Efficient Strategy and Language Modeling in Human-Machine Dialogues

D. Louloudis[1], K. Georgila[1], A. Tsopanoglou[2], N. Fakotakis[1] and G. Kokkinakis[1]

(1) Wire Communications Lab.,
University Of Patras, 265 00 Patras, Greece.
Tel. +30 61 991722 Fax. +30 61 991855
{dluludis, rgeorgil, fakotaki, gkokkin}
@wcl.ee.upatras.gr

(2) Knowledge S.A., LogicDIS Group,
N.E.O. Patron-Athinon 37, 264 41, Patras, Greece.
Tel. +30 61 452820 Fax. +30 61 453819
atsopano@knowledge.gr

ABSTRACT

Each time an Interactive Dialogue System (IDS) is adapted to a new domain, the language modeling and dialogue strategy modules must be modified to fulfil the new requirements. In this paper we present an algorithm for creating Stochastic Finite-State Netwoks (SFSN) for language modeling of dialogue states in an IDS. The resulting SFSNs are evaluated in terms of perplexity and recognition performance. Moreover, we present a method that enables the designer of the dialogue strategy to investigate system performance by employing diagnostic evaluation during the initial phases of a system's development. The recognition success rate taken from the previous language model evaluation combined with the proposed dialogue mathematical modeling, can be used to predict an IDS's behaviour by relating dialogue parameters (e.g. recognition success rate, number of turns, dialogue strategy) with the final system's performance. Thus the effort during global system assessment is reduced since we have diagnostic measures in advance.

Keywords: Interactive Dialogue Systems, language models, Stochastic Finite-State Networks, dialogue strategy, diagnostic evaluation.

1. INTRODUCTION

The first part of this paper contains the description of a method for creating stochastic networks. That is an algorithm introduced in [1] is now extended and evaluated in terms of perplexity and recognition performance. This technique has the advantage of the automatic creation of word/phrase classes during the construction of a Hidden Markov Model (HMM) that is transformed to a SFSN. In most existing systems, clusters are created manually or if automatic techniques are used, the clustering procedure is independent of the construction of the final models. Thus the language models require already formed clusters in order to become more compact and robust. Our algorithm does not require the preexistence of classes but creates them automatically and simultaneously with the construction of the HMM. The states and observations of the HMM correspond to the word/phrase classes and words/phrases respectively. The HMM is transformed to a SFSN where the nodes are the word/phrase classes and the arcs are the state-transition probabilities of the HMM. The observation probabilities of the HMM correspond to the probabilities within the classes (sub-networks) of the SFSN.

The use of stochastic automata to represent statistical language models has been recently proposed [2][3] with the aim to handle accurate language models in a one-step decoding procedure. In [2] a back-off *n*-gram language model is represented through a non-deterministic Stochastic Finite-State Automaton (SFSA), which is called Variable *N*-gram Stochastic Automaton (VNSA). In [3] the use of smoothed *K*-Testable Language in the Strict Sense (*K*-TLSS) regular grammars allowed the creation of a deterministic SFSA. In VNSA, and SFSA based on *K*-TLSS, the history size has a value of up to *N*-1 and *K*-1 respectively. Our algorithm produces a deterministic SFSN. Moreover, it is structured in such a way that allows for longer distance dependencies to be considered, and results in variable history sizes with no specific upper limit. That is the upper limit depends on the number of words/phrases of the sentences used as training data and the way these sentences are associated.

The second part of this work consists of the presentation of a method that enables the designer of the dialogue strategy to employ diagnostic evaluation during the initial phases of a system's development, which reduces the effort for global system assessment. The relationship between these two parts is that the recognition rate derived from the evaluation of the language models in each dialogue state is the input to the strategies used in the second part.

Although much work has been done on mathematical modeling of the dialogue control [4], very few results are available on the aspect of predicting a dialogue system's behavior by simply correlating a-priori knowledge such as recognizer' performance and dialogue strategy with the system's performance. Other authors, are trying to decide on optimum dialogue strategy either purely objectively [5] or taking into account users' perceptions using some form of a cost function and seeking for minimization of this function. Although the above techniques give a method for optimum strategy selection, they are based on information derived from actual dialogues obtained usually via black-box assessment. Furthermore, questionnaires filled out by system users may be used, in order to capture qualitative and subjective system characteristics.

In our work the notion of **SUB-TASK** is introduced, which is the part of the dialogue devoted to a single intermediate level task (e.g. supplying the system with the departure date of a trip). Although the term "turn" is usually defined as a stretch of speech spoken by one party in a dialogue, its use here indicates the set of system-user exchanges necessary for the completion of the sub-task. A sub-task includes one or more dialogue states (e.g. request for the departure time and confirmation). Thus the recognition rate of a sub-task is the weighted mean value of the recognition rates of each dialogue state of the sub-task. The use of weighted coefficients is justified by the fact that dialogue states have different levels of difficulty in recognition.

Regarding finite state IDSs a directed graph determines the dialogue flow and each sub-task is represented by a node. The transition from one node to another depends on the dialogue history, the current user answer and the dialogue strategy. Our approach is based on the assumption that if the designer has the characteristics of each node a-priori, and the graph topology, he can predict the behavior of the system without actually testing it. For IDSs, the global system variables of interest are the probability of success P and the expected number of turns μ. As an example, given a simple system, with n sub-tasks $S_1, S_2, \ldots, S_n$ executed in series with sub-task parameters P_i (probability of successfully finishing the sub-task i) and μ_i (average number of turns for the sub-task i), the probability of dialogue success for the system would be the product of P_i, $P=\prod P_i$ and the expected number of turns the sum of μ_i, $\mu=\sum_i \mu_i$. To analyze the behavior of the system, the designer should know in advance each sub-

task's behavior, given the dialogue control strategy chosen for the sub-task. We will investigate the following cases:

1. Immediate advance. There is no confirmation for the recognized item. Usually it is used when the recognition accuracy is very high.
2. Sub-task repetition until success. This strategy presumes confirmation by the user. It is assumed that if the recognition is correct, the user confirms the result and the dialogue is forwarded to the next sub-task, otherwise the same sub-task is repeated.
3. Sub-task repetition for a maximum of *m* times. The user confirms the recognition result. On success, the dialogue is progressed to the next sub-task. On failure, the sub-task is repeated for a maximum of *m* times.
4. Confidence level examination. The dialogue is forwarded to the next node if the confidence level of the recognition is greater than a prescribed threshold, otherwise the sub-task is repeated. In this case, no confirmation is used.

The structure of this paper is as follows: In section 2 the algorithm for the creation of the language model is described and evaluated. Section 3 explains how the diagnostic evaluation of dialogue strategies is implemented. Finally, in section 4, some conclusions are drawn and possible future work is investigated.

2. LANGUAGE MODEL

Algorithm description

At first a set of sentences is selected to train the initial HMM. These sentences can be derived from simulation experiments, from the system itself, from the application grammar, be manually created or be produced by a combination of these methods. Our algorithm takes the set of sentences for granted, regardless of how they are produced. However, as it will be shown in the tests carried out, the best results are obtained by mixing sentences taken from the use of the system with sentences derived from grammar-based networks. For every new sentence *S* the Viterbi algorithm is activated to check whether this sentence could be extracted by the current HMM. The probability assigned to the sentence *S* is compared with a threshold *T*, which is defined for the HMM.

If the probability assigned to *S* exceeds or is equal to *T* (Case 1), or if a part of *S* fits in an existing HMM path (Case 2), then unknown observations of *S*, that is words/phrases, are able to match existing states, i.e. word/phrase clusters, and become members of them. In this way, the clustering procedure takes place simultaneously with the construction of the HMM. Taking into consideration the modified clusters and sentences that are subsets of *S*, the HMM is updated. That is, the observation probabilities within the existing states (clusters) are reestimated and new states may be added (for the parts of sentences that cannot match existing states). Subset sentences of *S* are the sentences, all the words/phrases of which are contained in sentence *S*. The word/phrase order may be considered or not be taken into account. b. If the probability assigned to *S* is smaller than *T* and no parts of *S* fit in existing HMM paths, the already existing states (clusters) are not updated, but new ones are created to incorporate the subset sentences of *S* into the HMM. In either case (a) or (b), a new threshold for the updated HMM is estimated, which replaces *T*. Then a new sentence is selected, the probability of which is going to be compared with the updated threshold. The procedure iterates until no more sentences are available. Throughout the iterations, phrases may be formed (by using simple rules or by taking into consideration sophisticated syntactic and semantic restrictions), during each sentence's processing, that is before Viterbi is applied. After the final HMM has been constructed, it is transformed to a SFSN.

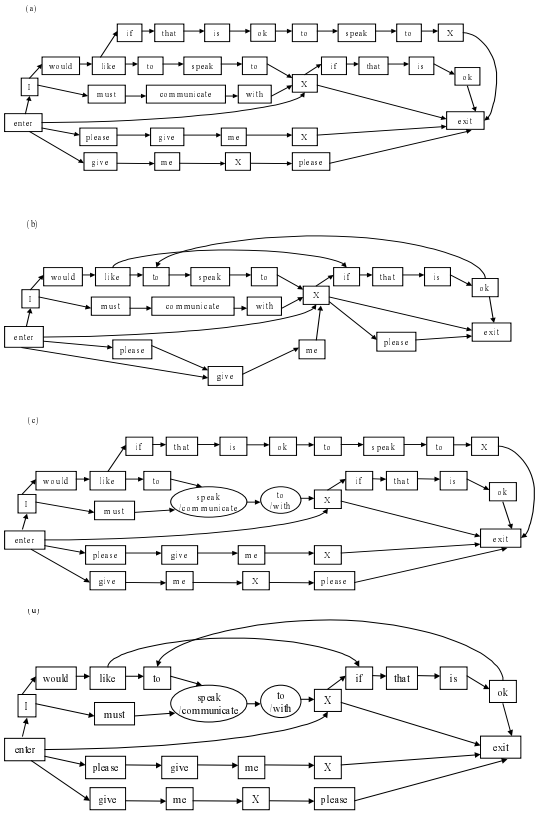

Figure 1. (a) Grammar-based network, (b) bigram, (c) hybrid network (WPO), and (d) hybrid network (NWPO).

The type of the HMM we use is discrete. Two types of transition probabilities are considered: transitions with equal probability from one state to another and probabilities derived from the number of times a word/phrase class appears after another. Thus if a word/phrase class *u* is followed by *n* word/phrase classes in the training data, then for the case of equal probabilities, the probability that a word/phrase class *w* occurs after the word/phrase *u* would be $P(w \mid u) = 1 / n$ **(1)**. On the other hand, if the number of times class *w* follows *u* is considered, then $P(w \mid u) = N(u, w) / N(u)$ **(2)** where $N(u, w)$ is the number of occurrences of class *w* after class *u* and $N(u)$ the number of occurrences of class *u*. In the same way, observations, i.e. words/phrases, can have equal probabilities within a state (class), or the probabilities are formed according to the frequency of occurrence of the words/phrases. In the former case if a word/phrase *w* belongs to a class $C(w)$, which has *n* members, then the probability of this word/phrase in the class is $P(w \mid C(w)) = 1 / n$ **(3)**. In the latter case $P(w \mid C(w)) = N(w) / N(C(w))$ **(4)** where $N(w)$ is the number of occurrences of word/phrase *w* and $N(C(w))$ the number of occurrences of class $C(w)$, that is the sum of occurrences of the words, which belong to class $C(w)$.

In case where the word/phrase order is retained (WPO–Word/Phrase Order), if *S* is the sequence of words/phrases $v_1, v_2, v_3, \ldots, v_n$ then a subset sentence of *S* would have the form $v_i, v_j, v_k, \ldots, v_m$ $1 \le i < j < k < \ldots < m \le n$. If the word/phrase order does not pose a constraint (NWPO–No Word/Phrase Order), the subset sentences of *S* are $v_i, v_j, v_k, \ldots, v_m$ $1 \le i, j, k, m \le n$. Every time Viterbi is activated, when we use the longest of the training sentences as the new sentence *S* and have the NWPO case, then more sentences become subset sentences directly, and the computation time is reduced. In Figure 1, a grammar-based network, the corresponding bigram and the two hybrid networks derived from our method are depicted. In Figure 1c (WPO) in

most paths the complete history is retained. However, in Figure 1d (NWPO) some part of history is lost due to the existence of loops. In general WPO allows for greater history size than NWPO.

In Case 2 where only a part and not the whole sentence matches an existing path straightforwardly or by shift, the candidate matches between new observations and existing states, may be accepted according to some criteria such as frequency of occurrence, position, number of words, word order, if a word/phrase sequence appears more than once in the path etc. If these criteria are very strict, then it is more likely that the candidate matches will be rejected, which will result in a model where grammatical structure supersedes stochastic features. On the other hand, loose criteria will allow matches that do not conform to grammatical rules and may also cause insertions of loops. That is the resulting network will come closer to the *n*-gram structure. Some additional criteria could also be added so that the clusters are correctly formed e.g. words are divided in functional and non-functional words or their Part-Of-Speech (POS) could be considered. Thus a functional word cannot be clustered with a non-functional one and words that do not have the same POS cannot belong to the same class. In the same way phrases of different types may not be allowed to be in the same cluster even if all the other criteria are met. These additional constraints (apart from POS) have been taken into account in tests and have resulted in improved performance.

Evaluation

In order to test our algorithm we used data from 3 different IDSs: ACCeSS (EU project LE-1 1802, a system for the automation of call center services of a car insurance company), IDAS (EU project LE-48315, an Interactive telephone-based Directory Assistance Services system), and a call-routing IDS developed by Knowledge S.A. We used data from 49 dialogue states (38 of ACCeSS, 7 of IDAS and 4 of the call-router).

Three sets of experiments were carried out. In the first one (Test 1) we considered as training data for our algorithm, the sentences derived from the grammars of the 3 applications. This aimed at comparing grammar-based networks with our hybrid models under the same conditions that is with exactly the same training data. The appropriate grammar was loaded according to the IDS and the dialogue state. We carried out experiments with word/phrase classes for both WPO and NWPO. Two types of probability estimations were considered. In the former case, which we call T1, equations (1) and (3) were used to compute the transition and within class probabilities respectively. In the latter case (T2), we applied equations (2) and (4). Phrases were formed without using sophisticated syntactic or semantic rules but by considering words with very strong correlation (e.g. *I would like to*, etc.). When we extracted the phrases for our training set, we modified the grammar networks to take the phrases into account so that we have phrase-based grammar networks too.

The precision and recall parameters are valid metrics for evaluating the performance of our algorithm regarding the clusters formed. We define as *C* the number of correct clusters formed by our method, *T* the total number of clusters, and *TC* the total number of correct clusters, which can be derived from the training data. Then: Precision = C / T and Recall = C / TC.

It is very crucial that the precision is high so that no ill-formed clusters are created, since this would result in associating irrelevant words/phrases and in the end in increasing perplexity. Thus very strong thresholds are set to ensure that only correct clusters are created. In Table 1, the precision and recall values are depicted. Computing the average is not an accurate but an indicative metric in our case since the 49 networks are not equivalent in structure. Sometimes a T1 network can have different precision and recall from the corresponding T2 network. We have observed that often the T2 networks have higher precision but lower recall than the T1 ones. That is they are more reliable in forming correct clusters but on the other hand as their probabilities are based on the exact number of occurrences, sometimes they fail to match words/phrases, which are strongly correlated but that do not have equivalent occurrences. In the same way in the WPO case the precision is higher since the word/phrase order is taken into consideration in forming clusters. However, networks derived from the NWPO case tend to have higher recall values. Moreover, phrase-based (P) networks generally outperform word-based (W) ones.

	Test 1		**Test 2**		**Test 3**	
	WPO	NWPO	WPO	NWPO	WPO	NWPO
	Precision					
W-T1	0.97	0.96	0.93	0.93	0.96	0.96
W-T2	0.98	0.97	0.94	0.93	0.97	0.96
P-T1	0.97	0.97	0.94	0.94	0.97	0.95
P-T2	0.98	0.97	0.95	0.95	0.97	0.96
	Recall					
W-T1	0.77	0.78	0.74	0.75	0.76	0.76
W-T2	0.77	0.77	0.74	0.74	0.75	0.75
P-T1	0.77	0.79	0.75	0.75	0.76	0.76
P-T2	0.76	0.78	0.73	0.74	0.75	0.76

Table 1. Precision and recall values.

	Perplexity Increase vs. grammars (%)		**Perplexity Reduction vs. bigrams (%)**	
	WPO	NWPO	WPO	NWPO
W-T1	7.34	8.57	17.11	15.85
W-T2	7.22	8.25	17.19	15.96
P-T1	6.89	8.18	17.36	16.13
P-T2	6.81	7.92	17.54	16.25

Table 2. The perplexity (%) in hybrid networks compared to grammar-based ones and bigrams (Test 1).

Table 2 depicts the average increase in perplexity of our hybrid networks compared to the grammar-based ones and the average reduction compared to bigrams. Perplexity in the grammar-based and in our hybrid networks is estimated by following paths backwards and multiplying the inverse branching factor at each step. Perplexity in grammar-based networks is smaller than in hybrid ones. However, a very small perplexity indicates that the language model is not robust against utterances not included in the training data. According to the experiments, T2 networks have lower perplexity than T1 ones. Networks of WPO case have lower perplexity values than the ones of NWPO case and phrase-based networks have generally lower perplexity than word-based ones.

In the second set of experiments (Test 2), we considered as training sentences data derived from the use of the system itself, to compare our models with bigrams. The reason is that the power of bigrams arises from the fact that they give reliable estimations when trained with real data. Thus it would not be appropriate to compare our models with bigrams using sentences derived only from grammars. Data is split into two parts (80% for training, 20% for testing) so that perplexity is computed by using a test set different from the training set. Since the test data may contain events not seen in the training sentences, smoothing techniques should be applied. We used the Witten-Bell discounting scheme. If we have a node *A* connected to a node *B*, then *n* is the number of occurrences of links "*A* *"

and *t* is the number of the distinct links "*A* *" that exist. We consider only the occurrences of the specific node and not of the word or phrase associated with it, because the word/phrase may appear in more than one nodes. For events that have been seen $P(w \mid h) = c / (n + t)$ (where *w* is a word, *h* is the history and *c* is the number of occurrences of *w* in the context *h*). For unseen events $P(w \mid h) = t / (n + t)$. Table 3 shows the average perplexity reduction in our hybrid networks compared to bigrams. The perplexity reduction vs. bigrams is a little higher in Test 1 compared to Test 2. A reasonable explanation would be that the performance of bigrams is better in Test 2 since the training sentences are real data derived from the use of the system itself and not by a grammar. The average precision and recall values for the clusters formed are shown in Table 1. There is a reduction compared to the values of Test 1 caused by the spontaneous nature of the training data in Test 2, which complicates clustering.

	Test 2		**Test 3**	
	WPO	NWPO	WPO	NWPO
W-T1	15.28	13.39	15.71	14.20
W-T2	15.42	13.63	15.85	14.44
P-T1	15.55	14.18	16.02	14.57
P-T2	15.69	14.22	16.15	14.91

Table 3. The average perplexity reduction (%) in hybrid networks compared to bigrams (Tests 2 and 3).

In the third experiment (Test 3) we considered as training sentences data derived from grammars mixed with sentences derived from the use of the system. Table 3 shows the perplexity reduction. Again smoothing was applied. Table 1 depicts the average precision and recall values for the clusters formed. There is a reduction compared to the values of Test 1 but an increase compared to Test 2 since sentences derived from grammars are included in the training data.

	Test 1		**Test 2**		**Test 3**	
W-G	78.0					
P-G	78.2					
	WPO	NWPO	WPO	NWPO	WPO	NWPO
W-T1	80.8	81.4	81.0	81.2	81.6	82.0
W-T2	82.0	82.2	81.6	82.0	82.4	83.0
P-T1	82.4	83.0	82.4	82.6	83.2	83.8
P-T2	82.8	83.2	82.6	82.8	84.0	84.4
W 2g	77.6		78.4		78.8	
P 2g	78.0		78.6		79.0	

Table 4. Keyword recognition accuracy (%).

In order to investigate how the networks produced by our algorithm affect recognition performance, tests were carried out with data from the call-routing dialogue system. We used 500 recordings spoken by real users, corresponding to the system prompt "*Who would you like to speak with?*". In Table 4 we can see the keyword accuracy for grammar-based (G) networks, hybrid ones and bigrams (2g). The keyword accuracy is the percentage of the sentences where the keyword (*name*) was recognized correctly. The hybrid networks give the best recognition rates due to the fact that they retain the predictability of the grammar-based networks and at the same time they are more robust for spontaneous speech. The columns correspond to the methods of building the models and the training data. This of course does not apply to grammar-based networks and that is why they have the same accuracy in all tests. If the best percentages of grammar-based networks, hybrid ones and bigrams are considered, the gain in recognition performance is 6.2% compared to grammar-based networks and 5.4% compared to bigrams.

3. DIALOGUE STRATEGY

Sub-task Strategy Analysis

For the following paragraphs it is implied that the recognizer's behavior is modeled by a random variable *R*, describing the recognition rate *r*, defined in the interval [0,1]. The probability density function (from now on pdf) for this random variable is $f_r(r)$ with cumulative distribution function $F_r(r)$ (from now on cdf). The average recognition rate of the recognizer is μ_r with standard deviation σ_r. In addition the random variable *Q*, describing the error rate is defined as $Q=1-R$. The cdf of *Q* is

$$F_q(q) = P(Q \le q) = P(1-R \le q) = P(R \ge 1-q) = 1 - F_r(1-q) = 1 - F_r(r)$$

It is proved that the pdf of the error rate *Q* is the same with the pdf of the success rate *R* since

$$f_q(q) = \frac{d}{dq} F_q(q) = \frac{d}{dq}[1 - F_r(1-q)] = -\frac{d}{dq} F_r(1-q) = \frac{d}{d(1-q)} F_r(1-q) = f_r(1-q) = f_r(r)$$

It also can be proved that the sum of the average recognition and error rates is one, and that the variance of the error rate is the same as the variance of the recognition rate. For the analysis of the strategies, the definition of the following events is necessary:

$$\mathrm{E} = \{\text{success}\} \qquad \overline{\mathrm{E}} = \{\text{failure}\}$$

$$\mathrm{E_i} = \{\text{success during the i}^{\text{th}}\ \text{turn}\}, \quad \mathrm{i} \in \mathrm{S} = \{1,2,\ldots\}$$

$$\mathrm{E} \cup \overline{\mathrm{E}} = \wp \quad \text{the space of the expirement}$$

$$\mathrm{E} \cap \overline{\mathrm{E}} = \phi \quad \text{the null set and}$$

$$\mathrm{E_i} \cap \mathrm{E_j} = \phi, \quad \mathrm{i} \neq \mathrm{j} \quad \mathrm{E_i} \cap \overline{\mathrm{E}} = \phi \quad \left(\bigcup_{\mathrm{i}} \mathrm{E_i}\right) = \mathrm{E}$$

where *S* is the space of the variable *i* which represents the allowed number of turns and depends every time the experiment. In addition, it is assumed that when there are confirmations, the items are recognized correctly.

Strategy 1 (immediate Advance): In this case $S=\{1\}$. The probability of success and failure, given the recognition rate is $P(E/R=r)=r$ and $P(\overline{E}/R=r)=1-r$

Using the total probability theorem, the probabilities of the events "success" and "failure" are $P(E)=\mu_r$ and $P(\overline{E})=1-\mu_r$

Strategy 2 (sub-task repetition until success): The probability of success on the first turn is *r*, on the second $(1-r)r$ on the third $(1-r)^2 r$ and so on. So the random variable *N* representing the number of turns necessary to observe the event *E*, given the recognition rate *r*, follows a geometric distribution. The space *S* for this strategy is $S=\{1,2,3,\ldots\}$.

The probability for the event E_i, given the recognition rate *R* is

$$P(E_i / R = r) = r(1-r)^{i-1}$$

Using the previous equation, the fact that $P(E/E_n)=1$ and the law of total probability, it is proved that

$$P(E/R) = 1, \quad P(\overline{E}/R) = 0 \quad r \neq 0$$

$$P(E) = 1, \quad P(\overline{E}) = 0$$

that is, eventually there will be a success, independently from the recognition rate *r*, for *r* different from zero.

Let *N* be a random variable representing the turn number at which the event *E* is observed. Then

$$f_{n/r}(n/r) = P(E/R=r) = P(N=n/R=r) = r(1-r)^{n-1} \qquad \textbf{(5)}$$

Since (5) is geometrically distributed, the expected number of turns and the variance for the sub-task given the recognition rate is

$$\mu_{N/R} = \frac{1}{r} \quad and \quad \sigma^2_{N/R} = \frac{1-r}{r^2}$$

Using the total probability theorem it is proved that

$$\mu_N = \int_0^1 \frac{1}{r} f_r(r)\,dr \quad and$$

$$\sigma^2_N = \int_0^1 \frac{2-r}{r^2} f_r(r)\,dr - \left(\int_0^1 \frac{1}{r} f_r(r)\,dr\right)^2$$

Strategy 3 (sub-task repetition for a maximum of *m* times): Following the same reasoning as in strategy 2, and

taking into account that for this strategy $S=\{1,2,\ldots,m\}$, it is proved that

$$P(E/R)=1-(1-r)^m,\quad P(\overline{E}/R)=(1-r)^m$$
$$P(E)=1-m_q^m,\quad P(\overline{E})=m_q^m \qquad (6)$$

where the last quantity is the m^{th} moment of the error rate Q.

If there is a constrain for the sub-task, that the probability of success is greater than a value P_0, then solving (6) for m gives the minimum value of m for which the condition is satisfied:

$$m\geq\frac{\log(1-P_0)}{\log(1-r)},\quad 0<r<1,\quad 0\leq P_0<1$$

Let N be a random variable describing the number of turns necessary for success or failure. Then the probability mass function (pmf), given the recognition rate r, would be

$$f_{n/r}(n/r)=\begin{cases}r(1-r)^{n-1}, & 1\leq n<m\\ (1-r)^{m-1}, & n=m\end{cases}$$

So the expected number of turns and the variance, given the recognition rate r and the threshold m, after some calculations become

$$\mu_{N/R}=\frac{1-(1-r)^m}{r}=\frac{P(E/R)}{r},\quad 0<r\leq 1,\quad m=1,2,\cdots$$
$$\sigma^2_{N/R}=\frac{1-r+(1-2m)r(1-r)^m-(1-r)^{2m}}{r^2}$$

If the recognition rate is not given, but instead its distribution is known, the following formulas can be proved

$$\mu_N=\int_0^1\frac{1-(1-r)^m}{r}f_r(r)\,dr$$
$$\sigma^2_N=\int_0^1\frac{2-r+r(1-r)^m-2rm(1-r)^m-2(1-r)^m}{r^2}f_r(r)\,dr-\left(\int_0^1\frac{1-(1-r)^m}{r}f_r(r)\,dr\right)^2$$

Strategy 4: (confidence level examination): The space S for this strategy is $S=\{1,2,3,\ldots\}$. Suppose that the recognizer's score is represented by a random variable X, with pdf $f_x(x)$ normalized in $[0,1]$, where $f_x(x)=0,\ x\notin[0,1]$.
In addition, suppose that the random variable representing the recognition rate R is a function of X, i.e. $R=g(x)$ [6]. Then the error rate is given *by* $Q(x)=1-g(x)$, also a random variable and function of X.

The decision regarding success and failure is given by the following rule {if x is greater than or equal to t, the recognition result is accepted, otherwise the result is discarded and the sub-task is repeated}. The threshold t is set by the system designer in advance.

The following events can be defined (given the threshold t): CA_t={Correct Acceptance}, FA_t={False Acceptance}, CR_t={Correct Rejection}, FR_t={False Rejection}. Then the following probabilities are defined as a function of t

$$P(CA_t/X=x)=\begin{cases}R=g(x), & x\geq t\\ 0, & x<t\end{cases}$$
$$P(FA_t/X=x)=\begin{cases}Q=1-g(x), & x\geq t\\ 0, & x<0\end{cases}$$
$$P(CR_t/X=x)=\begin{cases}Q=1-g(x), & x<t\\ 0, & x\geq t\end{cases} \qquad (7)$$
$$P(FR_t/X=x)=\begin{cases}R=g(x), & x<t\\ 0, & x\geq t\end{cases}$$

To find the probabilities of the events CA_t, FA_t, CR_t, FR_t when the confidence level distribution is known, the law of total probability is applied on (7) giving

$$P(CA_t)=\int_t^1 g(x)f_x(x)\,dx=E[g(x)/X\geq t]$$
$$P(FA_t)=\int_t^1(1-g(x))f_x(x)\,dx=1-F_x(t)-P(CA_t)$$
$$P(FR_t)=\int_0^t g(x)f_x(x)\,dx=E[g(x)/X<t]$$
$$P(CR_t)=\int_0^t(1-g(x))f_x(x)\,dx=F_x(t)-P(FR_t)$$

If the following events are defined (given the treshold t): S_t={Success}, F_t={Failure} and R_t={Repeat Sub-task}, then it is clear that $S_t=CA_t$, $F_t=FA_t$ and $R_t=FR_t\cup CR_t$ so the probabilities of the previously defined events are

$$P_s=P_{s,t}=P(S_t)=\int_t^1 g(x)f_x(x)\,dx$$
$$P_f=P_{f,t}=P(F_t)=1-F_x(t)-P(S_t)$$
$$P_r=P_{r,t}=P(R_t)=P(FR_t\cup CR_t)=P(FR_t)+P(CR_t)=F_x(t)$$

Then the probabilities of success and failure during the n^{th} turn are given by $P(E_n)=P_r^{n-1}P_s$ and $P(\overline{E}_n)=P_r^{n-1}P_f$
The probability of successfully finishing the sub-task is

$$P(E)=\sum_{k=1}^{\infty}P(E_k)=P_s\cdot\sum_{k=1}^{\infty}P_r^{k-1}=\frac{P_s}{1-P_r}=\frac{\int_t^1 g(x)f_x(x)\,dx}{1-F_x(t)} \qquad (8)$$

Lets define the random variable N representing the number of turns necessary for a failure or success. Then the probability mass function of N is
$f_n(n)=P(N=n)=P(E_n\cup\overline{E}_n)=P_r^{n-1}(P_s+P_f)=F_x(t)^{n-1}(1-F_x(t))$
which is clearly a geometric distribution. So the expected number of turns until success or failure becomes

$$\mu_N=\sum_{k=1}^{\infty}k\,f_n(k)=\frac{1}{1-F_x(t)} \qquad (9)$$

with variance

$$\sigma^2_N=\frac{F_x(t)}{(1-F_x(t))^2}$$

For all the relations above it is assumed that t is different from one.

Experimental Results

All the experiments described below were conducted on a call-router that was developed by Knowledge S.A., Patras, Greece. This system was installed at the headquarters of LogicDIS Group in Athens on July 2000. With a lexicon of more than 600 words, the system serves more than 300 employees and recognizes more than 100 departments and titles. It can resolve potential conflicts and ambiguities (for example staff that belongs to a department different from the requested one or employees with the same surname etc). The system is operating 8 hours per day, servicing a total of 200-250 calls daily. Although around 3,500 recordings were collected, not all of them were used for the evaluation for two reasons: Firstly, a percentage of them (around 10%) contained irrelevant user responses, mainly from users not familiar with the system. Secondly, since confirmations were used in some strategies, there was a selection of dialogues with 100% success rate in confirmations (although the actual success rate was around 98%) because it was necessary to eliminate the confirmation bias from the results. In all the experiments that are described below, the bootstrap method with 1000 samples was used to produce the recognizer's distribution.

Sub-task repetition for a maximum of *m* times: To test the first strategy, 1599 dialogues were used. The maximum number of turns after which the incoming call was forwarded to a human operator in order to be served was set to two. From the 1599 dialogues there were 124 with a second attempt for recognition. During this second turn, 116 dialogues were terminated successfully and 8 of them were routed to the operator. The average recognition rate was 92.33% with standard deviation 0.62%. Formulas with conditional

probabilities were used again to calculate the theoretical results. Theoretically, the probability of success was 1-(1-0.9233)^2=0.9941 and the expected number of turns 0.9941/0.9233=1.077. Experimentally the success rate was (1599-8)/1599=0.9950 and the average number of turns

$$\frac{(1599-124)\cdot 1+124\cdot 2}{1599}=1.0775$$

Table 5 summarizes the results.

		Theoretical	Experimental
Strategy 1	P(E)	99.41 %	99.50 %
	μ	1.077	1.077
Strategy 2	P(E)	95.21 %	95.48 %
	μ	1.2	1.175

Table 5. Theoretical and experimental results concerning the Call-Router Application.

Confidence level examination: To assess this strategy, 354 dialogues were tested. The recognizer produced a confidence level from 0 to 1000, which was normalized in range [0,1]. The decision threshold for the recognizer was set to 0.5. From the 354 dialogues, 338 terminated successfully, and the rest 16 with errors. In addition, during the 354 dialogues there were 62 requests for repetition of the sub-task, giving a total of 416 turns to be analyzed. The interval [0,1] was divided in 11 class intervals. 1000 bootstrap samples of vector pairs (*Cnf*, *R*) - *Cnf* stands for the confidence score - were created, each vector having 416 pairs of elements (*Cnf*, *R*) selected randomly with replacement from the original observations. For each class interval, the frequency of occurrence was counted, and averaged over the 1000 bootstrap samples. In addition for each class interval the recognition rate was calculated using the formula *R=C/(C+F)*, where *C* stands for correct recognition and *F* for failure. This value was also averaged over the 1000 bootstrap samples. The results are displayed in table 6.

Bin	Class Interval	Fi	pmfi	cdfi	Ri
1	0-0.099	5,00	01,17	01,17	59,33
2	0.1-0.199	14,95	03,49	04,66	39,28
3	0.2-0.299	12,01	02,81	07,47	41,95
4	0.3-0.399	16,14	03,77	11,24	56,45
5	0.4-0.499	23,05	05,39	16,63	56,16
6	0.5-0.599	29,96	07,00	23,63	83,30
7	0.6-0.699	35,08	08,20	31,83	88,73
8	0.7-0.799	40,00	09,35	41,18	89,72
9	0.8-0.899	54,90	12,83	54,01	98,10
10	0.9-0.999	45,96	10,72	64,73	95,63
11	1	150,96	35,27	100	99,35

Table 6. Experimental distribution of the confidence score.

Since experimental data were used, the formulas (8), (9) were modified to

$$\mu_N=\frac{1}{1-cdf_{0.499}},\quad P(E)=\frac{\sum_{\text{class interval}\geq 0.5}^{1} R_i\cdot pmf_i}{1-cdf_{0.499}} \qquad \textbf{(10)}$$

Substituting the values from Table 6 to equations (10) gives the theoretical results. Experimentally, the success rate is calculated to be 338/354=0.9548 and the average number of turns is 416/354=1.175. Table 5 summarizes the results.

4. CONCLUSIONS AND FUTURE WORK

In this paper we presented an algorithm for creating SFSNs for language modeling of dialogue states. Moreover, we described a method that enables the designer of the dialogue strategy to investigate system performance by employing diagnostic evaluation during the initial phases of a system's development. The recognition success rate taken from the previous language model evaluation combined with the proposed dialogue mathematical modeling, can be used to predict an IDS's behaviour by relating dialogue parameters with the final system's performance. Thus the effort during global system assessment is reduced.

Regarding the algorithm for the development of the language model, the tests carried out, proved the efficiency of our algorithm concerning precision and recall values for the clusters formed. In addition, they showed a considerable reduction in perplexity compared to bigrams, which if it is combined with the gain in recognition performance against both grammar-based networks and bigrams, makes our method appropriate for building efficient language models for IDSs. Future work will focus on modifying our algorithm so that it deals with higher order *n*-grams, which will result in lower perplexity values.

From the experimental results in the dialogue control strategies it is obvious that the theoretical models developed are very sound. The two most important dialogue variables, namely the task completion rate and the number of turns can be easily and accurately derived before installing the actual system. Some possible and interesting uses of the models are: A-priory analysis regarding a system's performance and behavior, identification of a system's problematic areas (where deviations from theoretical with experimental results are quite large) and system modeling for simulation runs.

Further work is necessary in order to enhance the efficiency of the models. Firstly, although user behavior can be concisely modeled in the random variable *R*, it is sometimes necessary to capture more details regarding system performance related to human behavior. From the conducted experiments it was obvious that in many cases the user's initial response was inappropriate, which is attributed to two major factors: The first one is that the user was unfamiliar with the system and his/her response was inappropriate or s/he thought s/he was discussing with a human operator and did not pay attention to the system's prompts. This problem can be partially solved with better system prompts. The other problem is related to the timing of the response. Too late or early response resulted in incorrect recognition. Barge–in capability or system audio signals can partially solve this problem. Secondly, it was observed that two types of errors were common which were related to the confirmations and speech synthesis capabilities of the system. The first type had the following characteristics: The recognition was correct, but when the user was asked to confirm it, s/he did not understand the synthesizer's output and a negative response was given. Consequently the turn was incorrectly repeated. The inverse procedure creates the second and more serious type of error i.e. the recognition is incorrect, but the user thinks that the system has made a correct recognition and gives a positive answer. This action does not terminate the dialogue, but the system fails to deliver correct information to the user. Both the above problems were created when an inferior synthesizer was employed in order to investigate the consequences of its operation.

5. REFERENCES

[1] K. Georgila, N. Fakotakis, G. Kokkinakis, "Building stochastic language model networks based on simultaneous word/phrase clustering", ICSLP 2000, Vol. 1, pp. 122-125.

[2] G. Riccardi, R. Pieraccini, E. Bocchieri, "Stochastic automata for language modeling", Computer Speech and Language, Vol. 10, pp. 265-293, 1996.

[3] G. Bordel, A. Varona, M.I. Torres, "K-TLSS(S) language models for speech recognition", ICASSP'97, Vol. 2, pp. 819-822.

[4] Y. Niimi, T. Nishimoto, "Mathematical Analysis of Dialogue Control Strategies", Eurospeech'99, pp. 1403-1406.

[5] E. Levin, R. Pieraccini., W. Eckert, "Using Markov decision process for learning dialogue strategies", ICASSP'98, Vol. 1, pp. 201-204.

[6] B. Rueber, "Obtaining confidence measures from sentence probabilities", Eurospeech'97, pp. 739-742.