Diss. ETH No. 16520

Activity Monitoring: Continuous Recognition and Performance Evaluation

presented by

Jamie A. Ward
BEng CS&E (hons.) Edinburgh
born 24th March 1979
citizen of United Kingdom

Timestamp: 15th June 2006 12:12 A.M.

7

Evaluation and optimization of performance[1]

Evaluating the performance of a continuous context recognition system can be a challenging problem. To-date there is no widely accepted standard for dealing with this, and methods and measures are usually taken from related fields such as speech and vision.

In this chapter we attempt to identify and characterise the errors typical to continuous context recognition. We introduce a means of quantifying these errors in an unambiguous manner. In an initial investigation, we score the errors in an example taken from previous work, and discuss the advantages that the proposed method provides over two of the most commonly used approaches.

---

[1]This chapter is based on work to appear in [61]

7.1. Introduction

In previous chapters, several different methods of recognising continuous activities were presented and compared. To help evaluate these a number of standard measures were used, such as precision and recall, calculated from the timewise, frame-by-frame, comparison of prediction sequences with their corresponding ground truth; and counts of insertions and deletions based on comparisons where the unit of measure is an activity event. Each strategy was chosen with the aim of providing the most relevant and critical information for analysis of the systems being presented. A combination of different evaluation strategies, such as used in the later result sections of Chapter 5, can give a fuller account of the information that might be necessary, both for the designer optimising a system, and for the developer charged with finding the most suitable recognition solution for an application.

Unfortunately, as also highlighted in Chapter 5, existing strategies of performance evaluation fail to capture - or obscure - some of the information that might be useful for the designer of an activity (or context) recognition system. One aspect of a typical continuous context recognition system which existing evaluation strategies fail to address is the problem associated with imprecise, variable duration event boundaries. This was initially tackled by the introduction of measures such as *overfill* and *underfill*, representing cases where recognised events spill over, or fail to cover, the boundaries marked by ground truth (see 5.5.6). Though sufficient for the purposes of the described experiments, these measures do not go far enough in more general categorisation of context performance. For one, there is no standard method in context for dealing with events that become fragmented into several smaller events; nor is there a method for dealing with cases where several events become merged into one.

7.1.1. Chapter contributions and organisation

The chapter is divided into four main parts. In 7.2 we motivate the work by highlighting the problems of existing measures when used on a context recognition example taken from previously published work. In 7.3 we provide a detailed analysis of these problems, together with an overview of evaluation methods in related fields and the issues involved with their use. Section 7.4 introduces our proposed methods to combat these problems. These are then, in 7.5.1, applied to the original example and used to fuel the discussion on how they might be used in practice.

7.2. Motivation

As a motivation for this work, consider Figure 7.1. 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 [30]. 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.

Visually, most of the non-*NULL* activities in (*a*) seem to correlate well with their ground truth: there are few insertions and only one completely deleted activity. The middle (*b*) and bottom (*c*) plots of Figure 7.1 tell a different tale: the abundance of insertions in (*b*) and the heavily fragmented output of (*c*) both contribute to the conclusion that these results are much poorer than that of (*a*). This is assuming, of course, that we are more interested in the correct ordering and contiguous nature of our prediction events than we are about their specific time durations.

7.2.1. Frame based analysis

When evaluating timewise prediction sequences a standard practice is to make frame-by-frame comparisons with ground truth. Counts of correct and incorrect frames can then be tallied for each class and entered into a confusion matrix [50]. From this a number of standard performance measures can be calculated, the most common of which is accuracy (overall correct rate) but also increasingly the dual metrics precision and recall.

However, frame-by-frame analysis has its limitations. These can be seen by refering again to the examples of Figure 7.1. The frame-by-frame confusion matrices for these, simplified to the summation of positive classes vs. *NULL*, are shown in Table 7.1. Note how the accuracies calculated for each of the examples tell nothing about their differences - they are all identical. In addition the confusion matrices for (*a*) and (*b*) are also 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
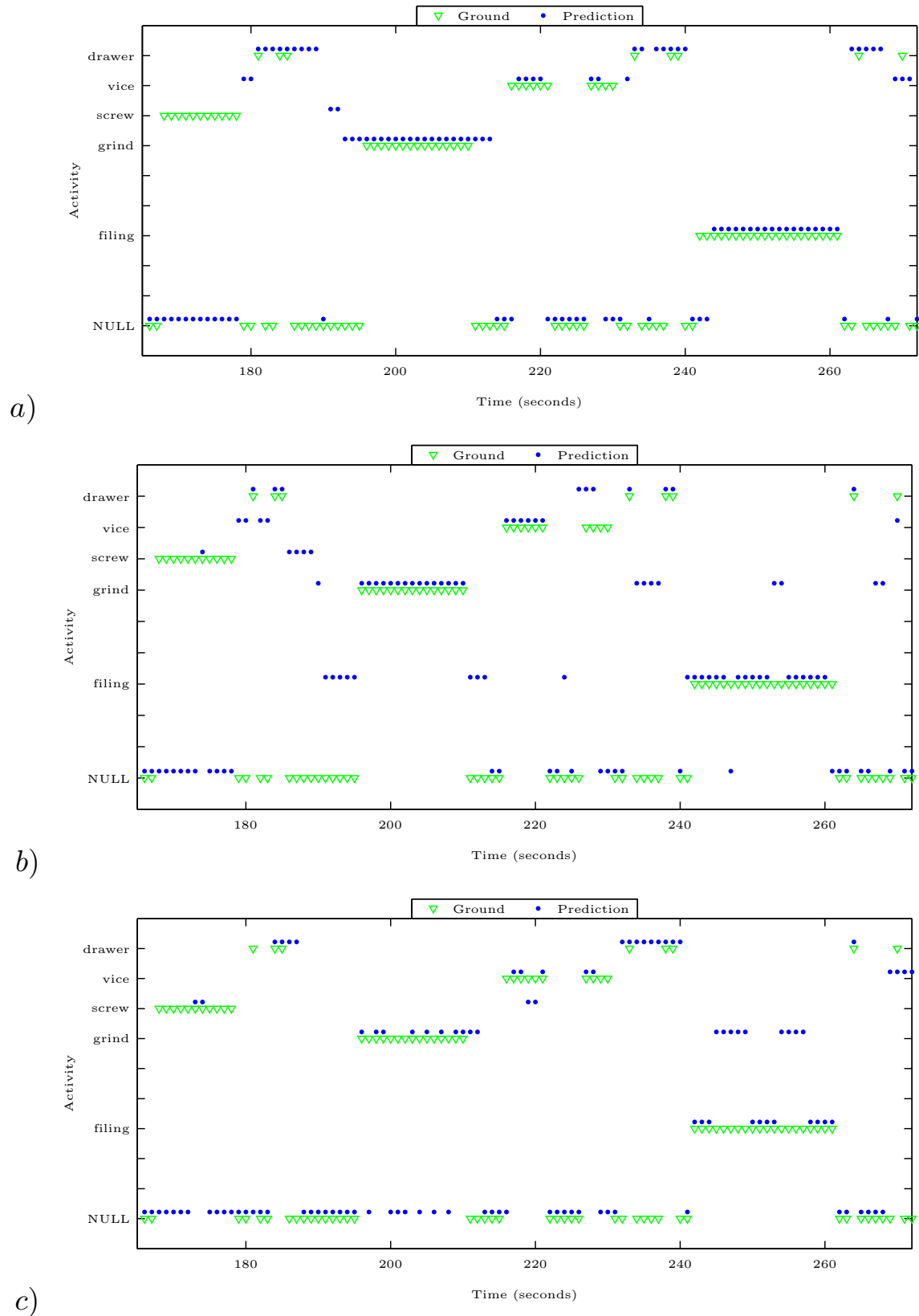
a)

b)

c)

Figure 7.1: *Multi-class continuous activity problem, examples (a − c). All examples have identical accuracy in frame-by-frame comparison. Note the prevalence of inserted events in b, and fragmented events in c.*

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.

7.2.2. Event analysis

Researchers in the fields of optical character recognition (OCR) [62, 63] 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. This is a standard approach which is also used in context recognition.

When these scores are calculated for the examples (see Table 7.2), the differences between (a) and (b) become much more appartent. This time (a) clearly has a lower insertion count than (b). However, if we look at example (c) - again a very different output from (a) - we are once again disappointed: its deletion, substitution and insertion counts 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 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: two or more events of the same class can be recognised as a single large event. In the examples given here this happens on two occasions, both involving the 'drawer' class (the first two instances of (a), and the third and fourth instances in (c)). In each case this error 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. In other instances it might be decided (by the system designer) that merging two separate events constitutes one correct event and one deleted event.

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 the examples given here, this is not always the case.

a)	Predictions			b)	Predictions			c)	Predictions			
	Pos.	Null			Pos.	Null			Pos.	Null	Total	
P	**46**(1)	17		P	**45**(5)	14		P	**32**(12)	20	64	
N	27	**16**		N	26	**17**		N	13	**30**	43	
	$acc_s = \mathbf{57.9\%}$				$acc_s = \mathbf{57.9\%}$				$acc_s = \mathbf{57.9\%}$		107	

Table 7.1: *Performance 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* $T_f = 107$.

	I_e	. D_e	S_e	err_e
a)	3	1	1	**45.5%**
b)	10	0	2	**109.0%**
c)	3	1	1	**45.5%**

Table 7.2: 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,* $T_e = 11$

7.3. Problem specification and existing methods

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 identify specific characteristics of performance that are common to continuous activity recognition.

7.3.1. 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 of the recognised sequence might be shifted in time. 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 that 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.

7.3.2. General considerations

Given two time series there can be no such thing as an optimal measure of similarity that 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 that emphasise correct ordering over the specific duration of symbols 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 that 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 most critical types of error. This means that evaluation methods that are successful in one field need not necessarily be so in another. Applying methods from one domain to another only makes sense if both domains have the same types of dominant error; in addition, similar relevance should be assigned to equivalent errors.

7.3.3. 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.) [33–35, 64], tracking of specific procedures (e.g. assembly tasks [65]), and the detection of changes in environmental conditions [36, 64]. 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 can 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. Direct frame-by-frame evaluation can be misleading in these 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. A recognition system designed to spot such situations will also be 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. Fragmentation breaks what should be one long event in the ground truth into several smaller 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' because 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 or situations while regarding the rest as instances of a 'garbage' or *NULL* class. A *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 that 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 sets of 'real life' data it is often impossible to perfectly time ground truth labels by hand. The definitions for start and stop times of an event are often arbitrary and imprecise. This is particularly so in domains such as activity recognition, where the notion of an 'event' is often difficult to define - 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 be working 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.

7.4. Error characterisation and representation

Following from the above observations we now present a character-
isation of the critical error types in continuous activity recognition.
Specifically we propose an approach that (1) includes event mergers
and fragmentation as errors in their own right; and (2) provides infor-
mation about event timing errors. This section presents both a defini-
tion 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 7.2 shows a three
class recognition problem broken up into 14 segments (denoted by the
vertical dotted lines). A segment is a variable-duration, contiguous se-
quence of frames during which neither prediction nor ground truth label
change. 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 context 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 tim-
   ing errors. Prediction and ground truth events are scored sepa-
   rately. The flowchart of Figure 7.3 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).
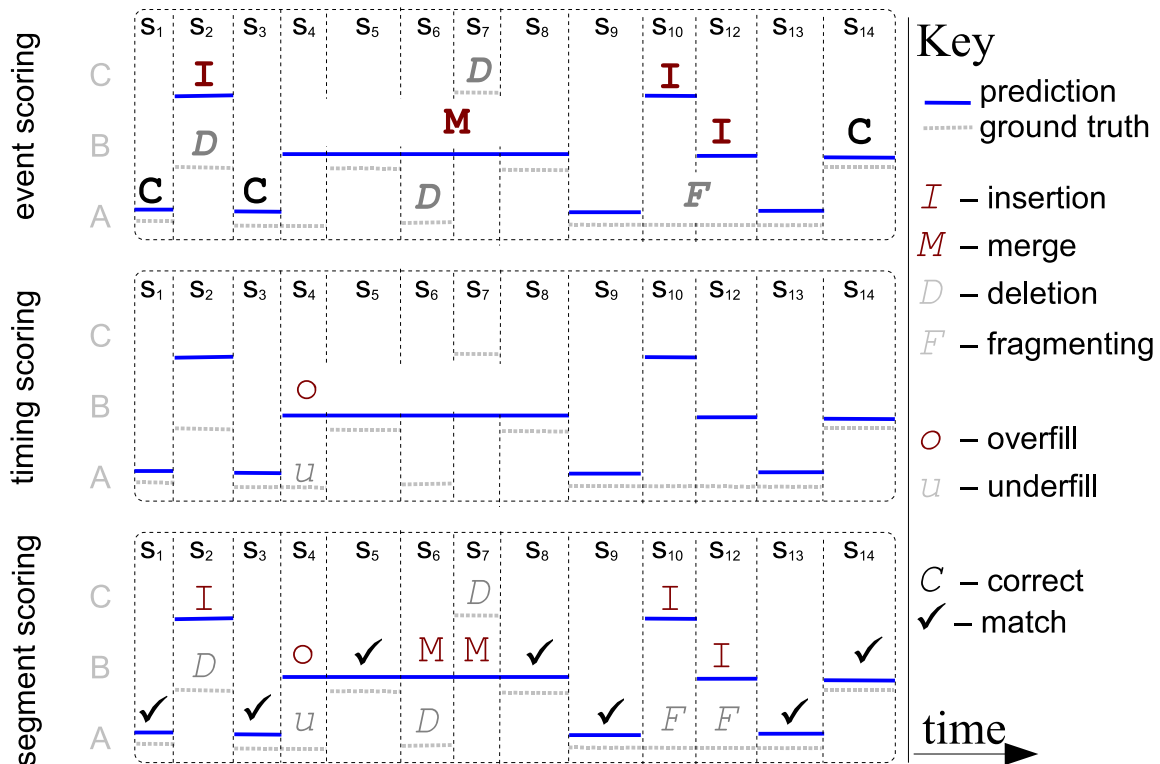
Figure 7.2: *Some possible error combinations for a single, three class* (A, B, C) *example (the special class* NULL *is not considered here): upper diagram shows event errors, middle diagram shows event timing errors, and lower diagram shows segment error pairs. Dotted vertical lines show how the sequence is broken up into segments* $s_{1..14}$. *Note how the merge event, covering* s_4 *to* s_8, *is made up of OU, match and MD segments.*

Prediction events are scored using the same algorithm, but with outputs: merge (M), insertion(I) and overfill (O).

3. Score the segment errors. The Figure 7.4 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.

In Figure 7.2 we take a single, three-class example and show the results from each of the event, timing and segment scoring algorithms. Note that, because we are initially dealing with the general multiple class problem (we do not give special consideration to *NULL*), all errors are substitutions between the three classes 'A', 'B' and 'C'. The error categories (event, timing and segment) and how to score them, are described in greater detail in the following sections.

7.4.1. 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 of the prediction sequence. A positive error in the *prediction* sequence can be defined as either:

Insertion - a prediction event containing no matches, or

Merge - a prediction event containing more than one match.

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 containing no matches, or

Fragmentation - a ground truth event containing more than one match.

Correct is only assigned when a corresponding prediction and ground truth event is free from *all* of the above errors. 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 7.2). On an event analysis these events cannot be said to be correct - but neither can they be called errors. Instead, we just call them *segment 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. It is for this reason that the event level scoring is carried out on two sequencies; on the ground truth (negative errors) and prediction (positive errors).

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 that can be applied to an event in addition to a correct, merge[2], or fragmenting score:

---

[2]Segment s_4 of Figure 7.2 shows one such example of this.

Underfill - *ground truth* event not completely covered by prediction.

Overfill - *prediction* event which spills over its ground truth boundary.

Four further sub-categories of timing errors might also be used: *delay*, noting an underfill at the beginning of an event; *shortening* an underfill at the end; *preemption*, an overfill at the beginning; and *prolongation*, an overfill at the end. In this thesis we focus only on the two main categories of overfill and underfill and leave treatment of the sub-categories for future work.

The algorithm for assigning both event errors and event timing errors is shown in Figure 7.3.

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.

7.4.2. Segment analysis

One aspect of performance that 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.

Event and timing error algorithm
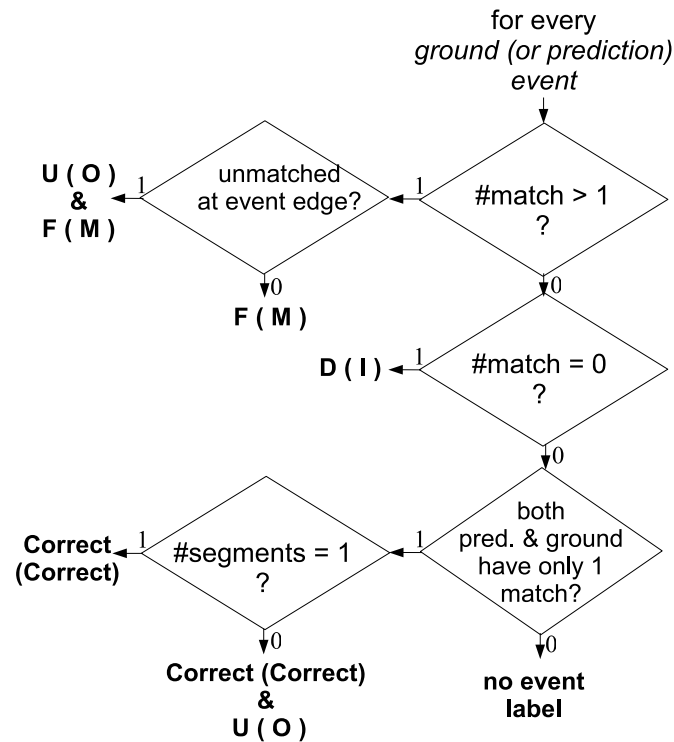
Figure 7.3: *Algorithm for assigning error labels to each ground truth event, and to each prediction event: for processing ground truth events, use F,D and U, for fragmenting, deletion and underfill; for processing prediction events, use bracketed labels (M), (I) and (O), refering to merge, insertion 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.*

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 Figure 7.4. In total there are six possible error types for non-matching segments based on the event combinations. Three of these involve segments forming part (or all) of event errors:

Insertion-Deletion (ID) - forms part (or all) of an inserted prediction and a deleted ground truth

Insertion-Fragmenting (IF) - an inserted prediction that lies somewhere between two matching segments of a fragmented ground truth

Merge-Deletion (MD) - forms part of a merge prediction, occuring somewhere between two matching segments of the same prediction event, causing a deletion in the ground truth

The remaining three error designations involve segments that form part of timing (or both timing and event) errors:

Insertion-Underfill (IU) - forms part (or all) of an inserted prediction and an underfilled ground truth (only assigned if the segment has not already been classified as IF)

Overfill-Deletion (OD) - forms part (or all) of an overfilled prediction and a deleted ground truth (only assigned if the segment has not already been classified as MD)

Overfill-Underfill (OU) - forms part (or all) of an overfilled prediction and an underfilled ground truth

The general rule is to take a non-matching segment, and name it according to the constituent event or timing error designations. Thus a segment that forms part of an insertion event in the prediction sequence, and a deletion in the ground truth, is classed as an *Insertion-Deletion (ID)*; the part of an insertion event which causes an underfilled segment in the ground truth is called an *Insertion-Underfill (IU)*. Similarly, a segment that forms part of an overfill timing error in the prediction and a deletion in the ground truth is classed as *Overfill-Deletion (OD)*; if the ground truth is merely an underfill, the classification is *Overfill-Underfill (OU)*.

Two exceptions to this rule occur when a timing error is assigned in addition to a merge or fragmentation event error. If a non-matching
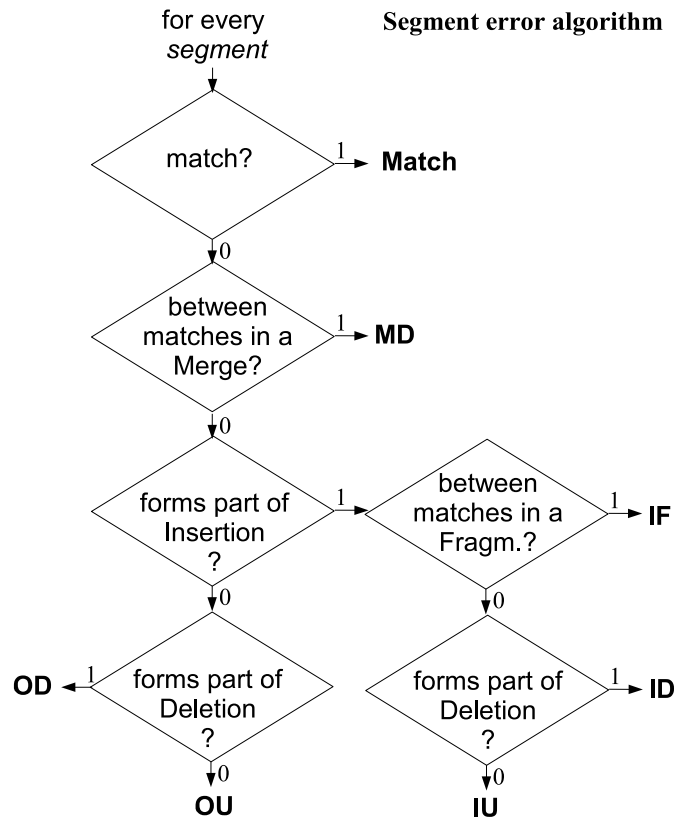
Figure 7.4: *Algorithm for assigning error pair labels to a segment based on its constituent event error designations:* Match *defined as a segment where ground truth and prediction agree; MD*=merge-deletion, *IF*=insertion-fragmentation, *ID*=insertion-deletion, *OD*=overfill-deletion, *OU*=overfill-underfill *and IU*=insertion-underfill.

segment lies between two matching segments of a merge event it is, by definition of merge, partly responsible for a deletion and should be called *Merge-Deletion (MD)*. However, if the predicted merge event is also an overfill and the segment lies outside of the matching segments then it should be refered to as either an *OD*, or *OU* segment (depending on the state of the corresponding ground event).

Similarly, if a non-matching segment lies between two matching segments of a fragmented event it can only be caused by an insertion in the prediction sequence and should be called *Insertion-Fragmenting (IF)*. However, if the fragmented ground event is also an underfill, and the segment lies outside of the matching segments, then it should be refered to as either an *IU*, or *OU* segment.

These pairings are codified and presented in Table 7.3, 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.

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

Table 7.3: *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

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 about the absolute time length, or frame counts, of these segments can also be included. This combined segment and frame count SET provides a representation of errors 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 7.4. 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.

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 in 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 7.4. For convenience we drop the 'N' suffix and the dual error designation from the errors involving $NULL$, referring to them directly as I, O, M, D,

	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	

Table 7.4: *Segment Error Table with* NULL*(N) as special case: full table (left) and reduced version (right)*

U and F. The remaining *substitution* errors retain the dual OU, IU, *etc.*, designators.

Multi-class SET

While the basic SET provides much more information about system performance than event level analysis alone, it still lacks the exact relation between errors of different classes as traditionally represented by confusion matrices. Where more detail is required regarding specific (or all) classes in a multi-class system, SET can easily be extended in a similar manner to that already shown for *NULL* in Table 7.4. By adding three additional columns (D,U and F) and three additional rows (I,O and M) for each class, SET can be extended for any number of classes.

7.5. Discussion

7.5.1. Application of method to worked example

We now apply the described error characterisations to our examples from Section 7.2 and give examples of how the event, timing and SET representations might look.

Event and timing results

Treating *NULL* as a special case, we present counts of the non-*NULL* class insertions, deletions, merge and fragmentation errors, for the examples of Figure 7.1, in Table 7.5. Comparing these counts with the insertion and deletion counts of the earlier event analysis in Table 7.2,

we can draw much the same conclusions. Notably, the new method allows us to clearly see the additional merge and fragmentation errors in example (c) . The poor timing performance of (a), with many overfilled events in comparison to 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 the idea of a 'substitution event' difficult to define at the event level. Instead, we defer to the segment analysis to provide this information.

#events		#timing(#frames)		#events		#timing(#frames)		#events		#timing(#frames)
I'	4			I'	12			I'	4	
D'	2			D'	2			D'	2	
M	1	Overfill 8 (18)		M	0	Overfill 1 (1)		M	1	Overfill 4 (6)
F	0	Underfill 4 (6)		F	1	Underfill 3 (11)		F	3	Underfill 4 (12)

a)   b)   c)

Table 7.5: *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*

Segment (and frame-by-frame) results

The segment and frame errors for the examples are presented in Table 7.6. Now clearly visible is the higher proportion of segments which form timing errors in (a) (Underfilling by *NULL, U* and Overfill onto *NULL, O*), than in the other examples. Correspondingly, a higher proportion forming event errors is found in (b) and (c). These examples 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.

These measures correspond to what a visual inspection of the output might confirm, and provide much more information than a basic frame-by-frame comparison.

Visualisation of SET

The pie-charts of Figure 7.5 give one possible manner in which the SET information might be presented. Another approach, perhaps better for

system comparisons, is to stack the errors in a barchart. For readability these need not necessarily show all of the different error types in the same graph - for example, the substitution errors might be treated together as a single error type, with perhaps a separate graph for all the others.

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 $1_{(1)}$	$4_{(12)}$	$1_{(7)}$	

Table 7.6: *SETs for positive classes (P) vs.* NULL *for the examples in Figure 7.1, with counts of segment errors and corresponding number of frames*

7.5.2. Significance and Limitations

As shown by the examples above, our scheme has three advantages over standard methods of performance evaluation 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, merging) which are ignored in conventional evaluation methods, even though they are widespread 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.

What require further investigation are the benefits of this additional error information. These are dependent on the application for which the recognition system is to be used. For a safety critical system, such
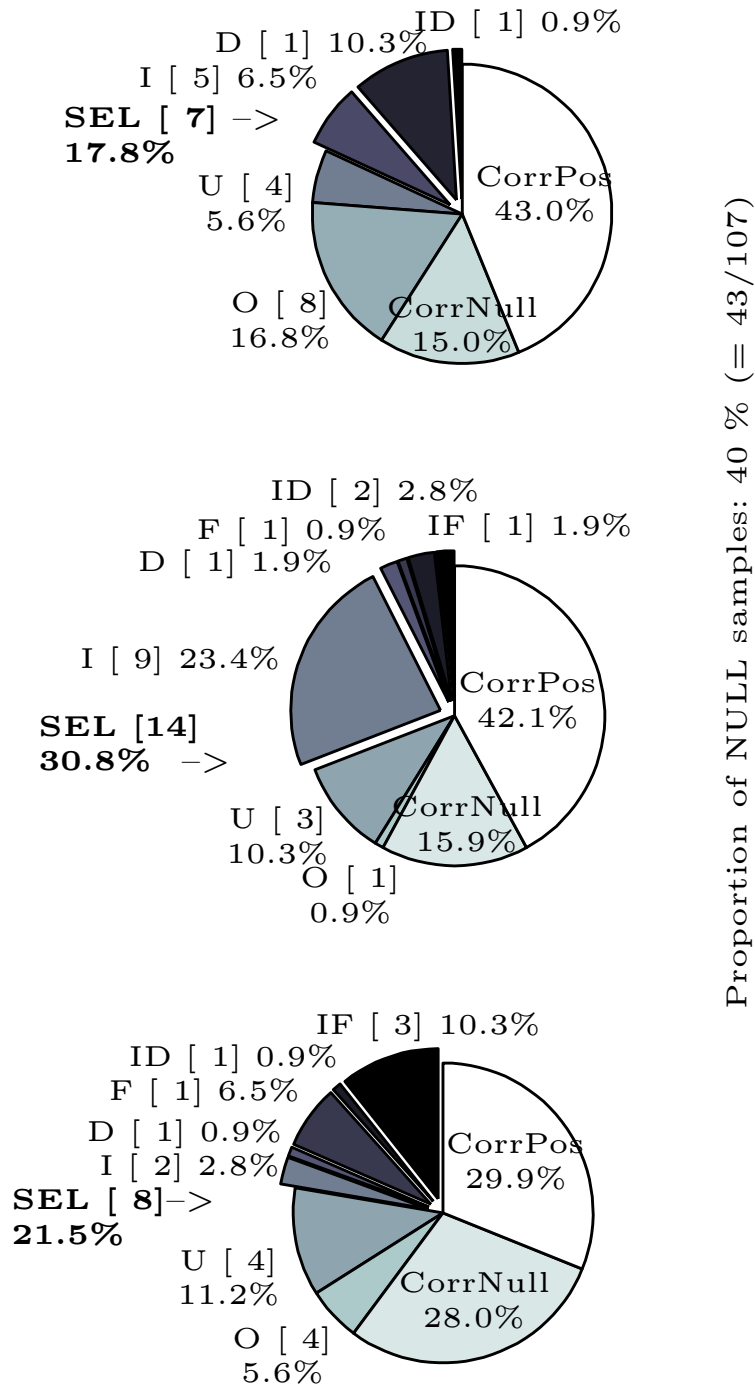
Figure 7.5: *Pie chart visualisation of SET information for the three examples (left (a) to right (b)). Errors given as a percentage of the total time (frames), with the number of segments given [in square brackets]. Exploded segments represent serious errors, the total of which is marked SEL*

as an accident avoidance monitor in an industrial setting, timing may be regarded as critical, thus making the minimization of overfill and underfill of recognized activities 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 here 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 work. 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 work consist a valuable contribution to the community.

7.5.3. Work in related fields

Some of the problem domains closest to continuous activity recognition are line detection in 2D Graphics [56] and video analysis [66, 67]. 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[56].

Some decision function based on a measure of closeness might also be used; perhaps utilising fuzzy error margins (as suggested at [68]). However this approach, as with weighting, requires the introduction of further parameters which only serve to further complicate the evalua-

tion 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.* [37] and by Müller [38]). 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[43] give an extensive overview). Some of the more common similarity measures are generally based on dynamic time warping (DTW)[69], or methods using longest common subsequences (LCS)[70]. Another useful method, as introduced by Perng *et al.*[71] 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.

7.6. Conclusion

In this chapter 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 aim of this chapter is primarily to highlight some of the problems in performance evaluation of context recognition systems, and to suggest a way in which these might be dealt with. The proposed methods of event and segment analysis, and the use of the SET, are intended as starting points from which further discussion in the community can commence. The examples given here are also intended as a preliminary study, and further evaluation of these methods on a wider range of context problems is desirable. Though it is the author's belief that these methods will find general use, it is likely that some revision will be necessary on encountering specific problems not envisaged here.

8

SET optimization for continuous activity recognition

This penultimate chapter aims to synthesize the main ideas presented in the thesis. In particular it provides a re-assessment of the earlier activity recognition methods using the ideas introduced in Chapter 7. Specifically, the Segment Error Table (SET) based strategy for performance evaluation is applied to the earlier results with the aim of provoking a discussion on its utility as a tool for system optimization.
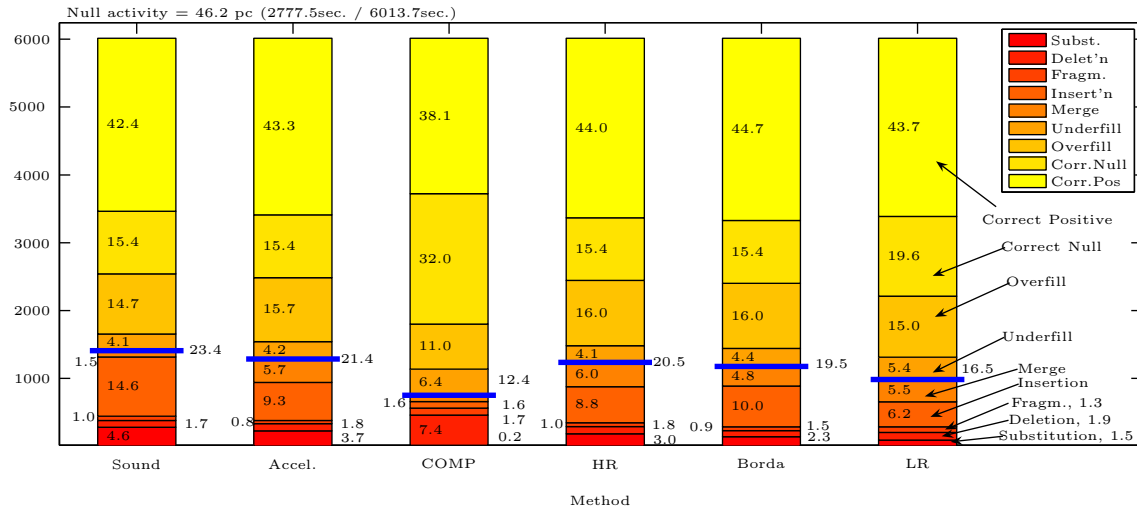
Figure 8.1: *Sound segmented results from Chapter 5, this time accounting for Merge and Fragmenting errors contributing towards the Serious Error Level (SEL)*

8.1. SET analysis of the wood workshop

The time based results of the work described in Chapter 5 and 6 were presented using barchart summaries of the confusion matrices. In these chapters, the idea of overfill and underfill was also introduced, which in the results presented incorporated frames which could be classified as merge and fragmenting errors. From the point of view of the experiment dealt with in this thesis, such errors are regarded as having little importance and the consequent combination of merge time with overfill, and fragmenting time with underfill, could be viewed as acceptable. However with the considerations of Chapter 7 in mind, the definition of *serious error level (SEL)* given in the earlier chapters should now be revised to incorporate merge and fragmenting. In doing so, a slight reduction in the levels of overfill and underfill, with a corresponding rise in serious error can be expected. The revised barcharts, using the full breakdown of overfill, underfill, merge, fragmenting, insertion, deletion and substitution timing errors, are given in Figure 8.1 and 8.2 for the user-dependent results of Chapter 5 and 6 respectably.

8.1.1. (Re-)analysis of results

Though the SEL rises as expected, it rises fairly consistently with all the methods, and the same broad conclusions can be drawn as given
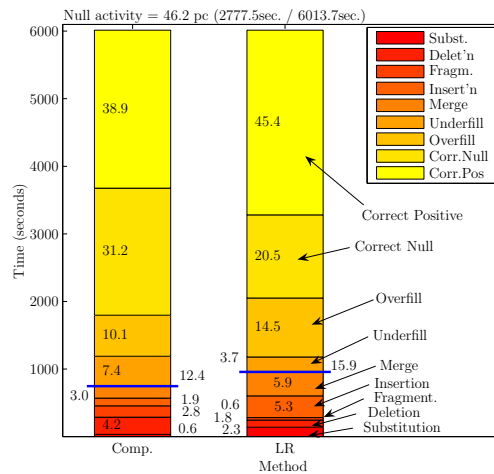
Figure 8.2: *Wrist only, sliding window results from Chapter 6, accounting for Merge and Fragmenting errors contributing towards the Serious Error Level (SEL)*

earlier, that is:

1. Sound based segmentation produces a reasonable recognition performance on its own, and when optimised for high true positives, still maintains less than 2% substitution errors (in user-dependent training).

2. Applying different fusion methods to combine sound and acceleration improves this performance considerably. Comparison of outputs (COMP) provides a 'cautious' recognition, preferring low instances of falsely recognised activities, and almost no substitution errors. Borda count and highest rank (HR) are better at detecting all positive activities, at the expense of insertions. While logistic regression (LR) provides a compromise in performance between the other ranking methods and COMP.

3. Using only wrist-worn sensors, and a simple sliding window segmentation which utilises both sound and acceleration, similar performance is observed to the sound segmented method, and consequently the above conclusions also hold.

When merge and fragmenting errors are taken into account, the following observations can also be made:

1. The user-dependent COMP. method of Figure 8.1 produces an equally low count of merge as it does insertion (both 1.6%). Its

COMP.

Class	T	I	M	D	F	C
NULL	740	41	118	84	21	635
hammer	20	0	0	0	0	20
saw	20	1	0	0	3	17
file	20	2	0	1	8	11
drill	20	2	0	0	3	17
sand	20	0	0	2	4	14
grind	20	2	0	6	3	11
screwd.	20	7	0	4	6	10
vise	173	38	16	35	5	133
drawer	440	11	20	110	0	330
Pos.	753	63	36	158	32	563

LR

Class	T	I	M	D	F	C
NULL	740	44	25	242	44	454
hammer	20	2	0	0	0	20
saw	20	6	0	0	4	16
file	20	7	0	0	7	13
drill	20	8	0	0	2	18
sand	20	0	0	1	3	16
grind	20	9	0	2	4	14
screwd.	20	13	0	3	6	11
vise	180	137	18	15	12	153
drawer	440	51	85	30	0	410
Pos.	760	233	103	51	38	671

Table 8.1: *Event error tables for (left) COMP. and (right) LR methods (using sound segmentation, with $T_{ia} = 0.3$, and user-dependent training). Total number of events (T), insertions (I), merges (M), deletions (D), fragmentations (F), and correct (C) are given for each class. Also given are the results when* NULL *is treated as a class, and the total positive class results (Pos.)*

count of fragmenting is also low (1.7%), with the bulk of its errors being attributed to deletions (7.4%). This helps confirms the 'cautious' nature of this method.

2. LR also produces a low count of fragmentations, but with a much larger degree of merging (1.3% versus 5.5% in Figure 8.1.)

This analysis can be furthered by looking at the absolute counts of class event errors for COMP. and LR, again using the user-dependent, sound-segmented cases, as given in Table 8.1.1. The COMP. method, for example, deletes many instances of drawer, yet very few of the other classes. These classes do however experience much fragmentation. This is in contrast to what might be assumed from the timing report of Figure 8.1. This is largely because the null insertions which cause these fragmentations, though numerous, have very short duration. A system where the occasional, short duration fragmentation can be tolerated, and where most of the classes to be recognised are of a reasonably long duration (i.e. longer than the typical 1-2 seconds of drawer usage), might therefore be suitable for the use of COMP.

In keeping with the report of Figure 8.1, the event response of the LR method in Table 8.1.1 produces a comparably similar number of fragmenting errors to COMP. Also in keeping with this summary, it has fewer deletions too - though the major reason for this is its much better response to the drawer class. For a system with many such short duration events, with a tolerance for a greater number of insertions, the LR method as used in this work would provide the better solution.

A further interesting observation is the results given for the drawer and vise classes of both methods in Table 8.1.1. These classes are unique in that they are both fairly short in duration and that they always occur in pairs: the opening of a drawer is always followed by a short pause before being closed again; and similarly the opening of a vise to place a piece of wood is usually accompanied by the vise being tightened again to hold it. Consequently, these are the only two classes which experience merge errors. In a similar way, as their duration is so short, these classes rarely exhibit fragmenting errors.

This is a facet of the dataset, and raises the (somewhat obvious) importance of selecting not only appropriate measures, but an appropriate dataset for whichever application is being developed. Any investigation into merge errors therefore would require focus on short duration, frequently occurring classes such as these. Conversely, investigation of fragmenting errors requires classes of fairly long duration (such as represented here by the other activity classes).

8.2. Parameter optimisation with SET

In Chapter 5 the response of the sound based segmentation was optimised using a plot of the precision and recall across values across a sweep of the IA threshold T_{ia}. In the analysis presented, the main criteria was for a high recall - on the premise that later fusion would improve precision. However, the use of precision and recall meant that overfill and underfill were not considered. Equally, the effect of T_{ia} on levels of the error categories was not investigated.

As an example of how a SET analysis might be used to investigate the effects of system parameters, Figure 8.3 presents the error barcharts for nine different settings of T_{ia} for two of the sound recognition setups used earlier in the thesis. The first graph (a) presents results for a partitioning and recognition method using only IA+LDA classification on frames. In (b), the frame classifications are smoothed over using a majority vote sliding window in a manner identical to that employed in 5.5.3.

Immediately it can be seen how the first method (a) has a consistently high level of fragmentation across all thresholds, in comparison to the low levels of (b). Even though (a) has very few deletions, it still retains less insertion time than the smoothed version of (b). These observations, not immediately evident in a precision-recall analysis, provide useful information that might allow a designer to further
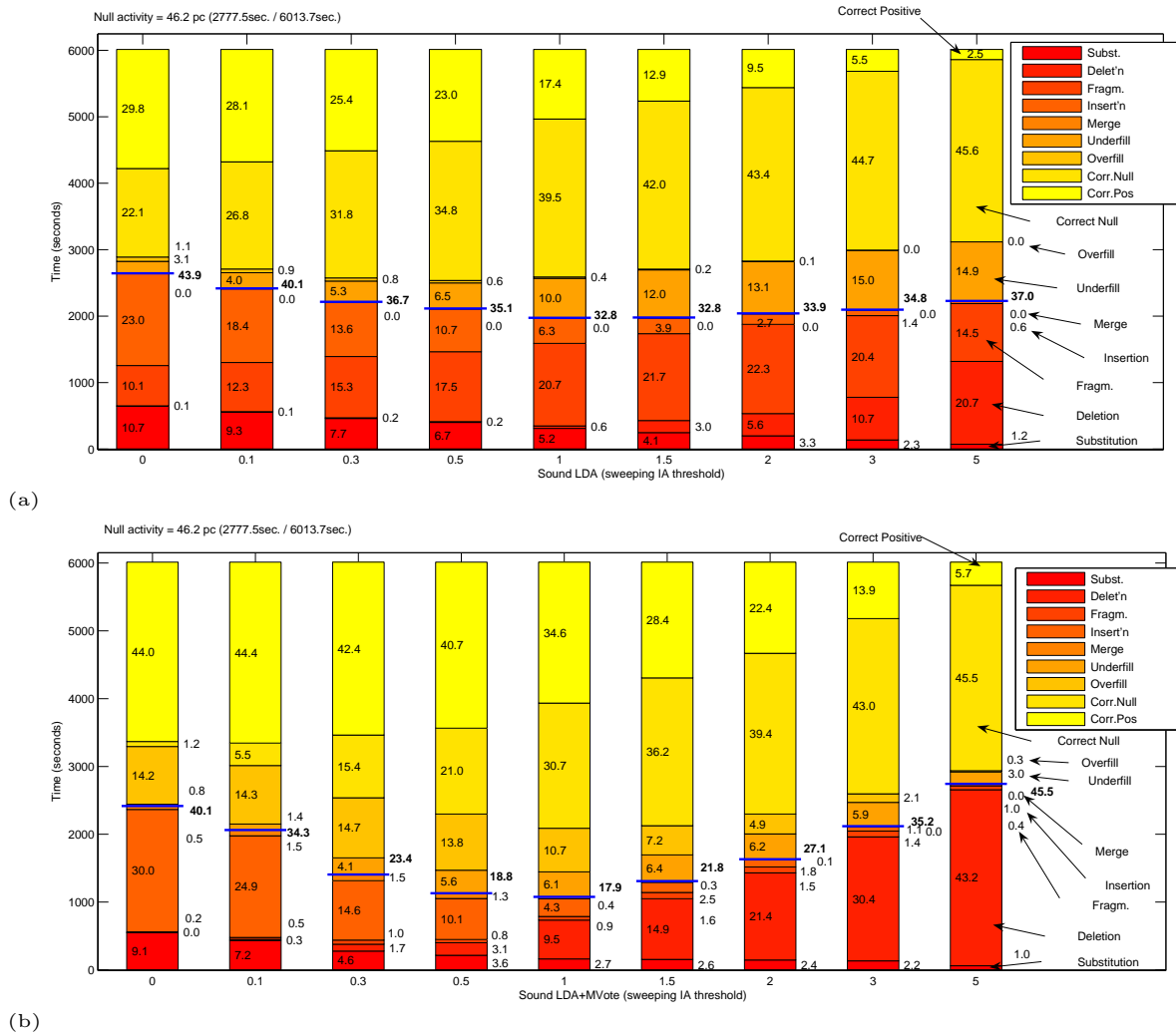
(a)

(b)

Figure 8.3: *Sound analysis across a sweep of T_{ia}, for (a) using LDA classification, and (b) using LDA plus majority vote smoothing. Note the prevalence of fragmenting errors in (a) over (b)*

fine-tune system parameters for different application needs. For example, knowing that many of the false negatives in (a) actually form part of fragmentations rather than deletions, might allow choice of such a setup for an application where a fragmented output is more acceptable than one where events are deleted.

8.3. Conclusion

Segment Error Table (SET) analysis provides a mechanism by which researchers of activity recognition can more completely assess the performance of their systems. This allows easier optimisation, or fine tun-

ing, of system parameters for specific applications. The information provided by SET goes over and above that provided by more traditional methods, such as confusion matrices and standard ASR-based event counts.

Though the basic conclusions drawn in the earlier chapters of the thesis still hold, the use of SET provides a more thorough analysis of those results. It allows researchers to take issues such as fragmentation and merging of output event into consideration in a much clearer, more direct manner, and provides a means by which these can be summarised.

Bibliography

[1] S. Feiner, B. MacIntyre, and D. Seligmann. Knowledge-based augmented reality. *Com. of the ACM*, 36(7):52–62, 1993.

[2] H.Junker, J.A.Ward, P.Lukowicz, and G.Tröster. *Benchmarks and a Data Base for Context Recognition*. ISBN 3-9522686-2-3, 2004.

[3] D. Abowd, A. K. Dey, R. Orr, and J. Brotherton. Context-awareness in wearable and ubiquitous computing. *Virtual Reality*, 3(3):200–211, 1998.

[4] T. Starner, B. Schiele, and A. Pentland. Visual contextual awareness in wearable computing. In *IEEE Int'l Symp. on Wearable Comp.*, pages 50–57, Pittsburgh, PA, 1998.

[5] C. Vogler and D. Metaxas. ASL recognition based on a coupling between HMMs and 3D motion analysis. In *ICCV*, Bombay, 1998.

[6] A. D. Wilson and A. F. Bobick. Learning visual behavior for gesture analysis. In *Proc. IEEE Int'l. Symp. on Comp. Vis.*, Coral Gables, Florida, November 1995.

[7] J. Schlenzig, E. Hunter, and R. Jain. Recursive identification of gesture inputs using hidden Markov models. *2nd Conf. on Applications of Comp. Vision*, pages 187–194, December 1994.

[8] J. M. Rehg and T. Kanade. Digiteyes:vision-based human hand tracking. Technical report, Carnegie Mellon, Dec 1993.

[9] Bussmann J. B. J., Martens W. L. J., Tulen J. H. M., Schasfoort F. C., van den Berg-Emons H. J. G., and Stam H. J. Measuring daily behavior using ambulatory accelerometry: The activity monitor. *Behavior Research Methods, Instrmts.+Comp.*, 33(3):349–356, 2001.

[10] Paolo Bonato. Advances in wearable technology and applications in physical and medical rehabilitation. *J. NeuroEngineering and Rehabilitation*, 2(2), 2005.

[11] Kamiar Aminian and Bijan Najafi. Capturing human motion using body-fixed sensors: outdoor measurement and clinical applications. *Computer Animation and Virtual Worlds*, 15:79–94, 2004.

[12] PH Veltink, HBJ Bussmann, W de Vries, WLJ Martens, and Rob C. Van Lummel. Detection of static and dynamic activities using uniaxial accelerometers. *IEEE Trans. Rehab. Eng.*, 4(4):375–386, 1996.

[13] Aminian K, Robert P, Buchser EE, Rutschmann B, Hayoz D, and Depairon M. Physical activity monitoring based on accelerometry: validation and comparison with video observation. *Med Biol Eng Comput.*, 37:304–8, 1999.

[14] M.L. Wetzler, J. R. Borderies, O. Bigaignon, P. Guillo, and P. Gosse. Validation of a two-access accelerometer for monitoring patient activity during blood pressure or ecg holter monitoring. *Clinical and Pathological Studies*, 2003.

[15] Uiterwaal M, Glerum EB, Busser HJ, and van Lummel RC. Ambulatory monitoring of physical activity in working situations, a validation study. *J Med Eng Technol.*, 22(4):168–72, 1998.

[16] J. Mantyjarvi, J. Himberg, and T. Seppanen. Recognizing human motion with multiple acceleration sensors. In *2001 IEEE Int'l Conf. on Systems, Man and Cybernetics*, volume 3494, pages 747–752, 2001.

[17] C. Randell and H. Muller. Context awareness by analysing accelerometer data. In *IEEE Int'l Symp. on Wearable Comp.*, pages 175–176, 2000.

[18] K. Van-Laerhoven and O. Cakmakci. What shall we teach our pants? In *IEEE Int'l Symp. on Wearable Comp.*, pages 77–83, 2000.

[19] Bernt Schiele Stavros Antifakos, Florian Michahelles. Proactive instructions for furniture assembly. In *UbiComp, 4th Int'l Conf.*, page 351, Gteborg, Sweden, 2002.

[20] Gaolin Fang, Wen Gao, and Debin Zhao. Large vocabulary sign language recognition based on hierarchical decision trees. In *Intl. Conf. on Multimodal Interfaces*, Vancouver, BC, November 2003.

[21] Holger Junker. *Human Activity Recognition and Gesture Spotting with Body-Worn Sensors.* Hartung-Gorre, PhD thesis, ETH Zurich, 2005.

[22] V. Peltonen, J. Tuomi, A. Klapuri, J. Huopaniemi, and T. Sorsa. Computational auditory scene recognition. In *IEEE Int'l Conf. on Acoustics, Speech, and Signal Processing*, volume 2, pages 1941–1944, May 2002.

[23] Michael C. Büchler. *Algorithms for Sound Classification in Hearing Instruments.* PhD thesis, ETH Zurich, 2002.

[24] B. Clarkson, N. Sawhney, and A. Pentland. Auditory context awareness in wearable computing. In *Workshop on Perceptual User Interfaces*, November 1998.

[25] Huadong Wu and Mel Siegel. Correlation of accelerometer and microphone data in the coin tap test. In *Instrumentation and Measurement, IEEE Trans.*, volume 49, pages 493–497, June 2000.

[26] Josef Kittler, Mohamad Hatef, Robert P.W. Duin, and Jiri Matas. On combining classifiers. In *IEEE TPAMI*, volume 20, pages 226–239, March 1998.

[27] Lei Xu, Adam Kryzak, and Ching Y.Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. In *Systems, Man, and Cybernetics, IEEE Trans.*, volume 22, pages 418–435, may/june 1992.

[28] Tin Kam Ho. Multiple classifier combination: Lessons and next steps. In *Hybrid Methods in Pattern Recognition, World Scientific*, 2002.

[29] Tin Kam Ho, J.J. Hull, and S.N Srihari. Decision combination in multiple classifier systems. In *IEEE TPAMI*, volume 16, pages 66–75, Jan 1994.

[30] 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.

[31] H.Junker, P.Lukowicz, and G.Tröster. Continuous recognition of arm activities with body-worn inertial sensors. In *IEEE Int'l Symp. on Wearable Comp.*, pages 188–189, 2004.

[32] K.Kunze, P.Lukowicz, H.Junker, and G.Troester. Where am i: Recognizing on-body positions of wearable sensors. In *LoCA*, May 2005.

[33] J. Lester, T. Choudhury, N. Kern, G. Borriello, and B. Hannaford. A hybrid discriminative/generative approach for modeling human activities. In *IJCAI*, 2005.

[34] Anjum 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.

[35] 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.

[36] Ling Bao and S. Intille. Activity recognition from user-annotated acceleration data. In *Pervasive, LNCS 3001*, 2004.

[37] 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.

[38] H. Müller, W. Müller, D. McG. Squire, S. Marchand-Maillet, and Thierry Pun. Performance evaluation in content-based image retrieval: Overview and proposals. Technical report, Uni. Geneve, Switzerland, 1999.

[39] Y.J. ZHANG. A review of recent evaluation methods for image segmentation. In *Intl. Symp. on Signal Processing and its Applications*, Aug 2001.

[40] J. R. Pierce. Wither speech recognition. *J.Acoust. Soc. Amer.*, 46:1049–1051, 1969.

[41] Roger K. Moore. Evaluating speech recognizers. *IEEE Trans. Acoust. Speech and Sig.Proc.*, 25(2):178–183, 1977.

[42] S. Greenberg. Whither speech technology? - a twenty-first century perspective. In *7th Intl. Conf. on Speech Com. and Tech. (Eurospeech)*, pages 3–6, 2001.

[43] 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.

[44] N. Kern, B. Schiele, H. Junker, P. Lukowicz, and G. Tröster. Wearable sensing to annotate meeting recordings. In *IEEE Int'l Symp. on Wearable Comp.*, pages 186–193, October 2002.

[45] M. Stäger, P. Lukowicz, N. Perera, T.v.Büren, G.Tröster, and T.Starner. Soundbutton: Design of a low power wearable audio classification system. IEEE Int'l Symp. on Wearable Comp., 2003.

[46] C. V. C. Bouten. A triaxial accelerometer and portable data processing unit for the assessment of daily physical activity. 44:136–147, March 1997.

[47] L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–16, Jan 1986.

[48] T. Starner, J. Makhoul, R. Schwartz, and G. Chou. On-line cursive handwriting recognition using speech recognition methods. In *ICASSP*, pages 125–128, 1994.

[49] Kevin P. Murphy. The hmm toolbox for MATLAB, http://www.ai.mit.edu/ murphyk/software/hmm/hmm.html.

[50] R. Duda, P. Hart, and D. Stork. *Pattern Classification, 2nd Edition*. Wiley, 2001.

[51] J.A.Ward, P.Lukowicz, and G.Tröster. Roc analysis of partitioning method for activity recognition using two microphones. In *Adjunct Proc. of the 3rd Int. Conf. on Pervasive Comp.*, volume 191, May. 8-13 2005.

[52] Tom Fawcett. *ROC Graphs: Notes and Practical Considerations for Researchers*. Kluwer, 2004.

[53] J.A.Ward, P.Lukowicz, G.Tröster, and T. Starner. Human activity recognition using body worn microphones and accelerometers for assembly tasks. In *To appear in IEEE Trans. Pattern Analysis and Machine Intelligence*, volume ?, page ?, ? 20'?

[54] E.M.Tapia, S.S.Intille, and K.Larson. Activity recognition in the home using simple and ubiquitous sensors. In *Pervasive, LNCS 3001*, pages 158–175, 2004.

[55] Foster Provost, Tom Fawcett, and Ron Kohavi. The case against accuracy estimation for comparing induction algorithms. In *IMLC, 15th Int'l Conf.*, 1998.

[56] I.T. PHILLIPS and A.K. CHHABRA. Empirical performance evaluation of graphics recognition systems. In *IEEE TPAMI*, volume 21:9, pages 849–870, 1999.

[57] C. J. Van Rijsbergen. *Information Retrieval, 2nd edition.* Dept. of Computer Science, University of Glasgow, 1979.

[58] M. Stäger, P.Lukowicz, and G. Tröster. Implementation and evaluation of a low-power sound-based user activity recognition system. 8th Int'l Symp. on Wearable Comp., 2004.

[59] G.Ogris, T.Stiefmeier, H.Junker, P.Lukowicz, and G.Trster. Using ultrasonic hand tracking to augment motion analysis based recognition of manipulative gestures. In *IEEE Int. Symp. on Wearable Computers*, 2005.

[60] J.A.Ward, P. Lukowicz, and G.Tröster. Gesture spotting using wrist worn microphone and 3-axis accelerometer. In *Soc-Eusai '05, Proceedings*, Oct. 12-14 2005.

[61] J.A.Ward, P.Lukowicz, and G.Tröster. Evaluating performance in continuous context recognition using event driven error characterisation. In *Submitted to LoCA'06*.

[62] A.F. Clark and C.Clark. Performance characterization in computer vision a tutorial. In *http://www.peipa.essex.ac.uk/benchmark/*, Essex, UK, 1999.

[63] T. Kanungo, G. A. Marton, and O. Bulbul. Paired model evaluation of ocr algorithms. Technical report, Center for Automation Research, Uni.Maryland, 1998.

[64] N. Kern, H. Junker, P. Lukowicz, B. Schiele, and G. Tröster. Wearable sensing to annotate meeting recordings. *Personal and Ubiquitous Computing*, 2003.

[65] M.Lampe, M.Strassner, and E.Fleisch. A ubiquitous computing environment for aircraft maintenance. In *ACM symp. on Applied comp.*, pages 1586–1592, 2004.

[66] 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.

[67] 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.

[68] NIST. *Proc. of TREC Video Retrieval Evaluation Conference (TRECVID)*. 2003.

[69] 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.

[70] 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.

[71] 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.

[72] O. Amft, H. Junker, and G.Tröster. Detection of eating and drinking arm gestures using inertial body-worn sensors. In *9th Int'l Symp. on Wearable Comp.*, Oct. 2005.

[73] H. Brashear, T. Starner, P. Lukowicz, and H. Junker. Using multiple sensors for mobile sign language recognition. In *IEEE Int'l Symp. on Wearable Comp.*, pages 45–53, White Plains, NY, 2003.

[74] D. Minnen, I. Essa, and T. Starner. Expectation grammars: Leveraging high-level expectations for activity recognition. In *IEEE Proc. Comp. Vision and Pattern Rec.*, June 2003.

[75] J.A. Ward, P.Lukowicz, and G.Tröster. A categorisation of performance errors in continuous context recognition. In *IEEE Int'l Symp. on Wearable Comp, Workshop for On-Body Sensing*, 2005.