Language Understanding Using n-multigram Models*

Lluís Hurtado, Encarna Segarra, Fernando García, and Emilio Sanchis

Departament de Sistemes Informàtics i Computació (DSIC),
Universitat Politècnica de València (UPV),
Camí de Vera s/n, 46022 València, Spain
{lhurtado,esegarra,fgarcia,esanchis}@dsic.upv.es

Abstract. In this work, we present an approach to language understanding using corpus-based and statistical language models based on multigrams. Assuming that we can assign meanings to segments of words, the *n-multigram* modelization is a good approach to model sequences of segments that have semantic information associated to them. This approach has been applied to the task of speech understanding in the framework of a dialogue system that answers queries about train timetables in Spanish. Some experimental results are also reported.

1 Introduction

Nowadays, the use of automatic learning techniques for Language Modelling is quite extensive in the field of Human Language Technologies. A good example of this can be found in the development of Spoken Dialogue systems. Very important components of these systems, such as the Language Model of the speech recognition component, the Language Model of the understanding component and the dialogue structure, can be modelled by corpus-based and statistical finite-state models. This is the case of the widely used n-gram models [1][2].

An n-gram model assigns a probability to a word depending on the previous n-1 words observed in the recent history of that word in the sentence. In these models, the word is the selected linguistic unit for the language modelization, and the recent history considered for the model always has the same length. Over the last few years, there has been an increasing interest in statistical language models which try to take into account the dependencies among a variable number of words; that is, stochastic models in which the probability associated to a word depends on the occurrence of a variable number of words in its recent history. This is the case of the grammar-based approaches [3][4][5], in which models take into account variable-length dependencies by conditioning the probability of each word with a context of variable length. In contrast, in segment-based approaches such as multigrams [6][7][8], sentences are structured into variable-length segments, and probabilities are assigned to segments instead of words. In other words, multigram approaches to language modelling take segments of

* Work partially funded by *CICYT* under project TIC2002-04103-C03-03, Spain.

J. L. Vicedo et al. (Eds.): EsTAL 2004, LNAI 3230, pp. 207–219, 2004.

words as basic units; these models try to naturally model the fact that there are certain concatenations of words that occur very frequently. These segments could constitute one of the following: relevant linguistic units; syntactic or semantic units; or a concatenation of words, which works well from the language modelling point of view (independently of the linguistic relevance of the segments).

Language Understanding systems have many applications in several areas of Natural Language Processing. Typical applications are train or plane travel information retrieval, car navigation systems or information desks. In the last few years, many efforts have been made in the development of natural language dialog systems which allow us to extract information from databases. The interaction with the machine to obtain this kind of information requires some dialog turns. In these turns, the user and the system interchange information in order to achieve the objective: the answer to a query made by the user. Each turn (a sequence of natural language sentences) of the user must be understood by the system. Therefore, an acceptable behavior of the understanding component of the system is essential to the correct performance of the whole dialog system.

Language understanding can be seen as a transduction process from sentences to a representation of their meaning. Frequently, this semantic representation consists of sequences of concepts (semantic units). Multigram models have been applied to language modeling tasks [6][8]; however, as the multigram approach is based on considering a sentence as a sequence of variable-length segments of words, it could be interesting to apply this methodology to language understanding.

In this work, we propose the application of multigram models to language understanding. In this proposal, we associate semantic units to segments, and we modelize the concatenation of semantic units as well as the association of segments of words to each semantic unit. This approach has been applied to the understanding process in the BASURDE dialog system [9]. The BASURDE system answers telephone queries about railway timetables in Spanish.

This paper is organized as follows: in Section 2, the concept of *n-multigram* is described. In Section 3, the task of language understanding is presented and in Section 4, the application of *n-multigram* to language understanding is proposed. In Section 5, the two grammatical inference techniques used in the experiments are illustrated. Finally, results of the application of the *n-multigram* models to a task of language understanding and some concluding remarks are presented.

2 The *n-multigram* Model

Let W be a vocabulary of words, and let $w = w_1\ w_2\ w_3\ w_4$ be a sentence defined on this vocabulary, where $w_i \in W, i = 1, \ldots, 4$. From a multigram framework point of view, the sentence w has the set of all possible segmentations of the sentence associated to it. Let S be this set of segmentations for w. If we use the symbol # to express the concatenation of words which constitutes the same segment, S is as follows:

$$S = \left\{ \begin{array}{ll} w_1\ w_2\ w_3\ w_4, & w_1\ w_2\ w_3\#w_4, \\ w_1\ w_2\#w_3\ w_4, & w_1\#w_2\ w_3\ w_4, \\ w_1\ w_2\#w_3\#w_4, & w_1\#w_2\ w_3\#w_4, \\ w_1\#w_2\#w_3\ w_4, & w_1\#w_2\#w_3\#w_4 \end{array} \right\}$$

From a multigram language model point of view, the likelihood of a sentence is computed by summing up the likelihood values of all possible segmentations of the sentence into segments [6] [8]; let w be a sentence, and let S be the set of all segmentations of w. The likelihood of the sentence $\mathcal{L}(w)$ given a multigram language model is:

$$\mathcal{L}(w) = \sum_{s \in S} \mathcal{L}(w,\ s) \tag{1}$$

where, $\mathcal{L}(w,\ s)$ is the likelihood of the sentence w given the segmentation s.

The likelihood of any particular segmentation depends on the model assumed to describe the dependencies between the segments. The most usual approach, called *n-multigram* [8], assumes that the likelihood of a segment depends on the n-1 segments that precede it. This approach can be seen as an extension of n-gram models of words to n-gram models of segments. Therefore, the likelihood of a segmentation is:

$$\mathcal{L}(w,\ s) = \prod_{\tau} p(s_{(\tau)} | s_{(\tau-n+1)} \cdots s_{(\tau-1)}) \tag{2}$$

where $s_{(\tau)}$ represents the τ-th segment in the segmentation s.

Due to the high number of parameters to estimate, it is convenient to define classes of segments. The formula presented above becomes:

$$\mathcal{L}(w,\ s) = \prod_{\tau} p\left(C_{q\left(s_{(\tau)}\right)} | C_{q\left(s_{(\tau-n+1)}\right)} \cdots C_{q\left(s_{(\tau-1)}\right)}\right)\ p\left(s_{(\tau)} | C_{q\left(s_{(\tau)}\right)}\right) \tag{3}$$

where q is a function that assigns a class to each segment (it is generally assumed that a segment is associated to only one class); $C_{q(s_i)}$ is the class assigned to the segment s_i; and $p\left(s_i | C_{q(s_i)}\right)$ is the probability of the segment s_i in its class.

Thus, an *n-multigram* model based on classes of segments is completely defined by: the probability distribution of sequences of classes, and the classification function q.

As we mentioned above, to obtain the likelihood of a sentence $\mathcal{L}(w)$, all the possible segmentations have to be taken into account. It can be computed as follows:

$$\mathcal{L}(w) = \alpha(|w|) \tag{4}$$

where $\alpha(t)$ is the likelihood of $w_1 \cdots w_t$, that is, the prefix of w of length t. Considering that the number of words in the segments is limited by l, we can define $\alpha(t)$ as:

$$\alpha(t) = \sum_{i=1}^{l} \alpha_i(t) \tag{5}$$

where $\alpha_i(t)$ is the likelihood of the sentence $w_1 \cdots w_t$ which only takes into account those segmentations whose last segment is composed by i words. This value can be calculated as:

$$\alpha_i(t) = \begin{cases} 1 & : \quad t = 0 \wedge i = 1 \\ \sum_{r=1}^{\min(l,t-i)} \alpha_r(t-i) \cdot p(w_{t-i+1}^{t} | w_{t-i-r+1}^{t-i}) & : 1 \leq t \leq |w| \wedge 1 \leq i \leq l \end{cases} \tag{6}$$

In the case of considering classes of segments (and assuming that each segment has only been associated to a class), (6) can be calculated by:

$$\alpha_i(t) = \begin{cases} 1 & : \quad t = 0 \wedge i = 1 \\ \sum_{r=1}^{\min(l,t-i)} \alpha_r(t-i) \cdot p(w_{t-i+1}^{t} | C_{q(w_{t-i+1}^{t})}) \cdot p(C_{q(w_{t-i+1}^{t})} | C_{q(w_{t-i-r+1}^{t-i})}) & : 1 \leq t \leq |w| \wedge 1 \leq i \leq l \end{cases} \tag{7}$$

where $C_{q(w_{m-i}^{m})}$ is the class associated to the segment $w_{m-i} \# \ldots \# w_m$ by the function q, and $p(w_{m-i}^{m} | C_{q(w_{m-i}^{m})})$ is the probability that this segment belongs to the category $C_{q(w_{m-i}^{m})}$.

3 Language Understanding

A language understanding system can be viewed as a transducer in which natural language sentences are the input and their corresponding semantic representation (frames) are the output. In [10], an approach for the development of language understanding systems that is based on automatic learning techniques has been presented. In this approach, the process of translation is divided into two phases: the first phase transduces the input sentence into a semantic sentence (a sequence of semantic units) which is defined in a sequential Intermediate Semantic Language (ISL). The second phase transduces the semantic sentence into its corresponding frames. Automatic learning techniques are applied in the first phase, and the second phase is performed by a simple rule-based system.

As the ISL sentences are sequential with the input language, we can perform a segmentation of the input sentence into a number of intervals which is equal to the number of semantic units in the corresponding semantic sentence. Let W be the vocabulary of the task (set of words), and let V be the alphabet of semantic units; each sentence $w \in W^*$ has a pair (u,v) associated to it, where v is a sequence of semantic units and u is a sequence of segments of words. That is, $v = v_1 v_2 \ldots v_{\mathrm{n}}, v_{\mathrm{i}} \in V, i = 1, \ldots, n \quad u = u_1 u_2 \ldots u_{\mathrm{n}}, u_{\mathrm{i}} = w_{\mathrm{i}_1} w_{\mathrm{i}_2} \ldots w_{\mathrm{i}_{|u_{\mathrm{i}}|}}$, $w_{\mathrm{i}_{\mathrm{j}}} \in W, i = 1, \ldots, n, j = 1, \ldots, |u_{\mathrm{i}}|$.

For example, in the BASURDE corpus, the sentence "me podría decir los horarios de trenes para Barcelona" (*can you tell me the railway timetables to*

Barcelona), whose translation in ISL could be *consulta <hora_salida> marcador _destino ciudad_destino* (*query <depart_time> destination_marker destination_city*), has the following pair associated to it, which is the output of the first phase of our understanding approach.

(u,v)=$(u_1u_2u_3u_4, v_1v_2v_3v_4)$ where:

Spanish		English	
u_1: me podría decir	v_1: consulta	u_1: can you tell me	v_1: query
u_2: los horarios de trenes	v_2: <hora_salida>	u_2: the railway timetables	v_2: <departure_time>
u_3: para	v_3: marcador_destino	u_3: to	v_3: destination_marker
u_4: Barcelona	v_4: ciudad_destino	u_4: Barcelona	v_4: destination_city

The output for the second phase is the following frame:

(DEPART_TIME)
DESTINATION_CITY: Barcelona

4 Applying *n-multigram* Models to Language Understanding

The first phase of our language understanding approach, that is, the semantic segmentation of a sentence w, consists of dividing the sentence into a sequence of segments ($u = u_1u_2 \dots u_n$) and associating a semantic unit $v_i \in V$ to each segment u_i. Due to the fact that the multigram approach considers sentences as sequences of segments of variable length, it seems especially appropriate for the semantic segmentation problem. In order to apply *n-multigrams* to language understanding, the following considerations must be taken into account:

- The training corpus consists of a set of segmented and semantically labelled sentences.
- The classes of segments are the semantic vocabulary V, which is obtained from the definition of the ISL for the understanding task.
- A segment could be assigned to several classes, so the classification function q must be adapted in order to provide a membership probability for each segment in each class.

4.1 Learning the Semantic *n-multigram* Model

In an *n-multigram* model, there are two kind of probability distributions that have to be estimated: a) the language model for the sequences of classes of segments and b) the membership probability of segments in classes.

The probability distribution (a) is estimated as an n-gram model of classes of segments from the sequences of semantic units $v = v_1v_2 \dots v_n, v_i \in V, i = 1, \dots, n$ of the training set.

A language model is learnt from the set of segments of words u_i of the training set associated to each class. These language models provide the membership probability of segments in classes (b). We have explored some approaches to automatically obtain these language models. In particular, we used n-gram mod-

els, the Prefix-Tree Acceptor (PTA)[1] estimated from the training set, and two Grammatical Inference (GI) techniques (MGGI and ECGI) which are described in section 5.

We define a function of segment classification $q : \Sigma^* \rightarrow 2^V$

$$q(s) = \{v_1, \ldots, v_{|q(s)|}\}, \quad \forall v_i \in V \tag{8}$$

where $q(s)$ is the set of all the classes such as $p(s \mid v_i) > 0$.

4.2 The Semantic Segmentation

Once the model that represents the sequences of semantic units and the probability distributions of membership for each class are learnt, the process to obtain the likelihood α is obtained through:

$$\alpha = \sum_{i=1}^{l} \sum_{v \in q(w_{t-i+1}^{t})} \alpha_i^v(t) \tag{9}$$

This formula (9) is an adaptation of (5) for the case of permitting the assignation of more than one class to each segment. $\alpha(t)$ represents the accumulated likelihood taking into account all the segmentations for the t first words of the sentence $(w_1 \ldots w_t)$. And $\alpha_i^v(t)$ represents the likelihood of all the segmentations of the t first words, taking into account only those that end with a segment assigned to the class v and that have length i. It is calculated as follows:

$$\alpha_i^v(t) = \begin{cases} 1 & : \quad t = 0 \wedge i = 1 \\ 0 & : \quad t = 0 \wedge i > 1 \\ \sum_{r=1}^{l} p(w_{t-i+1}^{t} | v) \cdot \sum_{v' \in q(w_{t-i-r+1}^{t-i})} \alpha_r^{v'}(t-i) \cdot p(v|v') & : \quad 1 \leq t \leq |w| \wedge 1 \leq i \leq l \end{cases} \tag{10}$$

The segmentation of maximum likelihood in terms of semantic units is obtained by using the Viterbi algorithm. Let $w = w_1 w_2 \cdots w_{|w|}$ be the sentence to analyze. The best segmentation is given by:

$$(u, v) = \underset{s \in S,\, q(s_\tau)}{\operatorname{argmax}} \mathcal{L}(w, s) \tag{11}$$

where S is the set of all segmentations of w, and $q(s_\tau)$ is the set of all classes that can be assigned to the τ-th segment of the segmentation s.

5 The Two GI Techniques: The MGGI Methodology and the ECGI Algorithm

As we mentioned above, we used two Grammatical Inference techniques in order to obtain the language models which represent the membership probability of

[1] The finite state automaton that only accepts the strings in the training set.

segments in classes. These techniques are the Morphic Generator Grammatical Inference (MGGI) methodology [11][12] and the Error Correcting Grammatical Inference (ECGI) algorithm [13][14]. In this section, we briefly describe these two GI techniques.

5.1 MGGI

The Morphic Generator Grammatical Inference (MGGI) methodology is a grammatical inference technique that allows us to obtain a certain variety of regular languages. The application of this methodology implies the definition of a renaming function, that is, each symbol of each input sample is renamed following a given function g. Different definitions of the function g will produce different models (stochastic regular automata).

Let R be a sample over the alphabet Σ. Let $\Sigma^{'}$ be a finite alphabet. Let h be a letter-to-letter morphism, $h : \Sigma^{'*} \rightarrow \Sigma^{*}$, and g a renaming function, $g : R \rightarrow \Sigma^{'*}$. The Regular Language, L, generated by the MGGI-inferred grammar, G, is related to R through the expression: $L = h(l(g(R)))$, where $l(g(R))$ is inferred from $g(R)$ through the 2-Testable in the Strict Sense (2-TSS) inference algorithm [15].

Next, we describe the MGGI methodology through an example. In order to better explain its performance, we show the 2-TSS model estimated from a training set and the finite automaton estimated through the MGGI methodology from the same training set:

Let $\Sigma = \{a, b\}$ be an alphabet and let $R = \{aabaa,\ abba,\ abbbba,\ aabbbba\}$ be a training set over Σ.

a) 2-Testable in the Strict Sense inference algorithm (its stochastic version is equivalent to bigrams):

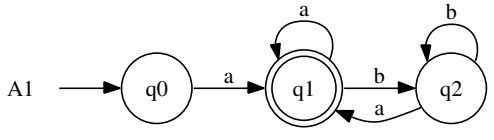


Fig. 1. The finite automaton inferred from R by the 2-Testable in the Strict Sense algorithm

The language accepted by the automaton of Figure 1 is: $L(A1) = a + a(b + a)^{*}a$. That is, the strings in $L(A1)$ begin and end with the symbol a and contain any segment over the alphabet Σ. For example, $a \in L(A1)$, $aaa \in L(A1)$, $ababa \in L(A1)$, $abbba \in L(A1)$, etc.

b) MGGI algorithm:

The renaming function $g : \Sigma^{*} \rightarrow \Sigma^{'*}$ is defined in this example as the relative position, considering that each string is divided into 2 intervals. This definition allows distinguishing between the first and the second parts of the strings.

$g(R) = \{a_1a_1b_1a_2a_2,\ a_1b_1b_2a_2,\ a_1b_1b_1b_2b_2a_2,\ a_1a_1b_1b_1b_2b_2a_2\}$

The language accepted by the automaton in Figure 2 is: $L(A2) = a^{+}b^{+}a^{+}$. That is, the strings in $L(A2)$ contain a sequence of 1 or more symbols a, followed

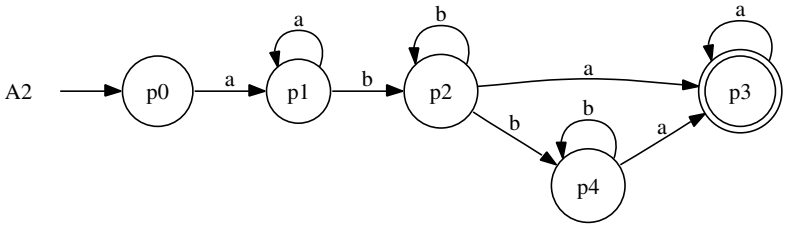


Fig. 2. The finite automaton inferred from R by MGGI algorithm, with a relative position renaming function

by a sequence of 1 or more symbols b, and followed by a sequence of 1 or more symbols a. For example $abbba \in L(A2)$ but $a \notin L(A2)$, $aaa \notin L(A2)$ and $ababa \notin L(A2)$. As can be observed from this example, the language inferred by the MGGI methodology is more similar to the strings in the training set than those inferred by the 2-TSS algorithm.

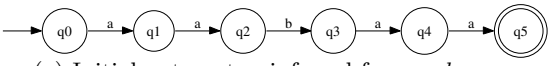
There is another interesting feature of the MGGI methodology. Given a task (a language to modelize), we can choose an adequate definition of a renaming function g. Different definitions of this function produce different models.

5.2 ECGI

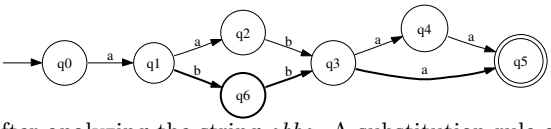
The ECGI algorithm is a Grammatical Inference algorithm that infers a finite-state model in an incremental way and is based on an error correcting parsing. The ECGI builds a finite-state automaton through the following incremental procedure: initially, a trivial automaton is built from the first training word. Then, for every new word, which can not be exactly recognized by the current automaton, the automaton is updated by adding to it those states and transitions which are required to accept the new word. To determine such states and transitions, an error correcting parsing is used to find the best path for the input word in the current automaton. The error rules defined for the analysis are: substitution, insertion and deletion of symbols. This way, the error correcting parsing finds the word in the current inferred language which is closest to the new word, according to the Levenshtein distance. To generate an automaton which is free of loops and circuits, some heuristic restrictions are imposed in the process of adding new states and transitions. Thus, the inference procedure attempts to model the duration and the position of the substructures that appear in the training data. Figure 3 shows the inference process of the ECGI algorithm using the same training set of the examples in Figures 1 and 2.

6 Experimental Results

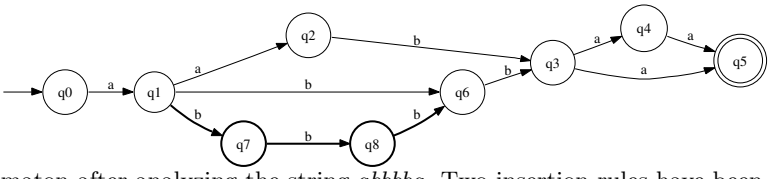
In order to evaluate the performance of the *n-multigram* models in a language understanding task, a set of experiments was conducted on the BASURDE [9] [10] dialog system, which answers queries about train timetables by telephone in



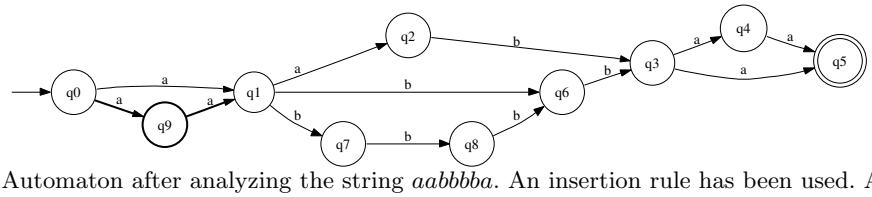
(a) Initial automaton inferred from *aabaa*



(b) Automaton after analyzing the string *abba*. A substitution rule and a deletion rule have been applied to accept this string. A new state, q_6, and some transitions have been added.



(c) Automaton after analyzing the string *abbbba*. Two insertion rules have been used. Two new states, q_7 and q_8, and some transitions have been added.



(d) Automaton after analyzing the string *aabbbba*. An insertion rule has been used. A new state, q_9, and some transitions have been added.

Fig. 3. The inference process of the ECGI algorithm using R

Spanish. The corpus consisted of a set of 215 dialogs, obtained through a Wizard of Oz technique [16]. These dialogs contained 1,440 user turns, with 14,902 words and a vocabulary of 637 different words. A cross-validation procedure was used to evaluate the performance of our language understanding models. To this end, the experimental set was randomly split into five subsets of 288 turns. Our experiment consisted of five trials, each of which had a different combination of one subset taken from the five subsets as the test set, with the remaining 1,168 turns being used as the training set.

We defined several measures to evaluate the accuracy of the strategy in both phases of the understanding process:

- The percentage of correct sequences of semantic units (%cssu).
- The percentage of correct semantic units (%csu).
- The semantic precision ($\%P_s$), which is the rate between the number of correct proposed semantic units and the number of proposed semantic units.
- The semantic recall ($\%R_s$), which is the rate between the number of correct proposed semantic units and the number of semantic units in the reference.
- The percentage of correct frames (%cf), which is the percentage of resulting frames that are exactly the same as the corresponding reference frame.

Table 1. Results of two language understanding models based on *bi-multigrams*: *bi-multigram*-BI y *bi-multigram*-TRI

Semantic Segmentation	%cssu	%csu	$\%P_s$	$\%R_s$
bi-multigram-BI	68.6	87.8	91.0	91.3
bi-multigram-TRI	69.2	87.8	91.4	90.9
Frame Transduction	%cf	%cfs	$\%P_f$	$\%R_f$
bi-multigram-BI	78.5	85.7	88.9	91.3
bi-multigram-TRI	79.0	86.1	89.6	91.0

- The percentage of correct frame slots (frame name and its attributes) (%cfs).
- The frame precision ($\%P_f$), which is the rate between the number of correct proposed frame slots and the number of proposed frame slots.
- The frame recall ($\%R_f$), which is the rate between the number of correct proposed frame slots and the number of frame slots in the reference.

A first set of experiments was performed using the orthographic transcription of user turns. In these experiments, different language understanding models based on multigrams were used. In all the experiments, the modelization of the sequences of segments (semantic units) was done by the *bi-multigram* probabilities. In other words, the probability that a segment of words representing a semantic unit appears depends on the previous segment (the term $p\left(C_{q(s_{(\tau)})}|C_{q(s_{(\tau-n+1)})}\cdots C_{q(s_{(\tau-1)})}\right)$ in formula (3)). The difference among the experiments consisted in which classification function (of the segments in classes) was chosen (the term $p\left(s_{(\tau)}|C_{q(s_{(\tau)})}\right)$ in formula (3)). Table 1 shows the results obtained using bigrams and trigrams of words to assign membership probabilities of the segments to each class.

Table 2 shows the results obtained by using the PTA, the MGGI and the ECGI techniques (described in section 5). These techniques obtain stochastic automata that represent the segments of words that can be associated to each class. These automata represent a modelization of the training samples and supply the membership probabilities associated to the segments. In order to increase the coverage of these models, we used a smoothing method for stochastic finite automata, which was recently proposed in [17].

These results for all techniques are similar. Nevertheless, the models obtained by GI techniques slightly outperform the n-gram models; this could be explained by the fact that the GI techniques better represent the structure of the segments of the training set.

In order to study the performance of the proposed models in real situations, we also performed a second set of experiments using the recognized utterances supplied by a speech recognizer from the same set of dialogs. The recognizer [9] used Hidden Markov Models as acoustic models and bigrams as the language model; its Word Accuracy for the BASURDE corpus was 80.7%. In these

Table 2. Results for the language understanding models: *bi-multigram*-PTA, *bi-multigram*-MGGI and *bi-multigram*-ECGI

Semantic Segmentation	%cssu	%csu	$\%P_s$	$\%R_s$
bi-multigram-PTA	69.0	88.0	91.5	91.1
bi-multigram-MGGI	69.3	88.3	91.6	91.3
bi-multigram-ECGI	68.9	88.3	91.4	91.8
Frame transduction	%cf	%cfs	$\%P_f$	$\%R_f$
bi-multigram-PTA	78.5	86.0	89.3	91.0
bi-multigram-MGGI	79.0	86.3	89.6	91.4
bi-multigram-ECGI	80.0	87.2	90.1	92.5

Table 3. Results for the understanding models based on *bi-multigrams*, with different models of membership probabilities, using the output of the recognizer

Semantic Segmentation	%cssu	%csu	$\%P_s$	$\%R_s$
bi-multigram-BI	44.3	74.3	81.6	82.5
bi-multigram-TRI	44.3	74.4	82.1	82.0
bi-multigram-AP	44.3	74.4	81.7	82.4
bi-multigram-MGGI	44.4	74.6	81.8	82.7
bi-multigram-ECGI	44.7	74.9	81.9	83.2
Frame Transduction	%cf	%cfs	$\%P_f$	$\%R_f$
bi-multigram-BI	54.9	69.6	77.6	81.7
bi-multigram-TRI	55.0	69.8	78.1	81.4
bi-multigram-AP	55.0	69.3	77.5	81.5
bi-multigram-MGGI	55.4	70.1	78.1	81.8
bi-multigram-ECGI	56.4	70.8	78.7	82.5

experiments, the models were the same as before; that is, they were estimated from the orthographic transcription of the sentences, but the test was done with the recognized sentences. Table 3 shows the results for the recognized dialogs, using the same understanding models based on *bi-multigrams* as in the previous experiments (Tables 1 and 2).

The results show a generalized reduction in the performance of the understanding system, if we compare them with those corresponding to transcribed utterances. This reduction is especially significant (over 20%) for %cssu and %cf. They respectively measure the percentage of sentences that are segmented perfectly and the percentage of sentences that are perfectly translated to their corresponding frames. Although less than 42% of the sentences are correctly recognized, the understanding system, *bi-multigram*-ECGI, is able to correctly understand 56% of the sentences. As far as recall and precision, the reduction is much lower, around 10% in most cases. The best model was the *bi-multigram*-ECGI. It obtained a Precision (P_f) and a Recall (R_f) of 78.7% and 82.5%, respectively, at frame level.

7 Conclusions

We have presented an approach to language understanding that is based on multigrams. In our language understanding system, segments of words represent semantic information. Therefore, considering a language modelization based on variable length segments is an appropriate approach. We have proposed different methods to assign segments to semantic units and we have applied our language understanding system based on multigrams to a dialog task. Our results are similar to those obtained by other approaches [10] and show that the proposed methodology is appropriate for the task. This modelization has the advantage that different models can be used to represent the segments associated to the semantic units; even different modelizations can be applied to different semantic units.

We think that the understanding system could be improved by adding information about the relevant keywords for each class in the classification function. It would also be interesting to develop techniques to automatically obtain the set of classes of segments, that is the set of semantic units, which in our system are manually defined.

References

1. Bahl, L., Jelinek, F., Mercer, R.: A maximum likelihood approach to continuous speech recognition. IEEE Trans. on **PAMI-5** (1983) 179–190
2. Clarkson, P., Rosenfeld, R.: Statistical language modeling using the CMU-cambridge toolkit. In: Proc. Eurospeech, Rhodes, Greece (1997) 2707–2710
3. Bonafonte, A., Mariño, J.B.: Language modeling using X-grams. In: Proc. of ICSLP, Philadelphia, PA (1996) 394–397
4. Bonafonte, A., Mariño, J.B.: Using X-Gram For Efficient Speech Recognition. In: Proc. of ICSLP, Sydney, Australia (1998)
5. Riccardi, G., Pieraccini, R., Bocchieri, E.: Stochastic automata for language modelling. Computer Speech and Language **10** (1996) 265–293
6. Deligne, S., Bimbot, F.: Language modeling by variable length sequences: theoretical formulation and evaluation of multigram. In: Proc. of ICASSP (1995) 169–172
7. Deligne, S., Bimbot, F.: Inference of variable-length acoustic units for continuous speech recognition. In: Proc. ICASSP, Munich, Germany (1997) 1731–1734
8. Deligne, S., Sagisaka, Y.: Statistical language modeling with a class-based n-multigram. Computer Speech and Language **14** (2000)
9. Bonafonte, A., et al: Desarrollo de un sistema de diálogo oral en dominios restringidos. In: I Jornadas en Tecnología del Habla, Sevilla (Spain). (2000)
10. Segarra, E., Sanchis, E., García, F., Hurtado, L.: Extracting semantic information through automatic learning techniques. IJPRAI **16** (2002) 301–307
11. García, P., Segarra, E., Vidal, E., Galiano, I.: On the use of the Morphic Generator Grammatical Inference (MGGI) Methodology in automatic speech recognition. IJPRAI **4(4)** (1990)
12. Segarra, E., Hurtado, L.: Construction of Language Models using Morfic Generator Grammatical Inference MGGI Methodology. In: Proc. of Eurospeech, Rhodes, Greece (1997) 2695–2698

13. Prieto, N., Vidal, E.: Learning language models through the ECGI method. Speech Communication (1992)
14. Prieto, N., Sanchis, E., Palmero, L.: Continuous speech understanding based on automatic learning of acoustic and semantic models. In: Proc. of ICSLP (1994) 2175–2178
15. García, P., Vidal, E.: Inference of k-testable languages in the strict sense and application to syntactic pattern recognition. IEEE Trans. on **PAMI-12** (1990) 920–925
16. Fraser, N.M., Gilbert, G.N.: Simulating speech systems. Computer Speech and Languages **5** (1991) 81–99
17. Segarra, E., et al: Achieving full coverage of automatically learnt finite-state language models. In: Proc. of EACL, Budapest (2003) 135–142