IJCLA VOL. 2, NO. 1–2, JAN-DEC 2011, PP. 175–190
RECEIVED 01/11/10 ACCEPTED 26/11/10 FINAL 15/02/11

Automatic Annotation of Referring Expressions in Situated Dialogues

NIELS SCHÜTTE, JOHN KELLEHER AND BRIAN MAC NAMEE

Dublin Institute of Technology, Ireland

ABSTRACT

To apply machine learning techniques to the production and interpretation of natural language, we need large amounts of annotated language data. Manual annotation, however, is an expensive and time consuming process since it involves human annotators looking at the data and explicitly adding information that is implicitly contained in the data, based on their judgment. This work presents an approach to automatically annotating referring expressions in situated dialogues by exploiting the interpretation of language by the participants in the dialogue. We associate instructions concerning objects in the environment with automatically detected events involving these objects and predict the referents of referring expressions in the instructions on the basis of the objects affected by the events. We judge the reliability of these predictions based on the temporal and textual distance between instruction and event. We apply our approach to an annotated corpus and evaluate the results against human annotation. The evaluation shows that the approach can be used to accurately annotate a large proportion of the utterances in the corpus dialogues and highlight those utterances for which human annotation is required, thus reducing the amount of human annotation required.

KEYWORDS: *reference resolution, situated dialogue*

1 INTRODUCTION

We present an approach to automatically annotating *referring expressions* in *situated dialogues*. A referring expression [1, Ch. 18] is an expression that occurs in natural language that is used to denote some kind of object that is discussed. For example in the sentence "Bob ate an apple", "Bob" is a referring expression that denotes some person named Bob, and "an apple" is a referring expression that denotes some apple.

The object that is being referred to is called the *referent* of the referring expression. An *anaphoric* referring expression is a referring expression that refers back to an object that has already been mentioned in the dialogue and is therefore in the linguistic context of the dialogue. An *exophoric* referring expression is a referring expression that refers to an object that has not previously been mentioned in the dialogue but that exists in some other context of the dialogue (e.g. the visual context). The process of *referring expression resolution* is the process of identifying the referents of referring expressions.

A situated dialogue is a conversation between at least two participants that takes places in an environment that is actively discussed as part of the dialogue. A typical example a of situated dialogue is a navigation task where one participant has to give instructions to a second participant to move through the environment the dialogue is situated in. Exophoric referring expressions are particularly common in this domain.

A computer system that participates in situated dialogues has to be able to resolve and produce exophoric referring expressions. There exist a number of approaches to this problem that can broadly be categorized as rule-based and machine-learning (ML) based approaches. Rule-based approaches use a number of (generally hand crafted) rules to perform the task. Grosz and Sidner [2] describe a rule-based approach to resolving reference in purely linguistic domains. Salmon-Alt and Romary present a rule-based approach to resolving reference in a multimodal domain [3].

ML based approaches on the other hand do not rely on prefabricated rules but set out to learn behaviour that is presented in the form of examples. Using ML is particularly attractive for dealing with referring expressions because using ML opens up the possibility to learn and discover strategies used by humans directly from data, which may be difficult to identify by introspection or manual analysis.

Supervised ML is a form of ML where algorithms learn a function that maps from inputs to outputs. Such algorithms require as training data a set of examples in which inputs are associated with the expected output.

Consequently, in order to train a ML algorithm to interpret or produce referring expressions, the algorithm requires a training set of examples that link spoken references to their intended referents in the world and that, furthermore, describe the conditions under which the reference was produced. These conditions may for example include the set of visible objects, the spatial relation of the speaker towards those objects and a records of previous references made by the speaker.

These training sets often have to be created manually by taking a set of inputs and annotating the expected outputs based on human judgment. This process is expensive and time consuming because it requires one or more human annotators to screen all of the examples and make a decision for each case. It is therefore desirable to find methods that can automatically perform at least parts of this process. This problem can be understood as a problem of information retrieval since the reference information must be (implicitly) contained in the data if human annotators are able to reproduce it.

Contribution: In this work we present an approach to automatically generating annotations for exophoric referring expressions in a situated task-based dialogue. We focus on identifying the referent of a referring expression, as this is a task that (unlike the determination of the set of visible objects for example), cannot be performed automatically in a straightforward manner and generally requires the attention of a human annotator. We predict the referent of a referring expression based on the interpretation of that expression in the dialogue. This is only possible if the referring expression can be related to some detectable action. We therefore only consider referring expressions in utterances that instruct the hearer to perform some specific task. In the experiments described in this work we focus on one specific kind of instruction, namely instructions to pass through a door.

Overview: In Section 2 we discuss corpora that are possible fields of application for our approach and introduce the corpus that is used in the example presented in this work. In Section 3 we present our approach to detecting the referents of referring expressions. In Section 4 we present the evaluation of the application of our approach to test data. Finally, in Section 5 we discuss these results and possible extensions of this work.

2 DATA

For this experiment we were interested in corpora featuring situated dialogue. In addition to information immediately related to the dialogue,

such as transcriptions and annotations, we were also interested in additional data related to the environment, such as maps and recordings of the actions of the participants.

There exist a number of freely available situated dialogue corpora. The TRAINS corpus [4], which contains dialogues between two participants planning train routes on a map, is an example of a corpus that incorporates the visual modality, and has transcriptions, but does not feature reference annotations. In addition to that, the corpus works with a static map, which is not dynamically updated, which makes it difficult to annotate referring expressions, because participants frequently talk about hypothetical scenarios. In addition to this, it also lacks a record of the planned routes.

Another visually situated corpus is the MAPTASK corpus [5]. This corpus is based on an experiment where one participant describes a route in a map to a second participant, who has access to a slightly different map. Navigation takes place at an abstract level which makes it hard to identify events at a level that would be relevant to this experiment.

The corpus considered in this work is the SCARE corpus [6]. This corpus consists of dialogues between two participants in a navigation task where the environment is perceived from a first person perspective. It contains transcriptions and reference annotations and is therefore a good example for learning referring expression resolution. Moreover, unlike the TRAINS and MAPTASK corpus, the SCARE corpus features recordings of all navigation steps, thereby enabling us to reconstruct actions performed by the player.

Fig. 1. Screenshot of a video recording from the SCARE corpus.

What differentiates the SCARE corpus and makes it particularly interesting to us, is that it does not take a remote approach with an external perspective, but is very situated, by putting the participants inside the environment. This means that the participants have a location in the environment, which restricts references and actions thereby creating the possibility of linking them. This is the key to our approach. The corpus was created in an experiment focusing on situated task-based dialogues. In this experiment one participant, the direction follower (DF), had to navigate through an environment simulated in a game engine, while the second participant, the direction giver (DG), had to give directions to the first participant to help them fulfil a given task. The details of the task and the layout of the world were known only to the DG. The DF navigated through the environment in a first person perspective, of which a live video feed was shown to the DG. Therefore both participants had the same perspective on the environment. The participants communicated through a voice connection.

The corpus comprised video and audio recordings of the dialogues, as well as transcriptions of the audio files that were annotated for reference, i.e. referring expressions that referred to objects in the environment were annotated with which object the expression referred to. In addition to that, demo files were provided that could be replayed in the game engine, thereby recreating the navigation movements in each dialogue.

3 EXTENDING THE SCARE CORPUS

As noted in Section 2, the SCARE corpus contains annotated dialogue transcriptions and a record of player movement. In order to automatically annotate referring expressions, we needed to create new data from the corpus. In particular, we had to identify a set of referring expressions and then determine the referent for each expression. We did this by establishing a correspondence between instructions that contain a referring expression in the dialogue and events in the world that could be caused by these instructions. The events we wanted to consider were not explicitly contained in the data, so we had to reconstruct them. Consequently, establishing a correspondence between instructions in the dialogue and events in the world involved 3 steps:

1. We detected a set of instructions.
2. We detected a set of events.

3. We established a correspondence between instructions and events and recorded values for different distance metrics between instructions and events.

Each of these steps is described in detail below. We then evaluated the correspondence against gold standard manual annotations. This evaluation is described in Section 4.

3.1 *Detecting the Instructions*

In this experiment we were interested in referring expressions that caused events we could detect by looking at the movement of the player in the environment. One class of such events is passing through doors. We therefore detected instances of the DG telling the DF to go through a door. We did this using a regular expression of this form:

```
[go|pass]through.*[door|one|that]
```
[1]

This expression fit instructions such as "go through the right door" or "pass through the next one". We collected instructions up to a length of seven words. The regular expression was defined by examining a small number of the dialogues in the SCARE corpus. In total we detected 135 referring expressions using this regular expression. This approach probably did not capture all instructions, but served as a good starting point.

3.2 *Detecting Events*

Once we had detected the set of instructions that we would use in our experiment we then had to detect the set of relevant events to match against the instructions. To do this, we replayed the demo files in the game engine and recorded the position and orientation of the player during the dialogues. We then aligned this information with time. By comparing this information with geometric information about the layout of the rooms, we were able to detect the moments when the player left a room and entered another room. This in turn enabled us to determine which door the player had passed at what point in time. Each passing of a door formed an event.

[1] $.*$ matches any sequence of characters, $[x|y]$ matches the sequence x or y.

3.3 *Establishing the Correspondence*

In this step we determined a correspondence between instructions and events for our example corpus. We aimed to identify for each instruction the specific event that occurred when the DF fulfilled the instruction. Events naturally occur slightly after the instruction has been produced because the DF needs time to interpret the instruction and to navigate into a position where it is possible to perform the required action. However, not every instruction is immediately succeeded by an event that fulfils the instruction. We see three main reasons for this:

1. The DF may misinterpret the instruction and begin to perform a different action.
2. The DF may not understand an instruction or find it ambiguous and ask the DG to clarify. In this case, the next event may follow after a longer delay, during which the participants come to an agreement about the next action, and may actually end up not fulfilling the original instruction because the participants decided on a different course of action.
3. A number of other events may occur between an instruction and the corresponding event because the DF has to fulfil a number of subgoals in order to be able to fulfil the instruction.

At first glance, two approaches in creating a correspondence are apparent: we can either start out with the events and search for an instruction to match each event; or we can start out with the instructions, and determine which event was caused by each instruction. The first approach immediately appeared less favourable because in the example dialogues, a great number of events are not directly caused by instructions. This happens when the DF is exploring the map on their own, or if the DG gives high level goals, such as returning to a previously visited room, which the DF can fulfil without being instructed in every step. We therefore decided to use the approach where we start with the instructions and then search for events that match these instructions.

We processed each dialogue incrementally by going through it from the beginning, picking up instructions and events as they occurred. An incoming instruction was processed by storing it on a FIFO queue. An incoming event was processed by removing the oldest instruction from the queue and associating it with the new event, and storing the resulting pair for later evaluation. This was based on the assumption that events were preceded by instructions. Roughly speaking this approach associates each instruction with the next event occurring after it.

We collected instructions in a queue, which enabled us to correctly interpret concatenated instructions (e.g. “go through this door and then go through the next one”) as two instructions to be executed sequentially.

Events that occurred while the instruction queue was empty were discarded as events that occurred without explicit instruction. Such events occurred often in the dialogues when the DF was asked to move to a previously visited location. In this case, the DF often could find the way on their own without having to be explicitly instructed for each step.

The matching process resulted in a set of pairs of instructions and events which was the basis for our experimental evaluation. The algorithm we used for this process is presented as Algorithm 1.

Algorithm 1 The algorithm for associating instructions and events.

Input:	**dialogue:** a temporally ordered set of events and instructions.
Data Structures:	**instructionQueue:** a Queue for incoming instructions, initially empty.
	correspondences: a list of instructions-event pairs.
Output:	a list of instructions-event pairs which may be related.

```
FOR the length of the dialogue
   e := select the next event or instruction
   IF e is an instruction
      push(e, instructionQueue)
   ELSE
      IF e is an event
         IF empty(instructionQueue)
            discard e
         ELSE
            i := pop(instructionQueue)
            a := associate(i,e)
            append(a, correspondences)
         ENDIF
      ENDIF
   ENDIF
ENDFOR
RETURN correspondences
```

Every time an instruction and an event were associated, we recorded the distance in time between instruction and event, and the number of

words spoken between them to facilitate evaluation. We derived these values from the time aligned dialogue transcriptions.

The output of the algorithm consists of a list of associated instructions and events. Each pair represents a possible causal relationship between an instruction and an event, and thereby a candidate for annotation. In the next step of the process each pair will be more closely examined, and it will be estimated how likely the pairing is to be a correct assignment.

Figure 2 illustrates the approach. Intervals of speech are represented as blocks below the time axis. Dark blocks represent instructions, while bright block represent speech that is not an instruction. Stars on the time axis represent events. The horizontal brackets delineate the intervals between the end of an instruction and the next event. The dashed vertical lines cut out intervals on the time axis and pieces of the speech blocks, which form the distance values.

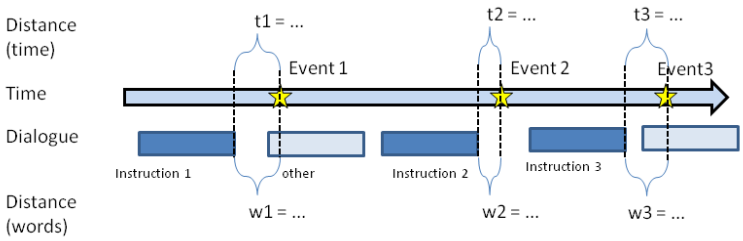


Fig. 2. Illustration of the instruction-event association and distance measuring process. Blocks represent intervals of speech, stars represent events.

4 EVALUATION

As mentioned in Section 2, referring expressions in the original corpus were annotated for reference. We therefore knew for each referring expression to which object it actually referred. This information formed the gold standard for the evaluation of our approach to reference resolution.

Once we had processed all the dialogues in the corpus we started the evaluation. As the first step we defined the baseline for the evaluation. We did this by taking the unmodified instruction/event pairs. In this set, each instruction was associated with the closest following event. This

is a relatively simple way of associating instructions and events since it assumes that each instruction was perfectly interpreted and fulfilled directly after the instruction, with no other events occurring between them. This is a very strong assumption, because misunderstandings between human communicators frequently occur. This results in the user executing a wrong action or not immediately performing the action. We therefore suspected that this initial association contained many false pairings.

We take this set of associations as the baseline in this experiment in the sense that this association is the most simple but plausible one that can be created without much effort.[2]

We then set out to detect likely false pairings by looking at the distance between instruction and event.

We used two basic approaches: If the distance between an instruction and the following event exceeded a given threshold, we would refuse to rate it, leaving the decision up to a human annotator ("late cut-off"). If the distance fell below a given threshold, we also did not rate the pair ("early cut-off"). In an actual annotation scenario, the examples that were not rated could be passed on to a human annotator who could judge them manually.

We ran the association algorithm (Algorithm 1) to create a set of instruction-event pairs. We subsequently judged the results by a number of different distances. The results for the time distances are presented in Table 1 , the results for word distances in Table 2. They show:

- the total number of cases (Column 1)
- the number of cases that were removed because of the cut-off criterion (Column 2)
- the percentage of removed cases (Column 3)
- the number of remaining cases (Column 4)
- the number of cases where the association between instruction and event was correct according to the gold standard (Column 5). This row is illustrated in Figure 3(a) for time distances and Figure 3(b) for word distances.
- the percentage of correct cases among the cases that were not removed (Column 6)

[2] A different measure that could serve as a basis for evaluating results later on would be the stochastic probability of picking the right referent when choosing randomly among the visible objects. In a related experiment [7] we determined this probability to be 57.4% for this corpus. However, in the current experiment we cannot assume that the intended referent is visible, therefore this approach is not directly applicable.

– the overall percentage of the correctly associated cases among the number of total cases (Column 7)

For early cut-off, we used the distances 5, 7.5, 10, 15 and 20 seconds and 5, 10, 20, 40, 50 and 60 words. The results are displayed in Table 3 and 4.

Table 1. Results for the different time distance values for late cut-off.

	Total	Removed		Remaining	Correct		Total correct
	#	#	%	#	#	%	%
Col. Nr.	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Baseline	135	0	0.0	135	93	68.9	68.9
Time 5s	135	88	65.2	47	34	72.3	25.2
Time 7.5s	135	61	45.2	74	61	82.4.	45.2
Time 10s	135	40	29.6	95	80	84.2	59.3
Time 15s	135	34	25.2	101	84	83.2	62.2
Time 20s	135	26	19.3	109	88	80.7	65.2

Table 2. Results for the different word distance values for late cut-off.

	Total	Removed		Remaining	Correct		Total correct
	#	#	%	#	#	%	%
Col. Nr.	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Baseline	135	0	0.0	135	93	68.9	68.9
Words 5	135	122	90.4	13	4	30.8	2.9
Words 10	135	102	75.6	33	21	63.6	15.5
Words 20	135	62	45.9	73	59	80.8	43.7
Words 40	135	38	28.1	97	80	82.5	59.3
Words 50	135	31	23.0	104	85	81.7	63.0
Words 60	135	26	19.3	109	86	78.9	63.7

To give an intuition about the significance of the different columns: Column (3) tells us for what fraction of the cases the algorithm refused to make a judgment. The figure basically tells us how much work is left for the human annotator. Column (6) tells us how many of the cases that were not removed were actually correct. This basically gives us a measure of the quality of the predictions made.

Table 3. Results for the different time distance values for early cut-off.

	Total	Removed		Remaining	Correct		Total correct
	#	#	%	#	#	%	%
Col. Nr.	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Baseline 0	135	0	0.0	135	93	68.9	68.9
Time 1s	135	7	5.2	128	93	72.7	68.9
Time 2s	135	8	5.9	127	93	73.2	68.9
Time 2.5s	135	13	9.6	122	89	73.0	65.9
Time 5s	135	47	34.8	88	59	67.0	43.7

Table 4. Results for the different word distance values for early cut-off.

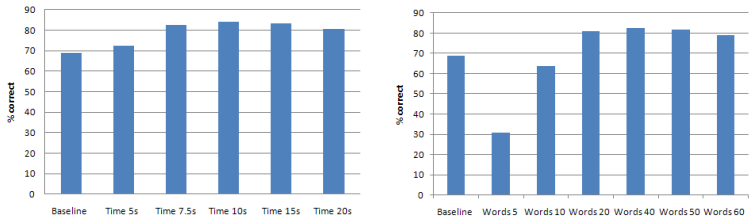
	Total	Removed		Remaining	Correct		Total correct
	#	#	%	#	#	%	%
Col. Nr.	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Baseline	135	0	0.0	135	93	68.9	68.9
Words 1	135	3	2.2	132	93	70.5	68.9
Words 2	135	4	2.9	131	93	71.0	68.9
Words 3	135	6	4.4	129	93	72.0	68.9
Words 4	135	7	5.2	128	92	71.9	68.1
Words 5	135	11	8.1	124	90	72.6	66.7
Words 7	135	16	11.9	119	86	72.3	63.7

Column (7) tells us which fraction of the total number of cases was correctly annotated according to the manual annotations from the corpus.

As we can see, the baseline alone delivers somewhat acceptable results. However, if we were to use the baseline approach in an actual annotation task, we would end up with false results with no indication of which results were doubtful decisions.

Using the cut-off approach removes cases while increasing the correctness of the remaining ones. This means that the cut-off strategy helps us identify cases that are likely to be incorrect.

For late cut-off we observe that low threshold values remove many cases while higher values remove less cases. We also observe that that the overall correctness of the remaining cases peaks at a certain point (around 10 words for the word distance cut-off and 40 seconds for the time distance cut-off) and decreases for greater values. This may seem counter-intuitive, but can be explained: early cut-off values remove the majority of cases, including many correct ones, and tend to preserve cases



(a) Proportion of correct judgements by time distance.

(b) Proportion of correct judgements by word distance.

Fig. 3. Graphs showing the distribution of correct judgements for late cut-off.

where instruction and event are very close together. As discussed earlier, it is a reasonable assumption that these cases tend to be incorrect matches.

The observations indicate that this is indeed the case and highlights the need for trying out the early cut-off approach.

In early cut-off, small values remove few cases and large values remove many. Again we observe a peak and subsequent drop in correctness. Early cut-off achieves at best a correctness around 73% while late cut-off achieves a correctness around 83%, which in both cases is a clear improvement over the baseline.

The results indicate that both early and late cut-off remove incorrect candidates, thereby increasing the correctness of the remaining candidate set. In addition to that we know early and late cut-off remove cases from opposing sides of the spectrum (cases where instruction and event are close together and cases where the opposite is the case). It is therefore likely that one approach captures cases the other does not cover. It is therefore promising to develop an approach that integrates both.

5 Conclusions and Future Work

We presented an approach towards automatically generating referring expression annotations for situated dialogues that exploits the interpretation of referring expressions by the participants of the dialogue. We demonstrated the approach for a specific type of references in a specific corpus. The approach can be generalized to other types of references in other corpora under two conditions: (1) The references must be contained in instructions that cause events involving the referents and (2) It must be possible to automatically detect these events.

On a conceptual level we can relate this approach to more general approaches that are based on intention recognition and perceived affordances [8]. In [9] Gorniak et. al. describe using intention recognition to improve reference resolution in the context of a game. In this work we somewhat reverse this approach: We take actions in the game as hypotheses about the intention of instructions (quasi hijacking the interpretation performed by the listener) and use the objects affected by the action as the referent of referring expressions in the instruction.

We explored different early and late cut-off values that give an indication for which suggested linkings might be unreliable. Deciding on a particular cut-off point, allows the algorithm to decide which cases are easy and reliably judged, and which cases are hard to judge, and should rather be inspected by a human annotator. However, it is not immediately clear how to derive cut-off values for new domains. It may be possible to directly transfer values between sufficiently similar domains. Another approach would be to manually create a gold standard annotation for a small subset of the domain and to determine values for this subset and transfer them to the whole domain.

Overall, the approach manages to produce at best a success rate around 80% if only one cut-off strategy is used.

To increase this value, we are investigating the use of cut-off windows instead of cut-off points. The results of the experiments suggest that very early events as well as very late events are poor candidates for annotation. Therefore it appears to be sensible to remove early as well as late events. On a trial basis we combined different good values for early and late cut-off and achieved a success rate around 93%, while still annotating about 45% of all cases (due to lack of space we unfortunately cannot present this data). While this is still quite a bit away from correctly annotating all examples, it still enables us to automatically annotate a sizeable subset of cases with good success rate.

The GIVE corpus [10] comprises a data set, that is very similar to the one we used, but is based on written instead of spoken language and features only monologue. In further work we may investigate how well our approach can be applied to this corpus and in how far results are transferable.

REFERENCES

1. Jurafsky, D., Martin, J.H.: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech

Recognition. second edition edn. Prentice Hall (2008)
2. Grosz, B.J., Weinstein, S., Joshi, A.K.: Centering: A framework for modeling the local coherence of discourse. Computational Linguistics **21**(2) (1995) 203–225
3. Salmon-Alt, S., Romary, L.: Reference resolution within the framework of cognitive grammar. In: Proceedings of the International Colloqium on Cognitive Science. Volume abs/0909.2626., San Sebastian, Spain (2000)
4. Heeman, P., Allen, J.: Trains 93 dialogues. Technical report, University of Rochester, Rochester, NY, USA (1995)
5. Thompson, H., Anderson, A., Bard, E.G., Doherty-Sneddon, G., Newlands, A., Sotillo, C.: The HCRC Map Task corpus: Natural dialogue for speech recognition. In: Proceedings of the workshop on Human Language Technology, HLT-93, Princeton, New Jersey, Association for Computational Linguistics (1993)
6. Stoia, L., Shockley, D.M., Byron, D.K., Fosler-Lussier, E.: Scare: A situated corpus with annotated referring expressions. In: Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008). (2008)
7. Schütte, N., Kelleher, J., Mac Namee, B.: Visual salience and reference resolution in situated dialogues: A corpus-based evaluation. In: Dialog With Robots. Papers from the AAAI Fall Symposium, Menlo Park, California, AAAI Press (2010)
8. Gorniak, P.: The Affordance-Based Concept. Ph.d., Massachusetts Institute of Technology (2005)
9. Gorniak, P., Orkin, J., Roy, D.: Speech, space and purpose: Situated language understanding in computer games. In: Twenty-eighth Annual Meeting of the Cognitive Science Society Workshop on Computer Games. (2006)
10. Gargett, A., Garoufi, K., Koller, A., , Striegnitz, K.: The Give-2 corpus of giving instructions in virtual environments. In: Proceedings of the 7th Conference on International Language Resources and Evaluation (LREC), Valletta, Malta, European Language Resources Association (ELRA) (2010)

NIELS SCHÜTTE
APPLIED INTELLIGENCE RESEARCH CENTRE,
DUBLIN INSTITUTE OF TECHNOLOGY,
IRELAND
E-MAIL: <NIELS.SCHUTTE@STUDENT.DIT.IE>

JOHN KELLEHER
APPLIED INTELLIGENCE RESEARCH CENTRE,
DUBLIN INSTITUTE OF TECHNOLOGY,
IRELAND
E-MAIL: <JOHN.D.KELLEHER@DIT.IE>

BRIAN MAC NAMEE
APPLIED INTELLIGENCE RESEARCH CENTRE,
DUBLIN INSTITUTE OF TECHNOLOGY,
IRELAND
E-MAIL: <BRIAN.MACNAMEE@DIT.IE>