FAST INCREMENTAL TECHNIQUES FOR LEARNING PRODUCTION RULE PROBABILITIES IN RADAR ELECTRONIC SUPPORT

Guillaume Latombe[1] *and Eric Granger*

Laboratoire d'imagerie, de vision
et d'intelligence artificielle
Dépt. de génie de la production automatisée
École de technologie supérieure
1100, rue Notre-Dame Ouest
Montreal, QC, H3C 1K3, Canada

Fred A. Dilkes

Defence R&D Canada – Ottawa
Dept. of National Defence
3701 Carling Avenue
Ottawa, ON, K1A 0Z4, Canada

ABSTRACT

Although Stochastic Context-Free Grammars appear promising for recognition of radar emitters, and for estimation of their respective level of threat in Radar Electronic Support systems, well-known techniques for learning their production rule probabilities are computationally demanding. In this paper, three fast incremental alternatives, called Graphical EM (gEM), Tree Scanning (TS), and HOLA, are compared from several perspectives – perplexity, generalization error, time and space complexity, and convergence time. Estimation of the execution time and storage requirements allows for the assessment of complexity, while computer simulation using a radar pulse data set allows to asses the other performance measures. Results indicate that gEM and TS may provide a greater level of accuracy than HOLA, and that computational complexity may be orders of magnitude lower with HOLA. Furthermore, HOLA is an on-line technique that allows for incremental learning of probabilities to reflect changes in operational environments.

1. INTRODUCTION

Radar Electronic Support (ES) involves the passive search for, interception, location, analysis and identification of radiated electromagnetic energy for military purposes. ES thereby provides valuable information for real-time situation awareness, for threat detection, for threat avoidance, and for timely deployment of countermeasures [2] [10]. Two critical functions of radar ESM are the recognition of radar emitters associated with intercepted pulse trains, and the estimation of the instantaneous level of threat posed by these radars. The recent proliferation and complexity of electromagnetic signals encountered in modern environments is greatly complicating these functions.

In conventional ES systems, radar signals are typically recognized using temporal periodicities within the pulse train in conjunction with histograms of the pulses in some parametric space, *e.g.*, carrier frequency, pulse repetition frequency, and pulse width. With the advent of automatic electronic switching designed to optimize radar performance, modern radars, and especially multi-function radars (MFR), are usually far too complex to be simply recognized in this way. MFR will continuously and autonomously change their transmitted signals in response to various events in their dynamically-changing environment. In order to exploit the dynamic nature of many modern radar systems, advanced signal processing algorithms based on Stochastic Context Free Grammars (SCFGs) have been proposed for modeling the behavior of radar systems [9] [8]. Such models can allow tracking the dynamic behaviour of radar emitter patterns, which can be exploited for recognition of radar emitters, and for estimation their respective level of threat.

Given prior knowledge of a radar's behavior, and a set of training sequences collected in the field, one challenge to the practical application of SCFGs is the task of learning probability distributions associated with the production rules. The most popular technique for learning the production rule probabilities is the Inside-Outside (IO) [1] [5]. Unfortunately, application of this algorithm to real-world tasks is limited due to the time and memory complexity per iteration, and to the large number of iterations needed to converge. For each iteration, IO has a time complexity which is cubic with the length of sequences in the training set, and cubic with the number of non terminal symbols in the grammar. Moreover, these techniques cannot incrementally learn new information that may emerge as the operational environment evolves.

In this paper, three fast incremental alternatives to the IO technique – Graphical EM (gEM) [7], Tree Scanning (TS), and HOLA [6] – are compared. Unlike IO, these algorithms involve pre-computing data structures, such as Earley charts, histograms, support graphs, etc., and can thereby lead to a significant reduction in time complexity per iteration. In addition, these algorithms are suitable for learning new information incrementally, without re-training from the start using all training data. Given the need for a learning procedure that offers both accurate results and computational efficiency, the performance of these techniques is examined from several perspectives – perplexity, estimation error, convergence time, time complexity, and space complexity. The data set used in our simulations describes electromagnetic pulses transmitted from a MFR system. The outcome of numerous computer simulations are combined to yield an average performance measure.

The rest of this paper is structured into four sections as follows. The next section provides some background information on grammatical modeling in the context of radar ES applications. In Section 3, the main features of the Graphical EM, Tree Scanning, and HOLA techniques are outlined. Then, the methodology used to compare these techniques, namely, the experimental protocol, data sets, performance measures, is described in Section 4. Finally, the results of computer simulations and complexity estimates are presented and discussed in Section 5.

2. GRAMMATICAL MODELING IN RADAR ELECTRONIC SUPPORT

In radar ES applications, pulsed radar signals are generated by a MFR in reaction to its current operating environment. For instance, when a radar detects or abandons targets it switches among its search, acquisition and tracking functions. The algorithm controlling the function of a MFR is designed according to stochastic automata principles, and the state transitions within the automata are driven by the

[1]**Corresponding author**. Tel.: 1-514-396-8800 #7687; fax: 1-514-396-8595. E-mail address: latombe@livia.etsmtl.ca.

1-4244-0469-X/06/$20.00

stochastic behavior of the targets [8]. The signals generated by a MFR may be viewed with two levels of data organization – the *pulse level* and the *word level*. Radar words can be defined as static or dynamically-varying groups of pulses that a MFR emits in different states. A sequence of several words may form a *phrase*, which corresponds to a state of the radar. The number of words per phrase, their structure, etc., varies according to the MFR.

A deterministic formal language ζ corresponds the set of all possible sequences over the symbols of a vocabulary. Grammatical modeling of a radar system's behavior is achieved if one assumes that symbols of a vocabulary correspond to words of a specific MFR, and that a language represents all possible combination of sequences that a radar could ever emit, from power-up to shutdown. Then, one can create a finite set of production rules to describe a particular language associated with a complex radar system [9] [8].

At a word level, most MFR systems of interest have a natural and compact description in terms of a type of grammar called Context-Free Grammars (CFG) [4]. A CFG G is a mathematical construction represented by the quadruplet $G = \{V, N, R, \epsilon\}$. It consists of a set of terminals symbols (*i.e.*, vocabulary) V, a set of nonterminals symbols N, a set of context-free production rules R, and a start symbol $\epsilon \in N$. A language generated by a grammar is a set of sequences of terminals that can be derived from ϵ by applying the R. A context-free production rule has the form $A \rightarrow \Gamma$. The left-hand-side must be a single non-terminal $A \in N$ and the right-hand-side may consist of any sequence of terminals and nonterminals. Therefore, a CFG allows to model long term dependencies established between the different words of a MFR sequence.

Given the behavior of MFRs and the imperfections of signals observed on a battlefield, it is not possible to design a robust deterministic CFG to model the behavior of a radar system. To allows for a robust modeling of the signal degradations, noise and uncertainties, an element of stochasticity is introduced into the definition of grammars by assigning probability distributions to the production rules. In Stochastic Context-Free Grammars (SCFGs) [4] every production for a non-terminal A has an associated probability value such that a probability distribution exists over the set of productions for A. Therefore, a SCFG G_s is defined as a pair (G, π) where G is a CFG and $\pi = (\pi_{A_1}, \pi_{A_2}, ..., \pi_{A_r})$ is a vector whose component π_{A_i} represents the distribution of probabilities of a non-terminal A_i being derived in a combination of symbols λ. Assuming that $P(\lambda|A_i)$ is the probability of A_i producing λ, then $\pi_{A_i} = (P(\lambda|A_i), P(\mu|A_i), ..., P(\sigma|A_i))$, where $0 \leq P(\lambda|A_i) \leq 1$ for λ, and $\sum_\lambda P(\lambda|A_i) = 1$.

3. A SURVEY OF FAST TECHNIQUES FOR LEARNING PRODUCTION RULE PROBABILITIES

The problem of learning production rule probabilities of a SCFG from a finite set of training sequences Ω can be formulated as an optimization problem. Given a SCFG G_s, and any set Ω of sequences drawn from $\zeta(G)$ (allowing for repetitions), one approach for estimating the probabilities of the grammar consists in maximizing the likelihood of Ω, $P(\Omega, \Delta_\Omega|G_s) = \prod_{x\in\Omega} P(x, \Delta_x|G_s)$, where x is a sequence of Ω, and Δ_Ω represents the set of the derivation trees Δ_x considered to form x [1].

The Expectation-Minimization (EM) algorithm called Inside-Outside (IO) [1] [5] is a well-known technique for learning the production rule probabilities. IO re-estimates probabilities of rules in an iterative manner such that the likelihood of Ω is maximized. However, the IO algorithm has a time complexity of $O(M_{nt}^3 L^3)$ per iteration and a space complexity of $O(M_{nt} L^2)$, where M_{nt} is the number of non terminals in a grammar and L the length of an input sequence. Application to real-world problems is therefore limited. Moreover, it is not possible to update production rule probabilities incrementally in order to reflect changes in the environment.

To accelerate IO, some authors have proposed alternate EM algorithms that incorporate chart parsing within a pre-processing phase. Re-estimation of probabilities can be significantly faster since the blind combination of rules, where any non terminal symbol could produce any combination of non terminals, is avoided. Among these authors, Sato and Kameya [7] have recently introduced an algorithm called the *graphical EM* (gEM) that completely separates the EM learning process from the parsing of Ω. During pre-processing, this algorithm creates a set of ordered support graphs from the chart of a CYK or Earley parser, to represent only the derivations that may possibly lead to a sequence. When probabilities are re-estimated, gEM only passes by the transitions described in support graphs.

Along the same lines as gEM, pre-processing with the *Tree Scanning* (TS) algorithm consists in creating the chart for each training sequence using a CYK or Earley parser. This chart is then used to extract all the possible derivation trees producing a sequence. Probability re-estimation is performed by computing the total likelihood of the sequence (*i.e.,* multiplying of the associated probabilities), and by extracting frequencies (*i.e.,* counting the production rules). This algorithm applies to cases (such as ES applications) in which grammars are not very ambiguous. The authors could not find it in the literature. Note that both gEM and TS give the same results as IO.

Oates and Heeringa [6] present a heuristic on-line algorithm called *HOLA* based on summary statistics. During pre-processing, HOLA exploits a chart parser to computes summary statistics – the distributions over the rules found after parsing Ω, and after parsing a set of sequences produced by the grammar. Then, during the iterative process, HOLA re-estimates probabilities of rules in an iterative manner, using a gradient descent approach, such the relative entropy between these two distributions is maximized.

gEM, TS and HOLA each involve pre-computing data structures, such as charts, histograms, support graphs, etc., and can thereby lead to a significant reduction in time complexity per iteration, at the expense of space complexity. In addition, these algorithms are suitable of learning new information incrementally. HOLA is fully incremental since a new training set Ω' can refine probabilities without retraining the start using $\Omega + \Omega'$. gEM and TS are however semi-incremental. Although new set Ω' can be incrementally incorporated into their data structures during pre-precessing, the iterative process must re-estimates new probabilities from scratch, using $\Omega + \Omega'$.

4. EXPERIMENTAL METHODOLOGY

In order to characterize the performance of the three techniques presented in Section 3, a fictitious MFR system called Mercury was considered. The Mercury MFR can be in one of five functional states – Search (S), Acquisition (Acq), Non-Adaptive Track (Na), Range Resolution (Rr), and Track Maintenance (Tm). If the radar is in Search, it can remain there, or move to the Acquisition once a target is detected. The target acquisition cycle involves transitions from Acquisition, to Non-Adaptive Track, to Range Resolutions, and finally to Track Maintenance. The radar can remain in any of these states for an unspecified length of time. Finally, the target acquisition or track can be abandoned at any point, at which time the radar returns to Search. A word-level CFG was designed for this MFR from its functional description, according to the Chomsky Normal Form. Mercury represents a low ambiguous grammar.

The transition probabilities needed from a SCFG were learned according to gEM, TS and HOLA techniques through computer simulation with a synthetic radar data set. A data set was generated for Mercury. It consisted of 400 sequences, where each sequence corresponds to the set of words that would be produced by this MFR during one target detection, while switching through all its internal states, starting and ending in Search mode. Mercury sequences had a size that ranged from 108 to 1540 words, with an average of 588 words. The duration of each state is set using gaussian distributions to approximate reality.

Prior to simulation trials, this data set was partitioned into four equal parts a training subset `Train`, a validation subset `Val`, a test subset `Test`, and an *ideal* test subset `TI`. Inside an ES system, the

MFRs words would first be detected from within the stream of intercepted radar pulses, and then sequences of words would be recognized using the SCFGs. Prior to sequence recognition, errors occur if (1) a word is incorrectly detected, (2) a word is missing or not detected, or (3) multiple words are detected simultaneously. If only the best-matching detection is retained for sequence recognition, these 3 cases are equivalent to incorrectly detected words (case 1). Accordingly, two noisy versions of TI, $\texttt{TN}_1$ and $\texttt{TN}_2$, were generated. Each noisy test set consists of the 100 sequences from TI. Then, to select incorrectly detected words to be modified, a Poisson process was applied with a mean of 50 words for $\texttt{TN}_1$ and 10 words for $\texttt{TN}_2$. Finally, a uniform law was used to determine the replacement words.

During each trial, the prior probabilities of a SCFG were initialized, and then training was performed over several iterations of Train, until the difference between the negative log likelihoods (gEM and TS) or relative entropy of words (HOLA)[1] of sequences on Val was lower than 0.001 for two successive iterations (the hold-out strategy). In order to assess the effect on performance of Train size, the number of sequences from Train that were used during learning was progressively increased from 25 to 100 sequences, by increments of 25, while Val, Test, TI, $\texttt{TN}_1$ and $\texttt{TN}_2$ were held fixed. At the end of each trial, SCFG probabilities associated with the minima of the negative log likelihoods (gEM and TS), or of the relative entropy of words (HOLA) on Val were stored.

Each independent trial was replicated 10 different times, with production rule probabilities initialized in 10 different ways – one in a uniform way, and the others in a random way. Test was then used to assess performance and select, for each technique, the best set of SCFG production rule probabilities among the different probability initializations. Finally, the performance on TI, $\texttt{TN}_1$ and $\texttt{TN}_2$ was measured with the best set of SCFG probabilities.

The performance of techniques was compared in terms of the amount of resources required during training, and the accuracy of results on the test sets. The accuracy of SCFG produced using gEM, TS and HOLA were assessed in terms of the perplexity, and the estimation error on radar states. In contrast, the amount of resources required to implement these techniques is measured using the time and space complexity and the convergence time. Estimation of the execution time and storage requirements allows for the assessment of complexity, while computer simulation using the Mercury data set allows to asses the other performance measures.

Perplexity (PP) is measured with $PP = 2^{-\frac{1}{n}\log_2 P(w_1^n)}$ where $P(w_1^n)$, is the probability of the sequence $\{w_1 \dots w_n\}$ being produced by a language [3]. Perplexity can be interpreted as the number of words that the model has to choose from to complete a given sequence. It is based on information theory, and is independent of sequence length. The closer this value is to 1, the more a model can predict the language. *Estimation error on states* is estimated by the ratio of incorrectly estimated states over all occurrences of the states in the tests.

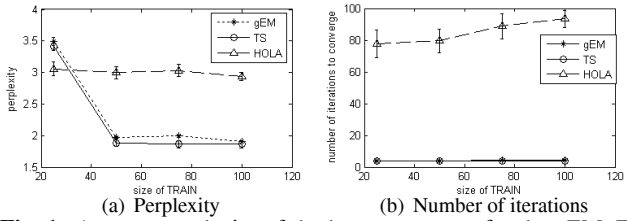
Time complexity can be estimated analytically from the time required during one iteration, to re-estimate production rule probabilities of a SCFG from a single training sequence. The result is a total average-case or worst-case running time formula, T, which summarizes the behavior of an algorithm as a function of key parameters. For simplicity, time complexity is estimated as a sum of the number of multiplications and divisions. The *average-case* is estimated by introducing the stochastic parameters, while the *growth rate* is obtained by making the parameters of the worst-case complexity tend to ∞. *Space complexity S* is estimated as the number of 8 bit registers needed during learning process to store variables. Only the average-case memory space required during preprocessing phase was considered. (The temporary memory space required during the iterative processes was neglected.) For gEM, one branch of a support graph consists in 3 vectors of 3 registers $([A, B, C]; [B, i, k]; [C, k, j]$, which means that the non-terminal A is expanded by BC to produce the subsequence $\{w_i ... w_j\}$), or 2 vectors of 2 and 3 registers $([A, a]; [a, i, i+1]$, which means that A is expanded by a to produce w_i). With TS, a production rule consists only in a vector of 2 or 3 registers ($[A, a]$ or $[A, B, C]$ for emission or transition rules). With HOLA only one register is needed for each rule (to representing its frequency).

Finally, convergence time is measured by counting the number of iteration I of Train needed for learning to end. This measure is independent from time complexity, since an algorithm may require several iterations to converge using very simple processing, or vice-versa. The overall time complexity associated with a learning technique is $T_{\text{tot}} = T_{\text{init}}(|\Omega|) + I \cdot T \cdot |\Omega|$, where $T_{\text{init}}(|\Omega|)$ is the time required during preprocessing to produce data structure. The product of I and T provides useful insight into the amount of processing required by technique to produce its best asymptotic performance.

[1] For HOLA, a maximum number of 100 iterations was also set.

5. RESULTS AND DISCUSSION

Figure 1 shows the average perplexity of the best parse trees and number of iterations achieved by the gEM, TS and HOLA techniques on Test, as a function of Train size. An average perplexity of about 1.9 is obtained when learning with TS, of just below 2.0 with gEM, and of about 3.0 with HOLA. This level of performance is attained when gEM and TS use a Train of 50 sequences, and when HOLA uses a Train of 25 sequences. In those specific cases, gEM and TS require on average about 4 iterations to converge, while HOLA requires on average about 80 iterations to converge.



(a) Perplexity (b) Number of iterations

Fig. 1. Average perplexity of the best parse trees for the gEM, TS and HOLA techniques versus Train size, for the Mercury MFR. (Error bars are standard error of the sample mean.)

In practice, the SCFG that yields lowest perplexity for the smallest amount of resources would be selected. The set of SCFG probabilities corresponding to the lowest perplexity across the 10 initializations, when Train size is 50 (best ratio PP/resources), were selected for gEM, TS, and HOLA. Table 1 shows the perplexity obtained on TI, $\texttt{TN}_1$, and $\texttt{TN}_2$ test sets. Although perplexity tends to grow with the level of noise for all three techniques, the perplexity obtained with gEM and TS is always lower than with HOLA. The confusion matrix for the same case is shown in Table 2. This table presents the estimation errors for the SCFGs obtained with gEM, TS and HOLA for each state of the Mercury MFR. Although the SCFG obtained with HOLA usually produced the greater number of estimation errors, all SCFGs behave consistently. When TI is processed, the only estimation error occurs when Na is estimated instead of Acq. However, the ability of SCFGs to estimate radar states degraded with noisy data.

Test subset	gEM	TS	HOLA
TI	1.8103	1.8285	2.4899
$\texttt{TN}_1$	3.1271	3.1784	4.0492
$\texttt{TN}_2$	9.8086	10.187	12.972

Table 1. Perplexity on TI, $\texttt{TN}_1$, and $\texttt{TN}_2$ for SCFGs obtained with gEM, TS and HOLA.

Tables 3 and 4 give the average time and space complexities, and corresponding growth rate, associated with IO, gEM, TS and HOLA. For IO, the average and worst-case time and space complexities are fixed, regardless of SCFG. In contrast, the average complexities for gEM, TS and HOLA differ considerably from the worst-case. For instance, the time and space complexities for gEM and TS grow exponentially with the inherent ambiguity of the SCFG. With HOLA, these complexities grow linearly with the number of

			Estimated States				
			S	**Acq**	**Na**	**Rr**	**Tm**
			TI / TN_1 / TN_2	TI / TN_1 / TN_2	TI / TN_1 / TN_2	TI / TN_1 / TN_2	TI / TN_1 / TN_2
Real States	**S**	gEM	4178 / 3500 / 2433	0 / 107 / 343	0 / 150 / 164	0 / 18 / 110	0 / 66 / 341
		TS	4178 / 3440 / 2399	0 / 109 / 359	0 / 173 / 157	0 / 35 / 96	0 / 66 / 312
		HOLA	4178 / 3380 / 2052	0 / 134 / 377	0 / 161 / 255	0 / 37 / 217	0 / 146 / 569
	Acq	gEM	0 / 10 / 22	1734 / 1530 / 1044	235 / 345 / 32	0 / 7 / 66	0 / 0 / 85
		TS	0 / 9 / 18	1734 / 1527 / 1027	235 / 333 / 347	0 / 6 / 89	0 / 11 / 64
		HOLA	0 / 3 / 35	1734 / 1452 / 800	235 / 355 / 281	0 / 25 / 92	0 / 52 / 213
	Na	gEM	0 / 0 / 0	0 / 8 / 7	736 / 613 / 412	0 / 77 / 212	0 / 0 / 34
		TS	0 / 0 / 0	0 / 8 / 5	736 / 613 / 411	0 / 78 / 213	0 / 0 / 33
		HOLA	0 / 0 / 9	0 / 11 / 8	736 / 571 / 315	0 / 71 / 167	0 / 47 / 135
	Rr	gEM	0 / 31 / 21	0 / 0 / 16	0 / 51 / 122	2073 / 1852 / 1267	0 / 55 / 288
		TS	0 / 31 / 25	0 / 0 / 14	0 / 64 117/	2073 / 1835 / 1256	0 / 54 / 309
		HOLA	0 / 31 / 66	0 / 0 / 17	0 / 39 / 165	2073 / 1783 / 958	0 / 115 / 403
	Tm	gEM	0 / 445 / 1196	0 / 68 / 118	0 / 364 / 523	0 / 74 / 179	6243 / 5389 / 4249
		TS	0 / 505 / 1236	0 / 69 / 119	0 / 316 / 511	0 / 74 / 175	6243 / 5376 / 4222
		HOLA	0 / 552 / 1284	0 / 116 / 259	0 / 415 / 548	0 / 135 / 455	6243 / 5122 / 3716
1-Error rate per state		**gEM**	**1 / 0.838 / 0.582**	**1 / 0.882 / 0.602**	**0.758 / 0.631 / 0.424**	**1 / 0.893 / 0.611**	**1 / 0.863 / 0.68**
		TS	**1 / 0.823 / 0.574**	**1 / 0.88 / 0.592**	**0.758 / 0.631 / 0.423**	**1 / 0.885 / 0.606**	**1 / 0.861 / 0.676**
		HOLA	**1 / 0.809 / 0.491**	**1 / 0.837 / 0.461**	**0.758 / 0.588 / 0.324**	**1 / 0.86 / 0.462**	**1 / 0.82 / 0.595**

Table 2. Confusion matrix on TI, TN_1, and TN_2 for SCFGs obtained with gEM, TS and HOLA.

SCFG rules. Overall, HOLA has the lowest time and space complexity of the three. The others can have a very low time complexity but require a substantial amount of memory to store data structures. The time and space complexity associated with the SCFG model for Mercury for the different algorithms are: $T_{HOLA} = 196 < T_{TS} = 7.32 * 10^3 < T_{gEM} = 8.24 * 10^4 < T_{IO} = 2.64 * 10^{13}$ and $S_{HOLA} = 392 < S_{TS} = 1.74 * 10^7 < S_{IO} = 3.49 * 10^7 < S_{gEM} = 6.97 * 10^7$.

Methods	Average-case	Growth rate
IO	$\frac{4}{3}M_{nt}^3 L(L^2-1) + (L^2+1)$ $+M_{nt}M_t(L+2) + 2M_{nt}^3$	$O(M_{nt}^3 L^3)$
gEM	$6\|\Delta l_e\| + 9\|\varphi_t\|\|\Delta l_t\|$	$O(M_{nt}^3 L^3)$
TS	$\|\Delta Tr\|\|Tr\|$	$O(M_{nt}^L L^3)$
HOLA	$2\|r\|$	$O(M_{nt}^3)$

Table 3. Estimates of time complexity of learning techniques. In this table, $|r|$ is the number of rules of the grammar, L is the size of a training sequence, M_t is the number of emission rules, $|\Delta l_t|$ and $|\Delta l_e|$ are the average numbers of sub-graphs in support graph corresponding to transition and emission rules, $|\varphi_t|$ is the average number of branches per sub-graph corresponding to a transition rule, $|Tr|$ is the average number of trees leading to a sequence and $|\Delta Tr|$ is the average size of a tree, that is the average number of production rules per tree.

Methods	Average-case	Growth rate
IO	$2M_{nt}L^2$	$O(M_{nt}L^2)$
gEM	$3\|\varphi_t\|\|\Delta l_t\|$ $+2\|\Delta l_e\| + 4M_{nt}L^2$	$O(M_{nt}^3 L^2)$
TS	$\|\Delta Tr\|\|Tr\| + M_{nt}L^2$	$O(M_{nt}^L L^3)$
HOLA	$4\|r\|$	$O(M_{nt}^3)$

Table 4. Estimates of space complexity of learning techniques.

6. CONCLUSION

Three fast incremental alternatives to the IO technique – gEM, TS, and HOLA – have been compared for learning the production rule probabilities of SCFGs, in order to recognize MFR systems and estimate their states in ES applications. Unlike IO, these techniques rely on pre-computed data structures to accelerate the probability re-estimation process, and are suitable for learning new information incrementally, without re-training from the start using all training data. Their performance has been measured in terms of resource allocation and accuracy.

The selection of either technique would ultimately depend on the specifics of the ES application, and effect a tradeoff between accuracy and computational efficiency. In computer simulations using a synthetic radar data set, gEM and TS have provided a greater level of accuracy than HOLA, yet converged after fewer training iterations. Unless the MFR system is modeled by a very ambiguous SCFG, these two techniques can learn probabilities rapidly, at the expense of significantly memory resources. On the other hand, the execution time and memory requirements of HOLA are orders of magnitude lower than that of gEM and TS. Its complexity is bounded by the number of SCFG rules, not the amount of training data. Furthermore, HOLA is an on-line technique that can update these probabilities incrementally on the fly in order to reflect changes in the environment.

7. REFERENCES

[1] J. K. Baker. Trainable grammars for speech recognition. In *Speech Communications Papers for the 97th Meeting of the Acoustical Society of America*, pages 547–550, June 1979.

[2] C. L. Davies and P. Hollands. Automatic processing for esm. In *Proc. IEE,129 (3), Part F*, pages 164–171, 1982.

[3] Y. Estve. *Intégration de Sources de Connaissances pour la Modélisation Stochastique du Langage Appliquée à la Parole Continue dans un Contexte de Dialogue Oral Homme-Machine*. PhD thesis, Université d'Avignon et des pays de Vaucluse, 2002.

[4] K. S. Fu. *Syntactic Pattern Recognition and Applications*. Englewood Cliffs, N.J.: Prentice-Hall, 1982.

[5] K. Lari and S.J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56, 1990.

[6] T. Oates and B. Heeringa. Estimating grammar parameters using bounded memory. In H. Fernau P. Adriaans and M. van Zaanen, editors, *Proceedings of the Sixth International Colloquium on Grammatical Inference (ICGI)*, pages 185–198, Springer-Verlag Heidelberg, 2002.

[7] T. Sato and Y. Kameya. Parameter learning of logic programs for symbolic-statistical modeling. *Journal of Artificial Intelligence Research*, 2:391–454, 2001.

[8] N. Visnevski. *Syntactic Modeling of Multi-Function Radars*. PhD thesis, McMaster University, http://soma.ece.mcmaster.ca/~visnev/Resources/PHDThesis/, 2005.

[9] N. Visnevski, F. A. Dilkes, S. Haykin, V. Krisnamurthy, and B. Currie. Non-self-embedding context-free grammars for multi-function radar modeling - electronic warfare application. In *Proceedings of the 2005 IEEE International Radar Conference*, May 2005.

[10] R. G. Wiley. *Electronic Intelligence: The Analysis of Radar Signals, 2nd ed.* Norwood, MA:Artech House, 1993.