ERROR DETECTION AND ERROR RECOVERY FROM SPEECH RECOGNITION: LANGUAGE ENGINEERING STRATEGIES

María Teresa López-Soto and José F. Quesada
University of Seville, Spain

ABSTRACT

This work presents a series of strategies developed for a Natural Language Processing (NLP) module, which receives as input a sequence of words generated by a speech recognizer. These strategies have proved to be very effective in detecting and correcting the errors generated at the speech recognition stage. These strategies can be classified into three blocks: lexico-morphological and grammatical flexibility, metrics. The model has been implemented into a real system (automatic telephone operator system); the results generated for this paper have been taken after parsing a Spanish corpus of 1,719 requests to the operator.

1. INTRODUCTION

Speech recognition and language engineering have recently been integrated in real applications: ticket reservation systems, monitor and control systems, help-desk systems, database access systems, etc. Yet, continuous, speaker-independent speech recognition technology is not always efficient as regards precision of the recognized input. Thus, it is necessary to develop those strategies which allow for the detection and correction of the errors generated at the recognition stage. This must be done augmenting robustness and efficiency of the final system.

This work presents a series of strategies based on language engineering techniques, which have proved to be very useful for the task of detecting and correcting errors generated at the speech recognition stage. We present a systemic study of the most common errors in speech recognition and propose a classification of these errors into four categories: under-recognition, over-recognition, close-recognition and mis-recognition.

We also discuss the general architecture of the system which integrates speech recognition (at the input), natural language processing (NLP) (grammar and semantic analyses) and dialogue management (DM). The experiments have been done with a prototype model of a telephone operator system; the experiments have been carried out in Spanish. The system can be described as a multi-layer model where communication between the speech recognizer and the NLP module is done through an N-best ordered set of hypotheses. The CTAC protocol is the representation structure, which permits the interaction between the NLP and the DM. This protocol is based on complex feature structures [5] and allows for the flow of information in a bi-directional manner (from the speech recognition module to the NLP and DM modules as well as in the reverse direction, by launching predictions to the speech recognizer).

We also analyze the NLP strategies that have been implemented to deal with the hypotheses generated by the recognizer. These strategies can be classified into three blocks:

1. Semantic adequacy: We use a semantic-oriented grammar augmented by a priority control mechanism, which monitors the semantic adequacy of the input string.
2. Lexico-morphological and grammatical flexibility. We have designed semantic tagging and relaxed parsing strategies, which include the techniques of analysis of partial strings and the control of void words.
3. Metrics to find the information focus, which allows for the selection of the semantically appropriate hypothesis.

The use of these strategies shows a very high rate of success for the detection and correction of speech recognition errors in the domain of requests made to the telephone operator.

2. CHARACTERISTICS OF THE CORPUS

1.1. Characteristics of Spoken Language. Spoken language is characterized by the following features:

1. Syntactic disorder: the canonical syntactic constituency may be disrupted. Sometimes, there is a serious interruption in the current of discourse, as in the case of the so-called "speech disfluencies" (repetitions, false starts, speech repairs [2]).
2. Vocabulary: the speaker freely expresses his/her emotions, state of mind, attitudes and other situational information by using idioms, colloquial expressions, colloquial terms, etc.

In the corpus designed for this work, the language that the speakers use shows a number of terms which merely reflect situational information. For the task we present here (commands to activate different telephone functions) those terms have been considered as lacking relevant information for the semantic representation of the sentence. As explained below, errors made at the recognition stage will determine the need to recover information from partial strings. However, speech disfluencies have not been dealt with in this work.

1.2. The Output of the Speech Recognizer. The NLP module receives as input the best hypotheses of a series of N-best hypotheses generated by the speech recognizer. It is widely known that even the most efficient and robust speech recognizers show an error rate of 5% (approximately). Thus, in the design of the NLP module we had to include the appropriate mechanisms to overcome the errors produced at the speech recognition stage. Speech recognition errors have been classified into the following four blocks:

1. Under-Recognition: under-recognition takes place when any fragment of the original sequence is deleted. The speech recognizer commonly under-recognizes monosyllabic words, such as prepositions and other grammatical words (determiners, adverbs, etc.).

(a)

```
OriginalSentence: ( hacer una llamada )
InputSentence: ( hacer llamada )
```

*(*OriginalSentence:make a telephone call*)
*(*InputSentence:make telephone call*)

2. Over-Recognition: over-recognition takes place when extra fragments, which were not uttered, are inserted. Very often, the speech recognizer generates two words when only one was uttered. If the term inserted corresponds to a grammatical word, the meaning of the sentence is not altered.
(b)
OriginalSentence: (tengo algun mensaje almacenado)
InputSentence: (tengo algun mensaje la almacenado)
*(*OriginalSentence:do I have any stored message*)
*(*InputSentence:do I have any stored the message*)

However, the meaning of the sentence may change:
(c)
OriginalSentence: (desactiva el modo no molesten)
InputSentence: (es activa el modo no molesten)
*(*OriginalSentence:deactivate the still command*)
*(*InputSentence: is activate the still command*)

In the second example, the verb "desactiva" (deactivate) has been recognized in the sequence "es activa" (*is activate).

3. Close-Recognition: close-recognition takes place when the speech recognizer generates sequences of words which are similar to the ones originally uttered. The words that have been closely recognized often have the same morphological root as the original ones.
(d)
OriginalSentence: (anotar en mi agenda un nombre)
InputSentence: (anota en mi agenda un nombre)
*(*OriginalSentence:including a new name in my directory*)
*(*InputSentence:include a new name in my directory*)

4. Mis-Recognition: mis-recognition takes place when the speech recognizer generates sequences of words which show very little relation to the original sentence uttered by the speaker.
(e)
OriginalSentence: (quiero hablar con Celinda)
InputSentence: (y si quien pasa Celinda)
*(*OriginalSentence:I want to talk to Celinda*)
*(*InputSentence:and yes who pass Celinda*)

3. THE CTAC PROTOCOL AND THE GENERAL ARCHITECTURE OF THE SYSTEM

The NLP module is the language understanding module of a telephone operator system with voice I/O. The speech recognizer generates a series of N-best hypotheses. The best hypothesis is then analyzed by the language understanding module, which generates a meaning representation in the shape of a CTAC structure [4]. This representation is processed by a dialogue management system (DM).

The CTAC protocol is a hierarchical representation of semantic types. The structure shows several gaps to be filled by the semantic information in the domain. This process is done at the grammatical stage, where non-terminal nodes are associated with different CTAC structures by means of functional equations. These equations are processed during unification to generate a parsing tree in which the root node provides a semantic representation (CTAC). The CTAC structure is retained at the DM stage. If the structure is not complete, the DM can start a communicative exchange with the user to obtain the missing information.

The CTAC structure contains four fields: `CLASS`, `TYPE`, `ARG`(ument) and `CONT`(ent). In this domain, parsed sentences are divided into `CLASS`: `Function` (for example, "activate" in "I want to activate call waiting") and `CLASS`: `Object` (for example, "John" in "I want to talk to John". The field `ARG` is reserved for subcategorization, in the present application the subcategorization are expressed in the arguments required by functions (in the example, the function "telephone_call" requires one of three possible argument: name, company, telephone number). `CONT` includes the lexical information, as shown in the following example. Recursive lists are permitted.
(f)

```
(CLASS: Function,
 TYPE: HacerLlamada,
 ARG: [Nombre]|[Empresa]|[Numero],
 Numero:
             (CLASS: Object,
              TYPE: Numero,
              ARG: [ ],
              CONT: [93 000 111]))
*(CLASS: Function,
 TYPE:PhoneCall,
 ARG: [Name]|[Company]|[Number],
 Number:
             (CLASS: Object,
              TYPE: Number,
              ARG: [ ],
              CONT: [93 000 111]))
```

4. PARSING STRATEGIES

4.1 Lexical Analysis: Void Words. As stated above, the terms deleted at the recognition stage usually correspond to grammatical words. The same happens when over-recognition takes place. The existence of these two kinds of error has led to the adoption of the so-called "concept-spotting" technique. Thus, the NLP module does not consider those terms which are not relevant to the understanding of the meaning of the sequence (grammatical words). This is possible because the system has a very restricted domain and the semantic representation obtained is usually the combination of a telephone function and its arguments.

To obtain the basic semantic representation, the lexical category VOID is assigned to all the grammatical words. The system includes the `ConfVoidWordsIgnore` configuration mechanism.

When the tool is activated, all VOID words are ignored, that is, they are not processed at the morphological level. In the following example, the user has been speaking to a colleague when ordering a telephone command.
(g)
Esto, Pepe... quiero que grabes el mensaje
*(*That is, Pepe... I want to store the message*)

The mechanism deletes those terms which are not relevant to the task of generating the following layer to the lexical level:

grabes mensaje
*(*store message*)

The final sequence can be matched to a complete semantic structure (`record_message`).

When the number of VOID words in the sequence is too high for a correct understanding of the message, the `VoidThreshold` control mechanism can be activated.

When this mechanism is activated, the system can reject those sequences which show a certain number of VOID words. Very often, the rejected sentences correspond to ill-formed sentences at the recognition stage.
(h)
no yo yo yo yo eh
*(*no I I I I eh*)

The mechanism prevents the construction of a parse tree when serious errors have been produced at the recognition stage (mis-recognition).

4.2. Analysis of Partial Strings. The NLP module shows a relaxation approach at the analysis stage [1,3,4]. On the one hand, spoken language frequently shows syntactic disorder. On the other hand, when recognition errors are produced, the meaning of the utterance may not be constructed from the whole sequence. The parser of the system includes the following mechanism:

`ConfParserSolutionFullString`

When the mechanism is activated, the parser only accepts the root nodes which spawn the whole of the sequence (classical manner).

The mechanism

`ConfParserSolutionPartialStrings`

accepts every root node obtained from the parsing of the sequence. This mechanism also incorporates several strategies for the deletion of redundant nodes. For two nodes associated to the same symbol, if the interval of the first node includes the interval of the second one, then the second one is erased. There is another filter which deletes the nodes embedded, even though they are not associated to the same symbol.

The activation of the analysis of partial strings is very useful when the DM has to recover missing information: all partial analyses are kept active while the DM selects the correct one to complete the CTAC structure.

5. AMBIGUITY

The system incorporates a powerful mechanism of disambiguation. Ambiguity is a linguistic phenomenon which poses serious This is necessary when the analysis of partial strings is active, so that there is a selection of parse trees. The disambiguation mechanism is done with three different algorithms:

1. Length. The system gives priority to the parse tree which contains a larger number of nodes in its interval.
2. Position. The system gives priority to the focus of information. This way, latter trees are preferred to earlier ones.
3. Global criterion. The final result is the combination of the two previous ones.

5.1. Length and Position Algorithms. To understand the functionality of these two algorithms let us consider the following sequence generated by the morphological analyzer as an example:
(i)
>> OriginalSentence:(quiero escuchar los mensajes almacenados)
>> InputSentence:(cero escuchar los mensajes almacenados)
*(*OriginalSentence:I want to listen to stored messages*)
*(*InputSentence:zero want to listen to stored messages*)
>> LexicalAnalysis (LNumber LListento LMessage LStored)

The Length and Position algorithms operate based on the following concepts:
1. First, numerical values are marked between every item generated by the morphological analyzer, as follows:

```
[0] LNumber [1] LListento [2] LMessage [3] LStoredMessage [4]
```

2. Each parse tree is delimited by the two extremes which define its interval. For our example, `LNumber` would be enclosed to the [0-1] extremes while `LStoredMessage` would have the [3-4] extremes as its interval.
3. The left and right extremes of the interval are called LExtreme and RExtreme, respectively

The Position algorithm is defined in the following equation:

$$Position = LExtreme + RExtreme$$

The Length algorithm is defined in the following equation:

$$Length = RExtreme - LExtreme$$

Finally, the P*L algorithm is calculated:

$$P*L = Position * Length$$

The values assigned by these two algorithms are provided under the analysis output.
(j)
>>InputSentence(cero escuchar los mensajes almacenados)
>>LexicalAnalysis(LNumber LListento LMessage LStored)
>>OutputParser:
* Number:P14P(LNumber(zero))
  Position = 1, Length = 1, P*L = 1
* StoredMessage:P170P(LListento(listen_to), LStored(stored),LMessage(message))
  Position = 5, Length = 3, P*L = 15

In the example (“Zero listen to old messages”), the disambiguation tool in the system selects the second parse tree (StoredMessage = “listen to old messages”), taking into account

the value provided by the Position and Length algorithms (P*L=15).

5.2. Global Criterion. There is a third mechanism for disambiguation in the system: the global criterion or G criterion. This algorithm is based on a combination of the Position and Length algorithms as a priority control over ambiguous analyses. The results obtained from the application of the P*L algorithms proved to be efficient in a 85% of the cases. However, there is 15% of the analyses which cannot get disambiguated by the application of the P*L mechanism only. The G criterion improves these results up to a 100%. We can see this with an example.

(k)
>> InputSentence: (borrar numero grabado)
*(*delete stored number*)
>> LexicalAnalysis (LDelete LAuxNumber LStored)
>>OutputParser:
* StoredMessage:P137P(LStored(stored))
Position = 5 Length = 1 P*L=5 G =8
* DeleteNumber:P45P(LDelete(delete),
LAuxNumber(number))
Position=2 Length=2 P*L=4 G=8

The activation of the G criterion is done with the `ActivateGCriterion` configuration command. The mechanism is applied to every single node in the analysis tree, and not only to root nodes. The formula for the algorithm is as follows:

$$G = (((Length - 1)*2)*Position) + (Complexity * 2)$$

The Complexity of every terminal node is obtained from its Length. That is, for every terminal node TN, we define its complexity in this way:

$$Complexity(TN) = Length(TN)$$

The Complexity of every non-terminal node will be taken from the summation of the Complexity values of its daughters.

$$Complexity(N) = Complexity(N1) + \ldots + Complexity(Nk)$$

where N refers to the node and N1, ..., Nk refers to the set of daughters.

The algorithm selects the analysis with a higher value. Let us see the function of the G Criterion with an example:

(l)
>> InputSentence (Antonio ... desactiva el modo llamada en espera)
*(*Antonio ... deactivate call waiting*)
>> LexicalAnalysis (LName LDisconnect LCallWaiting)
>> OutputParser:
* Name:P6P(LName(Antonio))
Position = 1,Length = 1, P*L = 1, G = 8
* OffCallWaiting(LOff(deactivate),
CallWaiting:P222P(call_waiting))
Position = 4,Length = 2, P*L = 8, G = 36

The G criterion has proved to be efficient in the selection of the correct analysis when the mechanism of analysis of partial strings is activated.

5. RESULTS

The system was trained with a corpus of 1,727 requests to the telephone operator; the speech recognizer processed 1,719 requests. In this section we present the results of the lexical and grammatical analyses on the corpus.

The mechanism of analysis of partial strings improves the performance of the system in 38.4% when the original sentences were tested and in 52.8% when the input to the NLP module is the output of the speech recognizer. The disambiguation mechanism operate in 32,0% of the cases for the original sentences and in 47,2% of the total for the recognized sentences.

We also present the results obtained after analyzing a corpus of 1,576 sentences which had not been previously tested by the system. The mechanism of analysis of partial strings operates in 38.4% of the cases and the disambiguation algorithms are triggered in 36.1% of the cases.

In the evaluation of the corpora we have also tested the "semantic adequacy" of the analysis. The "semantic adequacy" of the analysis checks that the root node generated corresponds to the expected CTAC structure. The analysis of the original sentences generate a correct CTAC structure in 91.3% of the cases. Semantic adequacy shows a 68.1% success for the recognized sentences and 79.8% for the unseen corpus.

ACKNOWLEDGMENTS

This work has been financed by the Spanish Ministry of Education and Science with the research code CICYT 95-0214-OP. Funding was also received from Telefónica Investigación y Desarrollo.

REFERENCES

[1] Carbonell, J. and Hayes, P. 1983. Recovery Strategies for Parsing Extragrammatical Language. In *American Journal of Computational Linguistics*, 9(3-4):123-146.

[2] Heeman, P. A. 1997. *Speech Repairs, Intonational Boundaries and Discourse Markers: Modeling Speakers' Utterances in Spoken Dialog*. Ph.D. Dissertation. Department of Computer Science, University of Rochester, New York. Communication of the *ACM*, 13(2):94-102.

[3] Hobbs, J., Appelt, D., Bear, J., Tyson, M. and Magerman, D. 1992. Robust Processing of Real-World Natural-Language Texts. In P. Jacobs. ed. *Text Based Intelligent Systems*, 13-33. Lawrence Erlbaum Associates, Hillsdale, NJ.

[4] López-Soto, M.T. and J. F. Quesada. 1998. Parsing Strategies for a Spoken Language Processing System. In *Procesamiento del Lenguaje Natural*, 23:8-15

[5] Quesada, J. F. 1998. Lexical Object Theory: Specification Level. In *Grammars*, 1:57-84.

[6] Seneff, S. 1992. A Relaxation Method for Understanding Spontaneous Speech Utterances. In *Proceedings of the Speech and Natural Language Workshop*, 299-304.