Adaptive Event Recognition with the use of Limited Training Data

Georgios Paliouras[+ *] and David S. Brée
Computer Science Department
University of Manchester
Oxford Road
Manchester, M13 9PL
UK

Abstract: This paper presents a novel event recognition system, which is capable of adapting itself to improve its performance on a small set of training data. The event recognition system is represented by a network of events, related to each other by temporal constraints. This symbolic representation is particularly suitable to the treatment of overlapping events, which have been overlooked in most of the work on event recognition. Additionally, a method for refining the temporal parameters of the recognition system is presented here. The method uses a small set of preclassified training examples to improve the performance of the system. The principle of minimal model change is used to overcome the sparseness of the training data. Particular emphasis is given to the issue of multiple positive examples, which is prevalent when allowing overlapping events. The new system has been applied to the thematic analysis of humpback whale songs with encouraging results.

Keywords: learning, adaptive systems, event recognition, network models

1. Introduction

The fascinating properties of the sounds emitted by humpback whales have attracted the attention of marine biologists during the past few decades. The usual method of analysis of the song is by examination of the spectrogram of a recording. A spectrogram is a plot of the acoustic signal in the frequency domain. It depicts the frequency components of the uttered sounds, as acquired by a Fourier transformation, against time. Figure 1 shows a piece of such a spectrogram. The basic sounds identifiable in the spectrogram and separated by silence are called *units*. In order to aid the analysis of the song, experts in the field have developed a hierarchical model which groups units into *subphrases*, *phrases* and *themes* [Payn71,Payn83]. Phrases and themes are identified in Figure 1.

The hierarchical model of the humpback whale song allows the identification and comparison of song components in different recordings. This is an instance of the class of *event recognition* tasks, which are the subject of this paper. The song components, i.e., units, subphrases, phrases and themes, are the events and recognition is achieved by applying the song model to the recording. Similar tasks occur in a variety of domains, ranging from speech recognition [Rabi89,Waib90,Lipp90] and signal understanding [Nii88,Less95] to process monitoring [Kock94] and fault diagnosis [Hewi89,Dous96].

In this paper, event recognition is viewed as the process of analysing a stream of symbolic, time-indexed data, in order to recognise interesting events that have occurred. In the case of the humpback whale song the input data consist of time-stamped occurrences of the basic song units. The aim of the system is to recognise the occurrence of other events in the sequence, i.e., subphrases, phrases and themes. Event recognition is

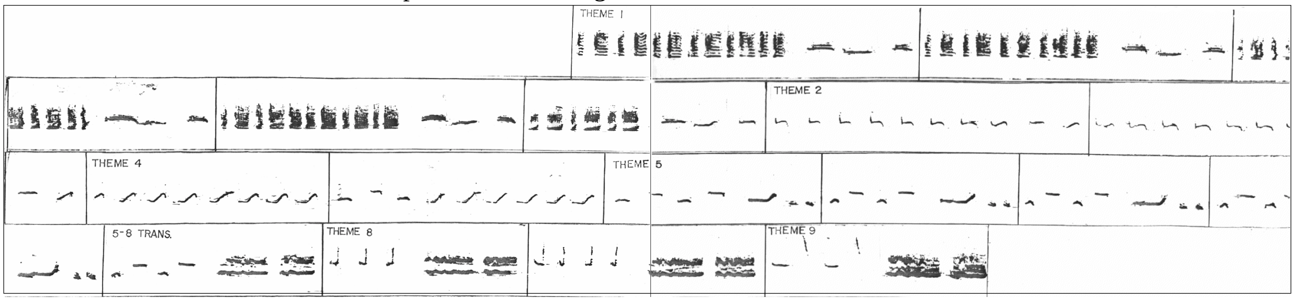

Figure 1. An example of a spectrogram from a humpback whale song, recorded in Hawaii in 1978. Reproduced with the permission of the Whale Conservation Institute, Lincoln, MA.

based on an *event model,* which can be thought of as a

[+] Current address: Institute of Informatics and Telecommunications, NCSR "Demokritos", 15310 Aghia Paraskevi, Attikis, Greece.

[*] This work was funded by Thomson Marconi Sonar UK Ltd. and the Engineering and Physical Sciences Research Council, UK.

temporal expert system. This model consists of a set of dependencies between events and temporal parameters, constraining the recognition of events. Section 2 describes in some detail the representation of the recognition system.

The manual assignment of the temporal parameters in the event recognition system is a tedious and error prone process. For this reason, we present here a method that partially automates the assignment of the temporal parameters. The initial values of the parameters are assumed to be incorrect, but near misses of the true values. Under this assumption, the parameters are refined using a small set of training examples. The relational nature of the event recognition task introduces a number of challenging issues, in addition to the small size of the training set. Such issues are the extent of supervision provided by the preclassified data and the construction of training examples from the continuous input stream. Section 3 examines these issues and provides solutions.

The method of refining the temporal parameters of the system was evaluated on the problem of analysing humpback whale songs to identify song components. The results obtained in these experiments, presented in section 4, are very encouraging, showing that the new method is useful for refining event recognition systems, especially under the constraint of a small training set. Section 5 briefly examines related work in event recognition and section 6 discusses the advantages and shortcomings of the methods presented in the paper.

2. Event Network

In order to facilitate parameter refinement, especially under the constraint of sparse data, the event recognition system needs to be represented in a simple and well-structured way. For this purpose, a network representation is proposed here, which organises events into a hierarchy. This representation is called '*event network'* and its main features are described below. A detailed description can be found in [Pali97].[1]

2.1 Hierarchical decomposition of events

The basic entities of the event network are events, which are divided into two types: *low-level* and *high-level*. Low-level events are the basic building blocks in the system, corresponding to the units in the humpback whale song. Their recognition is performed by a separate system, not considered in our work. For instance, assume the signal in Figure 2, which could be a simplified and cleaned up version of a small piece of a spectrogram.[2] The low-level event recognition system identifies four events in this spectrogram: A, B, C, D. The output of the low-level system is the sequence of recognised events, with their time stamps:

A(0,200), B(200,400), C(300,500), D(500,600),

where the numbers in brackets correspond to the start and the end of each *event occurrence*. The start time for the recording sets the zero point for time stamps. Time may be measured in any unit, e.g. msec. So A starts at time 0, the start of the recording, and ends after 200 msec. The recognition of an event is notified by the low-level recognition system, as soon as the event ends, so that the event can be associated with its time stamp. The above sequence of low-level event occurrences is the input to the high-level recognition system, which is the subject of this article.

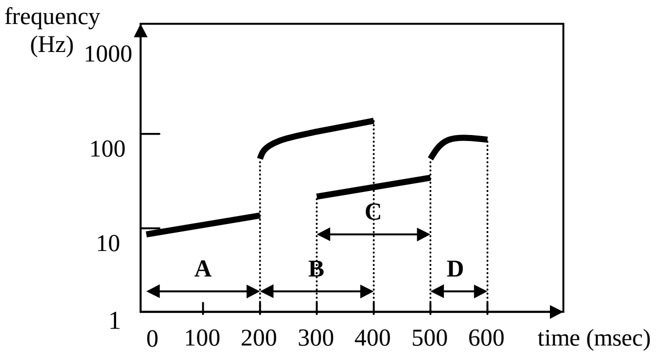

Figure 2. A hypothetical piece of a spectrogram.

Low-level events are combined into recognition rules for high-level events, e.g.

IF $A(i_A^-, i_A^+)$ **AND** $B(i_B^-, i_B^+)$ **AND** $i_B^- - i_A^+ = 0$
THEN $Z(i_Z^-, i_Z^+)$.

The notation (i_e^-, i_e^+) denotes the time stamp for the occurrence of event *e*, i_e^- being the start and i_e^+ the end time. The above rule is called the *definition* of event Z and the low-level events A, B are called its *subevents*. The time stamp of the event being defined is assumed to exactly cover the subevent occurrences. Thus, the high-level event Z(0,400) will be recognised, combining subevents A(0,200) and B(200,400).

High-level events can also be defined as combinations of other high-level events. For instance, Z could be the subevent of another event W. In this manner, a hierarchy of event definitions is constructed, forming an event network. The event network can be represented graphically as a Directed Acyclic Graph, e.g. the graph in Figure 3. Note that an event may be used as subevent in more than one high-level event definitions. This is the case with events A, B in Figure 3. In this case, the definitions of events Y and Z are said to *overlap*. Overlapping event definitions cause a serious problem in event recognition. The occurrence of A and B is not sufficient to say whether Y or Z is happening. Events C and D are necessary for distinguishing between Y and Z. Under certain conditions, e.g. noisy signal, both C and D might be recognised making it impossible to decide whether Y or Z should be recognised. This could be the case if the following sequence of events is recognised:

A(0,200), B(200,400), C(300,500), D(500,600)

[1] The term *temporal classification network*, instead of *event network*, was used in [Pali97].
[2] The term "cleaned up" refers to the removal of harmonics.

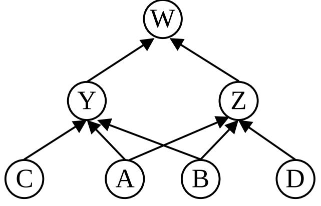

Figure 3. A simple event network.

and both the definitions of Y and Z are satisfied. In such situations, the system notifies the occurrence of both events, i.e., Y(0,500) and Z(0,600), using the same occurrences of subevents A and B.

The implicit assumption so far is that the recognition of a high-level event requires the occurrence of *all* of its subevents. In other words, event definitions correspond to rules with a conjunctive condition. Such rules are not sufficient for the representation of a realistic event recognition system. For this reason, two more types of event are used in the event network: *disjunctive* and *repeating*. Disjunctive events require simply that one of the subevents in the definition occurs. Repeating event definitions contain a single subevent, which is required to occur repeatedly. The number of repetitions of the subevent is constrained within a range, e.g. 20 to 30 times.

2.2 Temporal relations and parameters

As mentioned above, the event network represents a temporal expert system. Thus, event definitions are augmented with temporal parameters constraining the recognition of events. The example definition for event Z above contains such a constraint, i.e., $i_B^- - i_A^+ = 0$, which requires that subevent B starts when A ends. In general, the subevents in an event definition are not required to meet in time; there may be a gap between them or they may overlap. This is an important advantage of the explicit representation of temporal constraints in a symbolic system.

Three types of temporal constraint are used in an event network:

- A *precedence* qualitative constraint, *precedes*(x,y), which requires that subevent *x ends before y ends*. Note that x and y may still overlap.
- A quantitative constraint on the *duration* of subevents, *duration*(x,[d_1,d_2]), where [d_1,d_2] is an integer range of valid values for the duration of x.
- A quantitative constraint on the temporal *distance* between subevents, *distance*(x,y,[s_1,s_2]), [s_1,s_2] being the range of valid distances between x and y.

The use of ranges, instead of exact values, is needed to deal with noise and variation in real-world data. Disjunctive event definitions do not contain relations between subevents, since a single subevent is used for the recognition of the high-level event. Only the duration of the subevent is constrained. In repeating event definitions, there is just a single subevent which precedes itself. Thus, only one duration and one distance constraint are defined, which apply to all repetitions of the subevent.

The use of the *precedes* relation in conjunctive events is further constrained for efficient event recognition:

- No cycles in the precedence of subevents, e.g. *precedes*(A,B) and *precedes*(B,A), are allowed.
- Each subevent can only be used in two *precedes* relations: in one of them it should *precede* another, say *precedes*(B,C) and in the other it must be *preceded*, e.g. *precedes*(A,B).

In this manner a *precedence sequence* of subevents is constructed in each conjunctive definition. One of the subevents in this sequence does not precede any other and is called *terminal* subevent, since its end coincides with the end of the high-level event. Thus if Z is defined as the sequence of subevents A, B, C, where *precedes*(A,B) and *precedes*(B,C), C is the terminal subevent in the definition.

2.3 Recognition algorithm

In the design of the representation that was presented in this section, particular attention was paid to the performance of the recognition system. In particular two properties of the representation are used to achieve efficient recognition:

- The partial ordering of events in the event network.
- The precedence sequence in conjunctive events.

In brief, the recognition algorithm stores the occurrences of events in a dynamic database and for each incoming event occurrence it checks whether it could act as the terminal subevent in a definition. If not, no action is needed, otherwise the appropriate event definition is used to recognise potential occurrences of the corresponding event. These occurrences are appended to the database and recognition is propagated to the higher levels of the network. This algorithm is described in more detail in [Pali97].

3. Adaptation of the event network

The event network for a realistic event recognition problem may involve a large number of event definitions with an equally large number of temporal constraints. Incorrect assignment of values to these constraints may cause unacceptably high levels of misrecognition. However, manual setting of these constraint values can be very difficult and time-consuming. Therefore, any automation of this process would be welcome by the experts who design the recognition system. This section presents a method which starts with a set of initial constraint values and improves them to achieve correct recognition on a small set of preclassified training data. The initial constraint values may be only rough estimates of the correct values. These values will then be refined according to the performance of the system on the training data. Two versions of the *parameter refinement* method are presented in the following two subsections of this section. The two versions differ in the level of supervision which is provided by the training data.

3.1 Refinement under full supervision

The first version of the refinement method makes the assumption that the training data provide information about *all the high-level events which should be recognised*, given a sequence of low-level event occurrences. For instance, assuming the network in Figure 3 - where events A, B, C, D may correspond to the units of a whale song, Y and Z to phrases and W to a complete theme - the following sequence of unit occurrences may be part of the training data:

A(0,200), B(200,300), A(250,350), B(400,500),
C(410,500), D(500,600)

and supervision is provided in the form of phrases and themes that should be recognised, e.g.

Y(0,500), Z(250,600), W(0,600).

All relevant themes and phrases which should be recognised in the above sequence of unit occurrences are assumed to be provided in the training data. This mode of supervision is called here '*full supervision*.'

Note that the representation of the training data deviates significantly from the feature vector of typical classification problems, which is assumed in most of the machine learning and knowledge refinement literature, e.g. [Quin93] and [Craw90]. Here the data are presented in two separate streams of event occurrences:

- The *input stream,* consisting of unit occurrences in our examples.
- The *feedback stream,* consisting of themes and phrases that should be recognised.

No information is provided about which occurrences of low-level events should be combined to recognise a particular high-level event. To a certain extent this information can be deduced by the event network and the time stamps in the feedback data. For instance, it is clear that the unit occurrence C(410,500) can only be used in the recognition of Y(0,500), since event C participates only in the definition of Y. Similarly, A(0,200) is used in the recognition of Y(0,500), because there is no other event starting at time 0. However, the use of overlapping event definitions in an event network complicates matters, e.g. either of the two occurrences of B, B(200,300) and B(400,500), may be used to recognise Y(0,500). Thus, there may be more than one sequence of low-level events which could cause the recognition of one high-level event, e.g. for Y(0,500):

A(0,200), B(200,300), C(410,500),
A(0,200), B(400,500), C(410,500).

These alternative sequences are called '*multiple positive instances*.'[3] The refinement method is responsible for covering one or more of the alternative positives, depending on the extent to which the temporal parameters need to change. The principle of *minimal model change* dictates that the new values of the temporal parameters remain close to the original values.

In order to make the problem solvable by a machine learning method, the data pass through two pre-processing stages, which provide the required representation shift to the space of the temporal parameters. The first stage of pre-processing enumerates the sequences of subevent occurrences that could lead to the recognition of an event.[4] In the example of the event Y(0,500) above, the two alternative sequences of A, B, C will be generated. This process is initiated when the terminal subevent - C in this case - is recognised. The alternative subevent sequences are organised into a tree structure, called '*event support tree*' (EST). The EST for the above positive example for event Y(0,500) is presented in Figure 4.

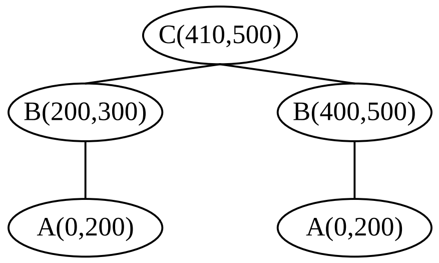

Figure 4. A simple event support tree for the event occurrence Y(0,500).

The important features of an EST are the following:

- Its root corresponds to the terminal subevent sequence.
- Each level of the tree corresponds to the occurrences of a single subevent.
- Each path from the root to a leaf is a sequence of subevents, which could lead to the recognition of the high-level event.

Each EST that is generated is considered a single example to be used for the refinement of the temporal parameters. The EST in Figure 4 is a positive example for event Y. In addition to positive ESTs, negative ones are constructed for those subevent sequences which lead to the recognition of events not in the feedback data. For instance, the sequence of unit occurrences

A(250,350), B(400,500), C(410,500)

could cause the recognition of Y(250,500), which is not in the feedback data and thus its recognition is undesirable.

In the second stage of pre-processing, a set of ESTs is collected to form a training set for the refinement of the temporal parameters in an event definition. Thus, all the ESTs for event Y - both positive and negative - are collected into a training set. This training set is also organised in a tree structure, which is called the '*relative event support tree*' (REST). The REST maintains the structural features of an EST, but instead of absolute time, the relative parameter values are used to label the nodes. Furthermore, lists of positive and negative examples are associated with each node to assist in the refinement decision. Figure 5 shows the REST for the running example of event Y(0,500).

[3] This term was coined in [Diet97].

[4] A user-defined window parameter is used to avoid combinatorial explosion.

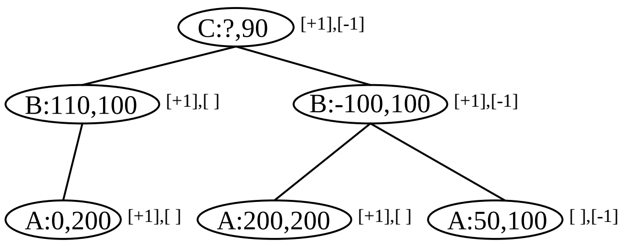

Figure 5. The relative event support tree for event Y.

Each node in the REST contains two parameter values: the first is the distance of the subevent occurrence from the occurrence that it precedes and the second is the duration of the subevent occurrence. For instance, the root of the tree does not precede any other subevent and its distance value is undefined (=?). Its duration is 90 msec, which is the difference between the end and the start of the event occurrence C(410,500). Similarly, the distance between B(400,500) and C(400,500) is 400-500=-100 msec and the duration of B(400,500) is 100 msec, resulting in the node B:-100,100 of the REST in figure 5. The lists next to the nodes of the tree contain labels for positive and negative examples. These labels are given by global example counters. In the example of figure 5 there is one positive (+1) and one negative example (-1). The positive example corresponds to two alternative paths through the tree, as explained above. The cumulative construction of the REST, allows the use of a batch learning method which is independent of the order in which examples are presented in the data.

Given the data in the form of RESTs, a number of alternative search algorithms can be applied to look for the optimal set of new parameter values. The algorithm that we developed performs a simple best-first search, traversing the tree from the root to the leaves. The optimality criterion combines two subcriteria:

- *purity*, which gives an indication of how many positive and negative examples are covered by each solution;
- *proximity*, measuring the closeness of a solution to the original values.

The user is allowed to determine the relative weight of these parameters, depending on the amount of noise in the data and his/her confidence in the original parameter values. The details of the algorithm can be found in [Pali97].

3.2 Refinement under partial supervision

In subsection 3.1 we have assumed that the feedback stream contains information about all of the events in an event network. This assumption cannot easily be satisfied when a realistically large event network is used. It is, therefore, desirable to reduce the amount of supervision required for refinement. This section presents briefly an extension to the original method, which makes the assumption that supervision is provided for only part of the training data. In particular only the top-level nodes in the network - the themes in the case of the whale song - are identified in the feedback data. In our running example, this would shrink the feedback data to a single event occurrence: W(0,600).

The problem that appears under this weaker supervision scenario is the labelling - as positive or negative - for the occurrences of all intermediate events in the network. In our example, we need to decide, which of the following occurrences of Y and Z should really be recognised:

Y(0,500), Y(250,500), Z(0,600), Z(250,600).

Almost any combination of the above two pairs of Y and Z occurrences could be used to recognise W(0,600), the only exception being the combination of Y(250,500) and Z(250,600), which does not give the right start time for W.

The solution that we adopted was to maintain a set of weights, one for each event occurrence in the network. These weights are called *recognition beliefs* and provide an estimate of whether an occurrence should be recognised or not. Recognition beliefs are initially set according to how easy it would be to recognise an event occurrence, i.e., how much the original model parameters should be modified, if at all, in order for the occurrence to be recognised. This initial model-based estimate is updated according to the feedback data. The recognition belief for the top-level event occurrences that are explicitly identified in the data is set to the maximum (1.0), while those events that should not be recognised are given minimum belief (0.0). This information is propagated backwards to their subevents to obtain the final belief for each event occurrence in the network.

After this two-stage calculation of the recognition beliefs, each training example is associated with a weight. At this stage, a heuristic could be applied to decide on a single set of event occurrences that would lead to the recognition of the required high-level events. For instance, a winner-takes-all strategy could be adopted. The problem with such heuristics is that they are local in nature and can thus cause brittleness, throwing away useful information, i.e., the actual weights. For this reason, we have opted for a simple modification to the optimality criteria used under full supervision, which allows the use of weights, resulting in finer distinctions between alternative solutions. The details of the extended refinement method can be found in [Pali97].

4. Experimental evaluation

The event recognition system that was briefly introduced in this paper was tested on a real-world problem: the recognition of song components in humpback whale songs. The structure of the song has been studied in detail in the biological literature, e.g. [Payn71,Payn83], and a number of interesting observations were made. The following are of particular interest here:

- In each recording season (roughly December to April) and for a particular population of humpback whales, the same song is sung by all individuals, with only small aberrations.

- The song is structured hierarchically into themes, phrases, subphrases and units, i.e., themes are broken down into phrases, which consist of subphrases and can be further decomposed into units.
- The temporal features of the song components, e.g. duration and distance, are highly variable, even for the same individual.

Each whale song consists of a sequence of themes, which differ structurally from each other. The task that was set for the event recognition system was to identify the themes and their structural components, i.e., phrases, subphrases and units. This task is named here '*theme analysis*' of the whale song. The data that we used for the test were provided by the Whale Conservation Institute in the form of spectrogram print-outs. These spectrograms corresponded to several hours of recording in one season in Hawaii. The same data were used in the detailed study presented in [Payn83].

The event network representation suited nicely the hierarchical structure of the whale song. Each theme was represented by an event network, using information from the study in [Payn83]. We also manually extracted sequences of unit occurrences from 10 of the spectrograms. Manual extraction was necessary due to the lack of a low-level event recognition system. The goal of our system was to learn the correct parameter settings, so that the subphrases, phrases and themes that are identified by the event networks are the correct ones. For this purpose, we performed a 10-fold cross-validation experiment, using the 10 songs that were coded from the spectrograms. In each of the ten runs, one of the songs was held out as test data and the rest were used for training.

The literature in marine biology does not provide detailed information on the temporal aspects of song components. For this reason, a parameter initialisation algorithm was developed, which uses a set of data to approximate the values of the temporal parameters in the event networks. The use of problem-specific information allows this algorithm to achieve close approximations of the real parameter values. Applying this algorithm on a subset of the 9 songs in the training data, the parameters were assigned initial values, which were further refined with the use of the remaining training data. For instance, 4 songs could be used to initialise the parameters and 5 to refine them. The performance of the algorithm for different sizes of the initialisation set is reported in the results below.

The initial set of parameters, generated by the above-mentioned algorithm is called an '*expert model.*' Similarly, the '*refined model*' is the set of refined parameters. In addition to those two, two more models were set up to act as standards for comparison: the '*perfect model*' is the set of correct parameter vales, which recognises correctly all events in the 10 songs and the '*goal model*' is the model that correctly recognises the events in the 9 songs of the training set. It should be noted that the goal model is an approximation of the parameters that we expect the refinement method to generate, but does not provide a uniquely correct solution.

Figure 6 presents the results of the experiment, using an evaluation metric called '*model displacement.*' This metric is simply the sum of absolute differences between the parameter values of a model and the perfect model. Results both under full and partial supervision are presented. The number of songs that are used in the initialisation of the parameters is varied on the x-axis of the diagram. When the initialisation set is small, e.g. 1 song, the expert model is far from the desired perfect model. As expected, for larger initialisation sets the expert model approaches the perfect model. When 8 of the 9 songs are used for initialisation the difference between the expert and the goal model is minimal. The goal model, which is generated on the 9 songs of the training set remains largely unaffected by the size of the initialisation set.

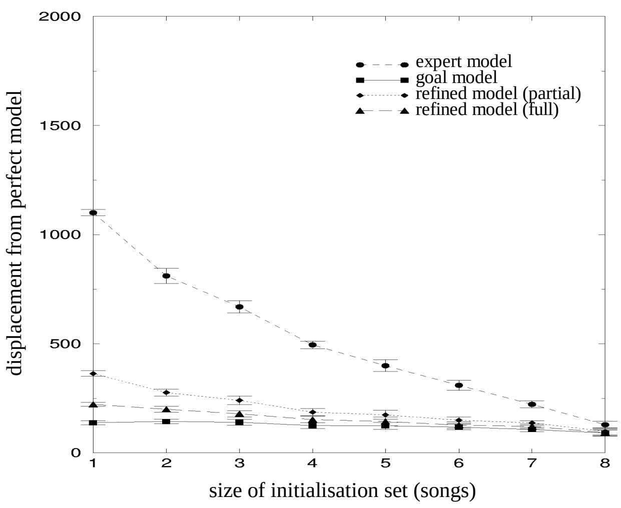

Figure 6. Displacement from the perfect model for different sizes of the initialisation set.

The goal and the expert model form an envelope, within which the refined model is expected to lie. The performance of the refinement method is judged by the distance of the refined model from the expert and the goal model. The closer the refined model is to the goal the better the method. On this basis, the results presented in Figure 6 are very encouraging. Both under full and partial supervision, the refined model is very close to the goal. The expected fall in performance under partial supervision is also small, when compared to the improvement that is achieved over the expert model. Under partial supervision, the amount of feedback provided for training is minimal: only the theme occurrences are given. Thus, the results of figure 6 suggest that our refinement method has handled the test problem very well.

These good results are reinforced by *the recognition* *accuracy* of the system on the unseen data, i.e., the tenth song that was not used for training. Recognition accuracy corresponds to the standard evaluation measure in machine learning literature, i.e., the proportion of objects being correctly classified. Figure 7 presents the results along this dimension. The picture is similar to that in Figure 6: the refined models achieve the desired performance. Under full supervision, the refined model outperforms even the goal model, which suffers small approximation errors.

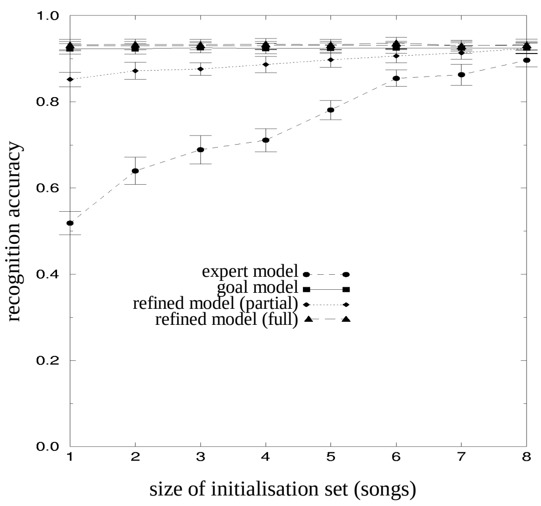

Figure 7. Recognition accuracy of the various models on the held-out song data.

5. Related work

Most of the work on event recognition, e.g. [Rabi89, Waib90, Lipp90] adopts numeric approaches such as Artificial Neural Networks and Hidden Markov Models. These methods can provide accurate and effective models, when there is enough data to estimate the necessary parameters. This was not the case in our work. Furthermore, these approaches cannot deal with overlapping events and do not provide an easy way to use domain knowledge, provided by a human expert.

Early symbolic signal understanding systems which also perform event recognition [Erma80, Lowe90] have meanwhile been abandoned, due their inefficiency. The main problem with these approaches was the use of a blackboard architecture, which is a flexible and powerful, but also inefficient, problem solving method. This architecture has been used in a more recent system which is presented in [Samm90]. In this system, the authors tried to improve recognition efficiency by structuring the use of the blackboard inference strategy into four distinct levels of event definitions. All the above-mentioned systems do not deal with overlapping events and are not adaptive, i.e., no learning takes place.

The event recognition system, as described in section 2, belongs in a small class of systems, grouped under the label '*temporal event recognition systems.'* Examples of this work can be found in [Kock94] and [Dous96]. These systems use a temporal logic to represent the event model and *Temporal Constraint Satisfaction* algorithms for event recognition. The first important difference of these approaches to the system presented here is that they are not adaptive. Additionally, there are practical differences in the recognition system, such as the kind of event that is used and the recognition algorithm. These differences are discussed in detail in [Pali97].

6. Conclusions

In this paper we presented an event recognition system, which is able to adopt itself in order to improve its performance on a small set of training data. In addition to the restriction of the small training set, the system was required to handle overlapping events, which are commonly ignored and are difficult to handle. The solution that we proposed was based on a simple, but well-structured hierarchical representation of events. This representation allows the use of explicit temporal constraints, making the handling of overlapping events easier.

Due to the small size of the training set, we restricted the scope of the learning method on the refinement of the temporal parameters. The parameters are initially assigned approximate values, which can be refined in the light of the training information. Particular attention was paid to the extent of supervision that the expert designer needs to provide, as this can be decisive for the usability of the system. We extended our method to handle a situation where a minimal amount of supervision is provided.

The evaluation of the method was performed on real data that was drawn from the "realm" of marine mammals. 10 songs of the humpback whale were coded and used to test our method. The results of the experiment are very encouraging. The refinement method consistently managed to improve the initially inaccurate model and reach close to the expected maximum on two different measures.

While these results are encouraging, they are just indicative, as only a single problem was used for evaluation. Further evaluation on more data and different problems is needed to verify the effectiveness of our system. This effort should be combined with the use of an automatic system for the recognition of low-level events, i.e., units in the case of the whale song.

Another direction for further work is to expand the scope of refinement to include the restructuring of the event network. Although the confidence of the human expert on the structure of the recognition system is higher than on the exact values of the temporal constraints, errors can also be made at that level. A prerequisite for a method that would perform such restructuring is the use of more training data, in order to make reliable decisions.

Acknowledgements

We are both indebted to the people doing work on the sounds of the humpback whale for their invaluable help, in particular to Roger Payne for providing the data and the much-needed insight on the acoustics of the whale sounds.

References

[Craw90] S. Craw and D. Sleeman. "Automating the Refinement of Knowledge-Based Systems." *Proceedings of the European Conference on Artificial Intelligence*, 167-172, 1990.

[Diet97] T.G. Dietterich, R.H. Lathrop and T. Lozano-Pérez. "Solving the Multiple Instance Problem with Axis-parallel Rectangles." *Artificial Intelligence*, 89:31-71, 1997.

[Dous96] C. Dousson. "Alarm Driven Supervision for Telecommunication Network II - on-line chronicle recognition." *Annales des Télécommunication*, 51(9-10):501-508, 1996.

[Erma80] L.D. Erman, F.Hayes-Roth, V.R. Lesser, and D.R. Reddy. "The Hearsay-II Speech-Understanding System: integrating Knowledge to resolve uncertainty." *ACM Computing Surveys*, 12(2):214-253, 1980.

[Hewi89] P.D. Hewitt, P.J.C. Skitt, and R.C. Witcomb. "A Self-organising Feedforward Network Applied to Acoustic Data." In J.G. Taylor and C.L.T. Mannion (eds.), *New Developments in Neural Computing*, pages 71-78, Adam Hilger, 1989.

[Kock94] S. Kockskämper, B. Neuman, and M. Schick. "Extending Process Monitoring by Event Recognition." In Proceedings of the 2nd International Conference on Intelligent Systems Engineering, pages 455-460, 1994.

[Less95] V.R. Lesser, S.H. Nawab, and F.I. Klassner. "IPUS: an architecture for the integrated processing and understanding of signals." *Artificial Intelligence*, 77:129-171, 1995.

[Lipp90] R.P. Lippmann. "Review of Neural Networks for Speech Recognition." In A. Waibel and K.F. Lee (eds.), *Readings in Speech Recognition*, pages 374391, Morgan-Kaufmann, 1990.

[Lowe90] B. Lowerre and R. Reddy. "The HARPY Speech Understanding System." In A. Waibel and K.F. Lee (eds.), *Readings in Speech Recognition*, pages 576-586, Morgan-Kaufmann, 1990.

[Nii88] H.P. Nii, E.A. Feigenbaum, J.J. Anton, and A.J. Rockmore. "Signal-to-Symbol Transformation: HASP/SIAP case study." In R. Engelmore and T. Morgan, (eds.), *Blackboard Systems*, pages 135157, Addison-Wesley, 1988.

[Pali97] G. Paliouras. *Refinement of Temporal Constraints in an Event Recognition System using Small Datasets.* Ph.D. Thesis, Computer Science Department, University of Manchester, 1997.

[Payn71] R.S. Payne and S. McVay. "Songs of Humpback Whales." *Science*, 173(3997):585-597, 1971.

[Payn83] K. Payne, P. Tyack, and R. Payne. "Progressive Changes in the Songs of Humpback Whales (Megaptera Novaeangliae): a detailed analysis of two seasons in Hawaii." In R. Payne, (ed.) *Communication and Behaviour of Whales,* pages 9-57, Westview Press, 1983.

[Quin93] J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[Samm90] T.M. Sammon Jr. "The Transient Expert Processor." *Conference Record - Asilomar Conference on Signals, Systems and Computers*, 612-617, 1990.

[Rabi89] L.R. Rabiner. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition." *IEEE Proceedings*, 77(2):257-285, 1989.

[Waib90] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang. "Phoneme Recognition using Time-Delay Neural Networks." In A. Waibel and K.F. Lee (eds.), *Readings in Speech Recognition*, pages 393-404, Morgan-Kaufmann, 1990.