Evaluating Performance in Continuous Context Recognition Using Event-Driven Error Characterisation

Jamie A Ward[1], Paul Lukowicz[2], Gerhard Tröster[1]

[1] Swiss Federal Institute of Technology (ETH)
Wearable Computing Lab
8092 Zürich, CH
[2] Institute for computer Systems and Netorks,
UMIT- University for Health Sciences, Medical Informatics and Technology,
Hall i. Tirol, Austria

Abstract. Evaluating the performance of a continuous activity recognition system can be a challenging problem. To-date there is no widely accepted standard for dealing with this, and in general methods and measures are adapted from related fields such as speech and vision. Much of the problem stems from the often imprecise and ambiguous nature of the real-world events that an activity recognition system has to deal with. A recognised event might have variable duration, or be shifted in time from the corresponding real-world event. Equally it might be broken up into smaller pieces, or joined together to form larger events. Most evaluation attempts tend to smooth over these issues, using "fuzzy" boundaries, or some other parameter based error decision, so as to make possible the use of standard performance measures (such as insertions and deletions.) However, we argue that reducing the various facets of a activity system into limited error categories - that were originally intended for different problem domains - can be overly restrictive. In this paper we attempt to identify and characterise the errors typical to continuous activity recognition, and develop a method for quantifying them in an unambiguous manner.
By way of an initial investigation, we apply the method to an example taken from previous work, and discuss the advantages that this provides over two of the most commonly used methods.

1 Introduction

As research interest into recognition of user activities - and more generally user context - continues to grow, so too does the demand for standard methods of evaluating and comparing performance of the different approaches. There are two main criteria involved in the development of such methods. One is in establishing open datasets to be used as a benchmark, of which to-date work is only just beginning.[3] The second criteria, and the one which is the focus of this

[3] For example, see the dedicated workshop on this topic at Pervasive '04 [8].

paper, is the issue of appropriate performance measures. While working on different recognition problems [23,12,16] and looking at related publications (such as [18,11,21,2]) we have found that existing evaluation measures, mostly taken from related fields such as automatic speech recognition (ASR), information retrieval (IR), and vision related fields such as optical character recognition (OCR), often fail to adequately reflect the specific problems of the activity recognition task, in particular with the non segmented, continuous case.

The performance evaluation of a continuous activity recognition system can be viewed as a problem of how to measure the similarity of two time series; the similarity of a prediction sequence to its corresponding ground truth. In the topic of activity recognition, these time series are made up of discrete events, each representing a particular activity which the system is designed to recognise. As such events are based on real world activities, or concern changes to a user's environment, it is often the case that they are of variable duration and have ambiguous start and stop times. This can lead to events being detected some time before or after they actually occur. It can also lead to single events being fragmented into multiple smaller events of the same class; or, alternatively, the merging of several real events into a single detected event.

Dealing with such traits poses a problem for satisfactory performance evaluation. Using existing methods of evaluation, designers have the choice of making a direct timewise (frame-by-frame) comparison of the ground and prediction sequences, thereby loosing information on the nature of events; or of performing a comparison of the events, at the expense of loosing information on the timing. In the later case, the definitions of the event errors - insertion, deletion, etc. - is further complicated by how to treat events which are fragmented or merged. Often the designer is forced to make a decision as to whether fragmented or merged events are undesirable or not, whether to ignore them or to count them as full inserted or deleted events.

1.1 Paper Contributions and Organisation

Our repeated encounters with such cases prompted us to investigate the problem of finding suitable evaluation measures in continuous activity recognition. In this paper we propose an alternative strategy for evaluation which combines the strengths of several existing methods without throwing away critical information that might be judged important by application developers wishing to use such activity recognition systems.

The paper is divided into four main parts. In section 2 we motivate the work by highlighting the problems of existing measures when used on a typical activity recognition example, as obtained from an earlier published work. In section 3 we provide a detailed analysis of these problems. Section 4 introduces our proposed categorisation and error scoring methods to combat these problems. Finally, in closing, we apply our methods to the original motivational examples, and use them to fuel the discussion on how they might be used in practice.

2 Motivation

As a motivation for this work, consider Figure 1. Plot (a) is an example of output from a multi-class, continuous activity recognition task which was carried out on a mock assembly scenario in the wood workshop of our lab [23]. The plot shows hand-labelled ground truth for five activities which we attempted to recognise in this experiment: use of a grinder, file, screwdriver, vice and drawer. The time where no relevant activity was performed is recorded as *NULL*. Plotted above the ground truth are the recognition system's predictions. This data is output on a timewise frame-by-frame basis, with each frame being one second in length.

For most of the non-*NULL* activities, these prediction sequences seem to visually correlate well with the ground truth. There are few insertions and only one completely deleted activity. Contrast this result with the middle(b) and bottom(c) plots of Figure 1. A casual visual assessment might report, due to the abundance of insertions in (b) and the heavily fragmented output of (c), that this data is much poorer than that of (a).

2.1 Frame based analysis

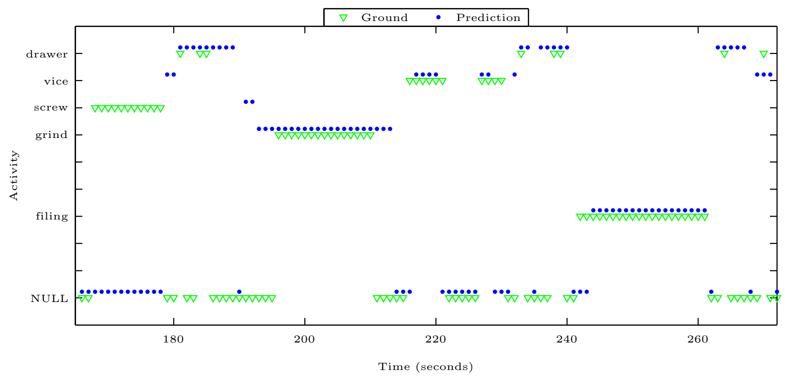
When evaluating such data quantitatively, a standard practise is to make a frame by frame comparison of the ground truth with the predictions. Counts of correct and incorrect matches can then be tallied for each class and entered into a confusion matrix [6]. From here a number of standard performance measures can be calculated, the most common of these being accuracy (the overall correct rate).

However, when this analysis is performed on the examples, as shown in the tables to the right of Figure 1, a somewhat unsatisfying result is obtained: they all have identical accuracy. This result seems contrary to what observation tells us. Furthermore, the confusion matrices for examples (a) and (b), simplified to the summation of positive classes vs. *NULL*, are very similar and tell us nothing about, for example, the prevalence of insertion errors in (b).

These results are not wrong - the numbers of frame errors in all three examples are in fact equal. What the visual analysis shows, and the frame analysis does not show, is that every positive frame forms part of an *event* - a contiguous sequence of same class frames. When judged from an event perspective, then the distribution of frame errors becomes more important. Many of the false positives in (a), for example, are joined to otherwise correctly classified sequences; however, in (b) they tend to form part of *event insertions* - an arguably more serious misclassification.

2.2 Event analysis

Researchers in the fields of optical character recognition (OCR) [5,13] and automatic speech recognition (ASR) both commonly employ counts of insertion (I_e), deletion (D_e) and substitution (S_e) event errors to measure performance.



a)

Frame errors

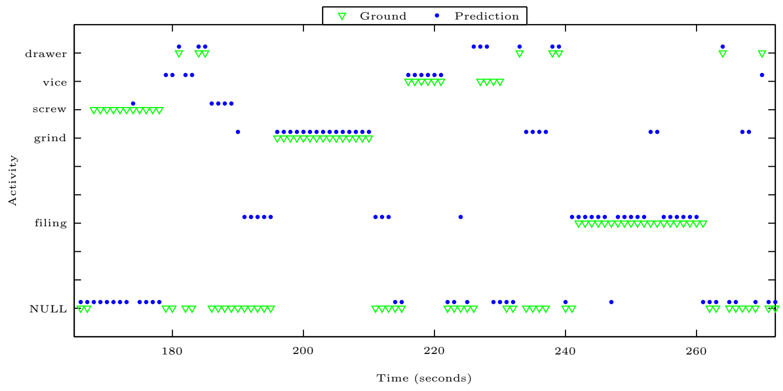
	P_{out}	N_{out}
P	**46**(1)	17
N	27	**16**

$acc_f = \mathbf{57.9}\%$

Event errors

$I_e =$	3
$D_e =$	1
$S_e =$	1

$err_e = \mathbf{45.5}\%$



b)

Frame errors

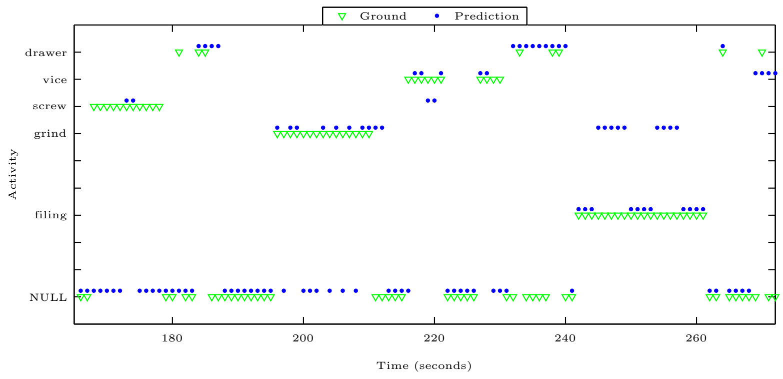
	P_{out}	N_{out}
P	**45**(5)	14
N	26	**17**

$acc_f = \mathbf{57.9}\%$

Event errors

$I_e =$	10
$D_e =$	0
$S_e =$	2

$err_e = \mathbf{109}\%$



c)

Frame errors

	P_{out}	N_{out}
P	**32**(12)	20
N	13	**30**

$acc_f = \mathbf{57.9}\%$

Event errors

$I_e =$	3
$D_e =$	1
$S_e =$	1

$err_e = \mathbf{45.5}\%$

Fig. 1. Examples $(a - c)$ from multi-class continuous activity problem. The tables give performance information using standard methods: *Frame errors* using binary confusion matrices of positive (P) vs. *NULL* (N) frames, where rows denote the ground truth and columns the output predictions. Positive substitutions are entered in brackets alongside True Positives (TP) in these matrices. Accuracy is calculated as: $acc_f = \frac{TP+TN-subst.}{T_f}$, with the total frames in each example being $T_f = 107$. *Event errors* are given as insertion (I_e), deletion (D_e) and substitution (S_e) counts. The *event error rate* is $err_e = \frac{I_e+D_e+S_e}{T_e}$, with total number of positive events, $T_e = 11$

These give indicators of the discrete event performance of a system, and seem a natural choice for evaluating a discrete sequence of activity events.

When these scores are calculated for each of the examples (see the lower tables of Figure 1), the differences between examples (a) and (b) become much clearer. Example (a) shows a relatively low insertion count in comparison with the very high number of insertions in example (b). However, if we look at example (c) - again a very different output from (a) - we are once again disappointed: the deletion, substitution and insertion counts of (c) are identical to those of (a).

There are two main problems underlying these results, neither of which are highlighted by any of the commonly used evaluation methods. The first problem is that many of the events are fragmented: several smaller segments, although correctly classified, only sparsely cover parts of the ground truth. Some of these segments are separated by small fragments of *NULL* (frame deletions); while others, such as the 'filing' event, are fragmented by insertions of another class (frame substitutions).

The second problem is that events can be merged together, (i.e.) two or more events of the same class can be recognised as a single large event. In the examples given here, this happens on only two occasions (the 'drawer' events of (a) and (c)). In each case, this error only affects two closely occurring events. For purposes of evaluation, the fact that these two separate events have been merged is simply ignored. They are both treated as correct. Alternatively, it might be decided that such a merging, reducing two or more events to one, is also a deletion of all merged events except the first.

In both of these cases, fragmenting and merging, it is clear that there are several ways one might choose to score the results, and here lies the problem: there is no standard definition for such errors. The existing designations of D_e, I_e and S_e, were developed for fields such as OCR which enjoy well-defined, discrete events. In continuous activity recognition, as highlighted by these examples, this is not always the case.

3 Problem specification

In order to develop more appropriate evaluation metrics, the problems illustrated in the previous section should first be formulated in a more systematic way. This section begins with a definition of the performance evaluation task. From this definition we discuss specific characteristics of performance which are common to continuous context recognition.

3.1 Definition of performance evaluation

In the most general classification problem we have n classes $(c_1, c_2, \cdots c_n)$ without a designated class for *NULL*. The ground truth consists of a number of m distinct events $(e_1, e_2, \cdots e_m)$, each mapping to one of the n classes. We assume the system to be time discrete with the smallest considered time unit being a frame. In most cases, a frame would correspond to the length of the sensor sampling window.

An ideal classifier would be one where every ground truth event, e_i, has a start time, stop time and label matching an event in the prediction sequence. Correspondingly, all constituent frames would also match.

Unfortunately such perfect alignment is rare. A typical recognition system deletes, inserts, and substitutes data. In addition even for correctly correlated data the start and stop frames might be shifted in the recognised sequence. The problem of evaluating such imperfect classification is equivalent to that of finding an appropriate similarity metric for the comparison of two time series. As we see it, this problem can be tackled on three levels:

1. *Frame by frame.* For each pair of corresponding time frames f (from the ground truth) and $\bar{f}$ (from the recognition system output) we perform a simple comparison of the class labels.
2. *Event-based.* Determine how many of the m ground truth events $(e_1, e_2, \cdots e_m)$ are accurately reflected in the $\bar{m}$ events $\bar{e}_1, \bar{e}_2 \cdots \bar{e}_{\bar{m}}$ produced by the recognition system. The difficulty of event based evaluation stems from the fact that neither the number of events nor their start and end points are necessarily identical in the ground truth and the recogniser output.
3. *Hybrid frame and event based.* A frame by frame comparison which takes into account the events to which individual frames are a part. Thus frame errors that merely cause the start and end points of events to be shifted are treated differently from frame errors that contribute to the deletion and insertion of events. This type of evaluation only makes sense if some prior event analysis has been carried out.

3.2 General considerations

Given two time series there can be no such thing as an optimal measure of similarity which holds for all applications. As a consequence there is no optimal, problem independent performance evaluation. Different application domains are subject to different performance criteria. In speech recognition, for example, it is more important that the system recognises what words have been spoken, and in which order, rather than how long it took to utter them. Consequently, methods which emphasise correct ordering of symbols over their specific duration are used to evaluate these systems. An input to a real-time system, on the other hand, would need to be extremely time sensitive. As such an evaluation metric which emphasises timing errors and delays, i.e. based on a direct timewise comparison, would be more appropriate.

For every domain, a specific metric must be chosen that characterises and highlights the type of error(s) most critical to that domain. This means that evaluation methods that are successful in one domain need not necessarily be so in another. Applying methods to a different domain only makes sense if both domains have the same type of dominant error types and similar relevance is assigned to equivalent errors.

3.3 Evaluation requirements of continuous activity recognition

The study of activity recognition encompasses a wide range of problems, including standard modes of locomotion (walking, standing, running, etc.)[18,15,11,21], tracking of specific procedures (e.g. assembly tasks [17]), and the detection of changes in environmental conditions[2,15]. While each of these problems have their own characteristic and relevant error types, there are a number of things that most continuous activity recognition tasks have in common:

Large variability in event length In many activity recognition tasks, event length can vary by an order of magnitude or more. A wood workshop assembly example includes such activities as sawing which can take minutes, as well as taking or putting away tools which take just a few seconds. Similarly, when recognising modes of locomotion, there can be instances of long uninterrupted walks, as well as instances of a user making only a few steps. A direct frame by frame evaluation can be misleading in such cases.

Fragmented events Long lasting events are often interrupted by the occurrence of short events. Thus a long sawing sequence might include one or two interruptions or an instance of the user changing the saw. A long walk might include a few short stops. Since the recognition system must be able to spot such situations, it is also prone to false fragmentation. As an example, a slight irregularity in the sawing motion might be falsely interpreted as an interruption, or a short instance of an entirely different activity. In addition to inserting a new event, fragmentation also breaks up one long event in the ground truth, producing several events in the recogniser output.

Event merging Trying to avoid false fragmentation can lead to a system that tends to overlook genuinely fragmented outputs. Thus two events of the same class separated by a short event of another class might be merged into a single long event of the first class. This in a sense is a 'double deletion' since it deletes the short event in the middle, and causes the two events of the outer class to become one.

Lack of well defined NULL class Many activity recognition tasks aim to spot a small set of interesting activities/situations while regarding the rest as instances of a 'garbage' or *NULL* class. This *NULL* class has the same function as the pauses in speech, or spaces in character recognition. The problem is that many activity recognition tasks have a *NULL* class which is complex and difficult to model. In the assembly task, for example, any motion made between the specific tool activities falls into this class. This includes everything from scratching one's head to unpacking a chocolate bar. As a consequence the *NULL* class model tends to be 'greedy', so that any unusual segment in an event (e.g. strange motion while sawing) tends to create a *NULL* event, thus contributing to the fragmentation problem.

Fuzzy event boundaries When collecting large, real life data sets it is often impossible to perfectly time ground truth labels by hand. The definition of start and stop times of an event are often arbitrary and by nature imprecise. This is particularly so for domains such as activity recognition, where even the event is often difficult to define - e.g. at which point does a walking event end and a running event begin? This leads to timing errors in the recognition, even if the system can be said to work perfectly. Similarly, in tasks where interesting events are separated by a greedy *NULL*, the lack of a well defined *NULL* model will inevitably result in some incursion into the boundaries of the correct events.

The importance of these different issues is dependent on the specific application for which the system is being evaluated. However, we believe that for most activity recognition tasks, one or more of these issues is important, and that they should be taken into account when evaluating these systems.

4 Error characterisation and representation

Following from the above observations we now present a characterisation of the critical error types in continuous activity recognition. Specifically we propose an approach which (1) includes event mergers and fragmentation as errors in their own right; and (2) provides information about event timing errors. This section presents both the definition of the proposed errors, and a precise method on how to score them. We then show how this information can be tabulated for presentation of a system's results. Additionally we show how the methods can be tailored for dealing with activity recognition systems that treat *NULL* as a special case.

Our evaluation method is based on partitioning the signal stream into what we call *segments*. As an example, Figure 3 shows a three class recognition problem broken up into 14 segments (denoted by the vertical dotted lines). A segment is a variable-duration, contiguous sequence of frames, during which neither prediction nor the ground truth label changes. That is, each boundary of a segment is defined by either the boundary of a ground truth, or of a prediction event.

From the point of view of performance evaluation such a segment definition has two advantages. The first is that there are no ambiguities in comparison: each segment can either have the prediction and the ground truth fully agree, or fully disagree. The second advantage is that from an analysis of these segments, an exhaustive definition of the event and timing errors appropriate to activity recognition can be derived. This strategy has three main steps:

1. Create the segment sequence and note each segment as matching or non matching. A *match* being when both the ground truth segment and its corresponding prediction segment have the same class label.
2. Use segment match information to score events and event timing errors. Prediction and ground truth events are scored separately. The left flowchart of Figure 2 shows the algorithm to do this for ground truth events, with possible outputs of fragmenting F, deletion D, underfill U, correct, and *no label* (a single matching segment event to which none of the other designations apply). Prediction events are scored using the same algorithm, but with the alternate outputs of: merge (M), insertion(I) and overfill (O).

3. Score the segment errors. The flowchart to the right of Figure 2 shows how this is done. Each (non matching) segment is assigned an error pair based on the ground and prediction events to which it forms part.

The example of Figure 3 shows three analyses of a 3-class (A,B,C) recognition example: one for counting event errors, one for counting event timing errors, and the third for counting segment errors. In each of the analyses, the same example prediction sequence is shown (with I,M or O assigned for each incorrect event or segment) against a possible ground truth (with D,F or U assigned to each incorrect event or segment). These error categories, and how to score them, are described in greater detail in the following sections.

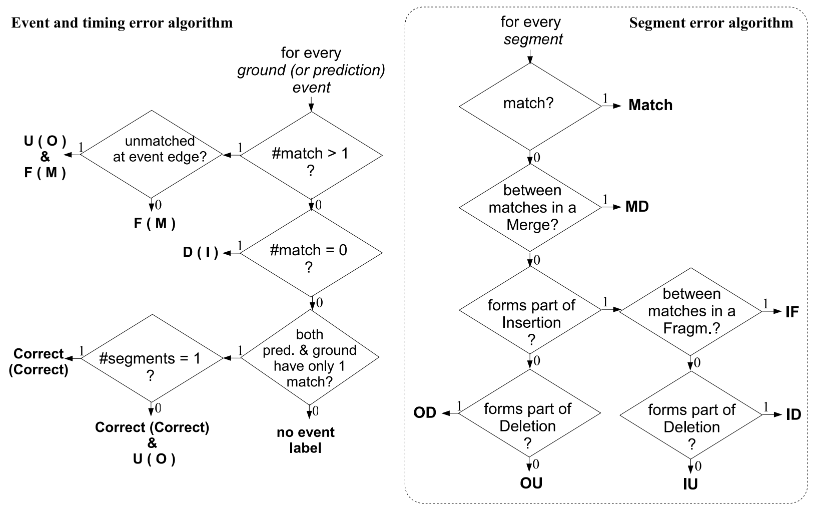

Event and timing error algorithm
for every ground (or prediction) event
#match > 1 ?
unmatched at event edge?
U (O) & F (M)
F (M)
#match = 0 ?
D (I)
both pred. & ground have only 1 match?
no event label
#segments = 1 ?
Correct (Correct)
Correct (Correct) & U (O)
Segment error algorithm
for every segment
match?
Match
between matches in a Merge?
MD
forms part of Insertion ?
between matches in a Fragm.?
IF
forms part of Deletion ?
OD
OU
ID
IU

Fig. 2. (Left) flowchart of algorithm for assigning error labels to each ground truth event, and to each prediction event: for processing ground truth events, use F,D and U, for fragmenting, deletion and underfill; for processing prediction events, use bracketed labels (M), (I) and (O), referring to merge, insertion and overfill errors respectively; *correct* or *no label* can be assigned to both. *#segments* refers to the number of segments that make up an event, *#match* refers to the number of matching segments in that event, with a *match* defined as a segment where ground truth and prediction agree. (Right, boxed) flowchart of algorithm for assigning error pair labels to a segment based on its constituent event error designations.

4.1 Event analysis

There are four types of event error, each falling into one of two divisions depending on whether they are part of the ground truth or prediction sequence. A *positive* error in the prediction sequence, can be defined as either:

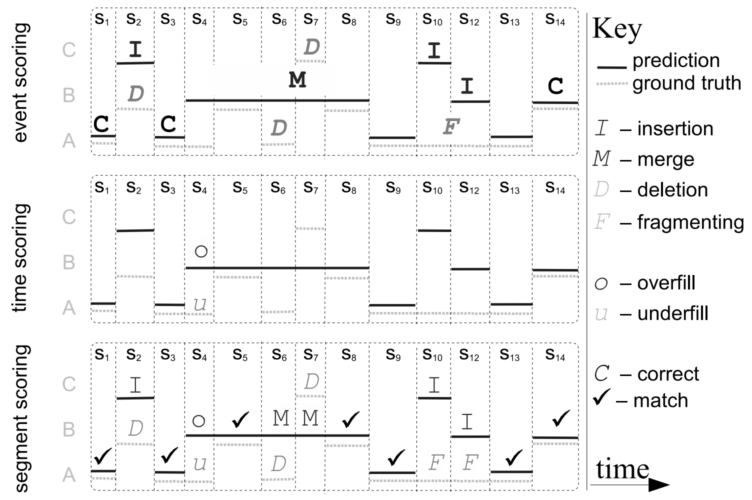


Fig. 3. Some possible error combinations for three class (A, B, C) example: upper diagram shows event errors, middle diagram shows event timing errors, and lower diagram shows segment error pairs. The dotted vertical lines show how the sequence is broken up into segments $s_{1..14}$. Event error labelling is shown in boldface (**C,I,D,M,F**), and is distinct from the segment labelling in that it applies to an entire event rather than just one segment.

***Insertion* -** a prediction event that contains no matching segment(s), or
***Merge* -** a prediction event that contains more then one matching segment.

A *negative* error, the failure to detect all or part of an event in the ground truth, is defined as either:

***Deletion* -** a ground truth event that contains no matching segment(s), or
***Fragmentation* -** a ground truth event that contains more than one match.

Correct is only assigned to where both prediction and the corresponding ground truth events are free from *all* the above categories. The example of Figure 3 shows 3 such correct event scores (s_1, s_3 and s_{14}). There are some cases where a single-segment, matched event is not assigned any designation (for example, see the merged ground events s_5 and s_8 in Figure 3). On an event analysis, these cannot be said to be correct - but neither can they be called errors. Instead, we treat these cases only as segment level matches.

Positive and negative errors are related: an insertion in the prediction sequence, for example, can result in the deletion or fragmentation of an event in the ground truth. This relationship is not always one-to-one however: a fragmentation might be caused by more than one insertion, possible of different classes. For this reason, the two scorings - positive and negative - are kept separate at event level.

Event timing Often an event might be judged correct (or merged, or fragmented) but fail to align completely with its boundaries. The prediction event might spill over the ground truth boundaries; or it might fall short of them. For these cases, we introduce two *event timing* error categories which can be applied to an event in addition to a correct, merge[4], or fragmenting score:

Underfill - *ground truth* event not completely covered by prediction.
Overfill - *prediction* event which spills over its ground truth boundary.

The algorithm for assigning both event errors and event timing errors is shown to the left of Figure 2.

Event error and timing error representation Counts of the four types of event error - insertion, deletion, merge and fragmentation - can be summed up for each class and presented in a simple table, one entry for each error type and each class. Similarly, counts of the timing event errors - overfill and underfill - can also be summed up and presented, in a separate table, alongside the specific time lengths (or number of frames) associated with them.

4.2 Segment analysis

One aspect of performance which event based scoring does not capture is the absolute time duration (in terms of frames or seconds) for each type of error. Additionally, subtle information such as the cause-effect relationship between prediction and ground truth errors is not captured. It can be shown that the following pairings are possible:

1. An event is deleted by insertions, merging, or overfilling of another class
2. An event is underfilled by either an overfill or an insertion of another class
3. An event is fragmented by insertion(s) of another class.

Rarely do event level comparisons allow a one-to-one relation between the prediction and ground truth. One deletion, for example, might be the result of a combination of an overfill plus several different insertions. Segments do allow such a relation. By definition, every segment forms part of exactly one prediction event and one ground truth event. The specific combination of event and timing errors for each ground truth and prediction can therefore be used to define the segment error type, as detailed in Figure2(right). In total, there are six possible error types for non-matching segments based on the event combinations: *insertion-deletion*(ID), *overfill-deletion*(OD), *merge-deletion*(MD), *insertion-underfill*(IU), *overfill-underfill*(OU) and *insertion-fragmentation*(IF).

These pairings are codified and presented in Table 1, which we name the Segment Error Table (SET). Prediction errors (insertion, overfill and merge) form the rows, while ground truth errors (deletion, underfill and fragmentation) make up the columns of this table.

[4] Segment s_4 of Figure 3 shows one such example of this.

Table 1. Possible segment error designations: rows represent prediction segment errors, columns ground truth errors; ID=*insertion-deletion*, OD=*overfill-deletion*, MD=*merge-deletion*, IU=*insertion-underfill*, OU=*overfill-underfill* and IF=*insertion-fragmentation*

	Deletion	Underfill	Fragmentation
Insertion	ID	IU	IF
Overfill	OD	OU	
Merge	MD		

Analysis of segments provides an unambiguous assessment of errors. In the simplest analysis, segment counts of the six different error types, ID, IU, IF, OD, OU, and MD are made and filled into the table. Additional information on the absolute time length, or frame counts, of these segments can also be included. Such a combined segment and frame count SET provides a representation of error that combines the temporal resolution of frame by frame evaluation with the descriptive power of event level evaluation.

NULL as a special case We can expand the table thus described to handle $NULL$ as separate from the other classes - a separation required for most activity recognition tasks. This is achieved by the addition of rows and columns denoting the six error combinations with respect to $NULL$, as shown to the left of Table 2[5]. The SET to the top left corner of the expanded table now only contains information regarding substitution errors between non-$NULL$ positive classes. The top right section of the table then gives a breakdown of false positive errors, while the bottom section gives information about false negative errors.

Table 2. Segment Error Table with $NULL$(N) as special case: full table (left) and reduced version (right)

	D	U	F	D_N	U_N	F_N
I	ID	IU	IF	ID_N	IU_N	IF_N
O	OD	OU		OD_N	OU_N	
M	MD			MD_N		
I_N	$I_N D$	$I_N U$	$I_N F$			
O_N	$O_N D$	$O_N U$				
M_N	$M_N D$					

	D	U	F	N
I	ID	IU	IF	I
O	OD	OU		O
M	MD			M
N	D	U	F	

In many continuous recognition scenarios we are not interested in whether a ground segment labelled $NULL$ has been completely deleted, fragmented or underfilled; likewise we are not interested whether a positive class deletion was caused by an insertion or an overfilling of $NULL$. In such situations, the error designations can be combined to produce a reduced table, as shown to the right of Table 2. For convenience, we drop the 'N' suffix and the dual error designation

[5] Similarly, such an expansion can also be carried out for every class in a system, leading to an enhanced SET with greater detail on the relations between different classes (i.e. similar to the confusion matrix).

from the errors involving *NULL*, referring to them directly as I, O, M, D, U and F. The remaining *substitution* errors retain the dual OU, IU, etc. designators.

5 Discussion

5.1 Application of method to worked example

We now apply the described error characterisations to our examples from Section 2, and give examples of how the event, timing and SET representations might look.

Event and timing results Treating *NULL* again as a special case, we present counts of the non-null class insertions, deletions, merge and fragmentation for the examples of Figure 1. Comparing the insertion and deletion counts of the earlier event analysis of Section 2 with those of 3, we can draw much the same conclusions. Notably however, the new method allows us to see clearly the additional merge and fragmentation errors which prevail in example (c) . The poorer timing performance of (a), with many overfilled events in comparison with the other examples, is also now evident.

The information regarding class substitution errors, however, has been lost in this representation - they are dissolved into pairs, such as insertion/deletion. The lack of a one-to-one relationship between prediction and ground truth errors makes such a joint 'substitution event' measure difficult to define at the event level. Therefore we defer to the segment analysis to provide this information.

Table 3. Event errors (for Positive, non-*NULL* classes only), I'=Insertion, D'=Deletion, M=Merge and F=Fragmentation; and event timing errors, Overfill and Underfill. Number of timing event errors are given together with the corresponding frame counts

#events	*a)*	*b)*	*c)*
I'	4	12	4
D'	2	2	2
M	1	0	1
F	0	1	3

#timing(#frames)	*a)*	*b)*	*c)*
Overfill	8 (18)	1 (1)	4 (6)
Underfill	4 (6)	3 (11)	4 (12)

Segment (and frame-by-frame) results The segment and frame errors for the examples are presented in Table 4. The major difference which becomes apparent is the higher proportion of segments forming part of timing errors (Underfilling by *NULL*, U and Overfill onto *NULL*, O) in (a), versus the higher proportion forming event errors in (b) and (c) . Note that examples (b), and in particular (c), contain fragmenting errors whereas (a) does not. Of merger errors, there are only two instances - in (*a*) and (*c*) - each of which involves only a single merge of two 'drawer' events.

Again, the information provided by the new method is clearly more detailed than that of the basic frame-by-frame analysis.

Table 4. SETs for positive classes (P) vs. *NULL* for the examples in Figure 1, with counts of segment errors and corresponding number of frames

a)	#segments (#frames)			
	D	U	F	N
I	$1_{(1)}$			$5_{(7)}$
O				$8_{(18)}$
M				$1_{(2)}$
N	$1_{(11)}$	$4_{(6)}$		

b)	#segments (#frames)			
	D	U	F	N
I	$2_{(3)}$		$1_{(2)}$	$9_{(25)}$
O				$1_{(1)}$
M				
N	$1_{(2)}$	$3_{(11)}$	$1_{(1)}$	

c)	#segments (#frames)			
	D	U	F	N
I	$1_{(1)}$		$3_{(11)}$	$2_{(3)}$
O				$4_{(6)}$
M				$1_{(4)}$
N	$1_{(1)}$	$4_{(12)}$	$1_{(7)}$	

5.2 Significance and Limitations

As shown above our scheme has three advantages over standard performance evaluation methods used in activity recognition:

1. It introduces the notion of segments as the largest continuous time slices in which no ambiguities occur in scoring the correctness of the predictions
2. Based on this notion it leads to an *unambiguous*, objective characterization of event level error.
3. It makes explicit different sources of error (timing, fragmentation merges) which are ignored in conventional evaluation methods, although they are wide spread in activity recognition systems.

The main limitation of the method concerns events with a large time shift between ground truth and the prediction. A prediction that is shifted by so much, that it has no overlap with the corresponding ground truth will be scored as an insertion and the corresponding ground truth event as a deletion.

The above advantages and limitations clearly follow from the algorithm described in this paper. As the algorithm is fully deterministic and an exact method rather then a heuristic, there is no further need of an empirical validation. There can also be little doubts concerning the benefits of having an objective,*unambiguous* method for scoring events. Even if it were to turn out that in most cases the scores produced by our method are very similar to what people have arrived at to date, having a consistent, objective scoring method is an undisputed methodological advantage.

What certainly does require further investigation are the benefits of the additional error information. They are obviously dependent on the application in which the recognition system is to be used. For a safety critical system, such as an accident avoidance system in an industrial setting, timing may be regarded as critical, and the minimization of overfill and underfill of recognized activities would clearly be desirable. On the other hand, for a system interested only in which activities are carried out, such errors would be less critical. Imagine, for example, a system monitoring the sequence of events as a mechanic repairs part of an aircraft engine. What is important then is that the number of insertions and deletions is kept low - that the system does not miss out any activities, and that it gets the sequence correct. If further information on the count of specific activities is required (how many bolts have been removed from the engine), then errors such as fragmenting and merge errors must also be kept to a minimum.

For a conclusive proof of the value of the information provided by our method an elaborate empirical study is needed. Such a study would need to consider a wide range of applications and preferably look at previously published activity recognition experiments and re-score their results using the above method.

For a meaningful study access to data from different groups would be required and the associated effort would beyond the scope of this paper. This is clearly a limitation and means that no authoritative statement can be made about the value of the additional error information. Nonetheless such benefits are very plausible. Considering the undisputed benefit of an objective scoring method we believe that this paper consist a valuable contribution to the community.

5.3 Work in related fields

Some of the problem domains closest to continuous activity recognition are perhaps line detection in 2D Graphics [22] and video analysis [7,10]. Consider the case of a 2D line: the ground truth indicates a single line, but the recognition system might return a sequence of shorter lines. Further, these might overlap with the ground line, or be slightly offset from it. Different approaches have been suggested to tackle this problem of fragmentation. One suggestion is to redefine the error measures to incorporate fragmented events as some lower weighted correct event[22].

Some decision function based on a measure of closeness might also be used; perhaps utilising fuzzy error margins (as suggested at TRECVID '03[20]). However this approach, as with weighting, requires the introduction of further parameters which only serve to further complicate the evaluation process. In addition, all of these approaches aim to "cover up" the problem rather than finding a way of presenting it as a result in itself.

In extreme cases, particularly in the vision domain, the problem of finding a suitable measure is sidestepped altogether in favour of showing typical example images (as commented by Hoover*et al.* [9] and by Müller [19]). This is an approach which has - out of necessity for lack of a standard measure - been used by researchers publishing in the activity domain. The trouble is that although valid for establishing the feasibility of a method with a small number of samples, it does not scale up well to comparative studies with large databases.

Time series matching methods More generally, the performance evaluation problem can be viewed as the matching of two time series - the prediction output with a trusted ground truth. Time-series similarity methods are used in an extremely wide variety of domains - astronomy, finance, chemistry, robotics, etc., to mention only a few. Even more vast is the number of performance measures that are introduced for every specific application (Keogh & Kassetty[14] give an extensive overview). Some of the more common similarity measures are generally based on dynamic time warping (DTW)[3], or methods using longest common subsequences (LCS)[1]. Another useful method, as introduced by Perng *et al.*[4] utilises 'landmarks' in the data, applying several different transformations (shifting, time warping, etc.) to approximate a more human perception of similarity. Though useful in measuring similarity, these methods do not provide a clear means of measuring phenomena such as event fragmenting and merging.

Rather than selecting some measure of "similarity", or parametrized boundary decision to fit existing error designations, we aim to characterise and present the errors as they are - in a quantifiable way which corresponds closely to that of the human observer.

6 Conclusion

In this paper we present a non-ambiguous scoring of event errors in a continuous activity recognition system. Observing the lack of a one-to-one relationship between events in the ground truth and those in the prediction sequence, we target errors in these two sequences separately: specifically, we define positive errors as *insertion (I)* and *merge (M)* events by the prediction sequence; and negative errors as *deleted (D)* and *fragmented (D)* events in the ground truth. Complementary to these, we introduce timing event categories which score whether a prediction event overfills its ground truth, or a ground event is underfilled by its prediction.

We introduce a timewise method of comparison based on the idea of segments - a segment being a contiguous section of time where neither ground truth nor prediction changes. This allows the representation of an unambiguous one-to-one relation between ground and prediction segments, which we have shown to produce a maximum of six possible error combinations, each assigned depending on the nature of the events to which each segment forms part: ID, IU, IF, OD, OU, and MD. These error pairings can be represented in the so-called *Segment Error Table (SET)*, with scoring on the number of segments, and their corresponding time durations (or number of frames).

The paper has presented a detailed description of the evaluation algorithm and demonstrated how the above mentioned properties follow from this algorithm. As the algorithm is deterministic and exact, no empirical study is needed to prove those properties. With respect to the usefulness of the additional information provided by our method only a simple illustrative example and a plausibility argument were given. As a consequence the main motivation behind this paper is to make the community aware of the existence and the properties of the method. We hope that this will lead to other groups adopting this method and/or to a wider discussion about appropriate evaluation standards. In the end we would like to see the emergence and adoption of a generally accepted, objective and informative evaluation method based on our ideas.

References

1. R. Agrawal, K.-I. Lin, H. S. Sawhney, and K. Shim. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In *21st Int'l Conf. on Very Large Data Bases*, pages 490–501, Zurich, CH, 1995. MKaufmann.
2. L. Bao and S. Intille. Activity recognition from user-annotated acceleration data. In *Pervasive, LNCS 3001*, 2004.
3. D. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *Proc. KDD Workshop*, pages 359–370, Seattle, WA, 1994.
4. C-S.Perng, H.Wang, S.R.Zhang, and D.S.Parker. Landmarks: a new model for similarity-based pattern querying in time series databases. In *ICDE*, 2000.
5. A.F. Clark and C.Clark. Performance characterization in computer vision a tutorial. In *http://www.peipa.essex.ac.uk/benchmark/*, Essex, UK, 1999.
6. R. Duda, P. Hart, and D. Stork. *Pattern Classification, 2nd Edition.* Wiley, 2001.
7. S. Eickeler and G. Rigoll. A novel error measure for the evaluation of video indexing systems. In *Int'l. Conf. on Acous., Speech & Sig. Proc.*, Jun 2000.
8. H.Junker, J.A.Ward, P.Lukowicz, and G.Tröster. *Benchmarks and a Data Base for Context Recognition.* ISBN 3-9522686-2-3, 2004.
9. A. Hoover, G. Jean-Baptiste, X. Jiang, P. Flynn, H. Bunke, D. Goldof, K. Bowyer, D. Eggert, A. Fitzgibbon, and R. Fisher. An experimental comparison of range image segmentation algorithms. *IEEE Trans. PAMI*, 18(7):673–689, 1996.
10. W. Hsu, L. Kennedy, C.-W. Huang, S.-F. Chang, C.-Y. Lin, and G. Iyengar. News video story segmentation using fusion of multi-level multi-modal features in trecvid 2003. In *ICASSP*, May 2004.
11. A. Ali J.K. Aggarwal. Segmentation and recognition of continuous human activity. In *IEEE Workshop on detection and recognition of Events in Video*, pages 28–35, Vancouver, Canada, 2001.
12. H. Junker, P. Lukowicz, and G. Tröster. Continuous recognition of arm activities with body-worn inertial sensors. In *Proc. IEEE Int'l Symp. on Wearable Comp.*, pages 188–189, 2004.
13. T. Kanungo, G. A. Marton, and O. Bulbul. Paired model evaluation of ocr algorithms. Technical report, Center for Automation Research, Uni.Maryland, 1998.
14. E. Keogh and S. Kasetty. On the need for time series data mining benchmarks: a survey and empirical demonstration. In *8th int'l. conf. on knowledge discovery and data mining*, pages 102–111, NY, USA, 2002. ACM.
15. N. Kern, H. Junker, P. Lukowicz, B. Schiele, and G. Tröster. Wearable sensing to annotate meeting recordings. *Personal and Ubiquitous Computing*, 2003.
16. K. Kunze, P. Lukowicz, H. Junker, and G. Troester. Where am i: Recognizing on-body positions of wearable sensors. In *LoCA*, volume 1, May 2005.
17. M. Lampe, M. Strassner, and E. Fleisch. A ubiquitous computing environment for aircraft maintenance. In *ACM symp. on Applied comp.*, pages 1586–1592, 2004.
18. J. Lester, T. Choudhury, N. Kern, G. Borriello, and B. Hannaford. A hybrid discriminative/generative approach for modeling human activities. In *IJCAI*, 2005.
19. H. Müller, W. Müller, D. McG. Squire, S. Marchand-Maillet, and T. Pun. Performance evaluation in content-based image retrieval: Overview and proposals. Technical report, Uni. Geneve, Switzerland, 1999.

20. NIST. *Proc. of TREC Video Retrieval Evaluation Conference (TRECVID)*. 2003.
21. V. Pavlovic and J. Rehg. Impact of dynamic model learning on classification of human motion. In *Comp. Vision and Pattern Rec. (CVPR)*, pages 788–795, 2000.
22. I.T. Phillips and A.K. Chhabra. Empirical performance evaluation of graphics recognition systems. *IEEE Trans. PAMI*, 21:9:849–870, 1999.
23. P.Lukowicz, J.A.Ward, H. Junker, G. Tröster, A.Atrash, and T.Starner. Recognizing workshop activity using body worn microphones and accelerometers. In *Pervasive, LNCS*, 2004.