Γεγονότων με τη χρήση πιθανοτικών, μέγιστων διαστημάτων. Αρχικά παρουσιάζουμε τον αλγόριθμο και τον ερευνούμε ενδελεχώς. Μελετούμε την ορθότητά του μέσα από μια σειρά μαθηματικών αποδείξεων περί της ευρωστίας (soundness) και της πληρότητάς του (completeness). Κατόπιν, παραθέτουμε εκτενή πειραματική αποτίμηση του υπό μελέτη αλγορίθμου και σύγκρισή του με συστήματα πιθανοτικής Αναγνώρισης Γεγονότων σε επίπεδο χρονικών σημείων. Τα αποτελέσματά μας δείχνουν ότι ο PIEC επιδεικνύει σταθερά καλύτερη Ανάκληση (Recall), παρουσιάζοντας, ωστόσο κάποιες απώλειες σε Ακρίβεια (Precision) σε ορισμένες περιπτώσεις. Για τον λόγο αυτόν, εμβαθύνουμε και εξετάζουμε συγκεκριμένες περιπτώσεις στις οποίες ο PIEC αποδίδει καλύτερα, καθώς και άλλες στις οποίες παράγει αποτελέσματα υποδεέστερα των παραδοσιακών μεθόδων σημειακής αναγνώρισης, σε μια προσπάθεια να εντοπίσουμε και να διατυπώσουμε τις δυνατότητες αλλά και τις αδυναμίες του αλγορίθμου. Τέλος, θέτουμε τις γενικές κατευθυντήριες γραμμές για περαιτέρω έρευνα στο εν λόγω ζήτημα, τμήματα της οποίας βρίσκονται ήδη σε εξέλιξη.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Αναγνώριση Σύνθετων Γεγονότων

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Τεχνητή Νοημοσύνη, Πιθανοτικός Συμπερασμός, Αναπαράσταση Γνώσης και Συλλογιστική, Πιθανοτικός Λογικός Προγραμματισμός, Αλγόριθμοι

To the memory of my father

George C. Vlassopoulos (13 May 1964 – 18 June 2016)

AKNOWLEDGMENTS

First of all, I would like to thank my supervisors at the National Centre for Scientific Research “Demokritos” Georgios Paliouras and Alexander Artikis for their constant guidance and support, thoughout our collaboration. Their patience, expertise and methodical thinking have taught me a lot, have helped me organize my work and overcome many obstacles. I would also like to thank my colleagues Nikos Katzouris, Elias Alevizos, Vaggelis Michelioudakis and Vaggelis Makris for their useful feedback, the fruitful discussions we have had and their extensive knowledge, which they profusely offered, whenever I needed help. I am also grateful to my professor Panagiotis Stamatopoulos, who trusted me and happily agreed to supervise my thesis.

I am especially grateful to the Makridakis family, who most kindly and generously offered a lot to me, including accommodation for my postgraduate studies, thus tremendously assisting my efforts. Special thanks to my dear Katerina Apostolopoulou, as well as my friends Leonidas Tsekouras, Manolis Pitsikalis, Antonia Tsili, Giorgos Kalampokis, Alex Troupiotis, Maria Georgara and Maria-Eleni Varela for being there for me, letting me share parts of this struggle with them and keeping me sane throughout. My therapist, Vassilis Manatos, also did a great job helping me realize my abilities and potential, and keep pushing even at times when the weight of my studies felt too heavy on my shoulders.

Last but by no means least, I want to thank my mother, Marina Gkioka, who always gives her everything for me and is always willing to advise me and encourage me to move forward, and my father, George Vlassopoulos, who inspired me to love learning, taught me to fight for what is worth in life and who would have been very proud to be holding this thesis in his hands and reading it. I devote this work to his memory.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

PREFACE

This thesis is a product of my collaboration with the National Centre for Scientific Research “Demokritos” and the Institute of Informatics and Telecommunications, in particular. This work has been funded by the “SYNTELESIS” project (ΣΥΝΤΕΛΕΣΙΣ: Καινοτόμες Τεχνολογίες και Εφαρμογές βασισμένες σε Διαδίκτυο των Πραγμάτων (IoT) και Υπολογιστικό Νέφος (Cloud)» (Ε-12181)).

1. INTRODUCTION

1.1 Motivation

As technology advances, more and more procedures and tasks that would traditionally be carried out using human labor tend to become automated using proper computational power and appropriate software. As a consequence, the amount, the formats, and the domain range of the data produced and stored keep increasing. Therefore, the efficient processing and knowledge extraction from these data becomes a necessity in an increasing number of different tasks. We need to be able to automatically detect and portray the occurrence of activities of interest in a wide range of domains, from city traffic monitoring to household activity supervision and from public space surveillance to fraud detection in credit card transactions.

Complex Event Recognition is an area of Artificial Intelligence that provides tools and algorithms that meet the need described above. More specifically, it focuses on detecting the occurrence of composite, non-trivial activities from data. These activities may vary depending on the context. For instance, we may be interested in detecting when two or more people are gathering together, or when two vessels are displaying suspicious activity. In order to recognize these activities, we need a set of rules that define them as temporal combinations of time-stamped data. These rules can be written by hand by domain experts or automatically learnt, using machine learning algorithms. The necessary data is collected from several heterogeneous, noisy and possibly interdependent sources. Examples of such sources are sensors, cameras, and the Web. The data may be subject to preprocessing before being given as input to a Complex Event Recognition mechanism.

Having to do with many and heterogeneous sources of data increases the risk of mutations or even loss of data. Sensors may fail, various hardware issues may impose delays to the transmission of the needed data. These are some of the factors that introduce noise in the event recognition procedure. In order to deal with noise and uncertainty in the data and the activity definitions, several probabilistic event activity recognition systems have been proposed (see [1], [2], [3]). These systems compute the probability that an activity of interest takes place at a specific timepoint. The work of Artikis, Makris and Paliouras [4] extends this approach to calculating probabilistic maximal intervals during which such an activity takes place, as well as a credibility rate for each one of them. Makris et al. incorporate the notions of probabilistic maximal interval and the credibility thereof into an algorithm called PIEC (Probabilistic Interval-based Event Calculus).

1.2 Contribution

The main contribution of this thesis is the reimplementation, correctness analysis and evaluation of the PIEC algorithm. More specifically:

- We implement the PIEC algorithm, from scratch, using the Scala programming language.
- We perform the correctness analysis of the PIEC algorithm. Specifically, we provide mathematical proofs that the algorithm is correct. To that direction, explicit proofs on PIEC’s soundness and completeness are given.

- We assess the performance of the proposed algorithm, using a benchmark dataset for human activity recognition. Special care has been taken for the experimental evaluation to be as exhaustive as possible. In particular, we compare our approach against three state-of-the-art probabilistic activity recognition methods. Our comparison is done in two ways: In an overall manner, where we run the involved algorithms over the entire testing dataset and compare the cumulative results, and in a case-by-case manner, where we isolate certain interesting scenarios from within the dataset and observe the behavior of the algorithms in these specific cases. Detailed discussion accompanies the evaluation results.
- We propose improvements to the PIEC algorithm, by providing alternative definitions for the credibility of a probabilistic maximal interval. We assess the new, proposed credibility definitions in a way similar to the assessment of the original version of the algorithm and we show that these alternative credibility definitions can lead to improved recognition results.
- The entire code developed during this M.Sc. project (i.e. the Scala implementation of PIEC, along with the code and data used in our experiments), as well as the respective documentation, are publicly available on GitHub[1].

1.3 Outline

The remainder of this thesis is organized as follows: First, we provide the theoretical foundations, the theories, methods and definitions upon which our research has been based and whose understanding is imperative for the study of our contribution. Subsequently, we give a brief overview of the latest literature and the current scientific advances that are related to our research. Afterwards, we proceed to our main contribution. We make an in-depth presentation and analysis of our methodology and algorithms, including mathematical proofs for the correctness of our method and thorough discussion. Next, we display a comprehensive assessment of the algorithms presented and examined, through extensive experimentation. We depict the outcome of our experiments using an abundance of diagrams, accompanied by detailed discussion. Finally, we conclude and set the general directions for future work.

[1] https://github.com/cvlas/Scala-PIEC

2. BACKGROUND

In the previous section, we introduced the concept of Complex Event Recognition, either with or without uncertainty. In this section, we discuss fundamental state-of-the-art techniques and algorithms that are used in Complex Event Recognition and upon which the contribution of this thesis is based.

2.1 The Event Calculus

The Event Calculus [5] is a Logic Programming formalism that allows for the efficient representation and reasoning about events and their effects ([6], [7]). It comprises *events*, *fluents*, a *time model*, and a set of *core axioms*. An *event* can be instantaneous or durative. A *fluent* F is a property and it is always accompanied by a value V that may change over time. The occurrence of an event can initiate or terminate a fluent. There are two ways of terminating a fluent: Either explicitly, using a termination statement for $F = V$ or implicitly, initiating F with a new value, for instance $F = V'$, with $V \neq V'$. The notion of time is crucial for the happening of events and the changing of fluents. In most cases, we use a linear *time model* with time points expressed as real or integer numbers. Finally, there is a set of *core*, *domain-independent* Event Calculus *axioms* that govern the holding of a fluent, based on the common sense law of inertia, i.e. a fluent $F = V$ that has been initiated at some point, continuously holds until it is terminated, either explicitly or implicitly.

We can use the Event Calculus to express involved dependencies between events and fluents. Fluents play the role of Long-term Activities (LTA), whereas events will be considered Short-term Activities (STA). LTA definitions along with the domain-independent core axioms of the Event Calculus can be expressed as first-order logic formulae.

In [1], a dialect of the Event Calculus is defined, called *Crisp-EC*. The main predicates of this dialect are shown in Table 1. Given this set of predicates, one can construct both the domain-independent Event Calculus axioms and the domain-specific LTA initiation and termination statements, as follows:

$$\begin{aligned} &\text{holdsAt}(F = V,\ T) \leftarrow \\ &\quad \text{initially}(F = V), \\ &\quad \text{not broken}(F = V,\ 0,\ T). \end{aligned} \tag{1}$$

$$\begin{aligned} &\text{holdsAt}(F = V,\ T) \leftarrow \\ &\quad \text{initiatedAt}(F = V,\ T_s), \\ &\quad T_s < T, \\ &\quad \text{not broken}(F = V,\ T_s,\ T). \end{aligned} \tag{2}$$

$$\begin{aligned} &\text{broken}(F = V,\ T_s,\ T) \leftarrow \\ &\quad \text{terminatedAt}(F = V,\ T_e), \\ &\quad T_s < T_e < T. \end{aligned} \tag{3}$$

$$\begin{aligned}&\text{broken}(F = V_1, T_s, T) \leftarrow \\ &\quad \text{initiatedAt}(F = V_2, T_e), \\ &\quad V_1 \neq V_2, \\ &\quad T_s < T_e < T.\end{aligned} \tag{4}$$

Axioms (1) and (2) dictate that a fluent holds at a time point T if it has been initiated at some point earlier and it has not been "broken" since. Axioms (3) and (4) describe what "broken" means. A fluent can be broken by explicit termination or implicitly, by initiating the same fluent with another value in the meantime. Here, "not" implements negation as failure. All axioms above implement the law of inertia described earlier.

Apart from the domain-independent axioms that govern the holding of a fluent, Crisp-EC also offers domain-specific axioms that describe the initiation and termination conditions for an LTA. An initiation statement looks like the following:

$$\begin{aligned}&\text{initiatedAt}(F = V, T) \leftarrow \\ &\quad \text{happens}(E, T), \\ &\quad \textit{conditions}[T].\end{aligned} \tag{5}$$

This statement declares that if event E happens at time point T and a (possibly empty) group of other temporal conditions is satisfied, then fluent $F = V$ is initiated at time T. Note that this does not, by any means, imply that $F \neq V$ at time point T. Similarly, a termination statement does not necessarily mean that $F = V$ at the time of termination.

Table 1: Main predicates of Crisp-EC

Predicate	Meaning
happens(*E, T*)	Event *E* occurs at time *T*
initially(*F* = *V*)	The value of fluent *F* is *V* at time *0*
holdsAt(*F* = *V, T*)	The value of fluent *F* is *V* at time *T*
initiatedAt(*F* = *V, T*)	At time *T* a period of time for which *F* = *V* is initiated
terminatedAt(*F* = *V, T*)	At time *T* a period of time for which *F* = *V* is terminated

2.2 Probabilistic Event Calculus

The Event Calculus can express complex event dependencies, where events correspond to STA and fluents correspond to LTA. Domain-independent axioms, as well as LTA definitions can be expressed as first-order logic formulae. Despite its high expressive power, the Event Calculus cannot efficiently handle noise, either in the input or in the LTA definitions. Noise and uncertainty are inherent in data coming from sensors and can naturally occur in many settings. To this direction, Skarlatidis et al. [1] have extended their Crisp-EC dialect, added probabilities to it and created a Probabilistic Event Calculus dialect called *Prob-EC*. This dialect has been built using the ProbLog Probabilistic Logic Programming language [8]. ProbLog is a probabilistic extension to Prolog, it allows for probabilistic facts of the form $p_i :: f_i$, where p_i is a real number between 0 and 1, and f_i is a (not necessarily ground) Prolog fact.

ProbLog's probabilistic facts are treated as independent random variables. This means that the probability of a clause that contains a lot of probabilistic facts in its body will be the product of the probabilities of these facts. Moreover, if an atom appears in the head of more than one rules, then its probability will be that of the disjunction of the respective rules. As described in [1] and [8], if we address a query q to a ProbLog program, its success probability is:

$$P(q) = P\left(\bigvee_{e \in proofs(q)} \bigwedge_{f_i \in e} f_i \right) \tag{6}$$

ProbLog has an efficient way of computing this probability, using Binary Decision Diagrams (BDD).

Crisp-EC axioms remain valid for the Prob-EC case, albeit with subtle modifications that have to do with ProbLog's negation [1].

Figure 1 shows an example of how Prob-EC performs its reasoning. Suppose that there are two people, Mike and Sarah and start walking together, at time 1. One frame later, Sarah stops walking and becomes "active" (recall the respective STA mentioned earlier in this chapter). Sarah remains active until time point 21, where she starts walking with Mike again. Then, at time 41 Sarah starts moving away from Mike.

Moving is defined as follows:

initiatedAt(*moving*(P_1, P_2) = true, *T*) ←
happens(*walking*(P_1), *T*),
happens(*walking*(P_2), *T*), (7)
holdsAt(*close*(P_1, P_2, 34) = true, *T*),
holdsAt (*similarOrientation*(P_1, P_2, 45) = true, *T*).

terminatedAt(*moving*(P_1, P_2) = true, *T*) ←
happens(*walking*(P_1), *T*), (8)
holdsAt(*close*(P_1, P_2, 34) = false, *T*).

Consider an input STA stream that looks like this:

0.70 :: happensAt(*walking*(mike), 1)

0.73 :: happensAt(*walking*(mike), 2)

…

0.69 :: happensAt(*walking*(mike), 21)

…

0.18 :: happensAt(*inactive*(mike), 41)

0.46 :: happensAt(*walking*(sarah), 1)

0.55 :: happensAt(*active*(sarah), 2)

…

0.58 :: happensAt(*walking*(sarah), 21)

…

0.32 :: happensAt(*walking*(sarah), 41)

Supposing that orientation and coordinates are computed with probability 1, we can see that at frame number 2 moving has already been initiated once. This means that rule (7) has been triggered and its probability is the product:

$$\begin{aligned} P(\text{holdsAt}(\mathit{moving}(\text{mike}, \text{sarah}) = \text{true}, 2)) &= \\ \text{P}(\text{initiatedAt}(\mathit{moving}(\text{mike}, \text{sarah}) = \text{true}, 1)) &= \\ \text{P}(\text{happens}(\mathit{walking}(\text{mike}), 1)) \times \text{P}(\text{happens}(\mathit{walking}(\text{sarah}), 1)) &= \\ 0.70 \times 0.46 &= 0.322 \end{aligned}$$

and thus, probability rises from 0 to 0.322. Then, rule (7) stops being triggered and no more moving initiations occur until frame 21. At the same time, there are no terminations for the moving LTA as rule (8) is not triggered, either. Thus, the probability remains as is, due to the law of inertia. At frame 21, Sarah appears to be walking with Mike again. They are still close to each other, therefore rule (7) will be fired again. This causes the probability of moving to further rise. Specifically, at the next frame, Prob-EC tries to calculate the probability of $\text{holdsAt}(\mathit{moving}(\text{mike}, \text{sarah}) = \text{true}, 22)$. Rule (2) forces Prob-EC to scan all previous timepoints in search for initiation conditions. In this particular case, Prob-EC finds two initiations. One at time point 1 – for brevity, let us name it “$init_1$” – and one at time point 21 – for brevity, let us name it “$init_{21}$”. Subsequently, Prob-EC calculates the probability of their disjunction.

$$\begin{aligned} P(init_1 \cup init_{21}) &= \text{P}(init_1) + \text{P}(init_{21}) - \text{P}(init_1 \cap init_{21}) \\ &= 0.70 \times 0.46 + 0.69 \times 0.58 - 0.70 \times 0.46 \times 0.69 \times 0.58 \\ &= 0.593 \end{aligned}$$

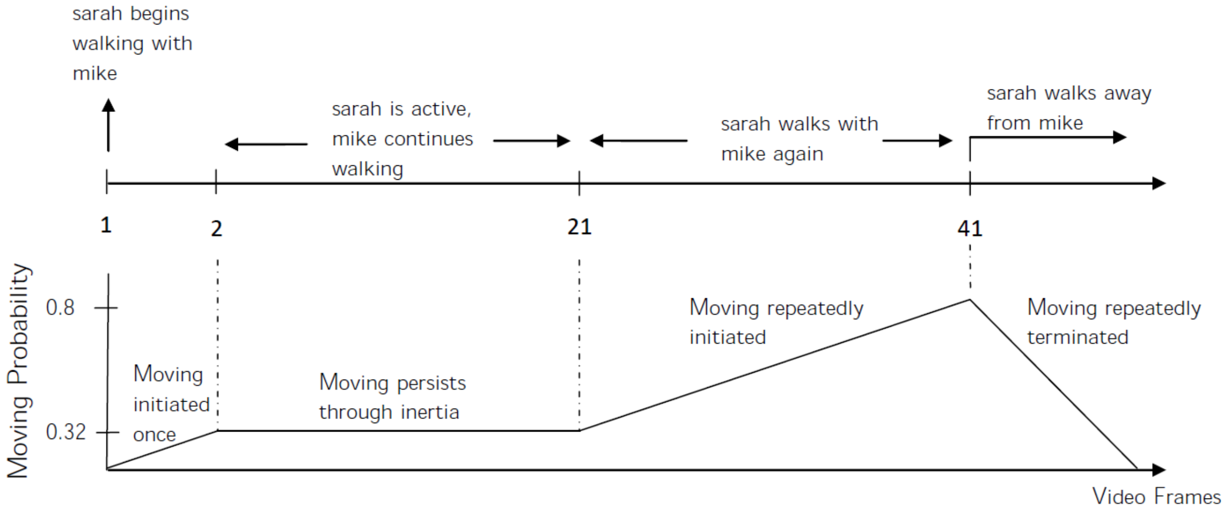

Figure 1: Example illustrating how repeated initiations and terminations of an LTA affect its probability calculated by Prob-EC.

We observe that the probability has further risen. As the initiation conditions persist, more initiations will be added to the disjunction and, therefore the probability that Mike and Sarah perform the moving LTA will increase. This behavior is compatible with our intuition. The more we receive indications that something has happened the more prompted we feel to accept that it has indeed happened.

At frame 41, the probability of "moving" between Mike and Sarah reaches 0.8. This is the point where Mike starts displaying inactive body movement and Sarah is walking away from him. As the distance between them exceeds the threshold imposed by rule (8), a termination for the moving LTA is triggered. Supposing that the distance between the actors involves no uncertainty at all – that is, the probability of $\mathrm{holdsAt}(close(\mathrm{mike}, \mathrm{sarah}, 34) = \mathrm{true}, 22)$ is 1 – the probability of this termination is

$$\mathrm{P}(\mathrm{happensAt}(walking(\mathrm{sarah}), 41)) = 0.32$$

Prob-EC calculates the probability of "moving" by taking into account all possible worlds in which Sarah did not walk away from Mike. The probability of these worlds is $1 - 0.32 = 0.68$. From rule (2), we have:

$$\mathrm{P}(\mathrm{holdsAt}(moving(\mathrm{mike}, \mathrm{sarah}) = \mathrm{true}, 42)) = 0.8 \times 0.68 = 0.544$$

Similarly to the multiple initiation case, we can have multiple terminations for an LTA. In this case the probability steadily drops approaching zero. Until the end of the video, Sarah keeps walking away from Mike and eventually leaves the scene. Therefore, the probability of "moving" drops to 0 and remains there until the end of the video. The probability fluctuations, as the result of the LTA initiations and terminations are evident in Figure 1.

2.3 Markov Logic Networks

Markov Logic Networks (MLNs) [9] is a state-of-the-art Statistical Relational Learning (SRL) framework that combines first-order logic representation with Markov Network modelling (see [10], [3]). Just as ProbLog does, Markov Logic Networks serve the need of supplementing the expressive powers of the Event Calculus with the ability to adequately handle uncertainty in the data or the LTA patterns.

MLNs perform probabilistic inference by softening the constraints imposed by the formulae of a knowledge base to the set of possible worlds, i.e. Herbrand interpretations. Every formula F_i is represented as a first-order formula and is associated with a weight value $w_i \in \mathbb{R}$. The higher the value of the weight w_i, the stronger the constraint represented by formula F_i. Contrary to classical logic, in MLN all worlds are possible with a certain probability. The probability of a world increases as the amount of formulae it violates decreases.

An MLN knowledge base may contain both hard and soft-constrained formulae. Hard constraints have an infinite weight and correspond to the part of knowledge that is considered to be certain. On the other hand, soft constraints correspond to imperfect knowledge the violation of which does not incur the exclusion of a world from the set of possible worlds. An acceptable world must at least satisfy the hard constraints.

Formally, a knowledge base L of weighted formulae, along with a finite domain of constants $\mathcal{C}$, is transformed into a ground Markov Network $M_{L,\mathcal{C}}$. All formulae are converted into clausal form and each clause is ground according to the domain of its variables. The nodes of $M_{L,\mathcal{C}}$ are Boolean random variables, each corresponding to a possible grounding of a predicate that appears in L. The predicates of a ground clause form a clique in $M_{L,\mathcal{C}}$. Each clique has its own weight w_i, as well as a Boolean *feature* that takes a value of 1 when the ground clause is true and a value of 0 otherwise. The ground $M_{L,\mathcal{C}}$ defines a probability distribution over possible worlds and is represented as a log-linear model.

As with every event recognition mechanism, our goal is to recognize LTA of interest, given streams of STA. For instance, we want to be able to recognize the “moving” LTA, based on a narrative of STA that represent people walking, running, etc. To that effect, discriminative MLNs [11] are used. Specifically, the random variables in $M_{L,\mathcal{C}}$ are split in two subsets: evidence variables X and query variables Y. The former subset corresponds to the input ground “happensAt” predicates, i.e. the STA stream. The latter corresponds to groundings of the “holdsAt”, “initiatedAt”, and “terminatedAt” predicates. The joint probability distribution of a possible query assignment $Y = \mathbf{y}$, conditioned over a given assignment $X = \mathbf{x}$ is defined as follows:

$$P(Y = y|X = x) = \frac{1}{Z(\mathbf{x})} exp\left(\sum_{i=1}^{|F_C|} w_i n_i(\mathbf{x}, \mathbf{y})\right) \quad (9)$$

Vectors $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{Y}$ represent a possible assignment of evidence X and query variables Y, respectively. $\mathcal{X}$ and $\mathcal{Y}$ are the sets of possible assignments that the evidence variables X and the query variables Y can take. F_c is the set of clauses produced from the knowledge base L and the domain constants $\mathcal{C}$. The scalar value w_i is the weight of the i-th clause and $n_i(\boldsymbol{x}, \boldsymbol{y})$ is the number of satisfied groundings of the i-th clause in $\boldsymbol{x}$ and $\boldsymbol{y}$. $Z(\boldsymbol{x})$ is the partition function that normalises over all possible assignments $\boldsymbol{y}' \in \mathcal{Y}$ of query variables given the assignment $\boldsymbol{x}$:

$$Z(\mathbf{x}) = \sum_{\mathbf{y}' \in \mathcal{Y}} exp\left(\sum_{i=1}^{|F_c|} w_i n_i(\mathbf{x}, \mathbf{y}')\right)$$

2.3.1 MLN-EC

A dialect of the Event Calculus that incorporates MLNs is MLN-EC [2]. MLN-EC combines the expressive powers of the Event Calculus with the probabilistic framework of Markov Logic Networks in order to deal with uncertainty. As with Prob-EC, the input to MLN-EC is a stream of STA, as well as a set of domain-specific LTA definitions. Then, along with a set of domain-independent axioms, MLN-EC generates a compact knowledge base upon which Markov Networks will be produced and probabilistic inference and learning can be performed (see [2] for more details).

Table 2: MLN-EC predicates

Predicate	Meaning
happens(*E*, *T*)	Event *E* occurs at time *T*
holdsAt(*F* = *V*, *T*)	The value of fluent *F* is *V* at time *T*
initiatedAt(*F* = *V*, *T*)	At time *T* a period of time for which *F* = *V* is initiated
terminatedAt(*F* = *V*, *T*)	At time *T* a period of time for which *F* = *V* is terminated

Table 2 presents the basic predicates used in MLN-EC. We can see that they are the same as those of Prob-EC without the “initially” predicate. The “happensAt” predicate expresses input evidence, i.e. the occurrence of a STA. The input stream of STA is delivered in the form of a narrative of “happensAt” predicates. Also, similarly to what we have already seen in Prob-EC, “initiatedAt” and “terminatedAt” predicates are used in the domain-specific LTA definitions to denote the initiation and termination points for the various LTA.

The domain-independent, internal MLN-EC axioms are the following:

holdsAt(*F* = *V*, *T*+1) ←
    initiatedAt(*F* = *V*, *T*). (10)

holdsAt(*F* = *V*, *T*+1) ←
    holdsAt(*F* = *V*, *T*),
    not terminatedAt(*F* = *V*, *T*). (11)

not holdsAt(F = V, T+1) ←
terminatedAt(F = V, T). (12)

not holdsAt(F = V, T+1) ←
not holdsAt(F = V, T),
not initiatedAt(F = V, T). (13)

Axiom (10) dictates that a fluent holds right after it has been initiated. Axiom (11) indicates that if a fluent holds at a specific time point and it is not terminated at this very time point, then it will hold in the following time point, as well. Axiom (12) states that if a fluent is terminated then it does not hold at the next time point and, finally, axiom (13) suggests that a fluent that does not hold and is not initiated, will continue not holding.

2.3.2 Inference in MLN-EC

Equation (9) cannot be directly computed, due to the normalization constant $Z(\boldsymbol{x})$. In order to perform inference in MLN-EC, two types of inference are used, namely marginal inference and maximum a-posteriori (MAP) inference.

Marginal inference computes the conditional probability that LTAs hold given a narrative of observed STA

$$P(holdsAt(LTA, T) = true \mid narrative\ of\ STA)$$

In other words, this probability value measures the confidence that the LTA is recognized. Given the high complexity of this computation, we can employ Markov Chain Monte Carlo (MCMC) sampling algorithms to approximate it. However, as pointed out in [2], the presence of deterministic dependencies (i.e. formulae with infinite weights) violates two very important properties of a Markov chain, namely *ergodicity* and *detailed balance*. This violation produces isolated regions in the chain, due to the existence of transitions with zero or near-zero probabilities. Traditional MCMC methods, like the Gibbs sampler [12], get trapped in regions and give poor results.

To overcome this issue, the state-of-the-art MC-SAT algorithm [13] is used. MC-SAT is a MCMC algorithm that combines satisfiability testing with slice-sampling [14]. Initially, a satisfiability solver is used to find the assignments that satisfy all the hard constraints -- i.e. clauses with infinite weights. Subsequently, a series of sampling steps takes place. At each step, the algorithm chooses the clauses that must be satisfied in the next step from the set of ground clauses that are satisfied at the current step, with probability proportional to their weight. Therefore, clauses with infinite or strong weights will be chosen with certainty or with high probability, respectively. Afterwards, instead of sampling from all possible states, MC-SAT restricts sampling only to the states that satisfy at least all chosen clauses. Thus, the algorithm does not get trapped in local regions and the resulting Markov chain satisfies both ergodicity and detailed balance.

Another type of inference used in MLNs is MAP inference. MAP inference determines the most probable assignment among all “holdsAt” instantiations that are compatible with the given narrative of STA, that is:

$$\underset{holdsAt}{\operatorname{argmax}}\{P(holdsAt(LTA, T) = true \mid narrative\ of\ STA)\}$$

In MLNs this task is equivalent to finding the truth assignment of all “holdsAt” instantiations that maximizes the sum of weights of satisfied ground clauses. This problem is equivalent to the NP-hard weighted maximum satisfiability problem. There has been a lot of effort in finding an approximate solution using local search algorithms, like MaxWalkSAT ([15], [16]) or using linear programming methods ([17], [18], [19]). Special attention has been paid to the LP-relaxed Integer Linear Programming method, proposed by Huynh and Mooney [17]. This method takes a Markov network, translates it into a set of linear constraints and solves it using standard linear optimization algorithms. See [20], [3], and [17] for more details.

2.4 Probabilistic Interval-based Event Recognition

Probabilistic Event Recognition methods like Prob-EC and MLN-EC, which were discussed above produce point-based output. In other words, these methods produce sequences of ground probabilistic “holdsAt” atoms, accompanied by a probability value, denoting that an LTA is deduced to have taken place at a certain time point with a specific probability.

However, as Artikis, Makris and Paliouras point out [4], an instantaneous indication of an activity through a sequence of probabilistic ProbLog “holdsAt” facts may be misleading, due to a possibly unreliable sensor or an inaccurate LTA definition or several other factors that may introduce noise. Erroneous LTA recognition may cause delays and impede the monitoring process. Therefore, we need a more robust recognition method. Reasoning in terms of temporal intervals instead of time points can add robustness to the recognition process.

2.4.1 The PIEC algorithm

The proposed method that performs event recognition and extracts probabilistic maximal intervals within which an LTA is deduced to have taken place is called *Probabilistic Interval-based Event Calculus (PIEC)* [4]. Figure 2 illustrates the big picture of the inference procedure.

Specifically, the process is split in two phases, the Point-based recognition phase and the Interval-based recognition phase. In the Point-based recognition phase, we take a stream of probabilistic ground facts (i.e. the STA) and give them as input to a point-based activity recognition method, like the ones presented in Chapter 2: Prob-EC or MLN-EC. Along with the input STA, we must provide the necessary domain-specific LTA definitions, in the form of initiation and termination rules. The output of Prob-EC or MLN-EC is a sequence of ground probabilistic “holdsAt” atoms that display the inferred probability that an LTA takes place at a specific time point. Afterwards, this sequence of probabilistic instantaneous LTA is forwarded to the Interval-based recognition phase, where the user supplies a probability threshold above which the probabilistic intervals are considered important and receives an output of probabilistic maximal intervals.

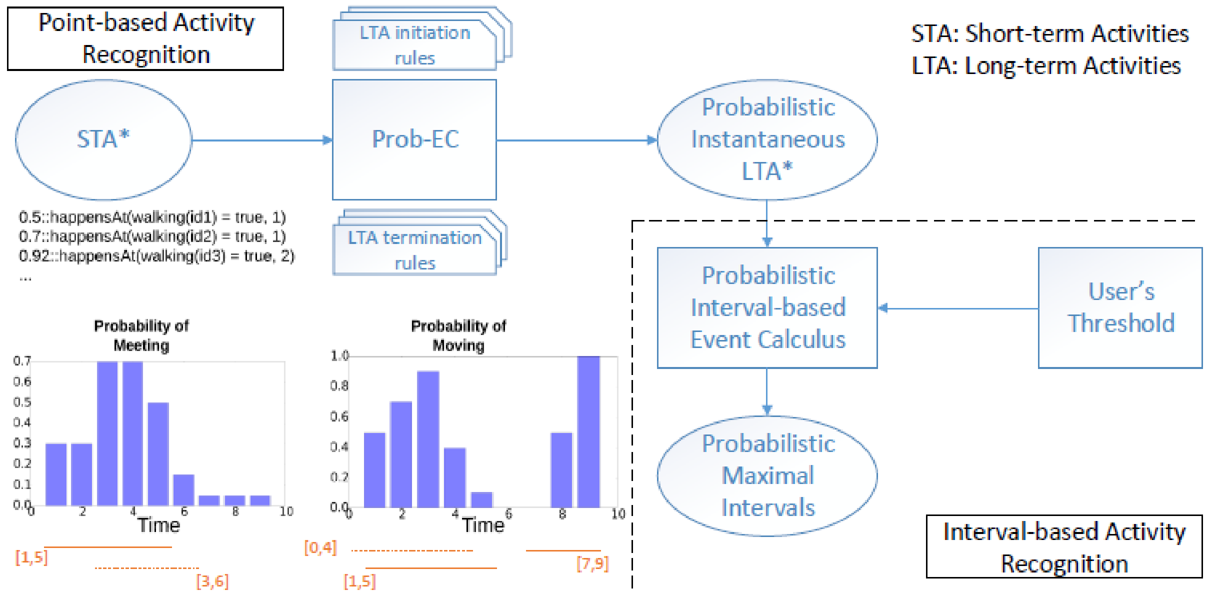

Figure 2: The big picture of Probabilistic Interval-based Activity Recognition

In the bottom left part of Figure 2 there is a visualized simple example of the extraction of intervals from points using PIEC. Specifically, the barplots show the instantaneous probabilities that LTA "meeting" and "moving" take place in a 10-timepoint-long time period. Below the barplots there are the probabilistic maximal intervals extracted by PIEC. It becomes evident that there can be more than one overlapping probabilistic maximal intervals. In that case, PIEC has a tie breaking mechanism by calculating a measure called the *credibility* of each of the candidate overlapping intervals and choosing the most credible amongst them. Solid lines correspond to credible intervals. Both the probability and the credibility of a probabilistic maximal interval will receive plenty of attention in the remainder of this thesis.

Before proceeding with the presentation of the PIEC algorithm, we provide a set of definitions that will appear useful later on.

Definition 1. *The probability of interval* $I_{LTA} = [i, j]$ *of an LTA with* $length(I_{LTA}) = j - i + 1$ *time points is defined as*

$$P(I_{LTA}) = \frac{\sum_{k=i}^{j} P(holdsAt(LTA = true, k))}{length(I_{LTA})}$$

In other words, the probability of an interval is equal to the average of the probabilities of the time points that it contains.

Another key concept of PIEC is that of a probabilistic maximal interval:

Definition 2. *A probabilistic maximal interval* $I_{LTA} = [i, j]$ *of an LTA is an interval such that, given some threshold* $\mathcal{T} \in [0,1]$, $P(I_{LTA}) \geq \mathcal{T}$, *and there is no other interval* I'_{LTA} *such that* $P(I'_{LTA}) \geq \mathcal{T}$ *and* I_{LTA} *is a sub-interval of* I'_{LTA}.

A consequence of the definition of a probabilistic maximal interval is that such intervals may be overlapping. Two examples are shown in Figure 2 – see the overlapping lines under the instantaneous probability evolution diagrams of “meeting” and “moving”. From a set of overlapping probabilistic maximal intervals, we keep only one, using interval credibility, defined as the product of interval length and probability:

$$Cred(I_{LTA}) = length(I_{LTA}) \cdot P(I_{LTA}) = \sum_{k} P(holdsAt(LTA = true, k)) \quad (14)$$

where *k* are the time-points of the interval I_{LTA}.

Algorithm 1 contains the steps of the Interval-based activity recognition procedure, in detail. First of all, the PIEC algorithm keeps four data structures (lists, in particular) to help keep the calculations simple and the complexity of the algorithm linear. These lists are presented in Table 3. Specifically, PIEC keeps a list of all the input instantaneous probabilities, named In, a list of the input probabilities subtracted by a user-specified probability threshold $\mathcal{T}$, named L, a list of the progressive sums of list L, named $prefix$, as well as a list of the progressive maximum prefixes from a certain time point until the end of the input, named dp.

For efficient interval estimation, PIEC relies on the following construct:

$$dprange[i,j] = \begin{cases} dp[j] - prefix[i-1], & if\ i > 0 \\ dp[j], & if\ i = 0 \end{cases} \quad (15)$$

$dprange[i,j]$ expresses the maximum sum that may be computed by adding all elements of L starting from i and ending in some $j^* \geq j$, i.e. $\max_{j^*}(L[i] + \cdots + L[j^*])$. Intuitively, $dprange$ is a maximality indicator for a certain interval $[i,j]$. $dprange \geq 0$ means that the interval $[i,j]$ is indeed a valid probabilistic interval, but it cannot be considered maximal yet, so the algorithm needs to further expand it and re-calculate the $dprange$ for the expanded interval $[i,j+1]$. $dprange < 0$ means that the interval $[i,j]$ is not a valid probabilistic maximal interval and the algorithm needs to keep searching, by sliding the current interval to the right and focusing on $[i+1,j]$. Algorithm 1 presents this procedure in detail.

In lines 2 – 9 of Algorithm 1, there is the initialization of the data structures enumerated earlier, as well as two pointers *start* and *end*, a data structure for the output intervals, and the $dprange$ matrix. Then, the algorithm starts sliding pointers *start* and *end* to the right, starting from 0 until the end of the input, calculating the $dprange$ at each (*start*, *end*) pair, according to (15). Line 12 of the algorithm implements the first branch of the definition of $dprange$, while line 14 implements the second one.

Depending on the value of $dprange$, the algorithm distinguishes two main cases, namely $dprange$ being negative or non-negative. In the non-negative case (see line 15), the algorithm knows that [*start*, *end*] is either a probabilistic maximal interval or, at least, the foremost sub-interval thereof. Therefore, it “flags” [*start*, *end*] (i.e. it labels it as a potentially maximal probabilistic interval) and slides the *end* pointer one step to the right of the timeline. In the corner case that the *end* pointer is pointing at the end of the timeline, PIEC has no means of expanding this interval, therefore it simply adds it to the output, as is (see line 17). In the negative $dprange$ case (line 20), PIEC knows that there is no element $end^* \geq end$ s.t. $dprange[start, end^*] \geq 0$. If we take a look at the definition of $dprange$, we will see that it depends on the value of dp, which – by definition – is a decreasing function. As a result, when PIEC computes a negative $dprange[start, end]$ and the previously examined interval, $[start, end - 1]$, is flagged, then it deduces that the previous interval (that is, $[start, end - 1]$) is indeed a probabilistic maximal interval and adds it to the output (line 22). In the corner case where both pointers *start* and *end* are pointing at the last time point of the timeline and the instantaneous probability at this point is greater than the threshold, then PIEC considers singleton interval $[n, n]$ to be probabilistic maximal (see line 24}). Either case, if PIEC comes across an interval with a negative $dprange$ it knows that it is neither a probabilistic maximal interval nor the foremost sub-interval thereof. Hence, it turns the binary flag to "false" (line 25) and slides the *start* pointer one step to the right (line 26). Full mathematical proof of the correctness of this procedure is provided later on.

2.4.2 Interval credibility

In the last line of Algorithm 1, we can see a call to a routine named "getCredible", that takes as input the full list of probabilistic maximal intervals that PIEC has produced. As we have already discussed, the way that PIEC performs its computations, leads to the existence of possibly overlapping probabilistic maximal intervals. The activity recognition task requires that every occurrence of an LTA be unambiguously represented. Overlapping intervals contain some level of ambiguity. If, for instance, we are in a situation similar to that depicted in Figure 3, where PIEC suggests that a certain LTA take place during the temporal intervals {[3, 25], [2, 27], [5, 28], [6, 29]}, we want to be able to keep just one of these intervals: the one that best fits to the actual occurrence of this LTA. In the specific case of Figure 3, the most credible interval is [5, 28], shown in bold.

Table 3: The lists of PIEC

Notation	Meaning
$\mathcal{T}$	Probability threshold
$In[0..n]$	The list of input instantaneous LTA probabilities
$L[i]$	$= In[i] - \mathcal{T}$, i.e. each element of In is subtracted by $\mathcal{T}$
$prefix[i]$	$= \sum_{j=0}^{i} L[j]$, i.e. the cumulative sum over L
$dp[i]$	$= \max_j(prefix[j]), j \in [i, n]$, i.e. the maximum prefix sum that can be reached from element i to n

Algorithm 1 Probabilistic Maximal Interval Estimation

```
Input: List In with instantaneous LTA probabilities and threshold T.
Output: List output of credible probabilistic maximal intervals.
function PIEC(List In[0..n], threshold T)
    L[i] ← In[i]−T for each 0 ≤ i ≤ n
    prefix ← computePrefix(L)
    dp ← computeDp(prefix)
    start ← 0
    end ← 0
    flag ← false
    output ← ∅
    dprange[start, end] ← 0
    while (start ≤ n and end ≤ n) do
        if start > 0 then
            dprange[start, end] ← dp[end]−prefix[start−1]
        else
            dprange[start, end] ← dp[end]
        if dprange[start, end] ≥ 0 then
            if (end = n and start < end) or (end = start = n and In[start] ≥ T) then
                add(start, end) to output
            flag ← true
            end++
        else
            if (start < end and flag = true) then
                add(start, end−1) to output
            if (start = end and In[start] ≥ T) then
                add(start, end) to output
            flag ← false
            start++
    return getCredible(output)
```

To achieve this, PIEC is accompanied by an auxiliary mechanism that takes all sets of overlapping probabilistic maximal intervals, computes their $credibility$ measure, and from each set picks the most credible interval. The ultimately chosen intervals are the final estimation of the algorithm as to when did an LTA take place, according to the input STA.

This mechanism is presented step-by-step in Algorithm 2. “getCredible” takes one argument, namely the list of possibly overlapping probabilistic maximal intervals, while it returns the list of credible probabilistic maximal intervals. If the input list contains more than one element, the algorithm first initializes some variables that are going to be needed later on. Specifically, in lines 5 – 8, “getCredible” initializes its output list to be empty, takes the first interval from the input and calculates its credibility. *currentEnd* is an auxiliary pointer that marks the end of the last interval examined and is useful in detecting overlaps. In line 7 *currentEnd* points to the end of the first input interval, while in line 8 *maxCredibility* is set to the credibility of the first input interval.

For every other input interval, “getCredible” checks if it is overlapping with the previous one (line 10). If there is an overlap, then it updates pointer *currentEnd* accordingly (line 14) and checks the credibility of the new interval. If the credibility is greater than the maximum credibility seen so far in the current group of overlapping intervals, then it appropriately updates the *maxCredibility* and the *maxCredInterval* (see lines 12, 13). If there is no overlap with the previous interval, this means that the algorithm has reached the end of a group and has already found the most credible among them. Therefore, in line 16 it adds *maxCredInterval* to the output and starts a new overlapping group, considering the current interval as the *maxCredInterval*, and its credibility as the *maxCredibility*. At the same time, it updates *currentEnd* to point to the last timepoint of the current interval. These actions can be seen in lines 17 – 19 of Algorithm 2. Ultimately, when the entire input has been processed, "getCredible" adds the last *maxCredInterval* to the output and returns it (lines 20, 21).

Algorithm 2 Credibility mechanism

```
    Input: List overlapping with possibly overlapping probabilistic maximal intervals.
    Output: List credible of credible probabilistic maximal intervals.
 1: function GETCREDIBLE(List overlapping[0..m])
 2:     if overlapping is empty or singleton then
 3:         credible ← overlapping
 4:     else
 5:         credible ← ∅
 6:         maxCredInterval ← overlapping[0]
 7:         currentEnd ← maxCredInterval[1]
 8:         maxCredibility ← probability(maxCredInterval) × length(maxCredInterval)
 9:         for i ← overlapping[1..m] do
10:             if overlapping[i][0] < currentEnd then
11:                 if probability(overlapping[i]) × length(overlapping[i]) > maxCredibility then
12:                     maxCredibility ← probability(overlapping[i]) × length(overlapping[i])
13:                     maxCredInterval ← overlapping[i]
14:                 currentEnd ← overlapping[i][1]
15:             else
16:                 add(maxCredInterval) to credible
17:                 maxCredInterval ← overlapping[i]
18:                 currentEnd ← overlapping[i][1]
19:                 maxCredibility ← probability(overlapping[i]) × length(overlapping[i])
20:         add(maxCredInterval) to credible
21:     return credible
```

2.4.3 Time and space complexity

In [4] it is maintained that PIEC has $O(n)$ time and space complexity, where n is the length of the input. Indeed, the data structures and pointers involved have been carefully chosen so as to aid the algorithm perform its calculation in at most two passes over the input probability array. More specifically, arrays L, $prefix$, and dp are computed in $\Theta(n)$ time. The detection of all the probabilistic maximal intervals requires a $O(2n)$ time because, at the worst case, pointers *start* and *end* will both have to slide through the entire input. The choice of the most credible interval is also a linear process, as the candidate intervals are temporally sorted based on their initial timepoint. Finally, as far as the memory is concerned, the sizes of the auxiliary data structures are linear with respect to n, therefore the entire memory consumption remains a linear function of n.

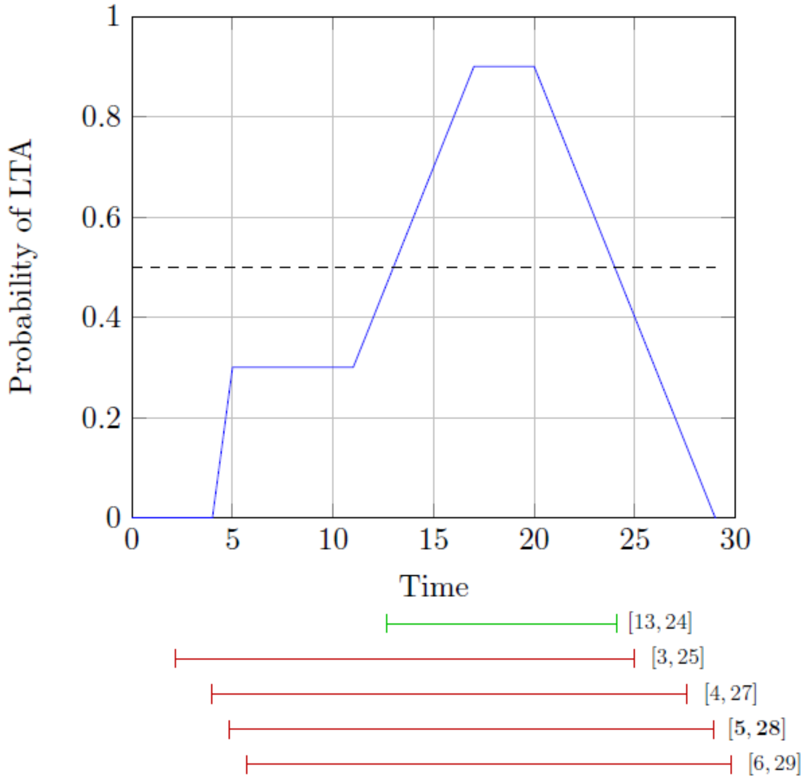

Figure 3: Sample case where PIEC computes several overlapping maximal intervals and chooses one of them as the most credible. The probability threshold has been set to 0.5 and is shown in the dashed line.

2.5 The CAVIAR dataset

All the experiments presented in this thesis have been conducted using the CAVIAR dataset[2] for human activity recognition. This dataset features 28 staged public space surveillance videos, totaling 26419 video frames. There are actors that perform several activities like walking, running, meeting, fighting, among others. The dataset has been fully annotated by its creators to provide the ground truth for all the activities included.

There are two categories of activities that appear in the CAVIAR dataset, namely Short-term activities and Long-term activities. The former, abbreviated as STA are simple actions that take place in a short period of time. The latter, abbreviated as LTA are composite activities, often combinations of STA and last longer. STA include walking, running, making gestures or being in motion without changing one's location (active), resting, appearing and disappearing from the video frame. On the other hand, LTA include meeting, two or more persons moving together, a person leaving an object unattended and two or more people fighting.

There are also auxiliary measurements concerning the position and orientation of the various tracked people and objects at each time point. This type of information is necessary for computing the distance between two entities, as well as the direction to which a person might be moving. Both distance and orientation are crucial for the definition of LTA like "meeting", "moving", "leaving object", and "fighting", because all of them involve people approaching other people or objects.

[2] http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/

3. RELATED WORK

There is a wide spectrum of technologies developed with view to detecting incidents or activities of interest from data streams. According to [21], these technologies form the field of Complex Event Processing (CEP). Unlike Active Database Systems (ADSMS) and Data Stream Management Systems (DSMS) (see, for instance [22], [23]), which run relational queries over data streams, CEP systems consider data streams as streams of (instantaneous or durative) events and use the temporal relations between them to detect interesting patterns.

In the literature ([24], [25]), CEP query languages are often divided into three categories, namely Logic-based, Tree-based and Automata-based. All dialects presented in the previous section are founded on the Event Calculus and pertain to the Logic-based category. Other examples of Logic-based languages include ([26], [7], and [27]). Tree-based models can be found in [28] and [29], whereas [30], [31], [32], [33], and [34] are examples of Automata-based languages. Bucchi et al. [21] state that state-of-the-art CEP query languages suffer from obscure semantics and problematic query evaluation. Specifically, the semantics of most CEP query languages tend to be too complicated, unintuitive or severely restricted and this makes the languages difficult to understand and evaluate. In order to deal with this issue, Bucchi et al. propose Complex Event Logic – CEL, a formal CEP language with well-defined semantics, as well as a formal evaluation framework for CEL, using an automata-based computational model called Complex Event Automata – CEA. In their work, Bucchi et al. present efficient evaluation algorithms and provide experimental results that suggest that their framework outperforms other state-of-the-art CEP systems.

According to [35], CEP is considered a subset of the Event Processing (EP) paradigm. Dayarathna et al. define EP as the computing that captures and processes the happening of real-world incidents, whereas they define CEP as the computing that performs calculations on complex events. Complex events are defined as aggregations or derivations on groups of events, and events are just real-world incidents. In this work, the software tools for EP are divided into three categories, namely Event Processing Platforms (EPP) (see, for instance SQLstream[3] and DataTorrent RTS[4] ([36], [37], and [38])), Distributed Stream Computing Platforms (DSCP) ([39] and [40]), and CEP Libraries. A CEP library should be an immediately responsive component, capable of identifying non-trivial, meaningful patterns among heterogeneous and seemingly unrelated events. Examples of CEP libraries are Esper[5], Siddhi [41] and Cayuga [42]. According to the work of Dayarathna et al., CEP has been used in a wide spectrum of modern, real-world applications, associated with the Internet of Things, as well as text, video and graph data stream processing and analytics.

This thesis focuses on probabilistic event recognition, and therefore we should take some time to briefly discuss systems that were designed with view to handling noise, either in the STA or in the LTA definitions. The parameters that must be taken into consideration when studying a complex event recognition system are:

---

[3] https://sqlstream.com/

[4] http://web.datatorrent.com/2014-Hadoop-World-Strata.html

[5] http://www.espertech.com/esper

- How knowledge is represented
- How time is modelled

In many systems, including the ones mentioned in the previous section, the knowledge is represented by means of Logic. For instance, Prob-EC is an event recognition system that applies the Event Calculus and the Probabilistic Logic Programming language ProbLog for the representation of knowledge. As far as the modelling of time is concerned, several event recognition systems calculate the probability of occurrence for an LTA at each timepoint, separately. These methods are called *timepoint-based*. On the other hand, there are other methods that try to recognize the occurrence of an LTA in terms of temporal intervals. These are called *interval-based* methods. Prob-EC is a timepoint-based probabilistic event recognition system that receives a sequence of timestamped STA with probabilities attached for every occurring instant.

Another MLN-based approach, presented by [10], is DEC-MLN. This dialect addresses the uncertainty that is caused by using incomplete LTA definitions. This uncertainty is expressed by means of weighted LTA definitions, which in turn are expressed using the Event Calculus, like the methods discussed in the previous Chapter. MLNs were also used in [43], where the LTA recognition was based on noisy STA, stemming from lower-level sensors-classifiers. A logic-based LTA recognition system is available on [44], where the proposed system attempts handling noise that is a result of unreliable sensors. The LTA definitions are uncertain, thus denoting an estimation of the credibility degree of a rule, whereas the STA are accompanied by a correct detection probability. Another logic-based uncertainty handling approach, called PEC, can be found in [45]. PEC presents a version of the Event Calculus appropriate for probabilistic recognition. It introduces an extension that maintains the format of the Event Calculus clauses, as well as the notion of possible worlds for the calculation of the activities.

Typically, in order to address uncertainty, systems that incorporate probabilistic graphical models are being used. These systems apply probabilistic graphical models during the processing of the data corresponding to the STA and aim at inferring LTA. Hidden Markov Models are used [46] with view to recognizing digits in speech signals. Moreover, Dynamic Bayesian Networks have been used for the recognition of audio-visual speech signals [47]. Also, in [48] we observe the application of Conditional Random Fields. The aforementioned models provide important information, even at the presence of noise, since they can handle it in a natural way. However, their ability to compactly and effectively define complex activities deteriorates, due to their increased complexity.

In addition to discrete time-point-based systems, temporal-relation-based systems have also been proposed in the Complex Event Recognition literature. For instance, in [49] data generated from low-level classifiers are being used in order to compute the most likely sequence of LTA. This system uses MLN for the LTA representation and the Allen's Interval Algebra for the reasoning upon temporal intervals. Another similar work is presented in [50]. This Event Recognition system considers LTA beginning and ending at uncertain timepoints. These uncertain timepoints are represented by means of uniformly distributed random variables, which are used in computing the LTA probabilities. Time is represented using an extended version of Allen's interval algebra that uses segmented intervals. The probability estimation is performed with respect to the LTA sequences, by calculating the probability sum of every mutually exclusive possible world. Pruning and caching techniques are also used in order to optimize the process.

In this work we focus on calculating the probability of an LTA taking place in a temporal interval. We model this probability as the arithmetic mean of the instantaneous probabilities of the LTA under examination. These instantaneous probabilities are computed using Prob-EC, which incorporates the Event Calculus axioms and the expressive powers of the ProbLog language. Afterwards, in order to deal with the noisy recognition, we apply the Probabilistic Interval-based Event Calculus (PIEC) algorithm, which is the main focus of this thesis. This algorithm computes probabilistic maximal intervals – that is, sets of one or more consecutive timepoints, which have a probability greater than or equal to a certain probability threshold and, at the same time, they are of maximum length. The algorithm is based upon the longest non-negative sum interval (LNNSI) computation problem [51], as well as the maximum sum interval (MSI). According to the LNNSI problem, we are asked, given a sequence of real numbers, to find a maximum length contiguous subsequence that has a non-negative sum. Hence, we transform our LTA recognition problem into a maximum length non-negative sum interval computation problem and, subsequently, we identify all maximal intervals whose probability is greater than or equal to a user-specified probability threshold.

4. CORRECTNESS OF THE PIEC ALGORITHM

In this chapter, we provide a series of mathematical proofs that the PIEC algorithm is correct. We investigate its soundness and completeness and try to prove that all intervals produced by PIEC are indeed probabilistic maximal and there are also no probabilistic maximal intervals that PIEC does not compute, given an input sequence of instantaneous probabilities. In what follows, we consider the full output of PIEC, without the filtering of the credibility mechanism.

4.1 Soundness

An interval $I_{LTA} = [i, j]$ satisfies the condition of a probabilistic maximal interval that $P(I_{LTA}) \geq \mathcal{T}$, if and only if the sum of the corresponding elements of list L is non-negative:

$$P(I_{LTA}) \geq \mathcal{T} \Leftrightarrow P([i,j]) \geq \mathcal{T} \Leftrightarrow \frac{\sum_{k=i}^{j} In[k]}{j-i+1} \geq \mathcal{T} \Leftrightarrow$$

$$\sum_{k=i}^{j} In[k] \geq \mathcal{T}(j-i+1) \Leftrightarrow \sum_{k=i}^{j} In[k] - \mathcal{T}(j-i+1) \geq 0 \Leftrightarrow \quad (16)$$

$$(In[i] - \mathcal{T}) + \cdots + (In[j] - \mathcal{T}) \geq 0 \Leftrightarrow \sum_{k=i}^{j} L[k] \geq 0$$

For brevity, in all proofs below, we assume intervals $[i, j]$ with $0 < i < j < n$, where n is the last time-point of the dataset. Therefore, such an interval may only be produced by line 22 of the Algorithm of PIEC.

Proposition 1. *Every interval computed by PIEC has probability greater or equal to the given threshold* $\mathcal{T}$**.**

Proof. Assume that PIEC computes the interval $[i, j]$ and that $P([i, j]) < \mathcal{T}$. Since PIEC computes $[i, j]$, then, by Algorithm 1, we have that:

$$dprange[i,j] \geq 0 \Leftrightarrow dp[j] - prefix[i-1] \geq 0 \Leftrightarrow$$

$$\max_{k \in [j,n]}(prefix[k]) - prefix[i-1] \geq 0 \Leftrightarrow \max_{k \in [j,n]}(prefix[k]) \geq prefix[i-1] \quad (17)$$

Moreover, since PIEC computes $[i, j]$, then:

$$dprange[i,j+1] < 0 \Leftrightarrow dp[j+1] - prefix[i-1] < 0 \Leftrightarrow$$
$$\max_{l\in[j+1,n]}(prefix[l]) - prefix[i-1] < 0 \Leftrightarrow \max_{l\in[j+1,n]}(prefix[l]) < prefix[i-1] \tag{18}$$

From inequalities (17) and (18), we have that:

$$\max_{l\in[j+1,n]}(prefix[l]) < prefix[i-1] \leq \max_{k\in[j,n]}(prefix[k]) \tag{19}$$

Consequently:

$$\max_{l\in[j+1,n]}(prefix[l]) < \max_{k\in[j,n]}(prefix[k]) \Leftrightarrow \max_{k\in[j,n]}(prefix[k]) = prefix[j] \tag{20}$$

From formulas (19) and (20), we have:

$$prefix[i-1] \leq prefix[j] \tag{21}$$

However, according to our assumptions, we have $P([i,j]) < \mathcal{T}$, and according to formula (16), we have that:

$$\sum_{k=i}^{j} L[k] < \mathcal{T} \Leftrightarrow prefix[j] - prefix[i-1] < 0 \Leftrightarrow prefix[j] < prefix[i-1] \tag{22}$$

Formulas (21) and (22) are contradicting and thus our initial assumption that $P([i,j]) < \mathcal{T}$ does not hold. ■

Proposition 2. For every interval $[i, j]$ computed by PIEC, there is no interval $[k, l]$ with

- $k = i$ and $l > j$, or
- $k < i$ and $l = j$, or
- $k < i$ and $l > j$,

and $P([k, l]) \geq \mathcal{T}$.

Proof. First case:

Assume that PIEC computes the interval $[i, j]$ and that there is an interval $[i, l]$ with $l > j$ and $P([i, l]) \geq \mathcal{T}$. Since PIEC computes $[i, j]$, then we have from formula (19) that:

$$\max_{m \in [j+1, n]}(prefix[m]) < prefix[i-1] \tag{23}$$

However, according to our assumptions, $P([i, l]) \geq \mathcal{T}$, which, according to formula (22), implies that:

$$prefix[i-1] \leq prefix[l] \tag{24}$$

Formulas (23) and (24) are contradicting, as $\max_{m \in [j+1, n]}(prefix[m]) < prefix[l]$ does not hold since $l > j$, and thus our initial assumption that there is an interval $[i, l]$ with $l > j$ and $P([i, l]) \geq \mathcal{T}$ does not hold.

Second case:

Assume that PIEC computes the interval $[i, j]$ and that there is an interval $[k, j]$ with $k < i$ and $P([k, j]) \geq \mathcal{T}$.

By taking a closer look on the PIEC algorithm, we observe that whenever it comes across an interval $[i, j]$ with $dprange[i, j] < 0$, it ignores it and slides the beginning of the interval one step to the right (see line 26 of the algorithm). If the *dprange* is non-negative, PIEC “flags” this interval and tries to expand it by sliding the end of the interval to the right (see line 19). If *dprange* becomes negative, while expanding a flagged interval, PIEC recognizes that it has just exceeded the ending of the desired interval and shapes its output accordingly (see line 22).

Therefore, since PIEC computes $[i, j]$, it must have gone through at least one negative *dprange* before it moves its start index to timepoint i. So, there has to be a timepoint k and a timepoint j', with $k < i$ and $j' \leq j$, such that:

$$dprange[k, j'] < 0 \Leftrightarrow dp[j'] - prefix[k - 1] < 0 \tag{25}$$

Also, since $j' \leq j$, we have that:

$$dp[j'] \geq prefix[j] \tag{26}$$

From (25), (26), we obtain:

$$prefix[j] - prefix[k - 1] < 0 \Leftrightarrow P([k, j]) < \mathcal{T}$$

which is incompatible with our assumption that $P([k, j]) \geq \mathcal{T}$.

Third case:

Assume that PIEC computes the interval $[i, j]$ and that there is an interval $[k, l]$ with $k < i$ and $l > j$ and $P([k, l]) \geq \mathcal{T}$.

Similarly to the second case, since PIEC computes $[i, j]$, there has to be a timepoint k and a timepoint j', with $k < i$ and $j' \leq j < l$, such that:

$$dprange[k, j'] < 0 \Leftrightarrow dp[j'] - prefix[k - 1] < 0 \tag{27}$$

Also, since $j' \leq j < l$, there has to be:

$$dp[j'] \geq prefix[l] \tag{28}$$

From (27), (28), we obtain:

$$prefix[l] - prefix[k - 1] < 0 \Leftrightarrow P([k, l]) < \mathcal{T}$$

which is incompatible with our assumption that $P([k,j]) \geq \mathcal{T}$. ■

Proposition 3 (Soundness)**.** Every interval computed by PIEC is a probabilistic maximal interval.

Proof. From Proposition 1 we obtain that every interval $I = [i,j]$ computed by PIEC has a probability above the specified threshold $\mathcal{T}$. From Proposition 2 we obtain that there is no other interval I' such that $P(I') \geq \mathcal{T}$ and I is a sub-interval of I'. Therefore, $[i,j]$ is a probabilistic maximal interval. ■

4.2 Completeness

Proposition 4 (Completeness)**.** There is no probabilistic maximal interval that PIEC does not compute.

Proof. First of all, we must observe that the last step of the PIEC algorithm (line 27) consists of taking all overlapping probabilistic maximal intervals and choosing the most credible among them. This, by definition, makes the algorithm incomplete with respect to probabilistic maximal intervals. Therefore, we will study the algorithm's completeness without filtering out the non-credible intervals.

Let us assume that there is a probabilistic maximal interval $[i,j]$ that is not computed by PIEC. As we have seen in Propositions 1 and 2, this assumption implies that $P([i,j]) \geq \mathcal{T}$ and that there is no super-interval $[k,l]$ whose probability is $P([k,j]) \geq \mathcal{T}$.

Whenever PIEC comes across an interval $[i,j]$ with $dprange[i,j] \geq 0$, which becomes $dprange[i,j+1] < 0$ at the next step, the algorithm deduces that $[i,j]$ is a probabilistic maximal interval and adds it to its output.

So, if interval $[i,j]$ is not computed by PIEC, then we must have either

$$dprange[i,j] < 0 \tag{29}$$

or

$$dprange[i,j+1] \geq 0 \tag{30}$$

From inequality (29), we have:

$$\begin{aligned} dprange[i,j] < 0 &\Leftrightarrow \\ dp[j] - prefix[i-1] < 0 &\Leftrightarrow \\ dp[j] < prefix[i-1] &\Leftrightarrow \\ \max_{k\in[j,n]}(prefix[k]) < prefix[i-1] & \end{aligned} \tag{31}$$

However, since $prefix[l] \leq \max_{k\in[j,n]}(prefix[k])$, for every $l \geq j$, we obtain:

$$prefix[j] \leq \max_{k\in[j,n]}(prefix[k])$$

Thus, (31) becomes:

$$\begin{aligned} prefix[j] < prefix[i-1] &\Leftrightarrow \\ P([i,j]) < \mathcal{T} & \end{aligned}$$

which is not compatible with our assumption that $P([i,j]) \geq \mathcal{T}$.

Now, as far as inequality (30) is concerned, if we take a look at lines 15-19 of Algorithm 1, we can see that, whenever the PIEC algorithm detects a non-negative *dprange*, it sets the flag to true and starts expanding the interval by incrementing its end point. Hence, inequality (30) means that our algorithm has flagged the interval $[i,j+1]$ and moves its *end* pointer to $j+2$. But if this was true, it would mean that there is an interval $[i,j']$, with $j' > j$ and $P([i,j']) \geq \mathcal{T}$. However, this is contradicting with our assumption that $[i,j]$ is maximal.

Therefore, both our initial assumptions that $[i,j]$ is maximal and $P([i,j]) \geq \mathcal{T}$ have collapsed. ■

5. EXPERIMENTAL EVALUATION

After having completed the correctness analysis of our algorithm, we focus on evaluating it on a benchmark dataset for video activity recognition. Specifically, we use the – publicly available – CAVIAR dataset[6]. Our goal is to check and illustrate the algorithm's ability to produce probabilistic, maximal intervals which correspond to the actual occurrences of the respective LTA's as accurately as possible.

We took Makris' Python implementation of PIEC and re-implemented it using the Scala programming language. We then repeated the experiments conducted in [4] in order to verify the equivalence between the two PIEC implementations. The results are presented below and appear to be identical to those in [4]. However, Makris' evaluation is limited to comparing PIEC to Prob-EC. We need to involve other state-of-the-art probabilistic Event Calculi, as well, in order to make the comparison more thorough and to achieve deeper understanding of our algorithm. To that effect, we further extended the evaluation by juxtaposing PIEC to MLN-based methods, like the ones discussed in Chapter 2. This comparison is performed in two levels: A general level, aggregating information from all 30 videos of the CAVIAR dataset, and a detailed level, focusing on interesting cases taken from one video at a time. Finally, we start exploring the field of alternative credibility definitions. The original interval credibility definition, made by Makris et al., is just one of many possible credibility strategies. We try several different ways to calculate interval credibility and observe how the performance of our algorithm is affected each time.

5.1 Comparison to Prob-EC

For the first round of experiments we used the state-of-the-art activity recognition system Prob-EC [1]. Prob-EC works in a timepoint-based manner, receiving a set of noisy[7], timestamped STA's and producing a similar set of probabilistic LTA timepoints. However, it is intuitively more convenient and less error prone to talk about *intervals* during which an activity takes place, rather than a collection of timepoints and this is exactly what PIEC algorithm does: Receiving a collection of probabilistic timepoints and producing probabilistic maximal intervals.

In all probabilistic point-based methods, like Prob-EC and MLN-EC, the intervals are being estimated by filtering the timepoints whose probability lies above a threshold, say 0.5, specified by the user.

In order to test the accuracy of PIEC, we ran our algorithm on the output of Prob-EC for the CAVIAR dataset, calculated its Precision, Recall and F-measure and compared its F-measure with that of the older, plain threshold approach. Figures Figure 4, Figure 5 and Figure 6 illustrate this comparison.

We are interested in observing the behavior of the two approaches over various levels of noise in the data. In other words, we want to see which setting is more accurate and noise-tolerant. To that effect, we have added artificial noise, using the Gamma probability distribution, with Gamma mean values ranging from 0.0 (no artificial noise) to 8.0 (high artificial noise), with a step of 0.5 in between.

---

[6] http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/

[7] Noise is expressed by means of a number $n \in [0,1]$ which is attached to each timestamped STA, denoting the probability that this STA actually happens at the given timepoint

The following figures display F-measures for both approaches with respect to noise levels. Specifically, Figure 4 shows the comparison between PIEC and the plain threshold approach for the “fighting” LTA. The 3 blue lines represent the results of the PIEC approach with 3 different thresholds: 0.5 (light slate blue), 0.7 (blue), and 0.9 (navy blue). Similarly, the 3 red lines correspond to the results of the simpler approach with threshold values of 0.5 (light coral), 0.7 (red), and 0.9 (maroon). Figures Figure 5 and Figure 6 show the comparison results for the “meeting” and “moving” LTA’s, respectively.

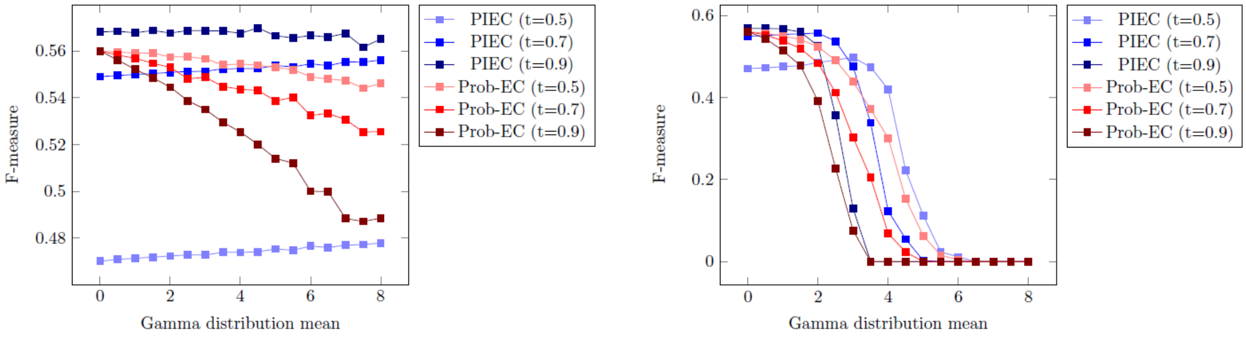

Figure 4: Testing results for the "fighting" LTA

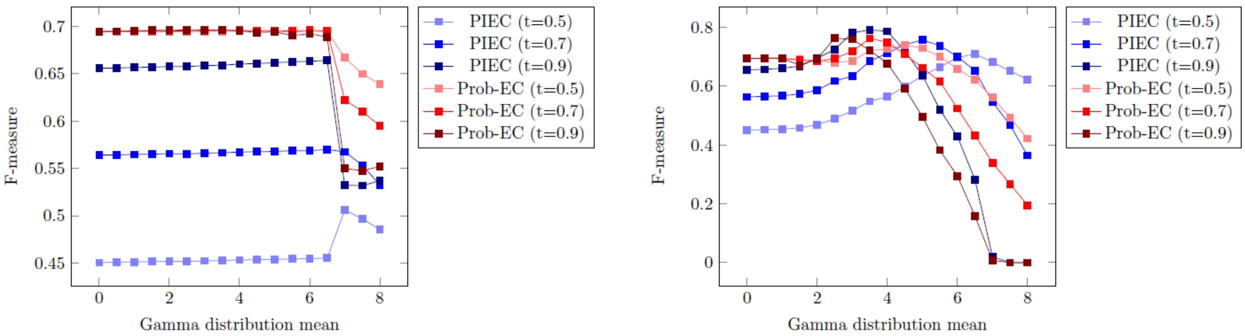

Figure 5: Testing results for the "meeting" LTA

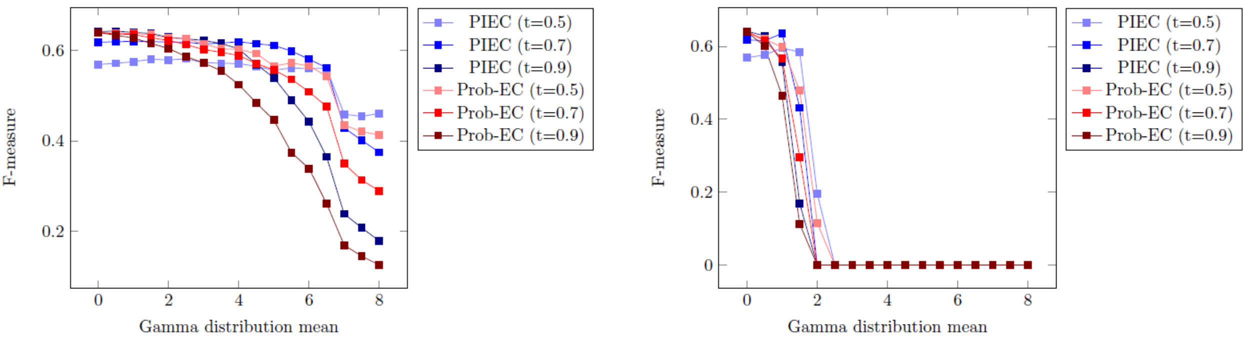

Figure 6: Testing results for the "moving" LTA

We observe that, in general, PIEC appears to achieve higher accuracy. Especially in the “fighting” case, blue lines are never below red lines that correspond to the same threshold value. They only become equal to each other in the rightmost part of Figure Figure 4, where the noise is so strong that, from some point onwards, both methods fail to recognize anything. Nevertheless, for milder noise levels, PIEC seems to consistently produce more accurate intervals for the occurrences of “fighting”.

On top of that, we can see higher PIEC thresholds leading to even better results, while on the simpler approach higher thresholds exhibit less noise-tolerant behavior. But this pattern only holds for the smooth noise scenario, on the left of Figure 4. When noise becomes significantly stronger, the smallest PIEC threshold appears to be the best option.

For the “meeting” LTA, the pattern seems somehow different. On the left of Figure 5, we observe that PIEC results are consistently worse than those of the plain threshold approach. We still witness, though, that higher PIEC thresholds lead to better results, while in the older approach thresholds work contrariwise. For a probability threshold of 0.9, the two approaches produce very similar results.

If we take a look at the right-hand part of Figure 5, we witness that, as noise becomes significantly stronger, PIEC surpasses the older approach for all 3 threshold values, while, at the same time, the smallest PIEC threshold once again proves to be the most noise-tolerant, for very high levels of noise. On the other hand, the effectiveness of the plain threshold approach drops quicker towards 0, especially for a threshold value of 0.9.

Finally, as far as the “moving” LTA is concerned, for smooth noise (Figure 6, on the left) the pattern seems similar to that of Figure 5, on the right. That is, for very small amounts of noise, the older approach gives better results, while, as noise gets stronger, blue lines surpass the corresponding red ones and, eventually, yet again PIEC with a threshold of 0.5 does best for the highest amounts of noise. The same pattern repeats on the right of Figure 6 with the only difference that, since the noise is very strong, both approaches quickly drop to 0.

Another interesting observation is that, no matter how better does PIEC behave, when the plain threshold approach reaches 0, for a given thresholds value, so does the PIEC approach, too. In other words, there is no case where, for a given noise amount and threshold value, one approach gives an F-measure of 0 and the respective F-measure of the other approach is greater than 0.

Overall, we can state that for moderate or high amounts of noise, PIEC seems to do significantly better, whereas for no or little noise, using the plain threshold approach seems a slightly better idea. These results are identical to those appearing on the evaluation section of [4], thus giving a strong indication that our Scala implementation of PIEC is equivalent to Makris’ Python implementation.

5.2 Comparison to MLNs

We now proceed to the comparison of PIEC to the results of the MLN-EC mechanism. We have taken two variants of the MLN-EC point-based probabilistic Event Recognition system; one that features manual rules and the Diagonal Newton weight learning method (hereinafter DN), and one that features rules learned by the OSLα structure learner (hereinafter OSLα) [20].

The output of MLN-EC is a sequence of instantaneous CE probabilities. We can take this sequence of probabilistic points and transform it into a sequence of intervals by either applying a probability threshold and building intervals from the points whose probability exceeds this threshold or by using the PIEC algorithm, whose purpose is to produce probabilistic maximal intervals from points.

In what follows, we start by making a general, overall performance comparison between DN/OSLα and PIEC for the LTA “meeting” and “moving” and for probability threshold values ranging from 0.1 to 0.9. Subsequently, we make an in-depth comparison, presenting specific cases (that is, pairs of entities involved in a LTA within a certain video) which appear to be of particular interest, either because PIEC is performing significantly better or because it is performing worse. In some of the cases where PIEC performs worse, we go to even greater depth by displaying all overlapping probabilistic maximal intervals to see if PIEC could have done better, had it used a different credibility definition.

5.2.1 Overall comparison

Figures Figure 7 to Figure 14 below, compare DN/OLSα with PIEC in terms of Precision, Recall and F-measure.

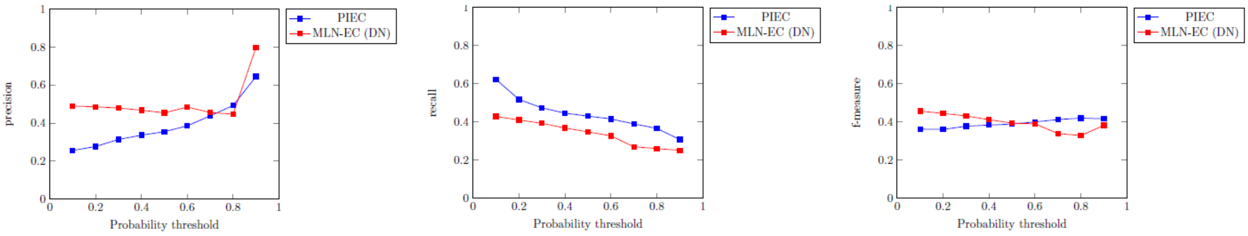

Figure 7: Micro Precision, Recall and F-measure comparison between DN and PIEC on CAVIAR's "meeting" LTA

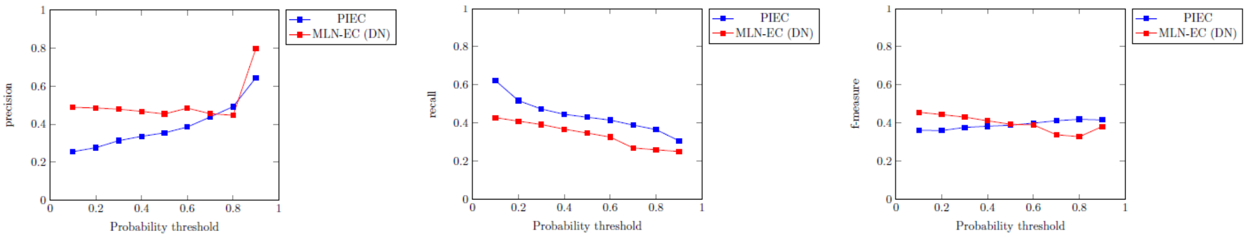

Figure 8: Macro Precision, Recall and F-measure comparison between DN and PIEC on CAVIAR's "meeting" LTA

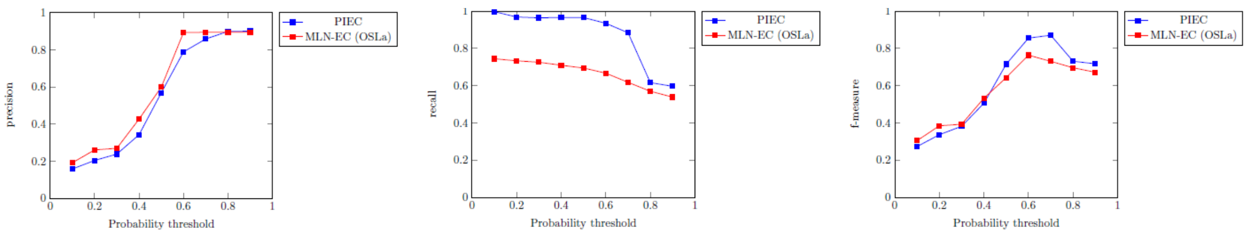

Figure 9: Micro Precision, Recall and F-measure comparison between OSLα and PIEC on CAVIAR's "meeting" LTA

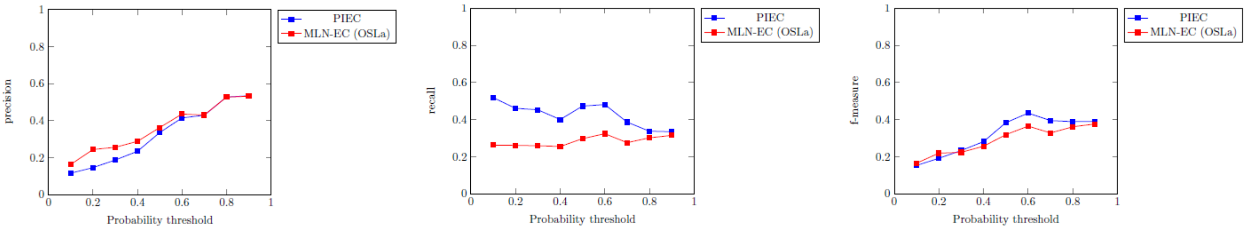

Figure 10: Macro Precision, Recall and F-measure comparison between OSLα and PIEC on CAVIAR's "meeting" LTA

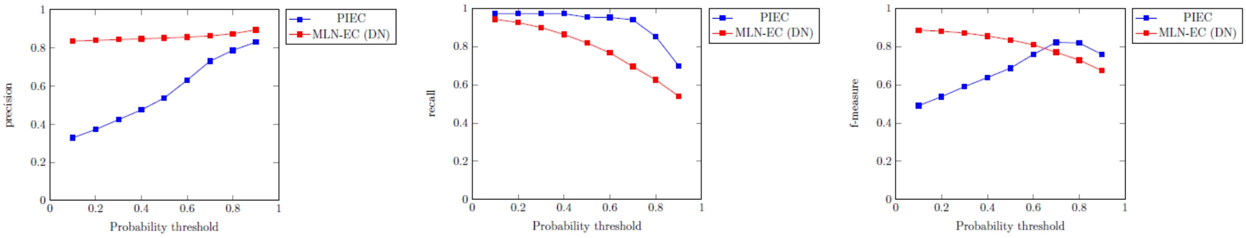

Figure 11: Micro Precision, Recall and F-measure comparison between DN and PIEC on CAVIAR's "moving" LTA

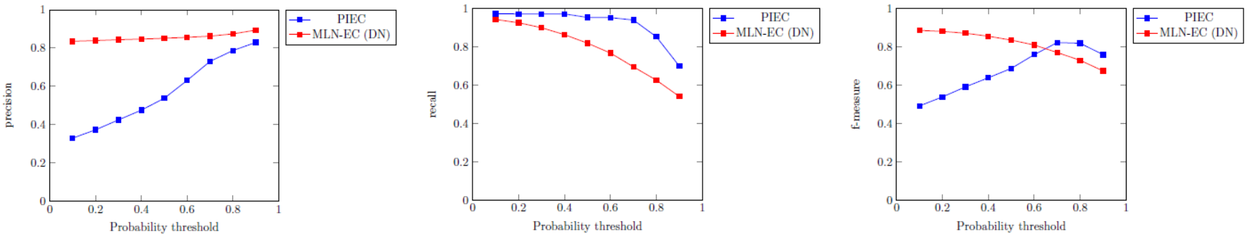

Figure 12: Macro Precision, Recall and F-measure comparison between DN and PIEC on CAVIAR's "moving" LTA

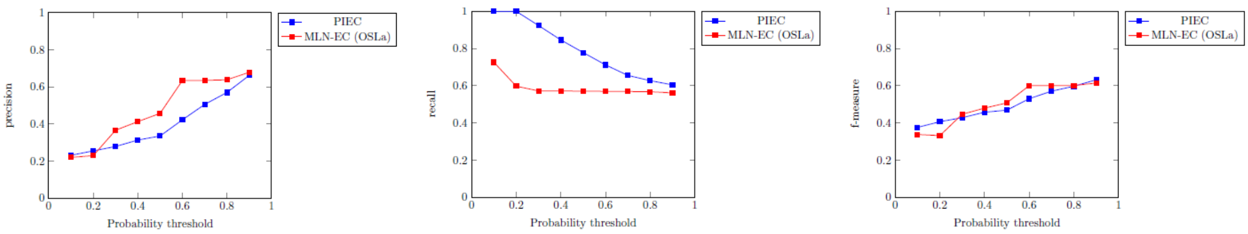

Figure 13: Micro Precision, Recall and F-measure comparison between OSLα and PIEC on CAVIAR's "moving" LTA

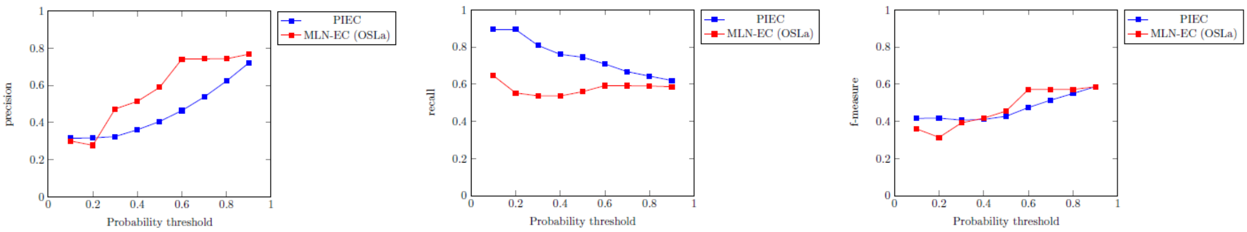

Figure 14: Macro Precision, Recall and F-measure comparison between OSLα and PIEC on CAVIAR's "moving" LTA

In Figures Figure 7 and Figure 8 we observe the comparison between PIEC and DN in terms of micro and macro Precision, Recall and F-measure, respectively. It is evident that, in all occurrences of the “meeting” LTA in the videos of CAVIAR, PIEC misses in Precision, while being superior in Recall. Overall, for threshold values ≥ 0.6 PIEC is better at recognizing the “meeting” LTA.

Figures Figure 9 and Figure 10 show the “meeting” LTA recognition measurements of OSLα and PIEC. The MLN-EC mechanism performs slightly better for low threshold values ($t \leq 0.4$), while for $t > 0.4$ PIEC performs better. On top of that, PIEC achieves the best F-measure of the two methods, for $t = 0.7$. This behavior is explained due to PIEC’s ability to achieve significantly higher Recall, despite the marginal superiority of OSLα in terms of Precision.

Subsequently, we shift focus on the “moving” LTA (see Figures Figure 11 and Figure 12). We ran DN and PIEC on all “moving” occurrences on the CAVIAR dataset. The results show that once again, DN achieves higher Precision, whereas PIEC achieves higher Recall. However, the difference in Precision is great, especially for small threshold values, which makes DN perform much better in small and intermediate threshold values ($t \leq 0.6$). On the other hand, PIEC’s quickly increasing Precision as the thresholds increase, combined with its almost perfect and slowly dropping recall lead to it taking the upper hand for high probability threshold values ($t \geq 0.7$). This behavior is almost exactly the same, both in the micro and in the macro case.

Finally, we repeat the comparison for the “moving” LTA, using the OSLα MLN-based activity recognition method this time. We executed our activity recognition methods on the “moving” cases of the dataset and observed a similar pattern as in the “meeting” LTA case. That is, as Figures Figure 13 and Figure 14 illustrate, the MLN-based activity recognition method achieves higher Precision scores and lower Recall scores, both in the micro and in the macro case. Thus, in terms of F-measure, the two methods display similar scores with OSLα being slightly better for intermediate threshold values ($0.3 \leq t \leq 0.8$) and PIEC being slightly better in extreme threshold values ($t \leq 0.2$ and $t = 0.9$). In addition, the highest F-measure (micro case) is achieved by PIEC, for a threshold value of $t = 0.9$. In the macro case, the highest F-measures for both methods come at $t = 0.9$ and happen to approximately coincide.

5.2.2 Cases where PIEC performs better

The overall comparison presented above gives the impression that neither PIEC nor the MLN-based activity recognition variants are generally dominant. There are LTA and threshold values for which MLN-EC seems to be doing better and other LTA and threshold values for which PIEC seems to be doing better. Thus far, we have abstracted away a lot of information concerning what happens in specific cases. That is, we are interested in investigating at which LTA occurrences does PIEC perform better and at which does it perform worse. To that effect, we compare the credible probabilistic maximal intervals that PIEC produces with the output of MLN-EC and the ground truth. For the completeness of the presentation, we also include the instantaneous probabilities produced by MLN-EC.

In Figures Figure 7: Micro Precision, Recall and F-measure comparison between DN and PIEC on CAVIAR's "meeting" LTA to Figure 14 above, we can observe that for threshold values ≥ 0.6, PIEC displays its best F-measures, outperforming the point-based alternative in all but the last cases. Especially for $t = 0.7$, PIEC’s performance seems either optimized or very close to the optimal (see Figures Figure 7, Figure 8, Figure 9,

Figure 11, and Figure 12). For this reason, we chose to elaborate more on the $t = 0.7$ case and explore PIEC's strengths and weaknesses in that case.

In the meantime, we have used the MLN-EC's output instantaneous probabilities to extract intervals, by taking all maximal sequences of timepoints in which the MLN-EC probability is greater than or equal to t. The following diagrams illustrate the credible probabilistic maximal intervals (i.e. the output of PIEC) in red, the MLN-EC intervals in green, the ground truth in blue, the output instantaneous probabilities of MLN-EC in black and the threshold t in a dashed gray horizontal line. Figures Figure 15 to Figure 26 below present an extensive selection of specific cases within videos of the CAVIAR dataset, where the PIEC algorithm appears to achieve significantly better activity recognition results than a point-based activity recognition method.

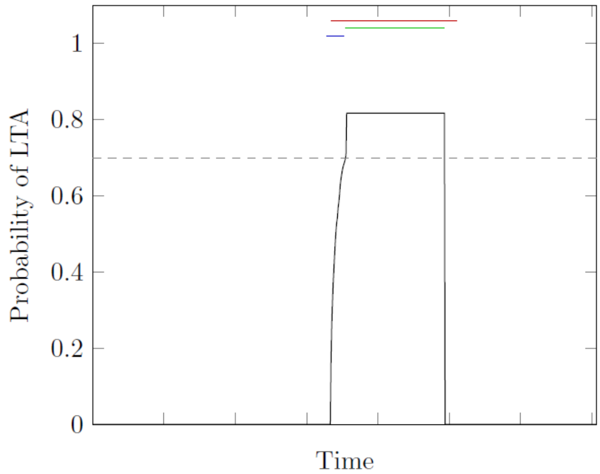

Figure 15: Video 19 – Meeting action between id1 and id2.

Specifically, in Figure Figure 15, we observe a case where DN produces high instantaneous probabilities for the occurrence of the "meeting" LTA outside of the annotated interval, shown in blue. This is an example of erroneous recognition by DN. However, due to its way of calculating probabilistic maximal intervals, PIEC manages to retrieve most of the annotated interval. In other words, PIEC achieves a decent Recall, where DN had a Recall of 0. This comes at the cost of a weak Precision, since PIEC also falsely recognizes the LTA to have occurred long after it has actually ended. This example is a very important one, as it helps us understand that if the underlying timepoint-based recognition method makes an erroneous recognition, PIEC can only improve the result to a certain extent.

Similarly, in Figure Figure 16 DN performs very poorly. Again, its instantaneous probabilities and its respective interval fall entirely outside of the annotated region. PIEC, on the other hand, achieves an excellent Recall, due to the relatively high instantaneous probability values that made the algorithm extend its probabilistic maximal intervals more to the left, without losing much on Precision. This is another example of erroneous timepoint-based recognition, where PIEC manages to significantly reduce the gap between the ground truth and the result of the MLN-based recognition mechanism.

Contrary to the previous two examples, in Figure Figure 17 we observe a more accurate recognition by DN. In the second video of the CAVIAR dataset, persons "id1" and "id2" appear to be moving together and DN manages to recognize this LTA with an excellent

Precision and a decent Recall. Here, PIEC improves the Recall to perfect, at the expense of a little worse Precision. Overall, PIEC does significantly better.

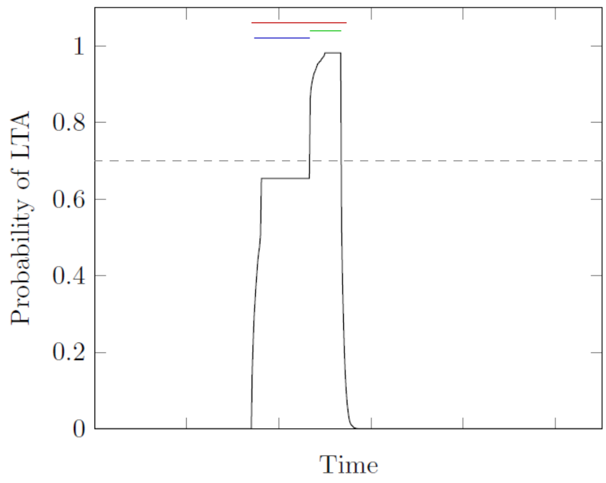

Figure 16: Video 24 – Meeting action between id0 and id1.

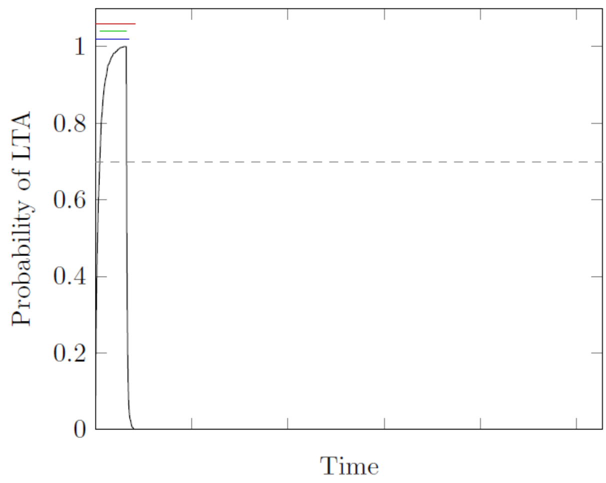

Figure 17: Video 02 – Moving action between id1 and id2.

In Figure Figure 18 we observe a case where the output instantaneous probabilities of DN display abrupt fluctuations. Video 14 from the CAVIAR dataset features two persons, “id1” and “id2” moving together. DN appears to be giving both extremely high and extremely low “moving” probability to timepoints where the LTA under examination is actually taking place. This results to fragmented DN intervals (shown in green) and, consequently, a significant loss on Recall, albeit with an almost perfect Precision. On the other hand, PIEC shows significant tolerance towards these abrupt probability oscillations and helps bridge the discontiguities of the DN intervals. The result is an enormously improved Recall with a slightly weaker Precision. Note that there is also another actual occurrence of the “moving” LTA that neither DN nor PIEC were able to detect.

In video 23 of the dataset, a group of people meets and moves together. There are two LTA that are of particular interest, namely “moving” between “id0” and “id1” (Figure Figure 19 – on the left) and “moving” between “id0” and “id3” (Figure Figure 19 – on the right). In the former case, we observe some probability fluctuations that yield multiple, discontiguous DN intervals and, hence, a lot of False Negatives and a loss in terms of

Recall. As we have already seen in a similar case, in Figure Figure 18, PIEC is able to bridge the recognition gaps and produce a single interval, that covers the entire ground truth (that is, it achieves 100% Recall) while at the same time sacrificing its Precision. Overall, PIEC performs better than DN, because the gain of Recall is greater than the loss of Precision. In the latter case, DN recognizes a sub-interval of the actual one, which means that DN achieves a perfect Precision, but a not so good Recall, as it only recognizes about half of the actual LTA occurrence stint. On the other hand, PIEC, led by the high, consecutive instantaneous probability values, produces an extended probabilistic maximal interval that corresponds to a much higher Recall, while preserving the perfect Precision.

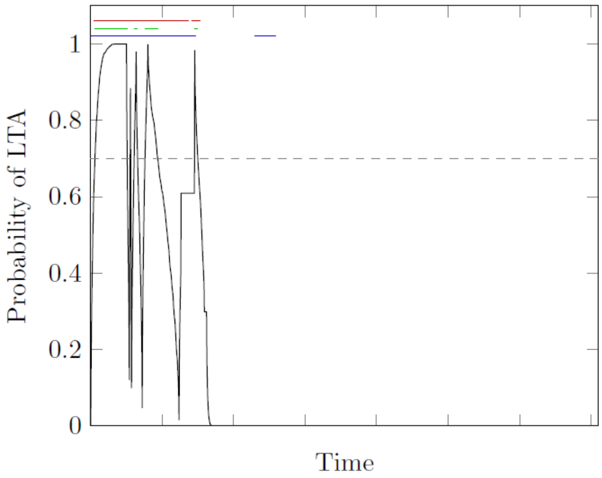

Figure 18: Video 14 – Moving action between id1 and id2.

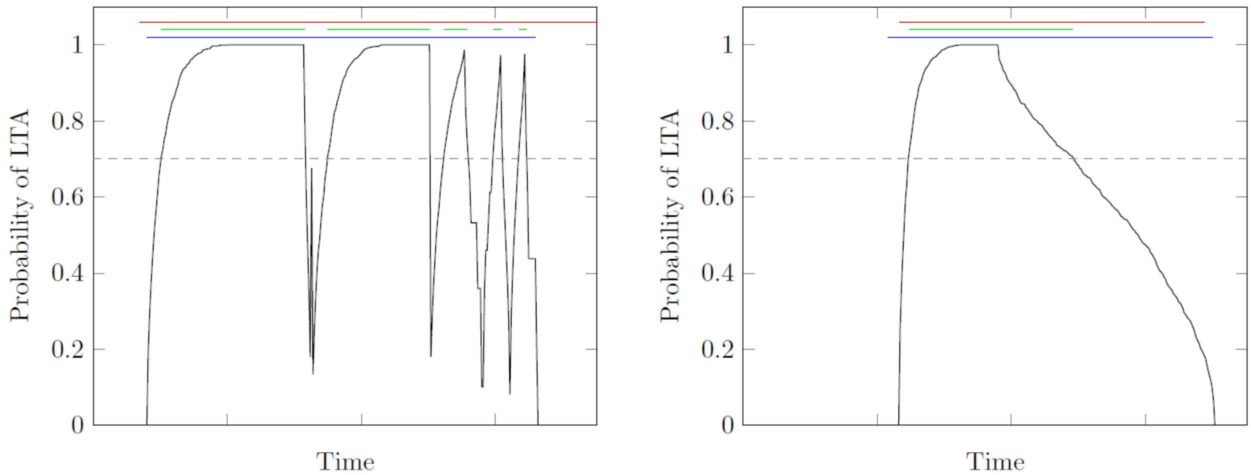

Figure 19: Video 23 – Moving action between id0 and id1 (Left). Moving action between id0 and id3 (Right).

Moving on to results from OSLα, Figure Figure 20 shows another case of heavy probability fluctuations. Again, we are confronted with fragmented OSLα intervals that cost a lot in terms of Recall. As with the previous two cases, PIEC is able to overcome this issue, due to the high instantaneous probability values that promote the recognition of longer intervals. Thus, PIEC once again performs better, by enormously improving the Recall, while keeping the perfect precision intact.

In Figure Figure 21, the “meeting” activity between entities “id1” and “id2” in the second video of the dataset leads OSLα to produce instantaneous probabilities that display very frequent, intense fluctuations. Hence, the OSLα intervals are heavily fragmented. PIEC, on the other hand, is fluctuation-tolerant and produces a long interval that includes the ground truth in its entirety. This results in PIEC achieving 100% Recall, albeit losing on Precision. Overall, PIEC performs better.

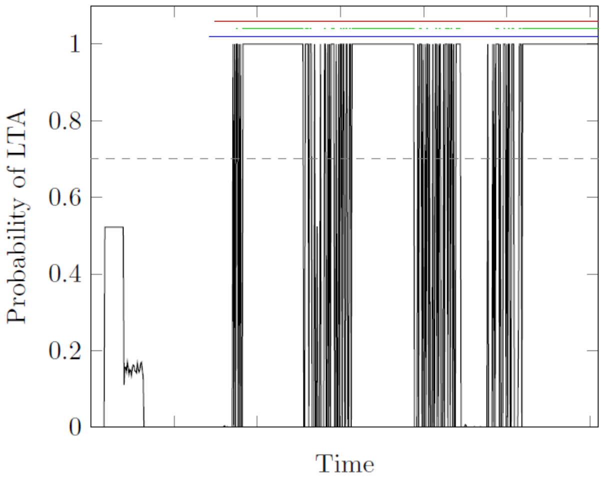

Figure 20: Video 01 – Meeting action between id4 and id5.

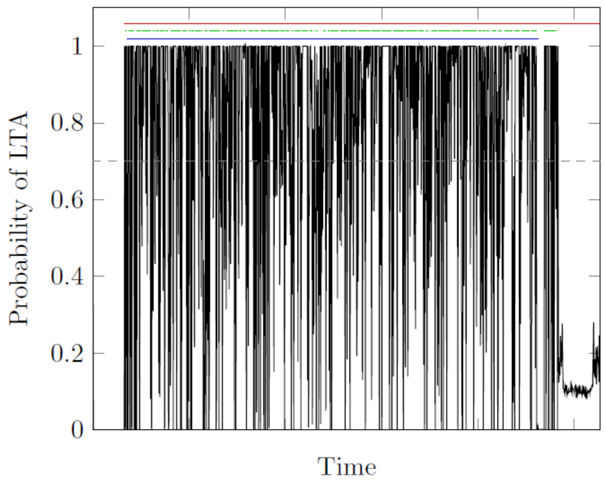

Figure 21: Video 02 – Meeting action between id1 and id2.

The instantaneous OSLα probabilities of Figure Figure 22 fluctuate heavily around the threshold, thus creating numerous, very short intervals where the OSLα probability is above the threshold. Since the probabilities do not far exceed the threshold, PIEC’s tolerance diminishes, making it fluctuate, as well – although less frequently than OSLα. Overall, PIEC performs better, with significantly more True Positives and fewer False Negatives, at the expense of a few more False Positives.

If we take a look at Figure Figure 23 we will once again observe heavy instantaneous probability fluctuations for OSLα. However, PIEC seems unaffected and much more stable than in the previous case. This happens because the upper edges of the oscillations far exceed the threshold (probabilities are close to 1) and, as follows from

Definition 1, PIEC produces longer intervals. This leads to much fewer False Negatives and hence a significantly better Recall, without affecting the Precision.

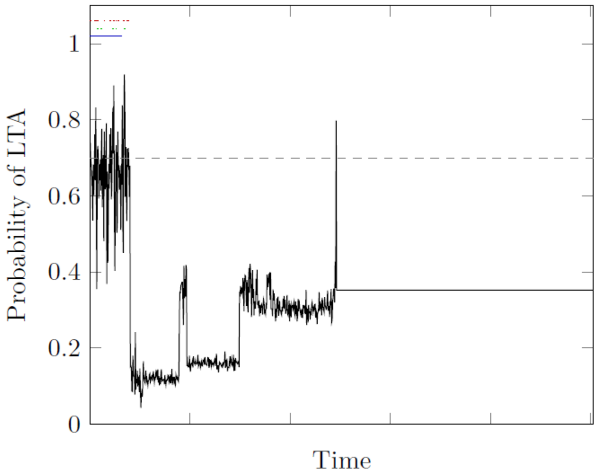

Figure 22: Video 13 – Meeting action between id0 and id1.

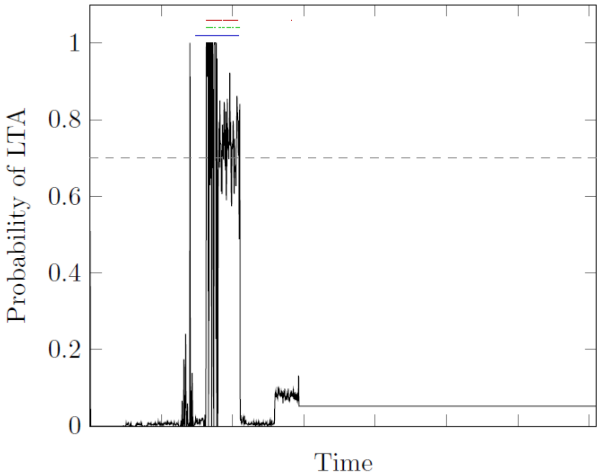

Figure 23: Video 14 – Meeting action between id1 and id2.

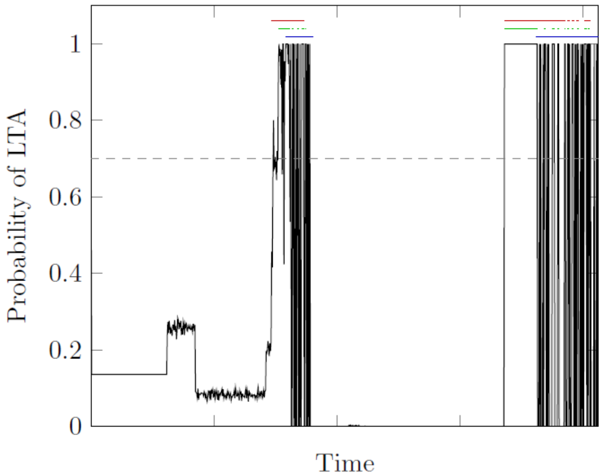

Figure 24: Video 20 – Meeting action between id1 and id2.

In video 20 of the dataset, entities “id1” and “id2” perform the “meeting” activity twice: Once around the middle of the video and once more at the end (Figure Figure 24). OSLα starts giving probabilities that heavily fluctuate from 1 to 0 around the first “meeting” occurrence. The recognition is not precise, as the OSLα intervals are many, small and discontiguous, and some of them lay outside of the annotated interval. Near the end of the video, the MLN-based algorithm erroneously gives a series of consecutive probabilities equal to 1, just before the start of the annotated interval. During the actual second occurrence of the “meeting” activity, OSLα again displays this severely oscillating behavior, leading to numerous, small and discontiguous intervals and producing a lot of False Negatives. In this case, the results of OSLα cost both in terms of Precision and in terms of Recall, as there are both False Positives and False Negatives. PIEC, as in most cases seen so far, manages to significantly increase the Recall without doing much harm in terms of Precision. Therefore, PIEC performs better and helps ameliorate some of the imprecisions of OSLα.

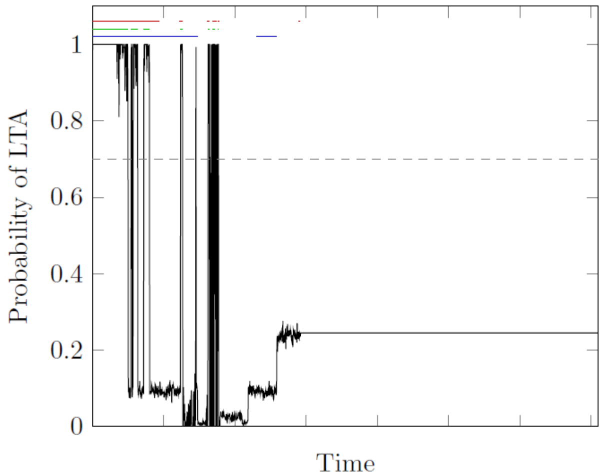

Figure 25: Video 14 – Moving action between id1 and id2.

In Figure Figure 25 we can see the “moving” activity between persons “id1” and “id2”, from the fourteenth video of the CAVIAR dataset. There are two “moving” stints: a longer one taking place at the beginning of the video and a shorter one taking place later, towards the middle of the video. Although initially OSLα shows stability and correctly recognizes the LTA, it soon starts producing oscillating probabilities both within and outside the first annotated stint. Both PIEC and OSLα appear to have similar, good Precision measures, but PIEC achieves a better Recall. However, none of the two methods recognizes the second “moving” stint.

We now return to video 23, where several people gather and move together, and focus on the “moving” activity between persons “id0” and “id2” (Figure Figure 26 – on the left), as well as persons “id0” and “id3” (Figure Figure 26 – on the right). In the first case, there is a long stint where the activity under examination actually takes place, shown in blue. OSLα correctly recognizes the first half of the stint, but then it starts producing abruptly fluctuating probabilities that drop as low as 0.2 and cause it to miss the majority of the second half of the stint. PIEC, on the other hand, shows better stability, despite it failing to recognize about a quarter of the actual LTA. On top of that, PIEC also produces some False Positives, as well. In other words, OSLα displays an excellent Precision, but a weak Recall, whereas PIEC loses in terms of Precision, but achieves a significantly better Recall. Overall, PIEC achieves a better F-measure than OSLα. In the second case, the actual "moving" stint is again long, spanning more than half the total duration of the video – that is the full length of the horizontal axis. OSLα seems to only recognize small fragments of this activity, again due to abruptly fluctuating output. The probabilistic maximal intervals of PIEC are discontiguous, as well, albeit fewer and longer. Both methods are perfectly precise, but they appear to be very weak in terms of Recall. Overall, PIEC performs slightly better, as it recognizes a slightly bigger part of the actual LTA occurrence.

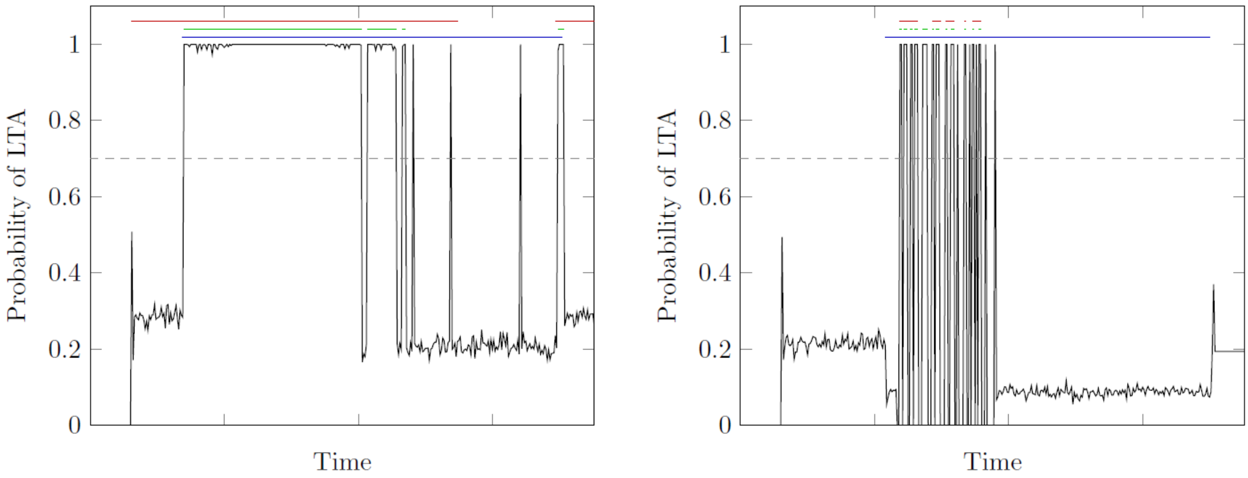

Figure 26: Video 23 - Moving action between id0 and id2 (Left). Moving action between id0 and id3 (Right).

Summary

To sum up, we observe that PIEC appears to be outperforming its point-based competition in cases of erroneous or incomplete recognition by the latter. When the intervals derived from the instantaneous probabilities calculated by MLN-EC fall partly or entirely outside of the corresponding annotated intervals, PIEC manages to achieve bigger overlaps, thus resulting in better Recall scores and overall better F-measures. PIEC also seems to be the preferred option in cases where the estimation of MLN-EC is precise but incomplete. That is, in cases where the intervals derived from point-based MLN-EC estimations fall within the annotated region but are fragmented due to abrupt probability fluctuations. PIEC then tends to produce more coherent intervals, with fewer and longer fragments, thus achieving an overall better performance.

5.2.3 Cases where PIEC performs worse

We continue the experimental evaluation analysis, by presenting several interesting cases where PIEC performs worse than the MLN-EC variations. Figures Figure 27 to Figure 40 display the same kind of information as the figures we have seen in the previous subsection with the only difference that in some particularly interesting cases, we also include the full list of probabilistic maximal intervals, among which PIEC had to choose the most credible. These extra intervals are shown in orange in the graphs below. This information will provide an extra insight into the cases where PIEC is outperformed and help us see the reasons behind this lack of performance. We will be able to see the role that the credibility mechanism plays and if the credibility definition appears to be suboptimal, we will try to define interval credibility anew.

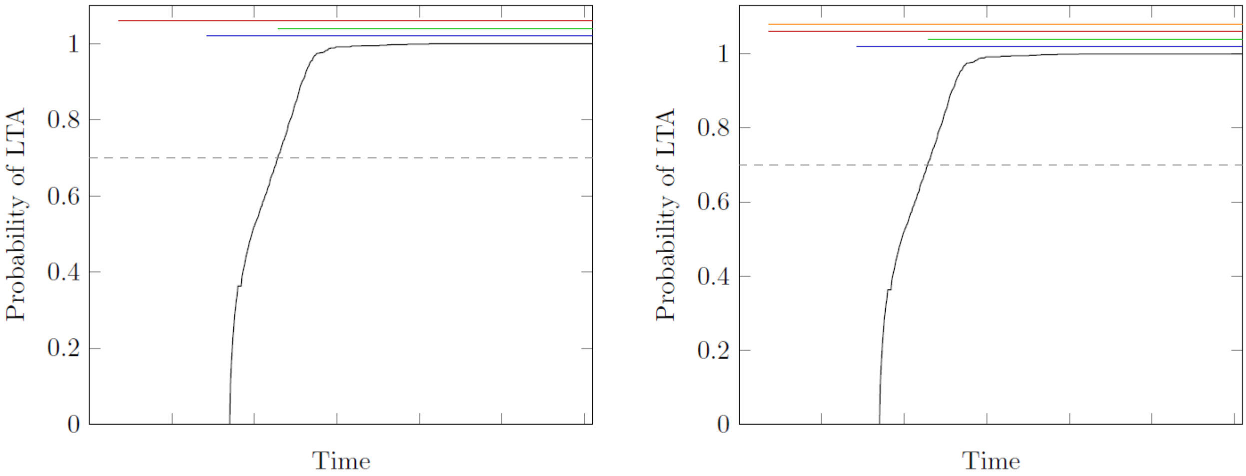

Figure 27: Video 01 – Meeting action between id4 and id5.

We begin this second part of our analysis with the recognition of the “meeting” LTA between entities “id4” and “id5”, from the first video of the dataset. The instantaneous probabilities are produced by the DN method and can be seen in Figure Figure 27, on the left. DN appears to be absolutely precise but misses a significant part from the beginning of the actual activity. On the other hand, PIEC achieves 100% Recall, but its precision is not perfect. More specifically, PIEC starts recognizing the activity long before it actually begins. This lack of Precision causes PIEC to achieve a weaker F-measure than DN in this case. On the second part of this figure, on the right, we can see another interval (in orange) that entirely coincides with the output of PIEC. This means that PIEC had only one candidate probabilistic maximal interval to pick as the most credible. Therefore, in this case there was nothing better the algorithm could do, at least in terms of credibility.

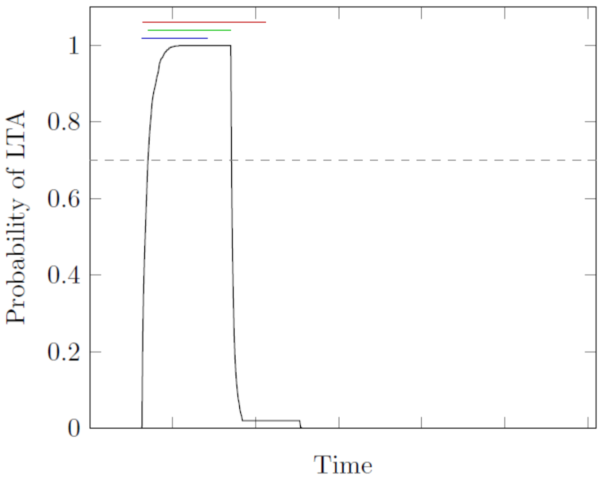

Figure 28: Video 01 – Moving action between id4 and id5.

In Figure Figure 28, we can see another case of PIEC underperforming. Again from Video 01, this time we are interested in recognizing the “moving” LTA between “id4” and “id5”. The DN method seems to recognize the LTA with a small delay. Thus, there is a part at the beginning of the LTA that DN fails to recognize, as well as a part after the end of the actual LTA, where DN keeps recognizing it, despite the fact that it has been terminated.

PIEC manages to increase the Recall to almost perfect, but it also increases the falsely recognized part after the end of the actual LTA. This behavior leads to a much weaker Precision and a weaker F-measure for PIEC.

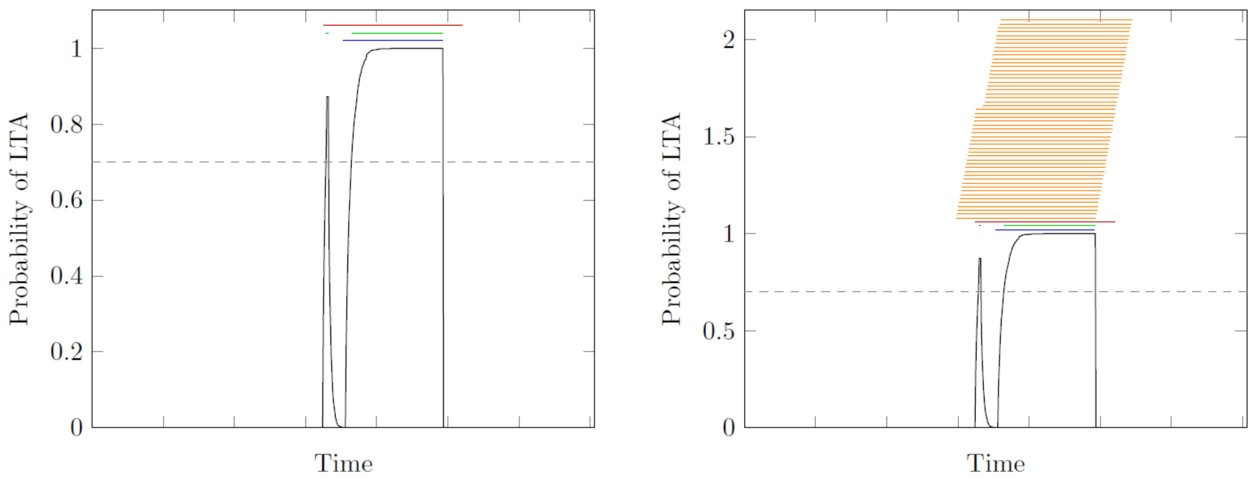

Figure 29: Video 19 – Moving action between id1 and id2.

On the left part of Figure Figure 29, we see the “moving” LTA recognition results using the DN method for the moving activity of entities “id1” and “id2”, in video 19. DN appears to approximate the ground truth well, with an almost perfect Precision and a very good Recall, as it manages to cover almost the entire ground truth interval. PIEC does not match the performance of DN. On the contrary, it yields many more False Positives, substantially undermining its Precision. Its perfect Recall is not enough to make up for the loss of Precision and, therefore, DN outperforms PIEC in terms of F-measure. On the right hand side of Figure Figure 29 we can see that, contrariwise to what we have seen in Figure Figure 27, there are numerous overlapping probabilistic maximal intervals from which PIEC has to choose the best fitting one, i.e. the most credible one. If we look at the candidate intervals more carefully, we can see that they form two groups: a group of longer intervals at the bottom half of the stack and a group of slightly shorter intervals at the top of the stack. The credibility mechanism of PIEC has chosen the latest among the longer intervals, that is the interval that lies at the top of the longer interval group. However, this is not the optimal choice. Since PIEC has a tendency of producing longer intervals when dealing with instantaneous probability values close to or equal to 1, sometimes it runs the risk of producing intervals way longer than needed and it, inevitably, suffers in terms of Precision. In this particular example, PIEC could have chosen from the group of shorter intervals (for instance, the earliest of them), which would lead to slightly fewer False Positives and, thus, to a slightly better Precision. In other words, this is a case where the credibility mechanism does a suboptimal job for PIEC.

In the twentieth video, there is the scenario of two people – namely “id1” and “id2” – performing the “moving” LTA. The recognition results are shown in Figure Figure 30. In this case, which is quite similar to the previous one, DN approximates very well the actual LTA occurrence. In fact, it only produces a handful of False Negatives and its Precision is perfect. PIEC once again suffers from excessive recognition, which severely undermines its performance, despite its perfect Recall.

In the next video, possibly among other LTA, we have the “moving” LTA between “id1” and “id3”. This is a case where none of our probabilistic activity recognition methods does well. Both DN and PIEC produce a surplus interval. However, as it can be seen in Figure

Figure 31, the output of PIEC is even longer than that of DN, which means that there are much more False Positives and, therefore, a much lower Precision.

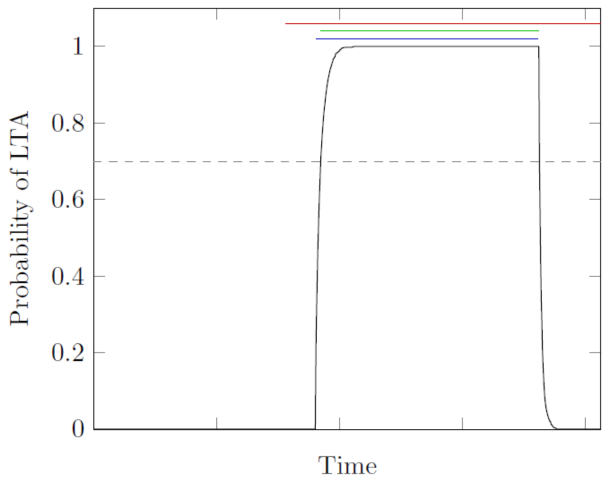

Figure 30: Video 20 – Moving action between id1 and id2.

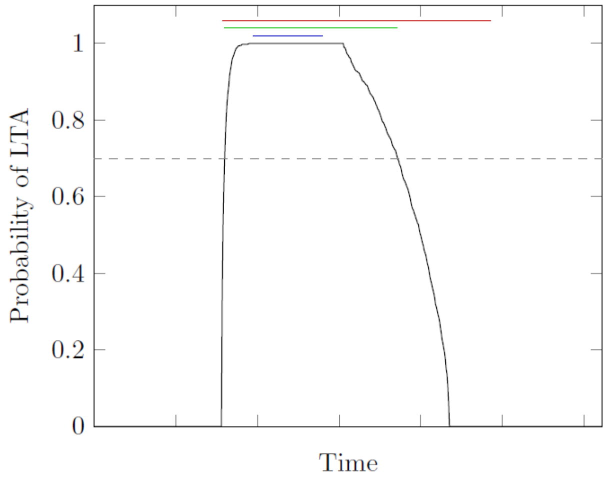

Figure 31: Video 21 – Moving action between id1 and id3.

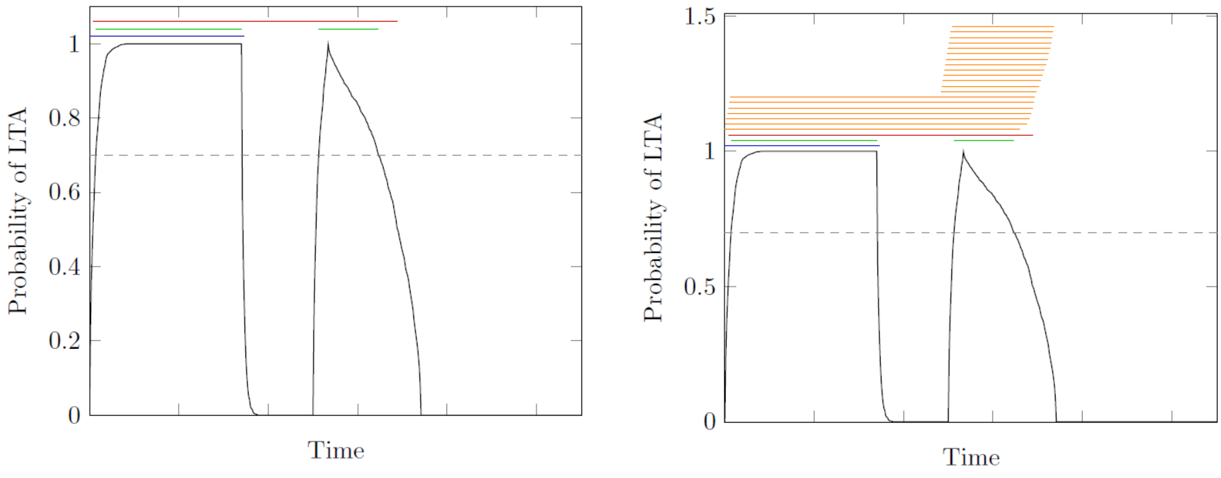

Figure 32: Video 24 – Moving action between id0 and id1.

On the left part of Figure Figure 32 we see the recognition results for the “moving” activity between “id0” and “id1” that takes place in the twenty-fourth video of the dataset. As far as the MLN-based activity recognition mechanism is concerned, we observe two recognized stints: one within the annotation and one outside the annotation. Although the first interval approaches very well the respective annotation, the existence of the second, false recognition damages the precision of DN. PIEC once more produces an excessive recognition stint, that includes both DN intervals, along with several other False Positives. On top of that, in this case PIEC also misses the very beginning of the actual occurrence of the LTA, which means that neither its Recall is perfect. On the right part of the figure, we can see all the candidate probabilistic maximal intervals of PIEC. We can observe that all candidate intervals are way longer than needed. This is due to the existence of falsely assigned very high probability values after the LTA is over. There is even a group of probabilistic maximal intervals that focuses solely on the falsely recognized region. However, PIEC does not make the optimal choice. There is an interval – the earliest one, on the bottom of the stack shown in orange – that begins from timepoint 0 and is slightly shorter than the subsequent 6. Had PIEC chosen this interval, it would have avoided all False Negatives and it would have slightly reduced its False Positives, thus leading to the improvement of both its Precision and its Recall. Therefore, we have come across one more case where the credibility mechanism leads to a suboptimal result, the other case having been discussed in Figure Figure 29.

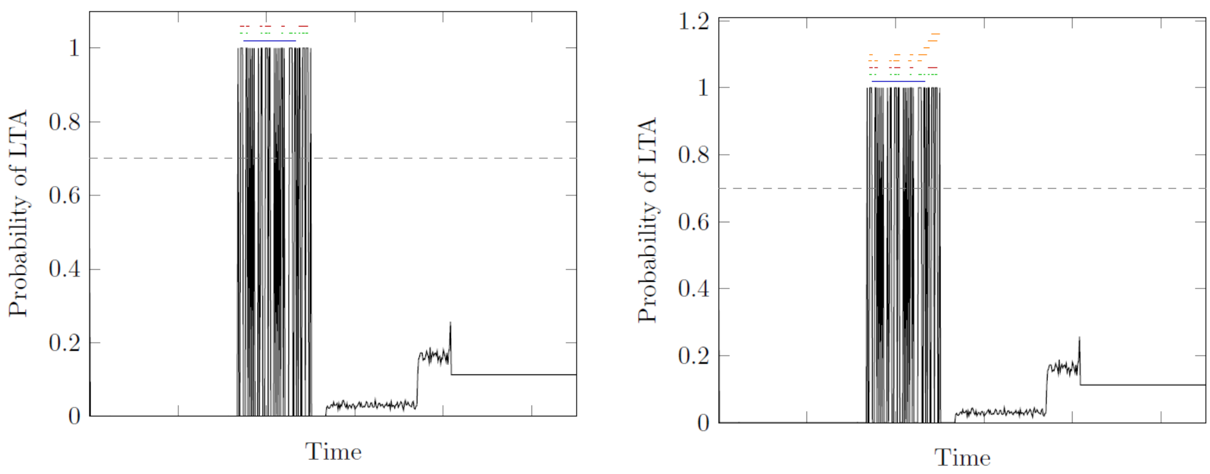

Figure 33: Video 24 – Meeting action between id0 and id1.

The remainder of this section features results from the OSLα MLN-EC variation. In Figure Figure 33 we can see the recognition results for the “meeting” activity between “id0” and “id1”, from video 24. On the left of the figure, we observe intense probability oscillations, around the annotated interval. These oscillations are translated into several, very small intervals, lying before, during and after the annotation. The output of PIEC seems fragmented, as well, with the only difference that the intervals are generally longer than those of OSLα. This behavior corresponds to fewer False Negatives, but also more False Positives. The longest interval that PIEC produces is outside of the annotated region. Therefore, the loss in terms of Precision is greater than the gain in terms of Recall and, thus, OSLα achieves a higher F-measure than PIEC. On the right of the figure, we can see multiple stacks of candidate, overlapping probabilistic maximal intervals for PIEC, delineated in orange. There is one such stack per output interval. Recall that the credibility mechanism finds groups of overlapping intervals and for every interval in such a group it calculates its credibility and then chooses the most credible among them. The rightmost stack comprises five probabilistic maximal intervals. One of them (the earliest) is entirely inside the annotated area, the second earliest is partially inside the annotated area and the rest lie outside the annotated area. As it can be observed from the figure, PIEC chooses the penultimate of this group of intervals, which is entirely outside the annotation. This is clearly not the optimal choice, as it could have chosen the earliest of the group, which would have led to significantly fewer False Positives and slightly fewer False Negatives, which in turn corresponds to a better Precision and Recall. The credibility mechanism appears to once again undermine the performance of PIEC.

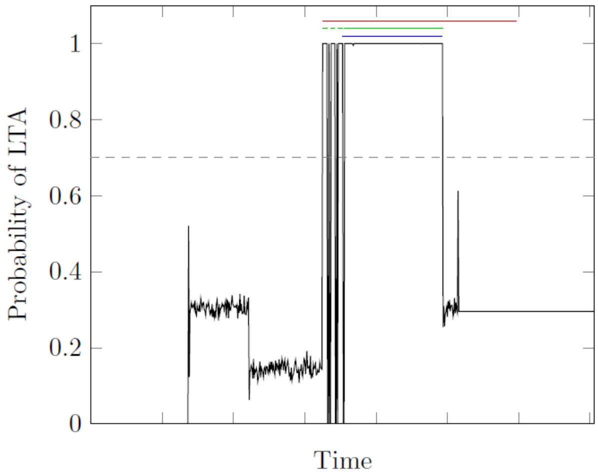

Figure 34: Video 19 – Moving action between id1 and id2.

In Figure Figure 34 we see the response of OSLα and PIEC to the task of recognizing the “moving” activity between entities “id1” and “id2”, from video 19. OSLα assigns high probabilities with occasional abrupt fluctuations to an area somewhat larger than and fully containing the annotation. Eventually, the MLN-based method achieves an almost perfect Recall and a decent Precision. PIEC, on the other hand, calculates a much longer interval that includes both the annotation and the intervals of OSLα. This may result to an absolutely perfect Recall but costs a lot on Precision. This has appeared to be the case in many examples so far. Recognizing excessively long intervals is unsolicited behavior and seems to be the main drawback of PIEC. To sum up, OSLα heavily outperforms PIEC in terms of Precision and PIEC marginally outperforms OSLα in terms of Recall. Hence, overall, OSLα does better.

In Figure Figure 35 we see a case where the MLN-based method achieves an almost perfect F-measure. Indeed, it assigns probabilities equal to 1, without abrupt fluctuations exactly upon the annotated region. That gives excellent Precision and Recall measures and an excellent F-measure, in total. Our interval-based activity recognition method shows once again its usual tendency of producing oversized intervals when the instantaneous probabilities involved are contiguous and much higher than the threshold, thus severely damaging its Precision. Based on what we have seen so far one might be wondering if there is any alternative probabilistic maximal interval the choice thereof as most credible would yield a better Precision and F-measure. The answer to this question can be found in the right-hand diagram of Figure Figure 35. There clearly is just one possible probabilistic maximal interval and PIEC can do nothing other than consider it credible and giving it to the output.

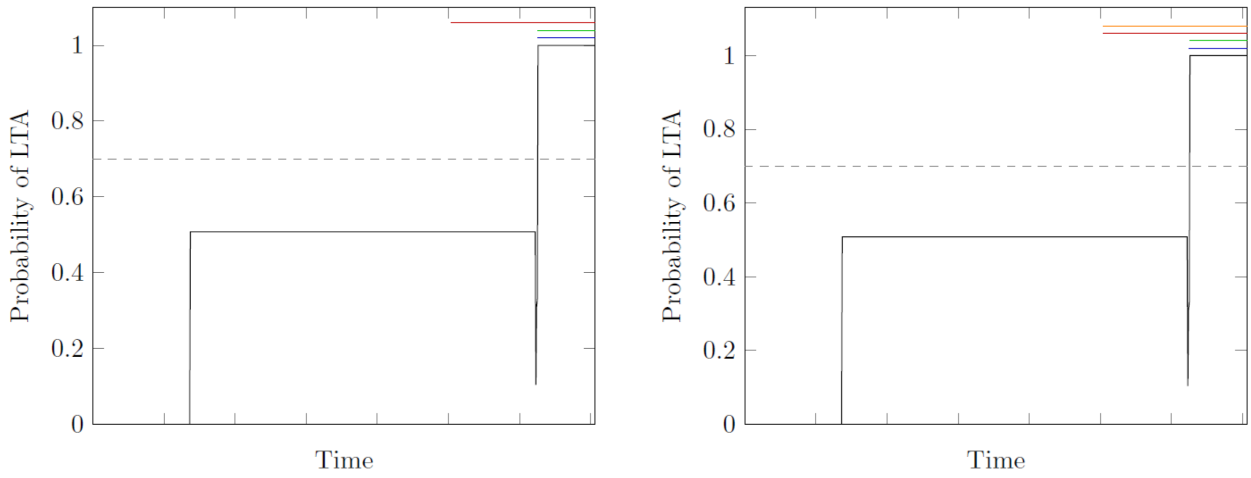

Figure 35: Video 19 – Moving action between id5 and id6.

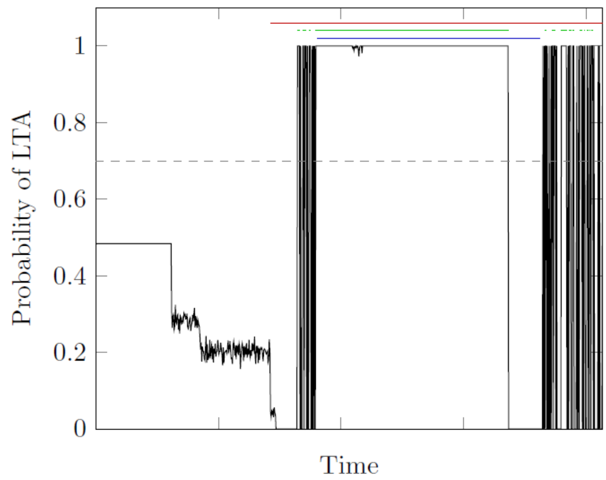

Figure 36: Video 20 – Moving action between id1 and id2.

In a case similar to that seen in Figure Figure 34, we examine the recognition results for the "moving" LTA between "id1" and "id2" that takes place in video 20 of the dataset (see Figure Figure 36). Here, OSLα starts by making some abrupt probability fluctuations before the start of the annotated interval and then produces a sequence of very high probabilities (close or equal to 1), during most of the annotation. At some point, towards the end of the annotation, it erroneously stops recognizing the LTA, only to produce a new series of oscillating probabilities, after the annotation interval has ended. The large concentration of probabilities close to 1 – contiguous or not – leads PIEC to produce an excessively long probabilistic maximal interval, that may incorporate the entire ground truth, but also give numerous False Positives. The gain of PIEC in terms of Recall is definitely smaller than its loss in terms of Precision. Therefore, the MLN-based activity recognition method performs better.

Figure Figure 37 shows a rather odd recognition example. Specifically, in video 21, entities "id1" and "id3" actually perform the "moving" LTA during a relatively short stint, shown in blue. Nevertheless, OSLα erroneously assigns high instantaneous probabilities (around 0.8, with mild fluctuations) for the "moving" LTA, long before the actual activity begins. Then, around the actual occurrence of the LTA, the MLN-based recognition mechanism starts producing probability values, heavily oscillating between 0 and 1. Several timepoints after the end of the annotation, OSLα seems to stabilize again around

0.8 for quite some time, before dropping to below 0.5. This way, we obtain falsely recognized, contiguous intervals outside the annotation and intense fragmentation during the annotation. This combination corresponds to poor performance for OSLα, with a severely damaged Precision and a poor Recall. PIEC, following its tendency to produce longer intervals, it manages to cover the entire second half of the annotation, significantly reducing False Negatives, but also Resulting PIEC intervals further expand earlier and later than the actual LTA occurrence, thus giving many extra False Positives. In other words, both methods suffer from many False Positives. Even though PIEC achieves a better Recall, OSLα appears to behave better, overall.

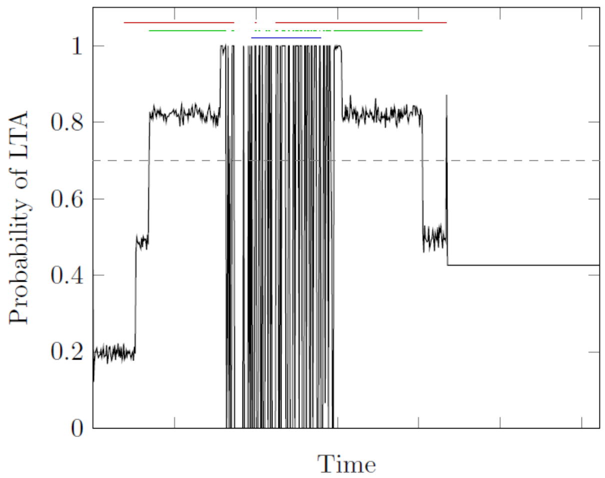

Figure 37: Video 21 – Moving action between id1 and id3.

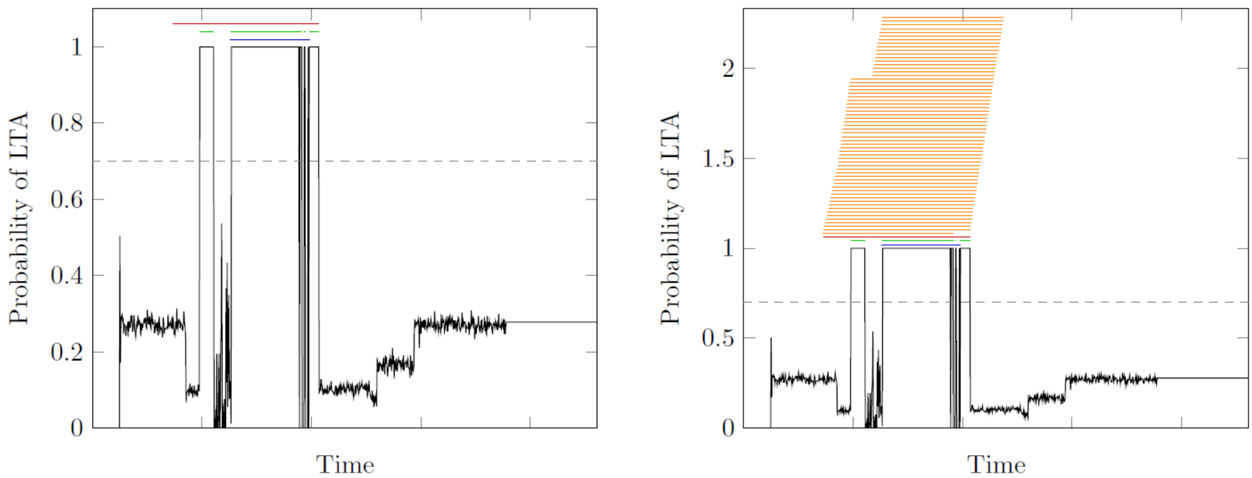

Figure 38: Video 22 – Moving action between id0 and id1.

Figure Figure 38 shows the recognition results for the “moving” action between entities “id0” and “id1”. Similarly to many of the cases seen so far, OSLα recognizes the LTA fairly accurately. We can see that OSLα produces three sequences of stable probability value equal to 1, plus some 0-to-1 oscillations, between the second and third sequences. Only the second sequence along with the oscillations correspond to the ground truth. The other two high probability sequences are false recognitions. In total, there are some False Positives before and after the annotation and a few False Negatives in the oscillating stint, towards the end of the annotated region. PIEC shows its usual behavior of constructing excessively long intervals that are able to achieve 100% Recall at the expense of many False Positives and poor Precision. Hence, OSLα performs better. On the right-hand side of the figure we can see a large group of overlapping probabilistic maximal intervals, depicted in orange, from which PIEC has to choose the most credible. Just like in Figure Figure 32, there are sub-groups of different length. The length of these intervals depends on the sequences of high instantaneous probabilities that lie underneath. Specifically, the earliest of the overlapping intervals – that is, the one that lies at the bottom of the stack, is created by OSLα's first and second high probability sequences. Above it, there is a subgroup of intervals that are affected by the entire probabilistic activity of the region. As it can be seen in the figure, PIEC's credibility mechanism has chosen the earliest interval of this subgroup as the most credible. The subgroup of shorter intervals that appear at the top of the figure are affected by the second and third high probability stints of OSLα. PIEC's choice appears to be again suboptimal. There are intervals in the top subgroup that could lead to much fewer False Positives and, hence, to a significantly increased Precision. In conclusion, this is another case where the credibility mechanism does not produce the optimal result.

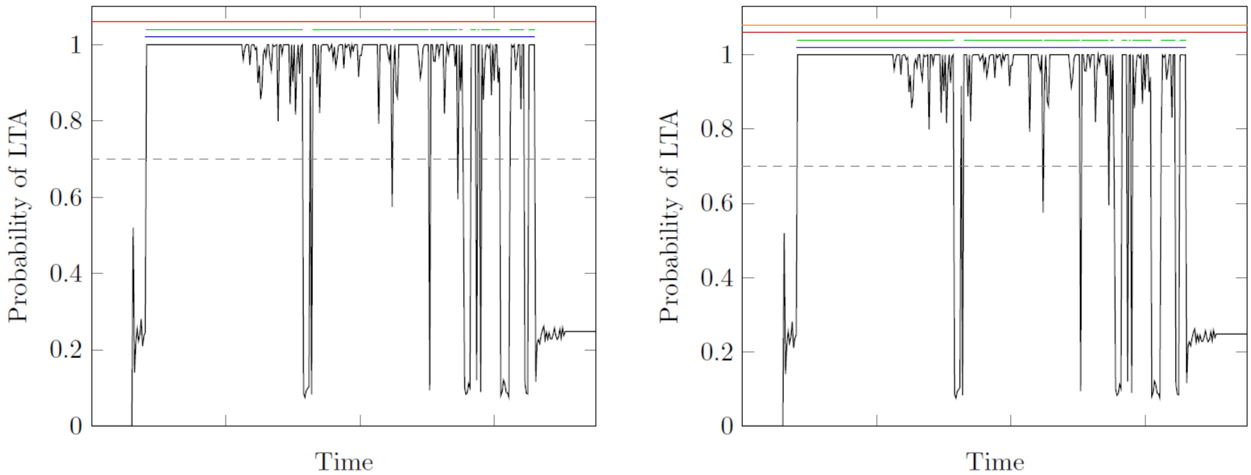

Figure 39: Video 23 – Moving action between id0 and id1.

Finally, in video 23 of the CAVIAR dataset, we focus on the "moving" activity between entities "id0" and "id1", as well as "id1" and "id3". The results of the recognition of these LTA is illustrated in Figures Figure 39 and Figure 40, respectively. More specifically, for the former case, OSLα seems to detect the LTA fairly accurately, as is produces high instantaneous probabilities throughout the annotated interval, albeit with some – milder or more abrupt – fluctuations. The more abrupt fluctuations lead to interruptions in the resulting OSLα intervals (see the green lines in Figure Figure 39). This means that, in spite of achieving a perfect precision, as there are no False Positives, its Recall is not Perfect. On the contrary, PIEC shows stability as it seems unaffected by the fluctuations of OSLα, but it produces a lot of False Positives. The high amount of probabilities close or equal to 1, leads PIEC to compute one, very long interval, which in fact spans the entire video. This way, PIEC achieves a perfect Recall, but its Precision is nowhere near the Precision of OSLα. On the whole, OSLα reaches a better F-measure than PIEC. Moreover, if we take a look at the right hand part of Figure Figure 39, we observe that PIEC has no means for improving its performance in this specific case, as there are no alternative probabilistic maximal intervals that might lead to improved measurements. In the graph, there is just one orange line, which is identical to the red line. This means that there is just one credibility contender, which is inevitably picked and shown in the output.

This case is similar to those appearing in Figure 27 and Figure 35, where there is only one credibility contender.

For the latter case, entities “id1” and “id3” perform the “moving” LTA. It can be observed in Figure Figure 40 that OSLα fails to recognize the majority of the LTA, despite the fact that, for the part that it does recognize, it is absolutely precise. As far as our interval-based activity recognition method is concerned, it starts recognizing the LTA prematurely and it stops the recognition at the same instant as OSLα does. This behavior leads to an even worse performance for PIEC. Nonetheless, in the right subfigure we can see a large stack of overlapping credibility contender intervals, in orange color. The credibility mechanism of PIEC chooses the earliest one (that is, the one at the bottom of the stack) as the most credible. The problem here is that the chosen interval is the interval with the smallest overlap with the ground truth. Since all intervals from the stack appear to be of equal size, then each subsequent interval should have a longer overlap with the ground truth and a shorter premature recognition stint. Therefore, it seems that the choice that would maximize the F-measure of the algorithm is the latest among the overlapping intervals (that is, the one at the top of the stack). In other words, the credibility mechanism of PIEC has made not only a suboptimal, but rather the worst possible choice.

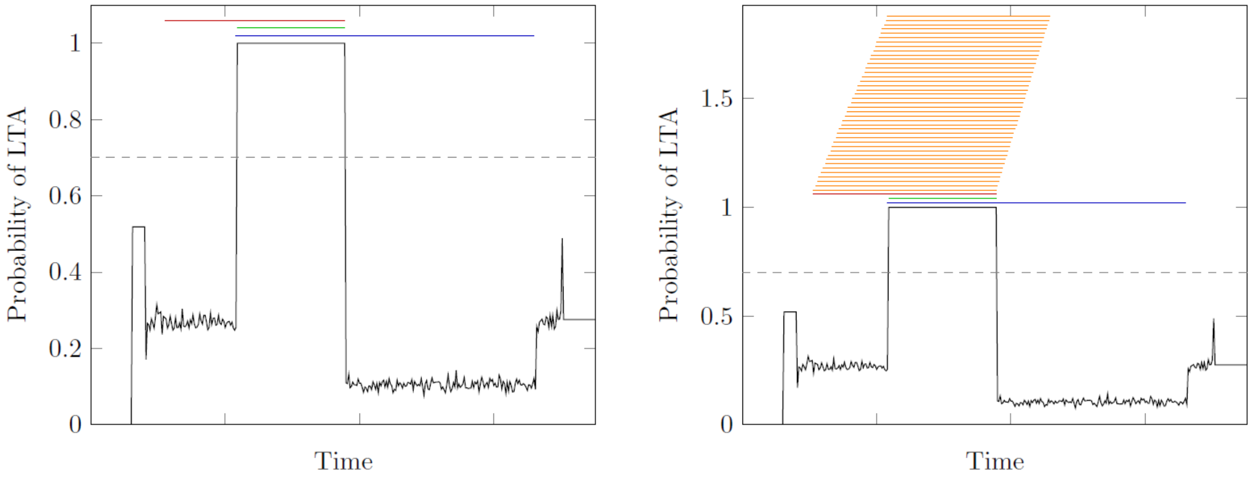

Figure 40: Video 23 – Moving action between id1 and id3.

Summary

To conclude, PIEC seems to be underperforming in the case of long sequences of very high (i.e. near 1) instantaneous probabilities. Due to the way interval probabilities are computed (recall Definition 1), the length of an estimated interval is directly proportional to the sum of the instantaneous probabilities and inversely proportional to the threshold value, $\mathcal{T}$. Long sequences of very high probabilities lead to big probability sums which, when combined with relatively low probability thresholds, in turn lead to very long output intervals. In almost all of the cases presented in this section, PIEC suffers from a severe loss of Precision, due to the existence of many False Positives, as a result of excessively long output intervals. Even though PIEC may still achieve better Recall scores, the loss in terms of Precision is big enough to overshadow any Recall improvements, leading to an overall inferior performance compared to the plain threshold approach.

5.3 Alternative credibility definitions for PIEC

In paragraph 2.4.2 we discussed the definition of credibility and presented the credibility mechanism step-by-step. Since maximal probabilistic intervals may be overlapping, the PIEC algorithm incorporates a routine for picking one – the most "credible" – amongst these overlapping maximal intervals. In PIEC's original version, the credibility of an interval was defined as the product of its probability times its length. From definition (14), this is equivalent to the sum of its instantaneous probabilities. We are now experimenting with different definitions for credibility. In this report, we present five new credibility definitions, namely:

1. "pick first": Among a group of overlapping probabilistic maximal intervals, pick the earliest as the most credible one.
2. "pick last": Among a group of overlapping probabilistic maximal intervals, pick the latest as the most credible one.
3. Define credibility as the average of the instantaneous probabilities, i.e.: $credibility = probability$.
4. Define credibility as the sum of the squares of the instantaneous probabilities.
5. Define credibility as the sum of the cubes of the instantaneous probabilities.

This credibility definitions list is by no means exhaustive. These definitions are indicative and are mostly based on intuition. For instance, the "pick first" strategy would improve PIEC's performance in cases like the one seen in Figure Figure 33, the "pick last" strategy would result in PIEC making the best possible choice in the case of Figure Figure 40, and strategies like the "sum of squares" and the "sum of cubes" aim at rewarding an interval that contains very high instantaneous probabilities by giving it a higher credibility value compared to a – possibly longer – interval with lower instantaneous probabilities. There is an abundance of other possible credibility definitions and this is a direction for further work, as it will be discussed in the respective section later.

The experiments whose results are presented below have been conducted with view to testing whether these new credibility definitions for PIEC can increase its performance. We have implemented each of these five new credibility definitions and tested the resulting PIEC variations on the CAVIAR dataset. The performance measurements (F-measures) for these PIEC variations are compared to these of PIEC with the original credibility definition, OSLα and DN. Tables Table 4 and Table 5 display the comparison results.

More specifically, Table Table 4 shows the results of running PIEC and all its variations on top of the point-based OSLα output, using a probability threshold $t = 0.7$[8]. As far as the "meeting" CE is concerned, the best F-measures are achieved by the original PIEC, the sum-of-squares, and the sum-of-cubes variations. These three variations appear to make the same credibility choices. Another credibility definition that seems to perform almost equally well is "pick last". It is significantly better than OSLα, although slightly worse than the three aforementioned definitions. For the "moving" CE, "pick last"

[8] As discussed earlier, probability threshold values other than $t = 0.7$ result in PIEC displaying inferior performance in this experimental setting, in general. Therefore, we have chosen this threshold value for our experiments.

credibility definition outperforms every other and is the only PIEC credibility definition that manages to surpass OSLα's F-measure (yet only in the micro F-measure case). Setting OSLα performance aside, we observe that all definitions but "pick first" are improvements to the original PIEC credibility definition.

Table 4: F-measure comparison between OSLα and PIEC with several different credibility definitions, for t = 0.7. Lines 1 and 3 correspond to micro F-measure values, while lines 2 and 4 correspond to macro F-measure values.

LTA	OSLα	Original	Pick first	Pick last	Cred = Prob	Sum of squares	Sum of cubes
Meeting	0.7311	**0.8711**	0.7519	0.8707	0.8020	**0.8711**	**0.8711**
	0.3282	**0.3948**	0.3149	0.3912	0.3738	**0.3948**	**0.3948**
Moving	0.6003	0.5706	0.5663	**0.6095**	0.5864	0.5774	0.5857
	0.5726	0.5149	0.5111	0.5560	0.5343	0.5212	0.5277

Table Table 5 displays the same information, for the DN case. Here, for the "meeting" CE, the weakest of the credibility definitions for the OSLα case, "pick first", seems to beat the others, and DN itself. However, not far behind "pick first" there are the original, the sum-of-squares, and the sum-of-cubes credibility definitions, again achieving the same F-measure. For the "moving" CE, "pick first" loses significant ground, while the original, the sum-of-squares, and the sum-of-cubes credibility definitions rise to share the top seed.

Table 5: F-measure comparison between DN and PIEC with several different credibility definitions, for t = 0.7. Lines 1 and 3 correspond to micro F-measure values, while lines 2 and 4 correspond to macro F-measure values.

LTA	OSLα	Original	Pick first	Pick last	Cred = Prob	Sum of squares	Sum of cubes
Meeting	0.3390	0.4125	**0.4161**	0.3777	0.4118	0.4125	0.4125
	0.1246	0.2703	**0.2767**	0.1517	0.2343	0.2703	0.2703
Moving	0.7696	**0.8218**	0.7452	0.7900	0.7727	**0.8218**	**0.8218**
	0.7313	**0.7850**	0.7210	0.7359	0.7275	**0.7850**	**0.7850**

Once more we have observed that the original, the sum-of-squares, and the sum-of-cubes credibility definitions achieve the exact same performance. However, this is not always the case. Specifically, if we take a look on Table Table 4, on the last two rows, that correspond to the "moving" CE, these three credibility definitions display different behavior. On top of that, the sum-of-squares and the sum-of-cubes variations seem to improve over the original. Therefore, measurements from both tables indicate that the sum-of-squares and the sum-of-cubes variations perform no worse than the original PIEC. On the other hand, "pick last" remains decent, beating DN, but is significantly weaker than in the OSLα case.

In what follows, we take two of the best performing credibility definitions according to what we have seen so far, namely the “pick last” and the “sum-of-cubes” and visualize their comparison to the original PIEC and the OSLα/DN methods. For the sake of brevity, from this point onwards we will refer to PIEC with the “pick last” credibility definition as “PIEC-PL” and to PIEC with the “sum-of-cubes” credibility definition as “PIEC-S3”.

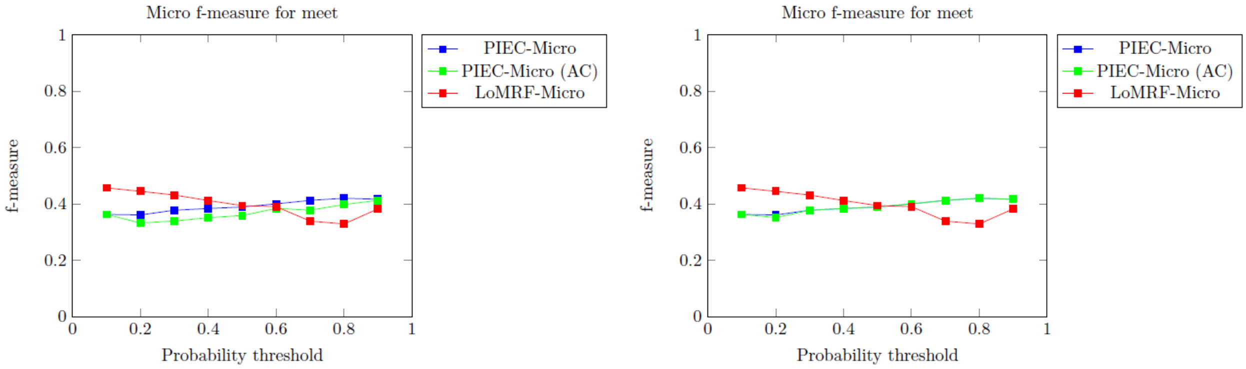

Figure 41: Micro F-measure comparison between DN with threshold, DN with PIEC (original credibility definition – blue), and DN with PIEC (new credibility definitions – green) on CAVIAR's meet LTA

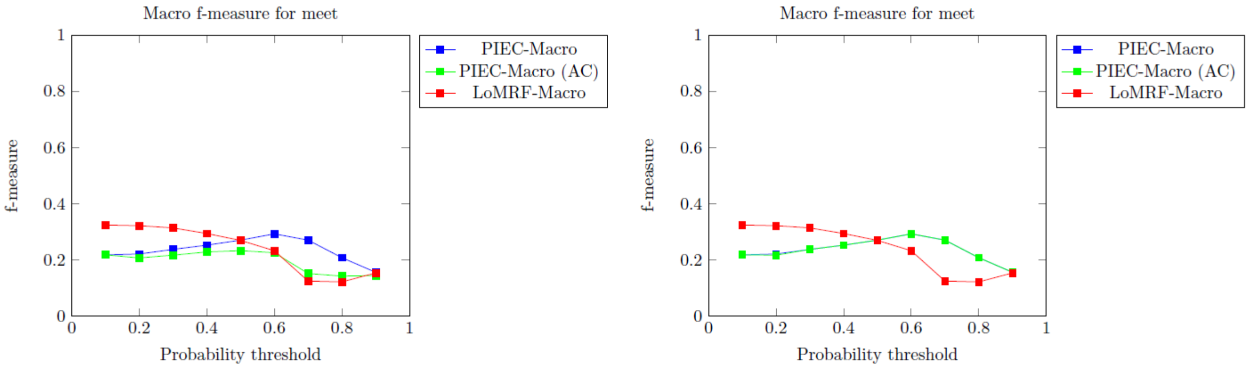

Figure 42: Macro F-measure comparison between DN with threshold, DN with PIEC (original credibility definition – blue), and DN with PIEC (new credibility definitions – green) on CAVIAR's meet LTA

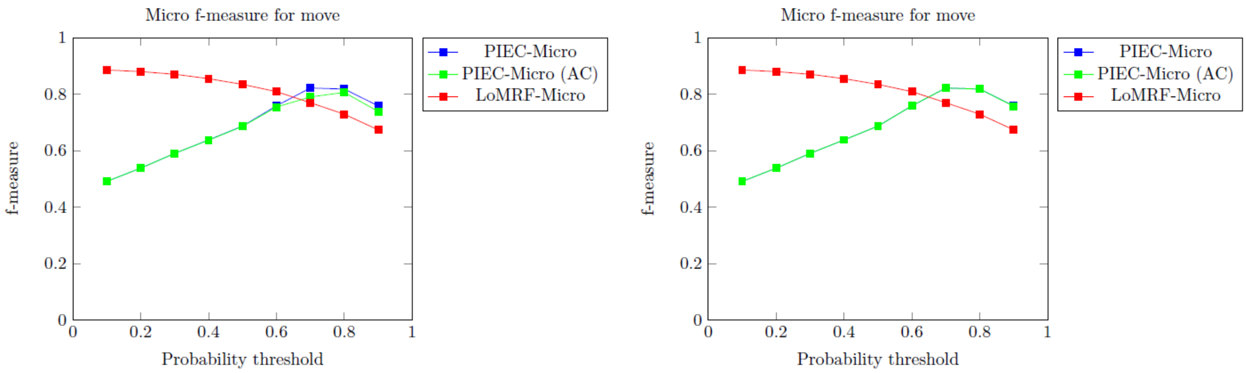

Figure 43: Micro F-measure comparison between DN with threshold, DN with PIEC (original credibility definition – blue), and DN with PIEC (new credibility definitions – green) on CAVIAR's move LTA

We start with the DN case in Figures Figure 41 and Figure 42, with the “meeting” CE, micro and macro F-measures. We can see that, even though DN starts off better, for probability thresholds ≥ 0.5 the original PIEC exceeds DN’s performance. PIEC-S3 is almost identical to PIEC, while PIEC-PL seems to perform worse, in accordance with the first two rows of Table Table 5. Overall, PIEC and PIEC-S3 seem to perform slightly better than DN.

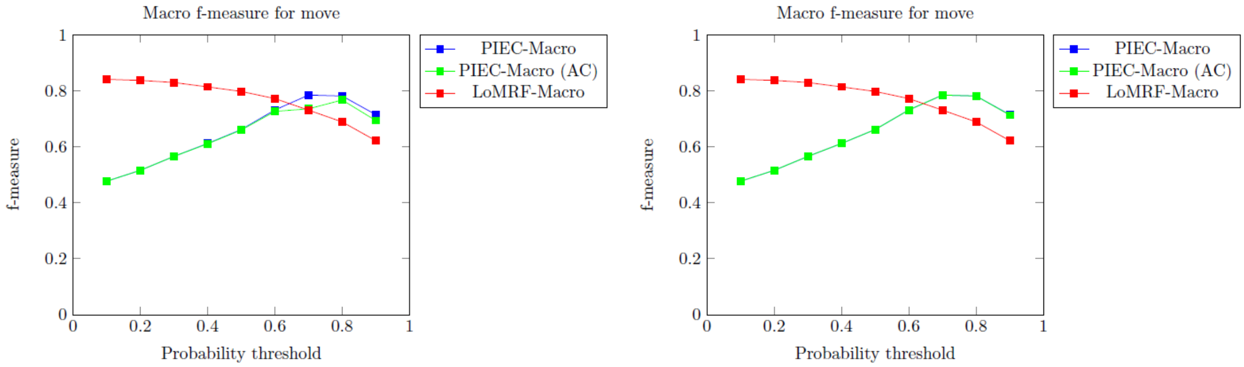

Figure 44: Macro F-measure comparison between DN with threshold, DN with PIEC (original credibility definition – blue), and DN with PIEC (new credibility definitions – green) on CAVIAR's move LTA

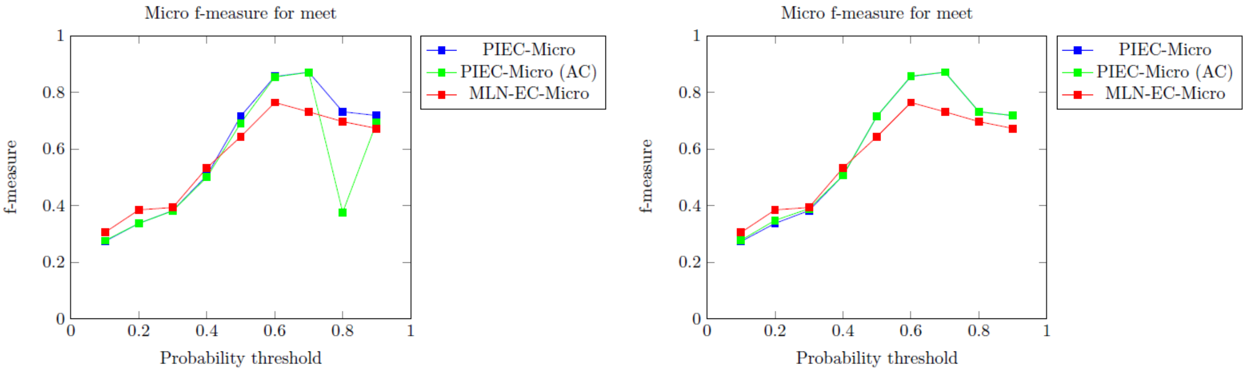

Figure 45: Micro F-measure comparison between OSLα with threshold, OSLα with PIEC (original credibility definition – blue), and OSLα with PIEC (new credibility definitions – green) on CAVIAR's meet LTA

In Figures Figure 43 and Figure 44, we can see the comparison for the “moving” CE case. DN appears to achieve much higher F-measures for small threshold values (≤ 0.5). From $t = 0.7$ onwards, PIEC and PIEC-S3 achieve better scores. PIEC-S3 shows identical behavior to the original PIEC. PIEC-PL shows a similar, yet somewhat inferior performance, as compared to PIEC and PIEC-S3.

As far as the OSLα case is concerned, in Figures Figure 45 and Figure 46 we can see that PIEC and PIEC-S3 strongly outperform OSLα, especially for threshold values 0.6 and 0.7. Also, PIEC-S3 performs no worse than the original PIEC. On the other hand, PIEC-PL appears to be somewhat weaker. On top of that, PIEC-PL displays a downward spike for $t = 0.8$ which is completely avoided using the original PIEC or its PIEC-S3 variant.

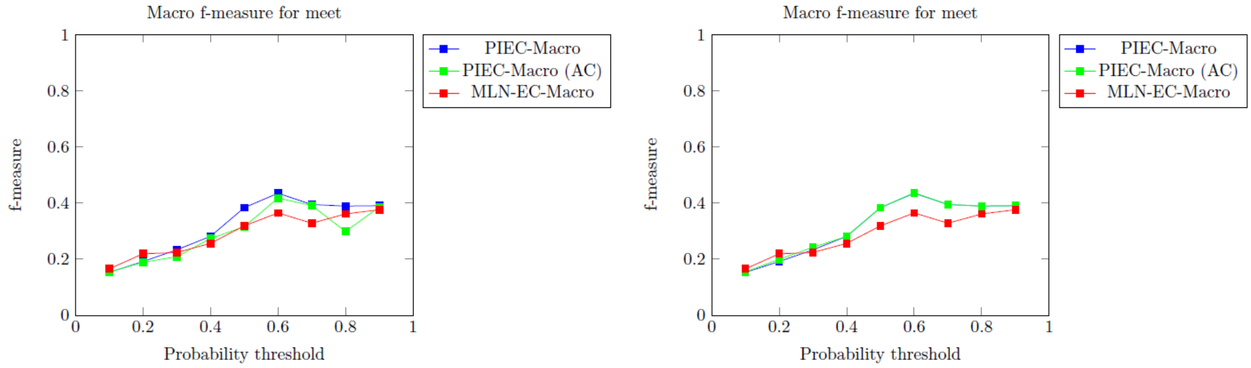

Figure 46: Macro F-measure comparison between OSLα with threshold, OSLα with PIEC (original credibility definition – blue), and OSLα with PIEC (new credibility definitions – green) on CAVIAR's meet LTA

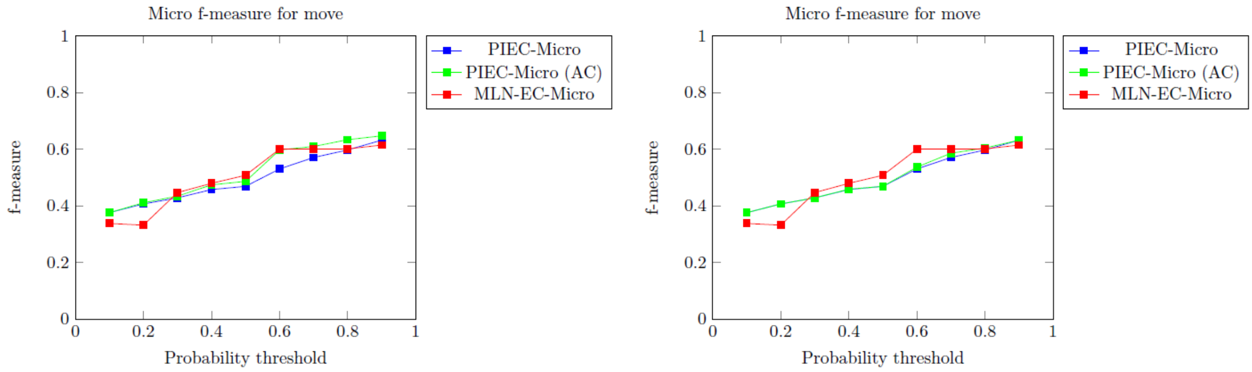

Figure 47: Micro F-measure comparison between OSLα with threshold, OSLα with PIEC (original credibility definition – blue), and OSLα with PIEC (new credibility definitions – green) on CAVIAR's move LTA

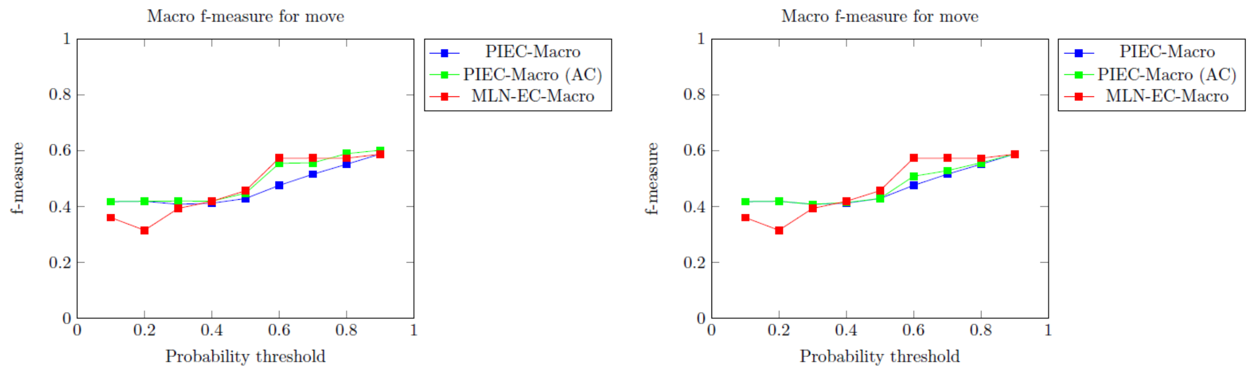

Figure 48: Macro F-measure comparison between OSLα with threshold, OSLα with PIEC (original credibility definition – blue), and OSLα with PIEC (new credibility definitions – green) on CAVIAR's move LTA

Finally, for the "moving" CE on the OSLα case, we observe a different situation. As pointed out earlier in Table Table 4, this is a case where PIEC-PL seems to be doing a significantly better job than both PIEC and PIEC-S3. In Figures Figure 47 and Figure 48, it becomes evident that OSLα generally outperforms PIEC and PIEC-S3, but PIEC-PL does better for high probability thresholds ($t \geq 0.8$), achieving equal or marginally higher F-measures. More specifically, the highest F-measure is achieved by PIEC and both its variants, for $t = 0.9$.

6. CONCLUSIONS AND FURTHER WORK

The Probabilistic Interval-based Event Calculus is a quick, sound, and complete algorithm for recognizing the occurrence of complex LTA, in terms of temporal intervals. It is able to handle intervals, when other activity recognition methods work in a timepoint-based manner, which runs the risk of producing misleading results. Thanks to its efficient data structures it is able to run in time linear with respect to the size of its input. It has been mathematically proven that every output interval it produces is indeed a probabilistic maximal interval and that it is able to correctly recognize all probabilistic maximal intervals for a given input of instantaneous LTA probabilities.

Subsequently, PIEC was thoroughly tested against state-of-the-art probabilistic activity recognition frameworks. The results of the comparison show that, even though there are cases where PIEC appears not to equal the performance of the traditional, timepoint-based approaches, in the general case it performs better than the older methods. A general pattern that can be observed by looking at the results is that PIEC manages to reduce False Negatives at the expense of some extra False Positives. It naturally follows that PIEC achieves a better Recall almost everywhere, whereas it seems to be slightly inferior in terms of Precision. Overall, PIEC seems to do better than the traditional activity recognition methods, especially for high probability thresholds (that is, $t \geq 0.7$). There also seems to be great room for improvement in several cases, if we consider alternative credibility definitions, as we have discussed at the end of the empirical evaluation section.

The work on interval-based event recognition can be extended in several ways. First of all, the field of alternative credibility definitions should be further investigated, as it might lead to new credibility strategies that increase PIEC's performance even more. We could also apply machine learning techniques in order to automatically specify the optimal credibility rule. Another possible future work direction is the introduction of a dynamically changing probability threshold, that could adapt to and effectively handle certain involved instantaneous probability patterns e.g. the abrupt probability fluctuations that OSLα frequently displays. Last but not least, an interesting approach could be to calculate maximal intervals of an LTA directly from known maximal intervals of its constituting STA. This way, we could recognize composite activities from events that occur in temporal intervals instead of timepoints. Such an approach could reduce or even overcome the shortcomings of the instantaneous activity recognition.

TABLE OF TERMINOLOGY

Ξενόγλωσσος όρος	**Ελληνικός Όρος**
Activity Recognition	Αναγνώριση Δραστηριοτήτων
Completeness	Πληρότητα
Complex Event Recognition	Αναγνώριση Σύνθετων Γεγονότων
Correctness	Ορθότητα
Credibility	Αξιοπιστία
Durative	Διαρκής
Event	Γεγονός
Event Calculus	Λογισμός Γεγονότων
Ground Truth	Πραγματικότητα
Heterogeneous	Ετερογενής/νείς
Instantaneous	Στιγμιαίος
Interval	Διάστημα
Long-Term Activities	Μακροπρόθεσμες Δραστηριότητες
Markov Logic Networks	Μαρκοβιανά Λογικά Δίκτυα
Maximal	Μέγιστος
Noise	Θόρυβος
Overlap	Επικάλυψη
Probabilistic Event Calculus	Πιθανοτικός Λογισμός Γεγονότων
Probability	Πιθανότητα
Reasoning	Συμπερασμός
Representation	Αναπαράσταση
Short-Term Activities	Βραχυπρόθεσμες Δραστηριότητες
Soundness	Ευρωστία
Statistical Relational Learning	Στατιστική Σχεσιακή Μάθηση
Threshold	Κατώφλι
Timepoint	Χρονικό Σημείο
Uncertainty	Αβεβαιότητα

ABBREVIATIONS - ACRONYMS

ADSMS	Active Database Systems
BDD	Binary Decision Diagrams
CE	Complex (or Composite) Events
CEA	Complex Event Automata
CEL	Complex Event Logic
CEP	Complex Event Processing
CER	Complex (or Composite) Event Recognition
Crisp-EC	Crisp Event Calculus
DSCP	Distributed Stream Computing Platforms
DSMS	Data Stream Management Systems
EC	Event Calculus
EP	Event Processing
EPP	Event Processing Platform
HLE	High-Level Events
LLE	Low-Level Events
LNNSI	Longest Non-Negative Sum Interval
LTA	Long-Term Activities
MAP	Maximum A Posteriori
MCMC	Markov Chain Monte Carlo
MLN	Markov Logic Networks
MLN-EC	Markov Logic Network Event Calculus
MSI	Maximum Sum Interval
PIEC	Probabilistic Interval-based Event Calculus
Prob-EC	Probabilistic Event Calculus
SDE	Simple, Derived Events
SLR	Statistical Relational Learning
SQL	Structured Query Language
STA	Short-Term Activities

REFERENCES

[1] A. Skarlatidis et al., A probabilistic logic programming event calculus, *Theory and Practice of Logic Programming*, vol. 15, no. 2, 2015, pp. 213-245.

[2] A. Skarlatidis et al., Probabilistic Event Calculus for Event Recognition, *ACM Transactions on Computational Logic*, vol. 16, no. 2, 2015, pp. 11:1-11:37.

[3] A. Skarlatidis, "Event Recognition Under Uncertainty and Incomplete Data", doctoral dissertation, Dept. Digital Systems, University of Piraeus, 2014.

[4] A. Artikis, E. Makris and G. Paliouras et al., A probabilistic interval-based event calculus for activity recognition, *Annals of Mathematics and Artificial Intelligence*, Aug. 2019.

[5] R. Kowalski and M. Sergot, A Logic-Based Calculus of Events, *New Generation Computing*, vol. 4, no. 1, 1986, pp. 67-96.

[6] A. Artikis, M.J. Sergot and G. Paliouras, "Run-time composite event recognition", *Proc. 6th ACM International Conference on Distributed Event-Based Systems (DEBS 2012)*, Berlin, Germany, July 16-20, 2012, pp. 69-80.

[7] A. Artikis, M.J. Sergot and G. Paliouras, An Event Calculus for Event Recognition, *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 4, 2015, pp. 895-908.

[8] A. Kimmig et al., On the Implementation of the Probabilistic Logic Programming Language ProbLog, *Theory and Practice of Logic Programming*, 2011, pp. 235-262.

[9] P. Domingos and D. Lowd, *Markov Logic: An Interface Layer for Artificial Intelligence*, Morgan & Claypool Publishers, 2009.

[10] A. Skarlatidis et al., "Probabilistic Event Calculus Based on Markov Logic Networks", *Rule-Based Modeling and Computing on the Semantic Web, 5th International Symposium, RuleML 2011 - America, Ft. Lauderdale, FL, Florida, USA, November 3-5, 2011. Proceedings*, pp. 155-170.

[11] P. Singla and P.M. Domingos, "Discriminative Training of Markov Logic Networks", *Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA*, pp. 868-873.

[12] G. Casella and E. I George, Explaining the Gibbs sampler, *The American Statistician*, vol. 46, no. 3, 1992, pp. 167-174.

[13] H. Poon and P.M. Domingos, "Sound and Efficient Inference with Probabilistic and Deterministic Dependencies", *Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA*, pp. 458-463.

[14] P. Damlen, J. Wakefield and S. Walker, Gibbs sampling for Bayesian non-conjugate and hierarchical models by using auxiliary variables, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 61, no. 2, 1999, pp. 331-344.

[15] H.A. Kautz and B. Selman and Y. Jiang, "A general stochastic approach to solving problems with hard and soft constraints", *Satisfiability Problem: Theory and Applications, Proceedings of a DIMACS Workshop, Piscataway, New Jersey, USA, March 11-13, 1996*, pp. 573-586.

[16] H.H. Hoos and Thomas Stutzle, *Stochastic Local Search: Foundations & Applications*, Elsevier/Morgan Kaufmann, 2004.

[17] T.N. Huynh and R.J. Mooney, "Max-Margin Weight Learning for Markov Logic Networks", *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2009, Bled, Slovenia, September 7-11, 2009, Proceedings, Part I*, pp. 564-579.

[18] J. Noessner and M. Niepert and H. Stuckenschmidt, "RockIt: Exploiting Parallelism and Symmetry for MAP Inference in Statistical Relational Models", *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA*.

[19] S. Riedel, "Improving the Accuracy and Efficiency of MAP Inference for Markov Logic", *Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence (UAI 2008), Helsinki, Finland, July 9-12, 2008*, pp. 468-475.

[20] E. Michelioudakis, "Online Structure Learning for Markov Logic Networks using Background Knowledge Axiomatization", Master's thesis, School of Electronic and Computer Engineering, Technical University of Crete, 2016.

[21] A. Grez, C. Riveros and M. Ugarte, Foundations of Complex Event Processing, *Computing Research Repository (CoRR)*, 2017.

[22] D.J. Abadi et al., "Aurora: A Data Stream Management System", *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, San Diego, California, USA, June 9-12, 2003*.

[23] A. Arasu et al., "STREAM: The Stanford Stream Data Manager", *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, San Diego, California, USA, June 9-12, 2003*.

[24] A. Artikis et al., “Complex Event Recognition Languages: Tutorial”, *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems, DEBS 2017, Barcelona, Spain, June 19-23, 2017*, pp. 7-10.
[25] G. Cugola and A. Margara, Processing flows of information: From data stream to complex event processing, *ACM Comput. Surv.*, vol. 44, no. 3, 2012, pp. 15:1-15:62.
[26] D. Anicic et al., “A Rule-Based Language for Complex Event Processing and Reasoning”, *Web Reasoning and Rule Systems - Fourth International Conference, RR 2010, Bressanone/Brixen, Italy, September 22-24, 2010. Proceedings*, pp. 42-57.
[27] F. Chesani et al., A Logic-Based, Reactive Calculus of Events, *Fundam. Inform.*, vol. 105, no. 1-2, 2010, pp. 135-161.
[28] Y. Mei and S. Madden, “ZStream: a cost-based query processor for adaptively detecting composite events”, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2009, Providence, Rhode Island, USA, June 29 – July 2, 2009*, pp. 193-206.
[29] M. Liu et al., “E-Cube: multi-dimensional event sequence analysis using hierarchical pattern query sharing”, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2011, Athens, Greece, June 12-16, 2011*, pp. 889-900.
[30] J. Agrawal et al., “Efficient pattern matching over event streams”, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pp. 147-160.
[31] R.S. Barga et al., “Consistent Streaming Through Time: A Vision for Event Stream Processing”, *CIDR 2007, Third Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 7-10, 2007, Online Proceedings*, pp. 363-374.
[32] A. Demers et al., A general algebra and implementation for monitoring event streams, *2012*.
[33] P.R. Pietzuch, B. Shand and J. Bacon, “A Framework for Event Composition in Distributed Systems”, *Middleware 2003, ACM/IFIP/USENIX International Middleware Conference, Rio de Janeiro, Brazil, June 16-20, 2003, Proceedings*, pp. 62-82.
[34] N.P. Schultz-Moller, M. Migliavacca and P.R. Pietzuch, “Distributed complex event processing with query rewriting”, *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems, DEBS 2009, Nashville, Tennessee, USA, July 6-9, 2009*.
[35] M. Dayarathna and S. Perera, Recent Advancements in Event Processing, *ACM Comput. Surv.*, vol. 51, no. 2, 2010, pp. 33:1-33:36.
[36] C. Ballard and IBM, *IBM Infosphere Streams Harnessing Data in Motion*, IBM redbooks, Vervante, 2010.
[37] S. Jayasekara et al., “Continuous analytics on geospatial data streams with WSO2 complex event processor”, *Proceedings of the 9th ACM International Conference on Distributed Event-Based Systems, DEBS '15, Oslo, Norway, June 29 - July 3, 2015*, pp. 277-284.
[38] A. Martin, A. Brito and C. Fetzer, “Real time data analysis of taxi rides using StreamMine3G”, *Proceedings of the 9th ACM International Conference on Distributed Event-Based Systems, DEBS '15, Oslo, Norway, June 29 - July 3, 2015*, pp. 269-276.
[39] L. Neumeyer et al., “S4: Distributed Stream Computing Platform”, *ICDMW 2010, The 10th IEEE International Conference on Data Mining Workshops, Sydney, Australia, 13 December 2010*, pp. 170-177.
[40] A. Toshniwal et al., “Storm@twitter”, *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*, pp. 147-156.
[41] S. Suhothayan et al., “Siddhi: a second look at complex event processing architectures”, *Proceedings of the 2011 ACM SC Workshop on Gateway Computing Environments, GCE 2011, Seattle, WA, USA, November 18, 2011*, pp. 43-50.
[42] L. Brenna et al., “Cayuga: a high-performance event processing engine”, *Proceedings of the ACM SIGMOD International Conference on Management of Data, Beijing, China, June 12-14, 2007*, pp. 1100-1102.
[43] R. Biswas, S. Thrun and K. Fujimura, “Recognizing Activities with Multiple Cues”, *Human Motion - Understanding, Modeling, Capture and Animation, Second Workshop, Human Motion 2007, Rio de Janeiro, Brazil, October 20, 2007, Proceedings*, pp. 255-270.
[44] V.D. Shet, D. Harwood and L.S. Davis, “VidMAP: video monitoring of activity with Prolog”, *Advanced Video and Signal Based Surveillance, 2005 IEEE International Conference on Video and Signal Based Surveillance (AVSS'05), 15-16 September 2005, Como, Italy*, pp. 224-229.
[45] F.A. D’Asaro et al., “Foundations for a Probabilistic Event Calculus”, *Logic Programming and Nonmonotonic Reasoning - 14th International Conference, LPNMR 2017, Espoo, Finland, July 3-6, 2017, Proceedings*, pp. 57-63.
[46] B.-H. Juang and L.R. Rabiner, Mixture autoregressive hidden Markov models for speech signals, *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 33, no. 6, 1985, pp. 1404-1413.

[47] A.V. Nefian et al., Dynamic Bayesian Networks for Audio-Visual Speech Recognition, *EURASIP J. Adv. Sig. Proc.*, vol. 2002, no. 11, 2002, pp. 1274-1288.
[48] J.D. Lafferty, A. McCallum and F. C.N. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data", *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pp. 282-289.
[49] V.I. Morariu and L.S. Davis, "Multi-agent event recognition in structured scenarios", *The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011*, pp. 3289-3296.
[50] N. Busany et al., "Interval-based Queries over Multiple Streams with Missing Timestamps", 2017.
[51] L. Allison, Longest biased interval and longest non-negative sum interval, *Bioinformatics*, vol. 19, no. 10, 2003, pp. 1294-1295.