TEL AVIV UNIVERSITY
RAYMOND AND BEVERLY SACKLER
FACULTY OF EXACT SCIENCES
SCHOOL OF PHYSICS & ASTRONOMY

אוניברסיטת תל אביב
הפקולטה למדעים מדוייקים
ע"ש ריימונד ובברלי סאקלר
בית הספר לפיסיקה ואסטרונומיה

Unsupervised Learning of Natural Languages

Thesis submitted towards the degree of

Doctor of Philosophy

by

Zach Solan

Submitted to the Senate of Tel Aviv University

May, 2006

This work was carried out under the supervision of

Professor David Horn

Professor Shimon Edelman

Professor Eytan Ruppin

To Noa, Karni, and Maly

Acknowledgments

First and foremost, I wish to express my deepest gratitude to my supervisors, Prof. Shimon Edelman, Prof David Horn, and Prof Eytan Ruppin, for their dedicated guidance, constant encouragement, infinite support and friendship. None of this work would have been possible without them. I do hope that our joint adventure, initiated more than four years ago, will continue to flourish.

I would like to thank the University Committee members Prof. Haim Sompolinsky and Dr. Shuly Wintner for their time and effort to review and inspect this research. I am deeply indebted to my friends and colleagues Ben Sandbank and Vered Kunik for their invaluable advice, encouragement and so many fruitful discussions. Far too many persons to mention individually have assisted me in so many ways during my joint project at the Tel-Aviv and Cornell Universities during the last four years. They have all my sincere gratitude, especially Yaniv Oshrat, Liat Segal, Roi Varshavsky, Uri Barakan, Roded Sharan, Tzachi Pilpel, Michal Lapidot, and Roni Katzir. I would like to thank Asa Ben-Hur and Shai Shen-Or for providing some of the bioinformatics data and Bo Pedersen for helping with the psycholinguistic testing. In addition I would like to thank Elhanan Borenstein, Alex Clark, Barb Finlay, Mark Johnson, Dan Klein, as well as the anonymous reviewers, for their comments that improved the final version of the PNAS paper. I owe a huge debt of gratitude to Michal Finkelman for helping me to find my way through the University maze. I would like to thanks Danny Shaked and Fred Horan for taking care of the system and answering all my questions and concerns. I wish to thank the Israel Science Foundation, the Horowitz Center for Complexity Science, The Dan David Prize Foundation, and The Adams Super Center for supporting my research.

The penultimate thank-you goes to my very dear parents for encouraging my scientific curiosity and believing in my skills since my early childhood, for always being present when I needed them, and for their continuous assistance in realizing my

dream.

The most heartfelt acknowledgment goes to my family: To my adorable daughters, Noa and Karni, born during these four magnificent years, who brought so much joy to our life, for sharing with me the amazing and miraculous process of language acquisition. And finally, my deepest thanks go to my wife and my best friend Maly, for being my inspiration, for being an invaluable source of advice and encouragement, for her willingness to understand the details of my work, for her constant support and love during my ups and downs, related to the ongoing rhythm of this work, and for experiencing with me the thrill of research.

Abstract

Many types of sequential symbolic data possess structure that is (i) hierarchical, and (ii) context-sensitive. Natural-language text or transcribed speech are prime examples of such data: a corpus of language consists of sentences, defined over a finite lexicon of symbols such as words. Linguists traditionally analyze the sentences into recursively structured phrasal constituents [80]; at the same time, a distributional analysis of partially aligned sentential contexts [44] reveals in the lexicon clusters that are said to correspond to various syntactic categories (such as nouns or verbs). Such structure, however, is not limited to the natural languages: recurring motifs are found, on a level of description that is common to all life on earth, in the base sequences of DNA that constitute the genome. In this thesis, I address the problem of extracting patterns from natural sequential data and inferring the underlying rules that govern their production. This is relevant to both linguistics and bioinformatics, two fields that investigate sequential symbolic data that are hierarchical and context sensitive. Given a corpus of strings (such as text, transcribed speech, nucleotide base pairs, amino acid sequence data, musical notation, etc.), the unsupervised algorithm developed as a part of this project recursively distills from it hierarchically structured patterns. The ADIOS (Automatic DIstillation of Structure) algorithm relies on a statistical method for pattern extraction and on structured generalization, two processes that have been implicated in language acquisition. It has been evaluated in the main scheme of grammar induction using artificial context-free grammars with thousands of rules, In on natural languages as diverse as English and Chinese, on coding regions in DNA sequences, and on protein data correlating sequence with function. This is the first time an unsupervised algorithm is shown capable of learning complex syntax, generating grammatical novel sentences, scoring well in standard language proficiency tests, and proving useful in other fields that call for structure discovery from raw data, such as bioinformatics.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Nothing surpasses the complexity of the human mind.

Leto II: Dar-es-Balat Records

HERETICS OF DUNE

1. GENERAL INTRODUCTION

1.1 Motivation

The study of language aims to characterize and explain the way humans acquire, produce and understand language, based on the main assumption that rules (grammar) are used to structure all three of these processes. For decades researchers have been trying to devise formal and rigorous detailed grammars which would capture the observed regularities of language. All such efforts consistently fall short of their goal, presumably because grammars tend to mutate, depending on how grammaticality is defined and on the purpose to which language is put; as Edward Sapir put it, "All grammars leak" [1].

Despite that, we cannot disregard the fundamental observation that the patterns of language that govern the production of those expressions that are indisputably well-formed are essentially rule-like. While the existence of syntactic rules seems necessary to explain linguistic productivity, it is less clear what form should such rules take and how they are to be acquired. Two distinct approaches to this basic issue exist. The rationalist approach claims that extensive innate knowledge of grammar is essential to explain the acquisition of language from positive-only data [19, 82]. The empiricist approach, in comparison, claims that general cognitive mechanisms such as pattern recognition and generalization can solve the language acquisition problem.[1]

The rationalist approach became dominant in linguistics when Noam Chomsky introduced the concept of innate language faculty which was based on the Argument from the Poverty of the Stimulus (APS). That is, the data available to the infant child is so impoverished and degenerate that no general, domain independent learning algorithm could possibly learn a plausible grammar without assistance. In this work I explore the question of how far a generalized learning mechanisms can reach without

[1] Another major difference between the rationalist and empiricist approaches has to do with the level of description: while rationalists see the language itself, such as text, as only indirect evidence about the human language faculty ("competence"), empiricists are interested in describing and accounting for the actual language data ("performance").

any prior knowledge.

Computational models of language acquisition or "grammar induction" are usually divided into two categories, depending on whether they subscribe to some variant of the classical generative theory of syntax, or operate within the framework of "general-purpose" statistical or distributional learning. The present work is rooted in the belief that polarization between statistical and classical (generative, rule-based) approaches to syntax hampers the integration of the stronger aspects of each method into a common powerful framework. On the one hand, the statistical approach is geared to take advantage of the considerable progress made to date in the areas of distributed representation, probabilistic learning, and "connectionist" modeling, yet generic connectionist architectures are ill-suited to the abstraction and processing of symbolic information. On the other hand, classical rule-based systems excel in just those tasks, yet are brittle and difficult to train.

In view of these considerations, this project set out to develop an approach to the acquisition of distributional information from raw input (e.g., transcribed speech corpora) that also supports the distillation of structural regularities comparable to those captured by Context Sensitive Grammars out of the accrued statistical knowledge. In thinking about such regularities, I adopt Langacker's vision: "As conceived in the present framework, the grammar of a language is simply an inventory of linguistic units" ([65], p.63). To detect potentially useful units, the algorithm that I shall describe identifies and processes sentences that are partially redundant in that they share the same word sequences. The detection of paradigmatic variation within a slot in a set of otherwise identical aligned sequences (syntagms) is the basis for the classical distributional theory of language [44], as well as for some modern NLP methods [98]. Likewise, the *pattern* — the syntagm and the *equivalence class* of complementary-distribution symbols that may appear in its open slot — is the main representational building block of my system, ADIOS (for Automatic DIstillation Of

Structure) [93, 94, 90].

My goal is to help bridge statistical and formal approaches to language [78] by placing the work on the unsupervised learning of structure in the context of current research in grammar acquisition in computational linguistics, and at the same time to link it to certain formal theories of grammar. Consequently, the following sections outline the main computational principles behind the ADIOS model, compare these to select approaches from computational and formal linguistics, report its performance across different tasks and describes its wide range of applications.

1.2 Outline of the thesis

In chapter 2 I introduce the thesis background as well as the aspects which inspired me during the design and the development of ADIOS. Chapter 3 describes in detail the MEX and ADIOS algorithms based on [91]. Chapter 4 presents the main results in several domains, such as artificial language natural language, genomics and proteomics. This chapter is based on the following publications: [93, 94, 90, 91, 92]. In Chapter 5, I survey the main related computational and linguistic approaches to grammar induction, where I compare the ADIOS approach to each one of the listed techniques. I then present the main psycholinguistic findings that have been implicated in language acquisition. Finally, I discuss the implications, the prospects, and the challenges of the ADIOS algorithm.

2. INTRODUCTION

Children acquire language spontaneously and rapidly, attaining within just a few years an impressive mastery over an infinitely productive means of conceptual thinking and communication. The brain mechanisms behind this uniquely human cognitive faculty are surprisingly poorly understood. Although the generative program in linguistics made significant inroads into exposing the structure of the language faculty, a computationally viable, behaviorally plausible, and neurally explicit theory of language acquisition and processing continues to be elusive. On the computational level, the principles that make possible the acquisition of rich language from impoverished stimuli, and the connections between language and the rest of cognition, are the subject of a continued controversy. On the behavioral level, the wealth of data from psychological studies of linguistic performance in acquisition and in various production and comprehension tasks waits to be integrated with the computational theories. On the level of neural mechanisms, little is known about the brain basis of language beyond its general topography, gleaned through the study of abnormal cases, and, more recently, through functional imaging. The language faculty thus remains a daunting enigma for cognitive scientists, and a formidable challenge for engineers who would imitate it. Although understanding how language works requires progress on all levels, the prospects for it depend most critically on innovation in theoretical/computational thinking.

2.1 The primacy of distributional information

Until recently, distributional information[1] present in a corpus could not be easily distilled and visualized, due to technical limitations. For that reason, most studies involving large corpora used to limit their consideration to simple statistics such as word frequencies, drawing just criticism from the opponents of distributional the-

[1] Distributional information quantifies the degree to which words tend to occur near each other in speech. One kind of such information is co-occurrence statistics, which captures the probability of target words to appear together within a small window (typically between 2 and 10 words wide).

ories. The situation is rather different in NLP, or natural language processing (a branch of computer science) and in psycholinguistics. In the former field, the integration of advanced statistical inference (including Bayesian methods), computational learning theory, efficient algorithms, and cheap hardware led to important conceptual progress, as well as to practical achievements [72]. In psycholinguistics, a rapidly growing body of data attesting to the heavy reliance of language users on distributional information suggests that corpus-based NLP may be in fact constitute a good starting point for modeling the representation and processing of linguistic information by the human language faculty. Distributional information has been experimentally implicated in comprehension, production, and acquisition of language by human subjects; some of the variables tallied are frequency of past tense versus past participle uses of a verb, frequency of transitive versus intransitive uses of a verb, frequency with which a verb and its complement are non-adjacent, and quite a few others; cf. [69], p.190.

Much more importantly, discussions of the role of distributional knowledge have begun to address issues dealing with higher order statistics than mere frequencies (although such concepts are still rarely mentioned explicitly in this literature). One example of such a statistic arises in connection with the concept of *prefabs* — sequences, continuous or discontinuous, of words that appear to be prefabricated, that is, stored and retrieved as a whole, rather than being subject to syntactic processing [102, 101]. Members of a two-word prefab, for example, are distinguished by a higher than expected *conditional* probability on each other — a second-order distributional quantity that can be related both to Bayesian inference and to the Minimum Description Length principle, as illustrated (in the context of vision) by [31]. A recent study involving two corpora found that about 55% of words in both spoken and written English are parts of prefabs [29]. It is no wonder, therefore, that mastery of a language depends on getting right the prefabs (“motherhood and apple pie” but not

"cherry pie") and the idioms ("take it to the bank," which is not the same as "carry it to the bank"). For this, the possession of distributional information is crucial.

2.2 The process of abstraction

Considerations of representational parsimony dictate that the explanation for the pattern of acceptable sentences in a language be as concise as possible. A reduced representation of linguistic knowledge need not, however, take the form of a meta-language such as a prescriptive rule-set or grammar [45]. Instead, syntax may constitute an abstraction, emerging from a corpus of language [50], yet coexisting within the same representational mechanism that embodies the data. The process of abstraction can be guided by principles such as complementarity of distributions: tokens that function similarly in some sense (phonological, morphological, syntactic or semantic) but represent systematic rather than free variation will form complementary distributions or classes (e.g., [43, 53]).

When syntax is seen as an abstraction from distributional data, one may expect both the amount and the nature of syntactic knowledge to differ across individuals. The differences could stem, among other factors, from variation in exposure to linguistic material, as well as from varying efficiency with which subjects process the material they encounter (that in addition to the generally accepted characterization of language as a dynamic, constantly evolving system [22, 45]). An immediate consequence of this stance on syntax is the prediction of individual differences in the classical task of grammaticality judgment, which is being corroborated by an increasing number of published studies [40, 89]. Individual differences are also predicted by this conceptual framework for the behaviorally much more relevant task: sentence comprehension. Here too, the prediction is borne out by experiments; for example, recent studies [17, 18] found that comprehension scores for sentences of varying controlled complexity differed between three groups of subjects: uneducated

native speakers of English, native speakers with graduate education, and non-native graduate speakers; surprisingly, the latter group yielded the highest scores (the subjects' performance also varied with syntactic complexity, semantic plausibility, and the difficulty of the scoring question). Crucially, these individual differences in syntactic aptitude were obtained under self-paced unlimited-time exposure to the written sentences, a provision that minimizes the effects of so-called extra-linguistic, "performance" factors.

In thinking about emergent regularities [50], or syntactic-semantic constructions [32], I adopt Langacker's vision:

> "... particular statements (specific forms) coexist with general statements (rules accounting for those forms) in a speaker's representation of linguistic convention, which incorporates a huge inventory of specific forms learned as units (conventional expressions). Out of this sea of particularity speakers extract whatever generalizations they can. Most of these are of limited scope, and some forms cannot be assimilated to any general patterns at all. Fully general rules are not the expected case in this perspective, but rather a special, limiting case along a continuum that also embraces totally idiosyncratic forms and patterns of all intermediate degrees of generality." [65], p.43.

Langacker's conception of grammar as an inventory of linguistic units, which is structured in the sense that some units function as components of others, is well-suited to serve as a basis for a psychologically motivated theory of language learning, due to its clear parallels with the notion of *unitization* that arises in cognitive psychology [34]. Recent developments in probability and information theory and in computational learning have rendered distributional [45, 46] methods of linguistic unitization both more tractable and more readily relatable to grammar-based formalisms [78].

A representative example of such a development is the construction-based ap-

proaches to syntax [23, 33], which posit a lexicon populated by units of various sizes, as envisaged by [65]. Constructions may be specified completely, as in the case of simple morphemes or idioms such as take it to the bank, or partially, as in the expression whats X doing Y?, where X and Y are slots that admit fillers of particular types [58]. Constructions offer an intriguing alternative to traditional rule-based syntax by hinting at the extent to which the complexity of language can stem from a rich repertoire of stored, more or less entrenched [42] representations that address both syntactic and semantic issues, and encompass, in addition to general rules, totally idiosyncratic forms and patterns of all intermediate degrees of generality ([65], p.46). Because constructions are by their very nature language-specific, the question of acquisition in Construction Grammar is especially poignant.

2.3 Machine learning

In the wide field of Machine Learning, a key distinction can be made between supervised and unsupervised learning techniques. In unsupervised learning, the goal is to find a pattern from the data alone, without being told ahead of time (e.g., by a teacher) what such a pattern may be. In supervised learning, in comparison, the learner is trained on data/output pairs provided by the teacher. Existing grammar acquisition methods (as any other learning algorithms) can be supervised or unsupervised. One form of supervision is the use of negative data (that is, examples labeled as ungrammatical by a teacher).

Very few unsupervised grammar induction methods work with positive examples of raw, untagged data (see Section 5.1.1 for a survey). Reduction of redundancy is a general (and arguably the only conceivable) approach to unsupervised learning [6, 7] . Written natural language (or transcribed speech) is trivially redundant to the extent it relies on a fixed lexicon. This property of language makes possible the unsupervised recovery of words from a text corpus with all the spaces omitted,

through a straightforward minimization of per-letter entropy [83].

Pushing entropy minimization to the limit would lead to an absurd situation in which the agglomeration of words into successively longer "primitive" sequences renders the resulting representation useless for dealing with novel texts (that is, incapable of generalization; cf. [100], p.188). I observe, however, that a word-based representation is still redundant to the extent that different sentences share the same word sequences. Such sequences need not be contiguous; indeed, the detection of paradigmatic variation within a slot in a set of otherwise identical aligned sequences (syntagms) is the basis for the classical distributional theory of language [44], as well as for some modern NLP methods [98]. The pattern — the syntagm and the equivalence class of complementary-distribution symbols[2] that may appear in its open slot — is the main representational building block of my system, ADIOS (for Automatic DIstillation Of Structure) [93]. My thesis aims to transform the idea of the emergence of distributional syntactic knowledge into a concrete computational model which is based on the following three principles.

2.4 The principles behind the ADIOS algorithm

The representational power of ADIOS and its capacity for unsupervised learning rest on three principles: (1) probabilistic inference of pattern significance, (2) context-sensitive generalization, and (3) recursive construction of complex patterns. Each of these is described briefly below.

2.4.1 Probabilistic inference of pattern significance.

ADIOS represents a corpus of sentences as an initially highly redundant directed graph, The algorithm, described in detail in section 3.2, identifies significant patterns that balance high compression (small size of the pattern "lexicon") against good

[2] The symbols may be letters or morphemes.

generalization (the ability to generate new grammatical sentences by splicing together various fragments each of which belongs to a different pattern).

2.4.2 Context sensitivity of patterns.

A pattern is an abstraction of a bundle of sentences that are identical up to variation in one place, where one of several symbols — the members of the equivalence class associated with the pattern — may appear (Figure 3.7). Because this variation is only allowed in the context specified by the pattern, the generalization afforded by a set of patterns is inherently safer than in approaches that posit globally valid categories ("parts of speech") and rules ("grammar"). The reliance of ADIOS on many context-sensitive patterns rather than on traditional rules may be compared to the central idea of Construction Grammar, mentioned above.

2.4.3 Hierarchical structure of patterns.

The ADIOS graph is rewired every time a new pattern is detected, so that a bundle of strings subsumed by it is represented by a single new vertex. Following the rewiring, which is context-specific, potentially far-apart symbols that used to straddle the newly abstracted pattern become close neighbors. Patterns thus become hierarchically structured in that their elements may be either terminals (i.e., fully specified strings) or other patterns. Moreover, patterns may refer to themselves, which opens the door for true recursion (Figure 3.7).

3. METHOD

In the past four years, I have been working on developing a novel computational approach to language acquisition and representation, which bridges the gap between statistical (corpus-based) and classical (generative, rule-based) methods, integrating the stronger aspects of each into a common powerful framework. The emerging approach takes advantage of the statistical, distributional information present in raw input (such as transcribed speech corpora), at the same time supporting also the distillation of structural rule-like regularities out of the accrued knowledge.

The key component of my approach is a novel unsupervised algorithm that discovers hierarchical, context-sensitive structure in language data, on the basis of the minimal assumption that the corpus at hand contains partially overlapping sentences at multiple levels of abstraction. Intuitively, a sequence of words that is common to several sentences can be abstracted into a higher-order component, or pattern; the sentences can then be rewritten using this new component, and the search for overlapping sequences repeated. Furthermore, if a bundle of aligned word sequences differ in but one place, the words that occupy that slot are equivalent (belong to the same class) in that they are interchangeable in the given context. These two insights, which have a long history in structural linguistics, require a proper algorithmic formulation if they are to serve as a basis for language acquisition and representation. In my algorithm, the decision concerning the significance of the candidate structures — patterns and equivalence classes — is given a solid computational basis (I use a novel context-sensitive probabilistic criterion (the Motif Extraction criterion) defined in terms of local flow quantities in a graph whose vertices are the lexicon entries and where the paths correspond, initially, to corpus sentences). New patterns and equivalence classes can incorporate those added previously, leading to the emergence of recursively structured units that also support generalization, by opening paths that do not exist in the original corpus.

In the following section I describe the MEX criterion which is used as a distillation tool for extracting the most significant patterns in the data (see section 3.1). I then continue by describing the main mechanism that governs the generation of candidates (patterns and equivalence classes) to be considered by the MEX criterion. To help the reader understand the algorithm, I have included throughout the section boxes with the relevant pseudo-code along with some simple examples that illustrate the operation of the algorithm.

3.1 The MEX algorithm

Consider a corpus of m sentences (sequences) of variable length, each expressed in terms of a lexicon of finite size N. The sentences in the corpus correspond to m different paths in a pseudograph (a non-simple graph in which both loops and multiple edges are permitted) whose vertices are the unique lexicon entries, augmented by two special symbols, begin and end. Each of the N nodes has a number of incoming paths that is equal to the number of outgoing paths. Figure 3.1 illustrates the type of structure that we seek, namely, the bundling of paths, signifying a relatively high probability associated with a sub-structure that can be identified as a pattern. To extract it from the data, two probability functions are defined over the graph for any given *search path* $\mathbf{S}(e_1 \rightarrow e_2 \rightarrow ... \rightarrow e_k) = (e_1; e_k)$.[1] The first one, $P_R(e_i; e_j)$, is the right-moving ratio of fan-through flux of paths at e_j to fan-in flux of paths at e_{j-1}, starting at e_i and moving along the sub-path $e_i \rightarrow e_{i+1} \rightarrow e_{i+2}... \rightarrow e_{j-1}$:

$$P_R(e_i; e_j) = p(e_j|e_i e_{i+1} e_{i+2}...e_{j-1}) = \frac{l(e_i; e_j)}{l(e_i; e_{j-1})} \tag{3.1}$$

where $l(e_i; e_j)$ is the number of occurrences of sub-paths $(e_i; e_j)$ in the graph. Proceeding in the opposite direction, from the right end of the path to the left, we define

[1] In general the notation $(e_i; e_j), j > i$ corresponds to a rightward sub-path of **S**, starting with e_i and ending with e_j. A leftward sub-path of **S**, starting with e_j and ending with e_i is denoted by $(e_j; e_i), i < j$.

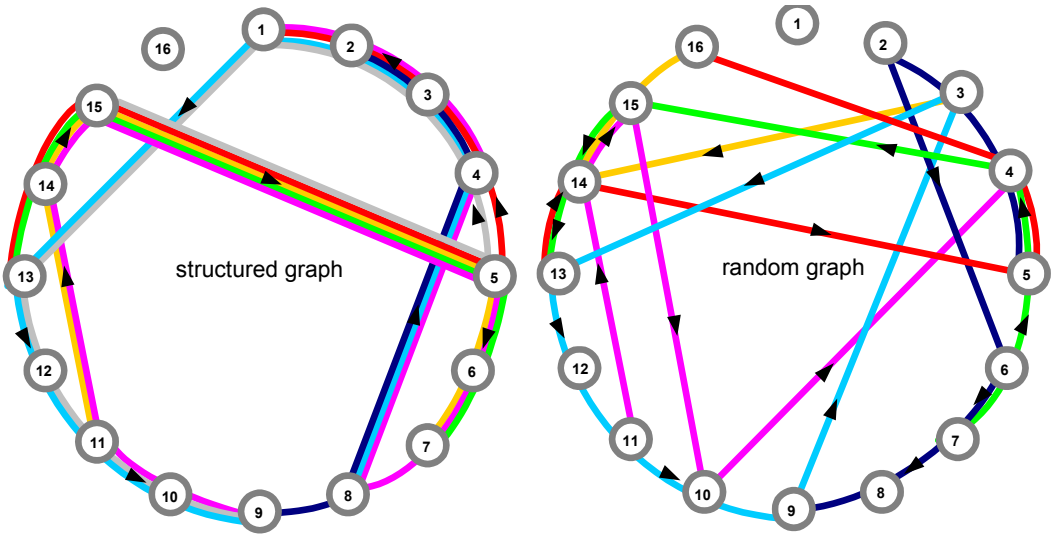

Fig. 3.1: Comparison between a structured graph, of the type expected to appear in my problem (left), and one of random connectivity (right).

the left-going probability function P_L:

$$P_L(e_j; e_i) = p(e_i | e_{i+1} e_{i+2} ... e_{j-1} e_j) = \frac{l(e_j; e_i)}{l(e_j; e_{i+1})} \tag{3.2}$$

and note that

$$P_R(e_i; e_i) = P_L(e_i; e_i) = \frac{l(e_i)}{\sum_{x=0}^{N} l(e_x)} \tag{3.3}$$

where N is the total number of vertices in the graph. Clearly, both functions vary between 0 and 1 and are specific to the path in question. The MEX algorithm is defined in terms of these functions and their ratios. In Figure 3.2, P_R first increases because some other paths join the search path to form a coherent bundle, then decreases at $e4$, because many paths leave it at $e4$. To quantify this decline of P_R, which is interpreted as an indication of the end of the candidate pattern, I define a *decrease ratio*, $D_R(e_i; e_j)$, whose value at e_j is $D_R(e_i; e_j) = P_R(e_i; e_j)/P_R(e_i; e_{j-1})$, and require that it be smaller than a preset *cutoff parameter* $\eta < 1$ (in the present example,

$D_R(e_1, e_5) = P_R(e_1, e_5)/P_R(e_1, e_4) < \frac{1}{3}$).

In a similar manner, the value of P_L increases leftward; the point e_2 at which it first shows a decrease $D_L(e_j; e_i) = P_L(e_j; e_i)/P_L(e_{j+1}; e_i) < \eta$ can be interpreted as the starting point of the candidate pattern. Large values of D_L and D_R signal a divergence of the paths that constitute the bundle, thus making a pattern-candidate. Since the relevant probabilities ($P_R(e_i; e_j)$ and $P_L(e_j; e_i)$) are determined by finite and possibly small numbers of paths ($l(e_i; e_j)$ out of $l(e_i; e_{j-1})$), we face the problem of small-sample statistics. It is useful therefore to supplement conditions such as $D_R(e_i; e_j) < \eta$ by a significance test based on binomial probabilities:

$$B(e_i; e_j) = \sum_{x=0}^{l(e_i;e_j)} Binom(l(e_i; e_{j-1}), x, \eta P_R(e_i; e_{j-1})) < \alpha; \alpha \ll 1, \tag{3.4}$$

The algorithm calculates both P_L and P_R from all the possible starting points (such as $e1$ and $e4$ in the example of Figure 3.2), traversing each path leftward and rightward, correspondingly. This defines a matrix of the form

$$M_{ij}(\mathbf{S}) = \begin{cases} P_R(e_i; e_j) & \text{if } i > j \\ P_L(e_j; e_i) & \text{if } i < j \\ P(e_i) & \text{if } i = j \end{cases} \tag{3.5}$$

One can write $M(\mathbf{S})$ in its explicit form, namely, as an instantiation of a variable-order Markov model up to order k, which is the length of the search-path:

$$\mathbf{M} \doteq \begin{pmatrix} p(e_1) & p(e_1|e_2) & p(e_1|e_2e_3) & \dots & p(e_1|e_2e_3 \dots e_k) \\ p(e_2|e_1) & p(e_2) & p(e_2|e_3) & \dots & p(e_2|e_3e_4 \dots e_k) \\ p(e_3|e_1e_2) & p(e_3|e_2) & p(e_3) & \dots & p(e_3|e_4e_5 \dots e_k) \\ & \vdots & \vdots & \vdots & \vdots \\ p(e_k|e_1e_2 \dots e_{k-1}) & p(e_k|e_2e_3 \dots e_{k-1}) & p(e_k|e_3e_4 \dots e_{k-1}) & \dots & p(e_k) \end{pmatrix}$$

Given the matrix $\mathbf{M}(\mathbf{S})$, the algorithm identifies all the significant $D_R(e_a; e_b)$ and

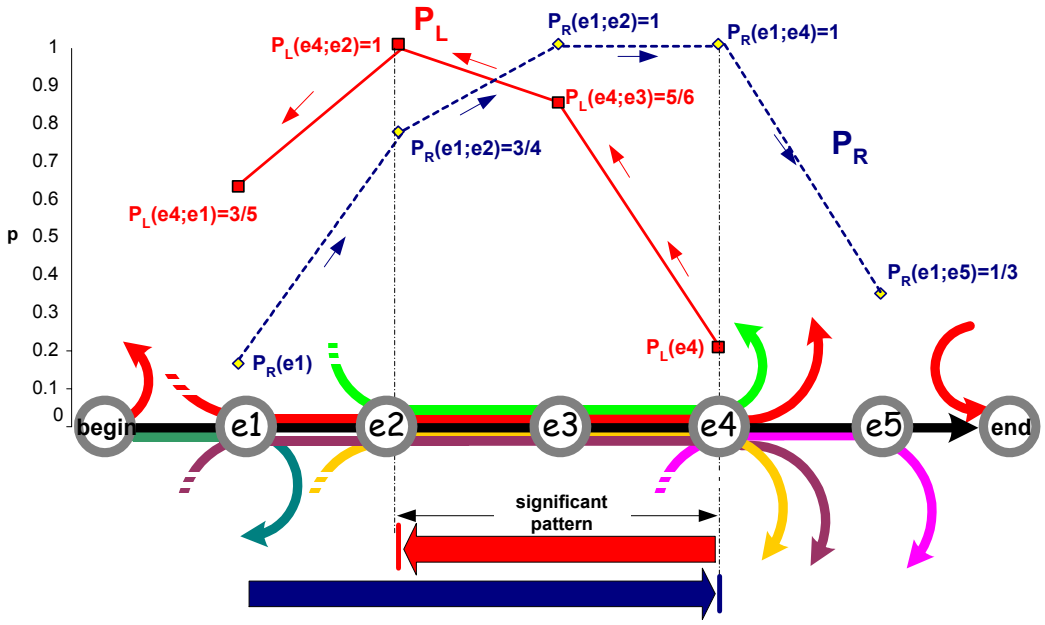

Fig. 3.2: The definition of a bundle that serves as a candidate pattern, whose beginning and end are signalled by the maxima of P_L and P_R. It becomes a candidate because of the large drops in these probabilities after the maxima, signifying the divergence of paths at these points.

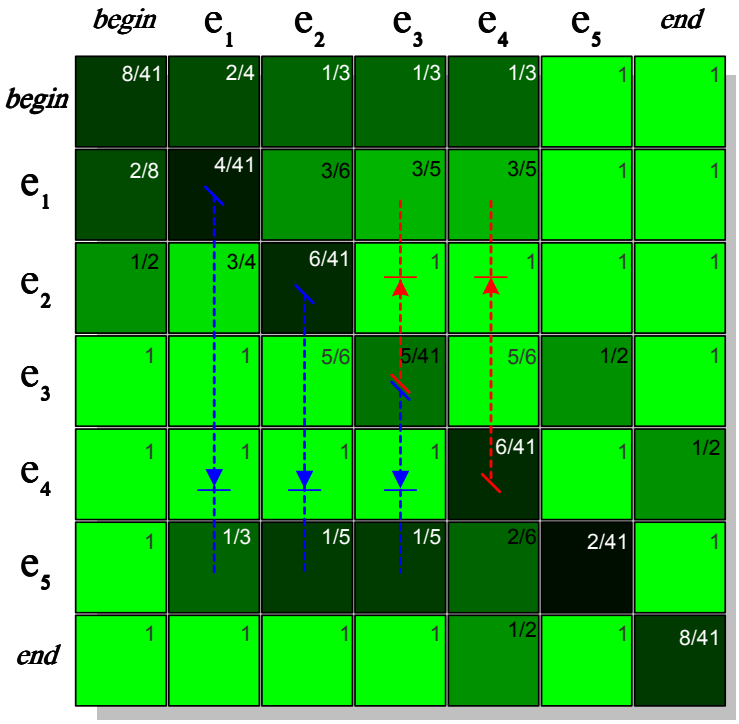

Fig. 3.3: An instance of a 7×7 M matrix based on the black search path in Figure 3.2. The blue and red arrows represent all the significant segments $D_R(e_a; e_b)$ and $D_L(e_d; e_c)$ ($\alpha < 0.01$), respectively. The values of the matrix elements appear in the upper right corners of the cells. The most significant pair of segments ($B(e_a; e_b)$, $B(e_d; e_c)$) for which $a < d < b < c$ is marked the *leading pattern* (in this example the leading pattern is $e_2 \rightarrow e_3 \rightarrow e_4$).

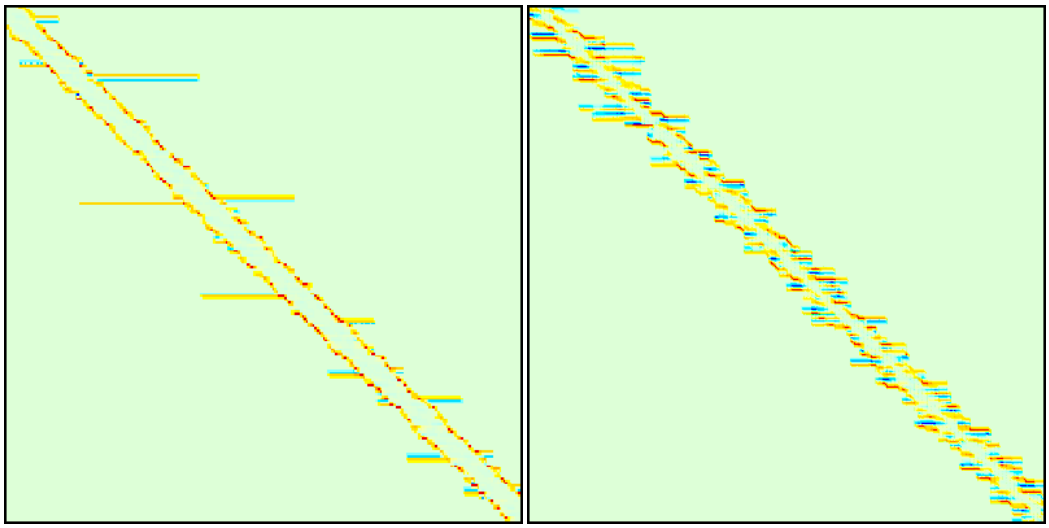

Fig. 3.4: Two instances of the M matrix computed from two different corpora: the SwissProt database sequence of amino acids (*left*), and the text of *Alice in the Wonderland* considered as a sequence of letters (*right*). Significant changes in the P_L and P_R values have been colored on a yellow/red scale and a cobalt/blue scale (increase and decrease, respectively); green is the neutral color. The protein O17433 is the search path used to construct the matrix on the left; the first paragraph of *Alice* is the search path used to create the matrix on the right. For visualization purposes, only the first 300 elements of the matrices are shown.

$D_L(e_d; e_c)$ ($1 \leq a, b, c, d \leq k$) and their coinciding pairs ($D_R(e_a; e_b), D_L(e_c; e_d)$), requiring that $a < d < b < c$. The pair with the most significant scores (on both sides, $B(e_a; e_b)$ and $B(e_d; e_c)$) is declared as the leading pattern (e_{d+1}; e_{b-1}). The leading pattern is returned as the outcome for the search path in question.

3.2 The ADIOS algorithm

As mentioned in section 3.1, each corpus sentence defines a separate path over the graph, starting at begin and ending at end, and is indexed by the order of its appearance in the corpus. Each one of these paths serves as a Search Path initially coinciding with one of the original corpus sentences.

The ADIOS algorithm consists of three modules, which operate serially on the pseudograph. Sentence loading (**1-Initialization**) is followed by an iterative search for significant *patterns* (**2-Pattern Distillation**), which are added to the lexicon as new units. The generalization mechanism (**3-Generalization**) generates more and more candidate patterns to be considered by the Pattern Distillation module. The structure of the initial graph is illustrated in Figure 3.5.

Algorithm 1 An overview of the ADIOS algorithm

1: **Initialization** (load all sentences)
2: **repeat**
3: **for all** $m = 1 : N$ **do** {N *is the number of paths in the graph*}
4: **Pattern Distillation(**m**)** (Identifies new significant patterns in search-path m using the MEX criterion)
5: **Generalization(**m**)** (Generate new pattern candidates for search-path (m))
6: **end for**
7: **until** no further significant patterns are found

After the algorithm has loaded all the paths onto the pseudograph, it starts considering candidate patterns by traversing, in each iteration, a different *search path* (initially coinciding with one of the original corpus sentences), seeking sub-paths that are shared by a significant number of partially aligned [44, 98] paths (see Algo-

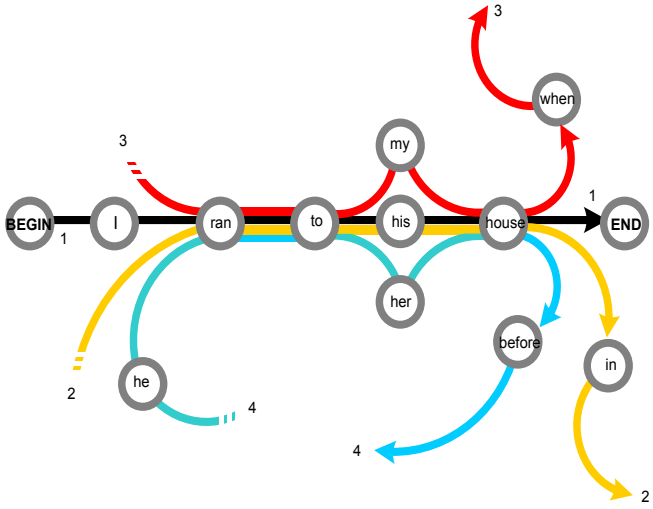

Fig. 3.5: The search path #1 (begin → i →→ run → to → my → house → end) is rendered as a solid black line connecting the special begin and end vertices. Three other paths (#2,#3,#4) join it along the vertices run,to, thus forming a bundle that may constitute a significant pattern subject to the MEX criterion described in Section 3.1.

Algorithm 2 Initialization

1: **for all** sentences $m = 1 : N$ **do** $\{N$ *is the number of sentences in the corpus*$\}$
2: **load** sentence m as path onto a pseudograph whose vertices $e_1 \dots e_{l(m)}$ are the unique words of the corpus starting at begin and ending at end symbols indexed by the order of their appearance in the corpus. $l(m)$ indicates the number of vertices in path m.
3: **end for**

rithm 3.2). The significant patterns (P) are selected according to a context-sensitive probabilistic criterion defined in terms of local flow quantities in the graph, stated in section 3.1 (known as the MEX criterion).

Generalizing the search path: the algorithm looks for an optional *equivalence class* (E) of units that are interchangeable in the given context (i.e., are in complementary distribution [44]). At the end of each iteration, the most significant pattern is added

Algorithm 3 Pattern Distillation

1: **for all** $m = 1 : N$ **do** $\{N$ *is the number of paths in the graph*$\}$
2: **for** $i = 1 : l(m)$ $l(m)$ is the number vertices in path n **do**
3: **for** $j = i + 1 : l(m)$ **do**
4: **find the leading significant pattern** perform MEX on the search segments (*i*,*j*), starting P_R at e_i and P_L at e_j; choose out of all segments the leading significant pattern P for the search path.
5: **rewire graph**
CREATE a new vertex corresponding to P.
6: **Mode A (context free):** replace the string of vertices comprising P with the new vertex P on all paths on which it occurs. [Figure 3.6C].
7: **Mode B (context sensitive):** replace the string of vertices comprising P with the new vertex P only on those paths on which P is significant according to the MEX criterion.
8: **end for**
9: **end for**
10: **end for**

to the lexicon as a new unit, the sub-paths it subsumes are merged into a new vertex, and the graph is rewired accordingly. Two rewiring modes are available: a context free Mode A, which replace the string of vertices comprising P with the new vertex P on all paths on which it occurs, and a context-sensitive Mode B, which replaces the string of vertices comprising P with the new vertex P only on those paths on which P is significant according to the MEX criterion. This highly context-sensitive approach to pattern abstraction, is unique to our model, allows ADIOS to achieve a high degree of representational parsimony without sacrificing its generalization power, although it requires enormous computational resources (see section 4.1.1.2). The entire process is governed by three parameters: α and η, which control pattern significance, and L, which sets the width of the context window where equivalence classes are sought (see Algorithm 3).

So that the MEX criterion can be applied to a generalized search path, we express it in terms of equivalence classes. Consider a situation in which a search-path contains an open slot where multiple alternative sub-paths coexist within a fixed context defined by the main path. As an example, consider a window of size $L = 3$, composed

of $e2$, $e3$ and $e4$, with a slot at $e3$. The generalized search path in this case consists of all the paths that share the context $e2, e4$ and branch into all possible vertices at location $e3$. We thus define $P(e_3|e_2;e_4) = \sum_\beta P(e_{3_\beta}|e_2;e_4)$, where each $P(e_{3_\beta}|e_2;e_4)$ is calculated by considering a different path going through the corresponding e_{3_β}. Likewise, we proceed to define $P(e_5|e_2e_3e_4) = \sum_\beta P(e_5|e_2;e_{3_\beta};e_4)$ and so on. In the example of Figure 3.6D, the generalized path is defined over a window of length 4 with one slot at the 3rd location; the context at the other locations is fixed. In Figure 3.6F there are two slots, in locations 2 and 3.

Algorithm 4 Generalization

For a given search path m ; $m \in 1 : N$
for $i = 1, \cdots l(m) - L - 1$ **do**
  slide a context window of size L along the search path from its beginning vertex to its end $l(m)$;
  examine the generalized search paths:
  for $j = i + 1, \cdots, i + L - 2$ **do**
    define a slot at location j;
    define the generalized path consisting of all paths that have identical prefix (at locations i to $j - 1$) and identical suffix (at locations $j + 1$ to $i + L - 1$);
    perform MEX on the generalized path;
  end for
end for
CHOOSE the leading P for all searches performed on each generalized path;
for the leading P define an equivalence class E consisting of all the vertices that appeared in the relevant slot at location j of the generalized path;
rewire graph CREATE a new vertex corresponding to P [Figure 3.6E] and replace the string of vertices it subsumes with the new vertex P on all paths where it occurs. We list here, and in the next rewiring step, only mode A; in mode B the replacement should occur only on the paths for which the new P is declared significant by MEX.

During the pass over the corpus, the list of equivalence sets is updated continuously; new significant patterns are found using the *current* equivalence classes (Figure 3.6(F)). For each set of candidate paths, the algorithm tries to fit one or more equivalence classes from the pool it maintains. Because a constituent can appear in several classes, the algorithm must check different combinations of equivalence

classes. The winner combination is always the largest class for which most of the members are found among the candidate paths in the set (the ratio η between the number of members that have been found among the paths and the total number of members in the equivalence class is compared to a fixed threshold as one of the configuration acceptance criteria). When not all the members appear in the existing set, the algorithm creates a new equivalence class containing only those members that did appear. Thus, as the algorithm processes more and more text, it "bootstraps" itself and enriches the graph structure with new SPs and their accompanying equivalence sets. The recursive nature of this process enables the algorithm to form more and more complex patterns, in a hierarchical manner. The search for patterns and equivalence classes and their incorporation into the graph are repeated until no new significant patterns are found. I estimate the computational complexity of the ADIOS algorithm to increase linearly with the size of the corpus (see section 4.3).

The final lexicon includes those of the original symbols not incorporated into larger units, and root patterns distilled by the algorithm (that is, the patterns that reside on the final graph, at the top level of the hierarchy). Due to the hierarchical process of pattern creation, each pattern is structured as a tree, whose leaves (terminals) are the original members of the lexicon and whose intermediate nodes are other patterns and equivalence classes (Figure 3.7). Note that the hierarchical construction and the resulting tree structure exclude cyclic recursion (loops) of patterns, although recursion may be introduced through pattern matching in a post-processing stage.

The final graph includes as many paths as all the original sentences, but it can also generate many new ones. To generate a sentence from a chosen path in the graph, all its root patterns are traversed. Each recursively encountered pattern is treated as a derivation or parse tree [49]: it is read from top (root) to bottom (terminals) and from left to right, while accruing the terminals (words from the original lexicon) and selecting one member from each encountered equivalence class (Fig-

Algorithm 5 Generalization boosted

For a given search path m ; $m \in 1 : N$
for $i = 1, \cdots l(m) - L - 1$ **do**
  slide a context window of size L along the search path from its beginning vertex to its end $l(m)$;
  for $j = 1, \cdots, K - L - 1$ **do**
    construct generalized search path
    for all do {all slots at locations j, $j = i + 1, \cdots, i + L - 2$}
      consider all possible paths through these slots that start at vertex i and end at vertex $l(m) - L - 1$
      compare the set of all encountered vertices to the list of existing equivalence classes, selecting the one $E(j)$ that has the largest overlap with this set, provided it is larger than a minimum overlap ω (set to 0.65 in all my experiments);
      reduce generalized search path
      for k, $k = i + 1, \cdots, i + L - 2$ and all j, $j = i + 1, \cdots, i + L - 2$ such that $j \neq k$ **do**
        consider the paths going through all the vertices in k that belong to $E(j)$ (if no $E(j)$ is assigned to a particular j, choose the vertex that appears on the original search-path at location j); for all j [Figure 3.6F];
        perform MEX on this reduced generalized path;
        extract the leading P; if the overlap of $E(j) < 1$ define a new equivalence class $E'(j)$ containing only those members that did appear in the set;
        rewire graph
        CREATE a new vertex corresponding to P [Figure 3.6G], and replace the string of vertices subsumed by P with the new vertex P on all paths on which it occurs;
      end for
    end for
  end for
end for

ure 3.7C). Because the equivalence relations only hold in the contexts specified by their parent patterns, the ADIOS representation is inherently safer than grammars that posit globally valid categories (such as "parts of speech" in a natural language). At the same time, because each rewiring of the graph brings closer far-apart units that used to straddle the newly abstracted pattern, the resulting representation can capture long-range structural dependencies among units.

Because patterns can be represented in the form of rewriting rules, which are context-free when Mode A is used (Figure 3.7D) and context-sensitive when Mode B is used (Figure 3.7G), the end product of an ADIOS run constitutes a grammar. As infinite recursion is not implemented in the current version of the algorithm, the representations learned by ADIOS are comparable in expressive power to finite Context Sensitive Grammars. This means that any grammar consisting of context sensitive rules can be loaded into an ADIOS instance (that is, translated into an ADIOS representation), provided that a limit is placed on the number of times each rule is invoked recursively. In learning, the results described in the following section show that this algorithm can acquire, from raw corpora, good operational approximations to those grammars that generate data rich with partially alignable sentences, including unconstrained natural-language data. Complex grammars in which inherent ambiguity [49] is exacerbated by the presence of multiple loops are dealt with effectively by acquiring more patterns.

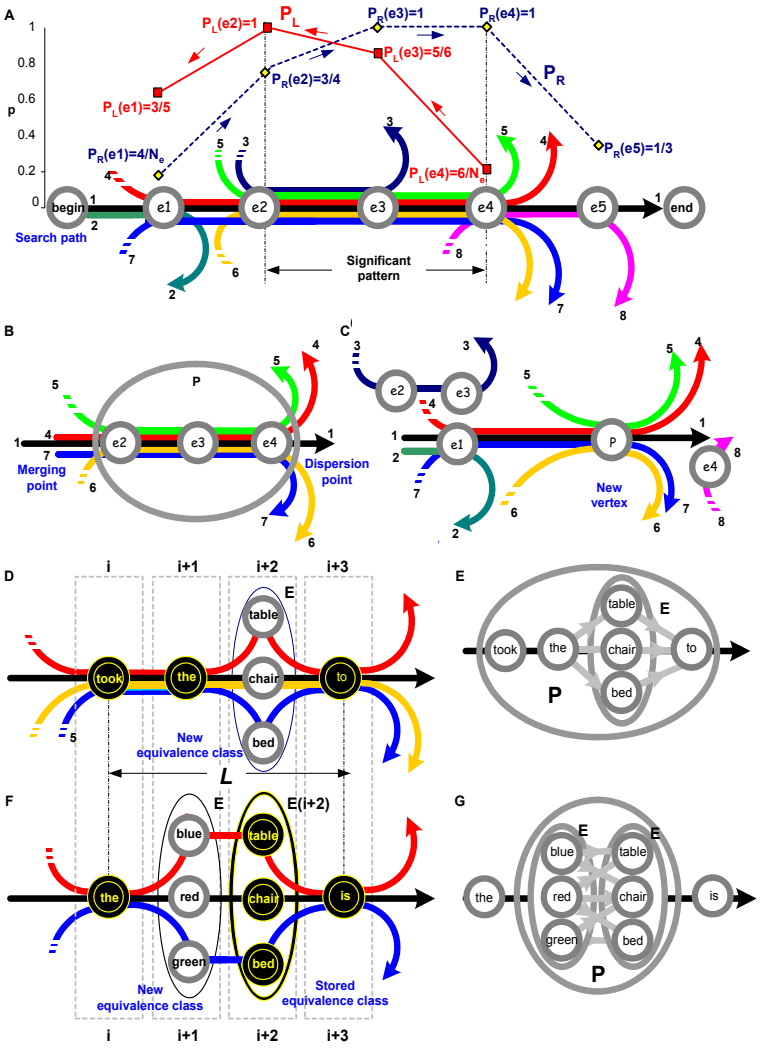

Fig. 3.6: (A), the search path #1 (begin → e1 → . . . → e5 → end) is rendered as a solid black line connecting the special begin and end vertices. Four other paths (#4,#5,#6,#7) join it along the vertices e2,e3,e4, thus forming a bundle that may constitute a significant pattern subject to the MEX criterion described in Section 3.2. Values of P_R and P_L, originating at e1 and e4, respectively, are displayed for the example shown here. (B), a significant pattern (P = e2 → e3 → e4) has been identified. (C), a new vertex is added to the graph, replacing the elements subsumed by P. Paths that belong to sequences not subsumed by it, such as #3 here, are left untouched. (D), the path is generalized: the algorithm picks among the set of path segments encountered in a window of size $L = 4$ those that differ in a single slot and are embedded in a common context (the relevant vertices are marked by open circles). The vertices in this slot form an *equivalence class* E. (E), the original search path is augmented by treating E as a single unit, resulting in the first generalization step (cf. Algorithm 3.2). (F), the just-detected $E(i+2)$ is used to find an additional equivalence class; it is specific to the current common context, thus enhancing the safety of generalization. (G), stacking two equivalence classes leads to further generalization (see section 3.2 for details).

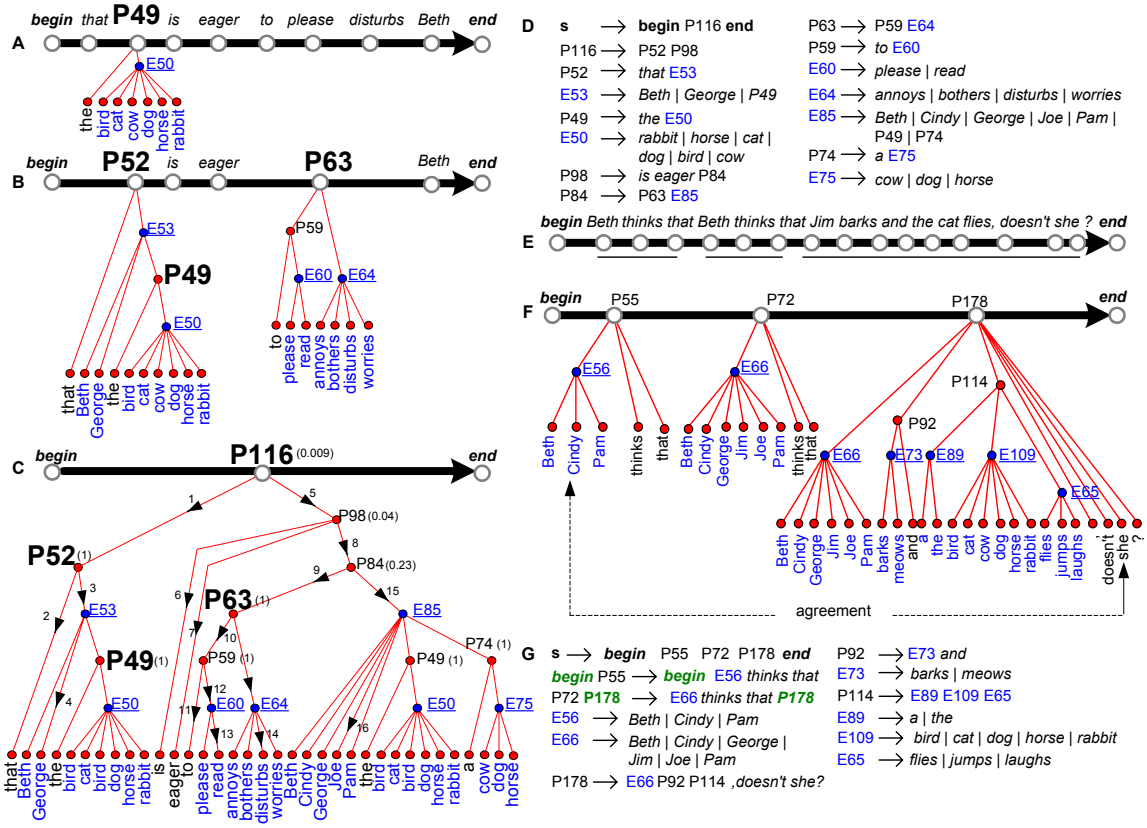

Fig. 3.7: Progressive abstraction of patterns constructs a forest of trees rooted in vertices of the graph (training data generated by a context-free grammar, TA1). (A), Pattern P49, consisting of the terminal the and equivalence class E50 = {bird, cat, cow, dog, horse, rabbit}, is distilled. (B), Further application of the algorithm yields equivalence classes (underlined) such as E64, which contain some verbs. (C), Pattern P116 can generate 896 novel sentences, eight of which appear in the training corpus (the *generalization factor*, 8/896, appears in parentheses). A novel sentence, such as that George is eager to read disturbs Joe, can be read off the leaf level of the tree (numbered arrows indicate traversal order during generation). Pattern P116 is a root pattern, that is, a unit situated on a final path. (D), The set of context-free productions (rewriting rules) that is equivalent to the tree of pattern P116. (E), The initial path through a sentence to which ADIOS was applied in the context-sensitive mode B. (F), The same path after three root patterns (P55, P72 and P178) have been distilled. Note how the two similar but not identical root patterns, P55 and P72, capture the difference between the equivalence classes E56 and E66 (indeed, Beth, for example, is equivalent to Jim in the context of P72, but not of P55). In this manner, ADIOS enforces long-range agreement between E56 and the phrase doesn't she (embedded in P178), and avoids over-generalization. (G), The two context-sensitive rules in this example are [begin P55 ⇒ begin E56 thinks that] and [P72 P178 ⇒ E66 thinks that P178].

4. RESULTS

In this chapter I discuss the performance of ADIOS in different domains. Section 4.1 presents the results in the linguistic domain, where the algorithm has been successfully tested both on artificial-grammar output (section 4.1.1) and on natural-language corpora such as ATIS [47], CHILDES [71], and the Bible [84] (section 4.1.2). In section 4.2 I describe the results from bioinformatics, where the algorithm has been shown to to extract from protein sequences syntactic structures that are highly correlated with the functional properties of these proteins (section 4.2.1), to distinguish between coding and non-coding regions of DNA (section 4.2.2.1) and to extract regulatory motifs in their promoters (section 4.2.2.2). I conclude with an estimate of the computational complexity of ADIOS.

4.1 Language

4.1.1 Artificial Language

4.1.1.1 Computational Grammar Induction

It is reasonable to require that the success of a learning algorithm be measured by the closeness — ideally, identity — of the learned and target grammars, G and G_0, respectively. Unfortunately, even for Context Free Grammars (CFGs), equivalence is undecidable [49]. Moreover, for natural languages G_0 is inaccessible. I thus opt for testing the implementation for generativity[1] as follows. In the artificial-grammar experiments, which start with a target grammar, a teacher instance of the model is

[1] Testing a learned grammar G for *strong generativity*, one comparing structural descriptions (parse trees) it assigns to novel strings to those produced by the target grammar G_0. A *weak generativity* criterion requires merely that G accept novel G_0-grammatical strings as such, and reject the ungrammatical ones. Unsupervised grammar induction algorithms that work from raw data are in principle difficult to test. Any "gold standard" that can be used to test strong generativity, such as the Penn Treebank, invariably reflects its designers' preconceptions about language, which are often controversial among linguists themselves. Thus, even the most conservative treebank-based evaluation metrics such as crossing brackets may present a skewed picture of the system's performance. We A human learner who exhibits perfect weak generativity — that is, who accepts and produces all and only those sentences respectively produced and accepted by the teacher — is, for all practical purposes, perfectly successful, which is why we opt for the so-called weak generativity. I note in this connection that hand-constructed grammars such as ATIS and the treebanks used in testing for strong generativity are liable to produce poor output when used in the generative mode.

first pre-loaded with this grammar (using the one-to-one translation of CFG rules into ADIOS patterns), then used to generate the training corpus $C_{training}$. After training, the learner generates a test corpus $C_{learner}$ and the teacher generates a target corpus C_{target}, the latter containing only novel sentences that do not appear in $C_{training}$. The two corpora, $C_{learner}$ and C_{target}, are then used to calculate *precision* (the proportion of $C_{learner}$ accepted by the teacher) and *recall* (the proportion of C_{target} accepted by the learner). A sentence is accepted if it precisely fits one of the paths in the ADIOS graph (that is, it can be generated by the path). In the natural language experiments, where no target grammar is available, the given corpus is split into two portions, one for training ($C_{training}$) and one for testing (C_{target}), and the same evaluation method is applied, except that precision must in this case be evaluated by an external referee (e.g., by a human subject). This evaluation method is unique (i) because it defines precision and recall more conservatively than is standard in the literature [61], and (ii) because it involves testing both the capability of the learner to *accept* all the grammatical sentences *and* its capability to *generate* only sentences that the teacher would deem grammatical.

I have conducted a series of experiments designed to evaluate the performance of ADIOS in grammar induction (Figure 4.1).

4.1.1.2 Learning a simple CFG

In the first study, I replicated one of the experiments of [3] ("A 2000 Sentences Sample", p.8). The aim of the original experiment was to reconstruct a specific context-free grammar (29 terminals and 7 rules) from a corpus of 2000 sentences using the EMILE 4.1 algorithm. The results of applying the ADIOS algorithm to a 2000-sentence corpus randomly generated from the given context-free grammar are shown in Table 4.1. The algorithm (used in its default Mode A, $\eta = 0.6$, $\alpha = 0.01$, recursion depth set to 15) yielded 28 patterns and 9 equivalence classes, and achieved 100% preci-

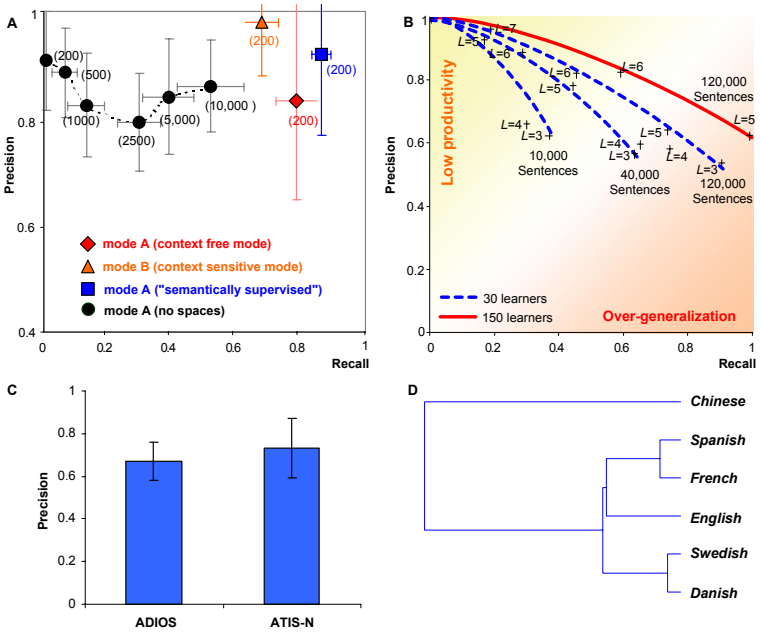

Fig. 4.1: (A), The performance of an ADIOS model trained on extremely small corpora generated by TA1. Optimal combinations of recall and precision (single learner, 30 trials, $\eta = 0.6$, $\alpha = 0.01$, $L = 5$, maximum recursion depth for the teacher here and below set to 10) are shown for four different conditions: (i) the default learning mode A (context free mode, see Box 2); with a 800-sentence training corpus (not shown), both precision and recall reach 90%; (ii) mode B (context-sensitive mode) (iii) a "semantically supervised" mode in which the equivalence classes of the target grammar are made available to the learner ahead of time; cf. evidence on the contribution of extra-linguistic knowledge to language acquisition [74]; (iv) bootstrap mode, starting from a letter level and training on corpora in which all spaces between words are omitted. To maintain a comparable level of performance, the bootstrap mode requires larger corpora (size, shown in parentheses: 200-$10,000$ sentences). (B), Using the ATIS Context-Free Grammar (4592 rules) [47] as the teacher of multiple ADIOS learners. Precision is defined by the mean over learners, while for recall acceptance by one learner suffices. Several corpus sizes, context window widths L and numbers of learners are compared. (C), Output generated by an instance of ADIOS that had been trained on the natural language ATIS-N corpus was judged to be as acceptable to human subjects as sentences from ATIS-N. Acceptability data (mean $\pm$ standard deviation) are from eight subjects. (D), A dendrogram illustrating the relationships among six different natural languages using pattern spectra. We define a pattern spectrum as the histogram of pattern types, whose bins are labeled by sequences such as (T,P) or (E,E,T), E standing for equivalence class, T for tree-terminal (original unit) and P for significant pattern. This plot was generated by applying hierarchical clustering to a table of Euclidean distances among histograms of patterns learned by ADIOS from online multilingual Bible texts [84], each consisting of approximately $31,100$ verses (single learner per language).

sion and 99% recall. In comparison, the EMILE algorithm, as reported in [3], induced 3000-4000 rules (the recall/precision performance of the EMILE algorithm was not stated). Table 4.1 shows a comparison between the induced grammar and its target grammar. The upper part of the table contains the extracted equivalence classes and their target counterparts, demonstrating the ability of ADIOS to identify most of the target classes (except one, E43). The lower part of the table shows that ADIOS distills a set of rules that is larger than the original one (but equivalent to it).

Next, I applied ADIOS to a small CFG and showed that it performs well even when only 200 sentences are used for training, as demonstrated in Figure 4.1A. Tables 4.1 to 4.6 show the performance of an ADIOS model trained on extremely small corpora (200 sentences) generated by the TA1 artificial grammar (listed in Table 4.6). The tables present the recall-precision values (with their standard deviations across 30 different trails) in four different running modes: **Table 4.2**, Mode A (context free); **Table 4.3**, mode B (context-sensitive mode); **Table 4.4**, "semantically supervised" mode, in which the equivalence classes of the target grammar are made available to the learner ahead of time (training in Mode A); **Table 4.5**, bootstrap mode, which starts from a letter-level training corpus in which all spaces between words are omitted (training in Mode A). In the first three experiments, the context-window length was varied while the other parameters were kept fixed ($\eta = 0.6$, $\alpha = 0.01$, corpus size 200). In the bootstrap mode, the algorithm must first segment the sequence of letters into words (applying only the MEX procedure without extracting equivalence classes), and only then use the identified words to extract the grammar. This two-stage process requires a larger corpus to attain a comparable level of performance (up to $10,000$ sentences in this example). Thus, in the last experiment L was kept fixed at 3, ω was lowered to 0.4, and the corpus size ranged from 200 to $10,000$ sentences. Performance was assessed by the F1 measure, defined as $2\cdot$recall$\cdot$precision/(recall + precision). The best recall/precision combinations appear in bold and are plot-

ted in Figure 4.1A. It can be seen that both context free mode and context sensitive mode reach similar F1 levels; however, while the context free mode yields better recall (83% versus 68%) the context sensitive mode gets higher level of precision (98% versus 80%). When semantic information is available to the learner ahead of time, it gives rise to a significant improvement in the learning performance (F1=0.89 versus 0.81), which parallels the documented importance of embodiment cues in language acquisition by children.

4.1.1.3 Learning a complex CFG

Because the ADIOS algorithm is greedy (the best available pattern in each iteration is immediately and irreversibly rewired), the syntax it acquires depends on the order of sentences in the training set. This is expected to affect the learning of a complex CFG, especially if it contains many loops. To assess this dependence and to mitigate it, I train multiple learners on different order-permuted versions of the corpus generated by the teacher. As Figure 4.1B illustrates, for the parameter values explored ($L = \{3, 4, 5, 6\}$; 30 or 150 learners; corpus size between 10,000 and 120,000 sentences), the optimal precision-recall trade-off for learning the ATIS CFG (4592 rules) [47] is obtained with a 150-learner cohort and L between 5 and 6.

Table 4.7 illustrates the recall and precision performance for learning the 4592-rule ATIS Context Free Grammar [47], using different parameter values (L = {3, 4, 5, 6}; 30 or 150 learners; corpus size between 10,000 and 120,000 sentences). Figure 4.2 presents a schematic illustration of the coverage of the target language by multiple learners, for various settings of L.

target grammar	**inferred grammar**
[NP_a] ⇒ John | Mary | the man | the child	E35 ⇒ child | man P34 ⇒ the E35 E37 ⇒ John | Mary | P34
[P] ⇒ with | near | in | from	E54 ⇒ with | near | in | from
[V_i] ⇒ appears | is | seems | looks	E39 ⇒ appears | is | seems | looks
[V_s] ⇒ thinks | hopes | tells | says	E51 ⇒ thinks | hopes | tells | says
[V_t] ⇒ knows | likes | misses | sees	E46 ⇒ knows | likes | misses | sees
[ADV] ⇒ large | small | ugly | beautiful	E49 ⇒ large | small | ugly | beautiful
[NP_p] ⇒ the car | the city | the house | the shop	E43 ⇒ house | shop
[S] ⇒ [NP] [V_i] [ADV] | [NP_a] [VP_a] | [NP_a] [V_s] that [S] [NP] ⇒ [NP_a] | [NP_p] [VP_a] ⇒ [V_t] [NP] | [V_t] [NP] [P] [NP_p]	E69 ⇒ P47 | P59 | P62 | P66 | P67 P40 ⇒ the city P41 ⇒ the car **P36** ⇒ John likes E37 P42 ⇒ the E43 E39 P44 ⇒ the house P45 ⇒ E37 E46 **P47** ⇒ P45 E37 **P48** ⇒ E37 E39 E49 P50 ⇒ E37 E51 that **P52** ⇒ P47 in the shop P53 ⇒ E54 P44 **P55** ⇒ P47 near P40 P56 ⇒ E54 P41 P57 ⇒ E54 P40 **P58** ⇒ P50 P50 P45 the shop **P59** ⇒ P45 the shop **P60** ⇒ P41 E39 E49 **P61** ⇒ P42 E49 **P62** ⇒ P45 P40 **P63** ⇒ P50 P50 P48 P64 ⇒ E54 the shop **P65** ⇒ P50 P62 P64 **P66** ⇒ P45 P44 **P67** ⇒ P45 P41 **P68** ⇒ E69 P53 **P70** ⇒ P38 E49 **P71** ⇒ E69 P57

Tab. 4.1: A comparison between the target grammar of Adriaans and Vervoort (*left*) and the grammar induced by a single ADIOS instance (*right*). Root patterns appear in bold.

Tab. 4.2: TA1 experiment, Mode A (Context-Free)

corpus size	L	recall	precision	F1
200	9	0.3 ± 0.2	0.9 ± 0.1	0.45
200	8	0.4 ± 0.2	0.93 ± 0.09	0.59
200	7	0.6 ± 0.1	0.9 ± 0.1	0.71
200	6	0.7 ± 0.1	0.9 ± 0.2	0.78
200	5	0.78 ± 0.08	0.8 ± 0.2	0.80
200	**4**	$\mathbf{0.83 \pm 0.06}$	$\mathbf{0.8 \pm 0.2}$	**0.81**
200	3	0.84 ± 0.06	0.6 ± 0.2	0.71

Tab. 4.3: TA1 experiment, Mode B (Context-Sensitive)

corpus size	L	recall	precision	F1
200	9	0.5 ± 0.2	0.8 ± 0.1	0.65
200	8	0.6 ± 0.1	0.78 ± 0.09	0.66
200	7	0.61 ± 0.07	0.9 ± 0.2	0.72
200	6	0.6 ± 0.1	0.8 ± 0.2	0.68
200	5	0.61 ± 0.09	0.8 ± 0.1	0.69
200	4	0.69 ± 0.05	0.9 ± 0.1	0.79
200	**3**	$\mathbf{0.68 \pm 0.06}$	$\mathbf{0.98 \pm 0.04}$	**0.80**

Tab. 4.4: TA1 experiment, Mode A, "semantically supervised"

corpus size	L	recall	precision	F1
200	8	0.86 ± 0.06	0.7 ± 0.2	0.80
200	7	0.89 ± 0.04	0.8 ± 0.2	0.84
200	6	0.90 ± 0.04	0.8 ± 0.2	0.85
200	5	0.90 ± 0.03	0.8 ± 0.2	0.83
200	4	0.92 ± 0.03	0.8 ± 0.2	0.83
200	**3**	$\mathbf{0.92 \pm 0.03}$	$\mathbf{0.9 \pm 0.2}$	**0.89**

Tab. 4.5: TA1 experiment, Mode A, "no spaces"

corpus size	L	recall	precision	F1
200	3	0.01 ± 0.01	0.91 ± 0.09	0.01
500	3	0.07 ± 0.04	0.89 ± 0.08	0.12
1000	3	0.13 ± 0.06	0.8 ± 0.1	0.23
2500	3	0.30 ± 0.07	0.79 ± 0.09	0.43
5000	3	0.39 ± 0.08	0.85 ± 0.1	0.53
10000	3	0.5 ± 0.1	0.86 ± 0.09	0.65

Tab. 4.6: The TA1 grammar, consisting of 50 terminals and 28 rules

σ	$\Rightarrow$ s1 | s2 | s3 | s4
s1	$\Rightarrow$ prec np2 vp ptag
s2	$\Rightarrow$ frec np2 vp ftag
s3	$\Rightarrow$ frec iv6 iv55
s4	$\Rightarrow$ *that* np2 iv5 iv6 iv4 np2
np	$\Rightarrow$ art noun | propn
np2	$\Rightarrow$ *the* noun | propn
propn	$\Rightarrow$ p vp2 | p
pp	$\Rightarrow$ p *and* p vp6 | p p *and* p vp6
vp	$\Rightarrow$ iv *and* com
vp2	$\Rightarrow$ *who* tv np
com	$\Rightarrow$ np iv2
rec	$\Rightarrow$ p vp5 *that* rec | p vp5 *that*
frec	$\Rightarrow$ pf vp5 *that* rec
ftag	$\Rightarrow$, *doesn't she ?*
prec	$\Rightarrow$ pp *that* rec
ptag	$\Rightarrow$, *don't they ?*
iv5	$\Rightarrow$ *is* iv5-ex
iv55	$\Rightarrow$ *is* iv55-ex
iv6	$\Rightarrow$ *to* iv6-ex
art	$\Rightarrow$ *the* | *a*
noun	$\Rightarrow$ *cat* | *dog* | *horse* | *cow* | *rabbit* | *bird*
p	$\Rightarrow$ *Joe* | *Beth* | *Jim* | *Cindy* | *Pam* | *George*
pf	$\Rightarrow$ *Beth* | *Cindy* | *Pam*
vp5	$\Rightarrow$ *believes* | *thinks*
vp6	$\Rightarrow$ *believe* | *think*
iv	$\Rightarrow$ *meows* | *barks*
iv2	$\Rightarrow$ *laughs* | *jumps* | *flies*
iv5-ex	$\Rightarrow$ *easy* | *tough* | *eager*
iv55-ex	$\Rightarrow$ *easy* | *tough*
iv6-ex	$\Rightarrow$ *please* | *read*
iv4	$\Rightarrow$ *annoys* | *worries* | *disturbs* | *bothers*
tv	$\Rightarrow$ *scolds* | *loves* | *adores* | *worships*

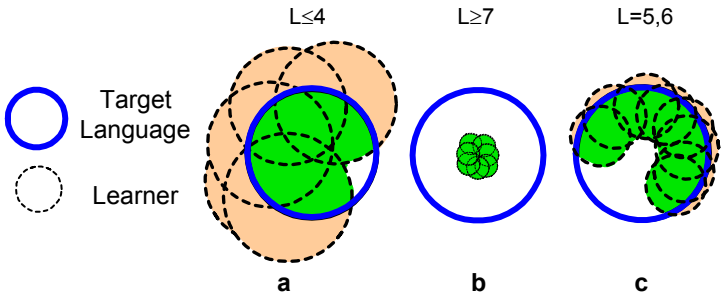

Fig. 4.2: High values of L bring the learners into a low-productivity region where precision is high, but the coverage of the target language is low. For low values of L, the learners tend to over-generalize and thus acquire an inaccurate language that, however, does cover most of the target. The proper balance is achieved by setting L to an intermediate value, so that the learners cover a large portion of the target language, yet remain within the language boundaries.

Tab. 4.7: ATIS-CFG recall and precision

corpus size	No. of learners	L	η	recall	precision	F1
10000	30	3	0.6	0.380	0.623	0.472
10000	30	4	0.6	0.308	0.657	0.420
10000	30	5	0.6	0.180	0.920	0.301
40000	30	3	0.6	0.643	0.568	0.603
40000	30	4	0.6	0.660	0.596	0.627
40000	30	5	0.6	0.456	0.780	0.576
120000	30	3	0.6	0.910	0.538	0.676
120000	30	4	0.6	0.750	0.580	0.654
120000	30	5	0.6	0.747	0.640	0.689
120000	30	6	0.6	0.465	0.818	0.593
120000	150	3	0.6	1.000	0.538	0.700
120000	150	4	0.6	1.000	0.580	0.734
120000	150	5	0.6	1.000	0.640	0.780
120000	150	6	0.6	0.600	0.820	0.693
120000	150	7	0.6	0.230	0.970	0.372

4.1.2 Natural Language

4.1.2.1 Generativity of the learned natural language grammar

To test the ability of ADIOS to generate acceptable novel sentences after learning from a natural language corpus, I trained it on $12,700$ sentences from ATIS-2 (a natural language corpus of size $13,043$ [47]) and tested its recall level on the 343 remaining sentences. The small size of the training corpus results in a relatively low recall of 40% (under the strict definition that requires an exact match). Figure 4.1C compares the acceptability of ADIOS-generated sentences with original sentences from the ATIS-2 corpus. Notably, the output generated by ADIOS is on the average as acceptable to human subjects as the original corpus sentences. The human-judged precision ($\approx 70\%$, as shown in the plot) is remarkable; for comparison, the ATIS-CFG grammar, hand-constructed to fit the ATIS-2 corpus (with recall of 45% on same data) produces over 99% ungrammatical sentences when used in a generative fashion.

Because the target grammar of a natural language is inaccessible, precision must be evaluated by human subjects (referees), while recall can be evaluated by the same method described in the section *Language: computational grammar induction*. In the present experiment, the ADIOS algorithm was trained on the ATIS-2 natural language corpus. This corpus contains $13,043$ sentences of natural speech, in the Air Travel Information System (ATIS) domain. The ADIOS algorithm was trained on $12,700$ sentences ($C_{training}$); the remaining 343 sentences were used to evaluate recall (C_{target}). Two groups of learners (30, 150) were trained ($\eta = 0.6$, $\alpha = 0.01$, $L = 5$) on different, order-permuted, versions of the corpus (several representative acquired patterns appear in Figure 4.3 along with their *generalization factors*). After training, each learner generated 100 sentences, which were then placed together into a single corpus (the $C_{learners}$ test-corpus). Precision of the ADIOS representation (mean $\pm$ std dev) was estimated by having eight human subjects judge the acceptability of 20 sentences taken from $C_{learners}$ and of 20 sentences taken from the original ATIS-2 corpus ($C_{training}$).

The subjects had no indication which sentence belonged to which corpus; the sentences appeared in a random order and each subject judged a different set of sentences. Altogether, 320 sentences were evaluated. The original ATIS-2 corpus was scored at $70 \pm 20\%$ precision while the ADIOS-generated sentences attained $67 \pm 7\%$ precision. Recall was calculated using the C_{target} corpus. Sets of 30 and 150 learners achieved 32% and 40.5% recall respectively. Interestingly, this result (30% recall and 70% precision), which was obtained from a natural language corpus of 10,000 sentences, is predicted by the theoretical curve that appears in Figure 4.1B, which was derived by training ADIOS on 10,000 artificial sentences.

4.1.2.2 Select syntactic examples

The present work is data- rather than theory-driven in that I refrain from making *a priori* assumptions about the kind of "grammar" that the algorithm is expected to produce (cf. the quote from Langacker [65] in section 1.1). Clearly, however, the recursively structured, parameterized patterns learned by ADIOS, and their use in processing and generating novel sentences, do resemble certain features of some extensively studied syntactic formalisms. A few select syntactic examples are illustrated in Figures 4.4 4.5 4.6 4.7.

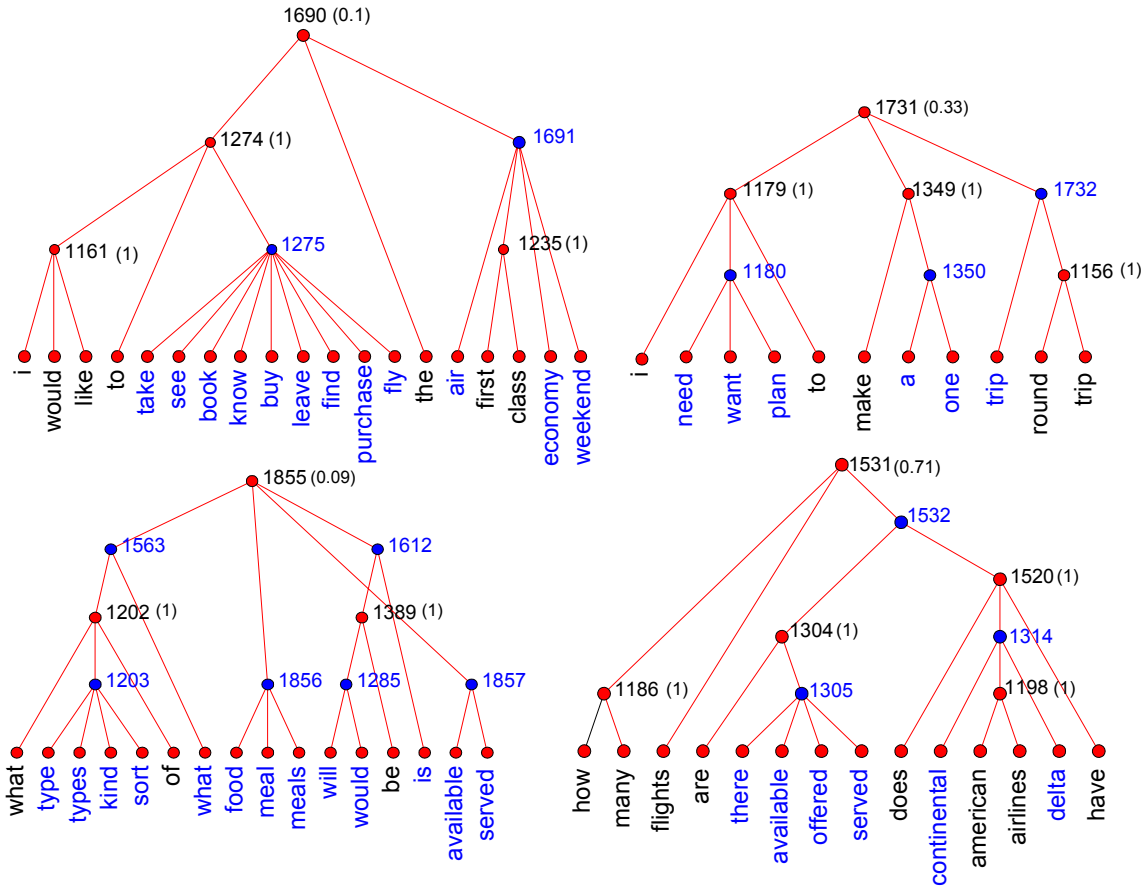

Fig. 4.3: Four simple patterns extracted from the ATIS natural language corpus. Some of the sentences that can be described/generated by patterns #1690, #1731, #1855 and #1531 are: I would like to book the first class; I plan to make a round trip; what kind of food would be served ; how many flights does continental have . None of these sentences appear in the training data, illustrating the ability of ADIOS to generalize. The numbers in parentheses denote the generalization factors of the patterns and their components (e.g., pattern #1690 generates 90% new strings, while pattern #1731 generates 66% new strings).

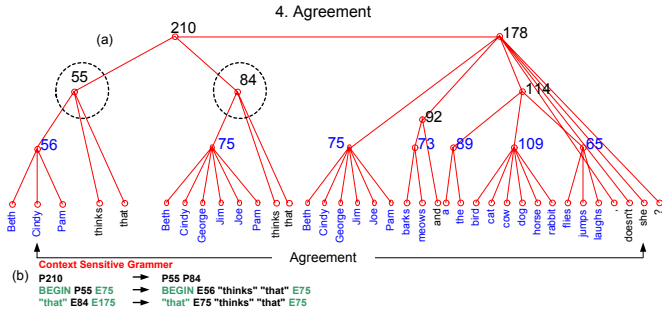

Fig. 4.4: a: because ADIOS does not rewire all the occurrences of a specific pattern, but only those that share the same context, its power may be comparable to that of Context Sensitive Grammars. In this example, equivalence class #75 is not extended to subsume the subject position, because that position appears in a different context (immediately to the right of the symbol begin). Thus, long-range agreement is enforced and over-generalization prevented. *b:* the context-sensitive rules corresponding to pattern #210.

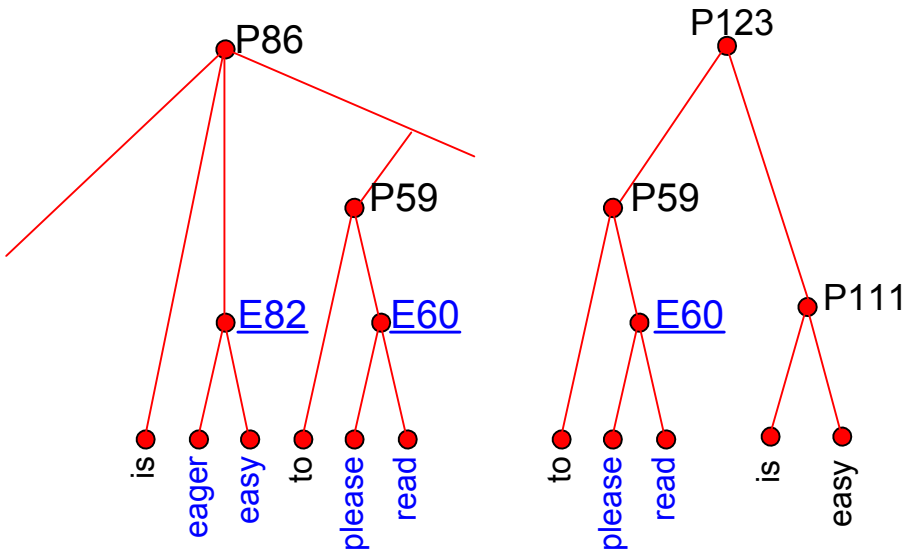

Fig. 4.5: Left: trained on sentences exemplifying "tough movement", ADIOS forms patterns that allow it to represent the correct phrases (is easy to read, is easy to please, is eager to read, is eager to please, to read is easy and to please is easy), but not the incorrect ones (*to read is eager or *to please is eager).

4.1.2.3 *Meta analysis of languages other than English*

Languages other than English: Applying ADIOS to the Parallel Bible [84] an online multilingual Bible texts each consisting of approximately $31,100$ verses (single learner

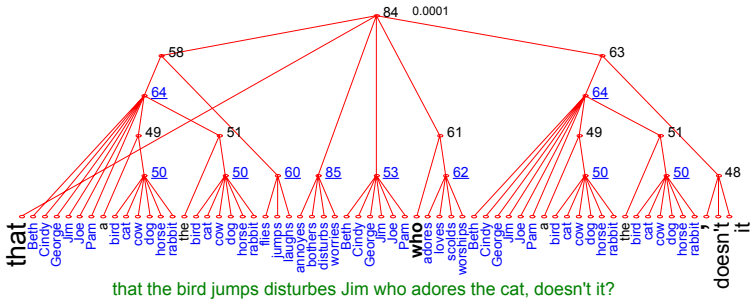

Fig. 4.6: a pattern (presented in a tree form), capturing a long range dependency. This and other examples here were distilled from a 400-sentence corpus generated by TA1.

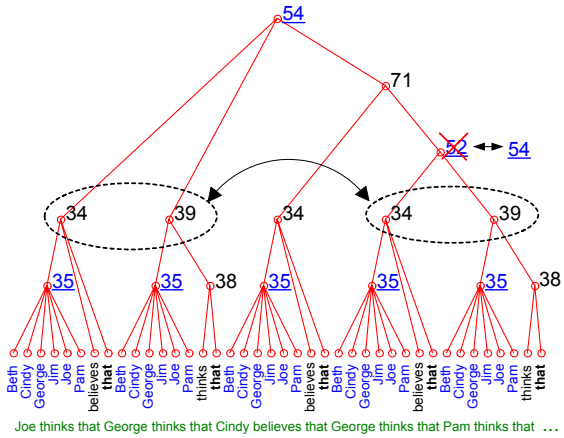

Fig. 4.7: the ADIOS pattern representation facilitates the detection of recursive structure, exemplified here by the correspondence between equivalence classes #52 and #54.

per language). I compared six different languages through a meta-analysis of their respective ADIOS grammars.

To visualize these typological relationships, I considered the pattern spectrum representation, defined as follows. I first listed all the significant patterns extracted from the data during the application of the ADIOS algorithm. Each of these consists of elements that belong to one of three classes: patterns (P), equivalence classes (E), and original words or terminals (T) of the tree representation. I next computed the proportions of patterns that are described in terms of these three classes as TT,

TE, TP, and so on, as shown in Figure 4.8. Thus, each language is represented by a single vector whose elements are the relative frequencies of the pattern classes in that language. All the vector elements sum to one. Comparing the spectra of the six languages, I derived a dendrogram representation of the relative syntactic proximity between them, which is shown in Figure 4.1D. This plot was generated by applying a hierarchical clustering (Matlab procedure LINKAGE, default parameters) to a table of Euclidean distances among histograms of patterns learned by ADIOS It corresponds well to the expected pattern of typological relationships suggested by classical linguistic analysis [38].

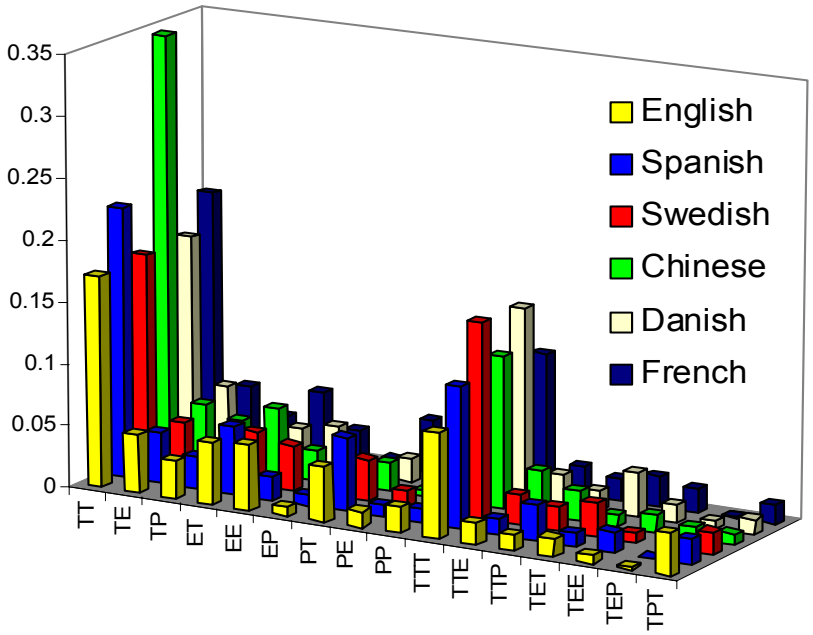

Fig. 4.8: A comparison of pattern spectra obtained from six languages (Chinese, English, Danish, French, Spanish and Swedish). The extraction of patterns was based on the same corpus, the Bible (66 books containing 33K sentences), in its six versions, available online at http://www.umiacs.umd.edu/ resnik/parallel/bible.html. It can be seen that natural languages have a relatively large percentage of patterns that fall into TT and TTT categories (known as collocations).

L1	**L2**
$P1 \Rightarrow pel - \mathbf{X} - rud$	$P1 \Rightarrow pel - \mathbf{X} - jic$
$P2 \Rightarrow vot - \mathbf{X} - jic$	$P1 \Rightarrow vot - \mathbf{X} - tood$
$P3 \Rightarrow dak - \mathbf{X} - tood$	$P1 \Rightarrow dak - \mathbf{X} - rud$

X(2)={wadim, kicey}
X(6)={wadim, kicey, puser, fengle, coomo, loga}
X(12)={wadim, kicey, puser, fengle, coomo, loga, gople, taspu, hiftam ,benez, vamey, skiger}
X(24)={wadim, kicey, puser, fengle, coomo, loga, gople, taspu, hiftam ,benez, vamey, skiger, benez, gensim, feenam, laeljeen,chila, roosa, plizet, balip, malsig, suleb, nilbo,wiffle}

Tab. 4.8: Definition of the L1 and L2 languages.

4.1.2.4 Psycholinguistics

4.1.2.4.1 Learning "nonadjacent dependencies" Gómez [35] showed that the ability of subjects to learn an artificial language L1 of the form $\{aXd, bXe, cXf\}$, as measured by their ability to distinguish it implicitly from L2=$\{aXe, bXf, cXd\}$, depends on the amount of variation introduced at X (symbols a through f here stand for 3-letter nonsense words, whereas X denotes a slot in which a subset of 2-24 other nonsense words may appear). Within the ADIOS framework, these non-adjacent dependencies translate into patterns with embedded equivalence classes. I replicated the Gómez study by training ADIOS on 432 strings from L1 (30 learners, $|X| = 2, 6, 12, 24$, $\eta = 0.6$, $\alpha = 0.01$). Training with the context window parameter L set to 3 resulted in performance levels (rejection rate of patterns outside of the learned language) that increased monotonically with $|X|$, in correspondence with the human behavior. Interestingly, when trained with $L = 4$, *adios* reaches perfect performance in this task.

The two languages used in [35], L1 and L2, are defined in Table 4.8. *Pel, vot, dak, tood* are all nonsense words that form three-element sequences, in whose middle slot, denoted by X, a subset of between 2 and 24 other nonsense words may appear. In the ADIOS terms, X thus stands for an equivalence class with 2-24 elements. The Gómez study was reproduced by training ADIOS on 432 strings from L1, using 30 learners and various sizes of X. Performance was evaluated in the same manner as

in the Gómez study. The test set consisted of 12 strings: 6 from L1 (which should be accepted) and 6 from L2 (which should be rejected). The results are as follows: when L is set to 3 ($\eta = 0.6$, $\alpha = 0.01$), and $|X|$ is set to $2, 6, 12, 24$ elements, ADIOS accepts all the sentences of L1 while rejecting $14 \pm 27\%$, $50 \pm 17\%$, $86 \pm 14\%$, $82 \pm 17\%$ sentences of L2, respectively. Performance level increases monotonically with $|X|$, in accordance with human data. Training with $L = 4$ yielded 100% acceptance rate for L1 and 100% rejection rate for L2, irrespectively of $|X|$, indicating a perfect ability of the algorithm to capture the non-adjacent dependency rule with the proper choice of parameters.

4.1.2.4.2 Grammaticality Judgments A single instance of ADIOS was trained on the CHILDES [71] corpus, using sentences spoken by parents to three year old children. It was then subjected to five grammaticality judgment tests. One of these, the Göteborg multiple-choice ESL (English as Second Language) test, consists of 100 sentences, each containing an open slot; for each sentence, the subject selects one word from a list of three choices, so as to complete the sentence grammatically. In this test, ADIOS scored at 60%, which is the average score for 9th grade ESL students.

I have assessed the ability of the ADIOS model to deal with novel inputs[2] by introducing an *input module* (described below). After training on transcribed speech directed at children (a corpus of 300,000 sentences with 1.3 million words, taken from the CHILDES collection [71]), the input module was subjected to grammaticality judgment tests, in the form of multiple choice questions. The algorithm[3] identified 3400 patterns and 3200 equivalence classes. The input module was used to process novel sentences by forming their distributed representations in terms of activities of existing patterns (a similar approach had been proposed for novel object and scene representation in vision [27]). These values, which supported grammaticality judgment, were computed by propagating activation from bottom (the terminals) to top

[2] Including sentences with novel vocabulary items that are not fully represented by the trained system.
[3] An earlier version of ADIOS [93], which did not use the full conditional probability matrix of eq. 3.1.

(the patterns). The initial activities a_j of the terminals e_j were calculated given a stimulus $s_1, \ldots, s_k$ as follows:

$$a_j = \max_{l=1..k} \left\{ P(s_l, e_j) \log \frac{P(s_l, e_j)}{P(s_l)P(e_j)} \right\} \tag{4.1}$$

where $P(s_l, e_j)$ is the joint probability of s_l and e_j appearing in the same equivalence class, and $P(s_l)$ and $P(e_j)$ are the probabilities of s_l and e_j appearing in any equivalence class. For an equivalence class, the value propagated upward was the strongest non-zero activation of its members; for a pattern, it was the average weight of the children nodes, on the condition that all the children were activated by adjacent inputs. Activity propagation continued until it reached the top nodes of the pattern lattice. When this algorithm encounters a novel word, all the members of the terminal equivalence class contribute a value of $\epsilon = 0.01$, which is then propagated upward as before. This enables the model to make an educated guess as to the meaning of the unfamiliar word, by considering the patterns that become active. Figure 4.9 shows the activation of a pattern (#185) by a sentence that contains a word in a novel context (new), as well as other words never before encountered in any context (Linda, Paul).

I assessed this approach by subjecting a single instance of ADIOS to five different grammaticality judgment tests reported in the literature [68, 67, 4, 75]; see Figure 4.10 (left). The results of one such test, used in English as Second Language (ESL) classes, are described below. This test has been administered in Göteborg (Sweden) to more than $10,000$ upper secondary levels students (that is, children who typically had 9 years of school, but only 6-7 years of English). The test consists of 100 three-choice questions (Table 4.9), with 65% being the average score for the population mentioned. For each of the three choices in a given question, my algorithm provided a grammaticality score. The choice with the highest score was declared the

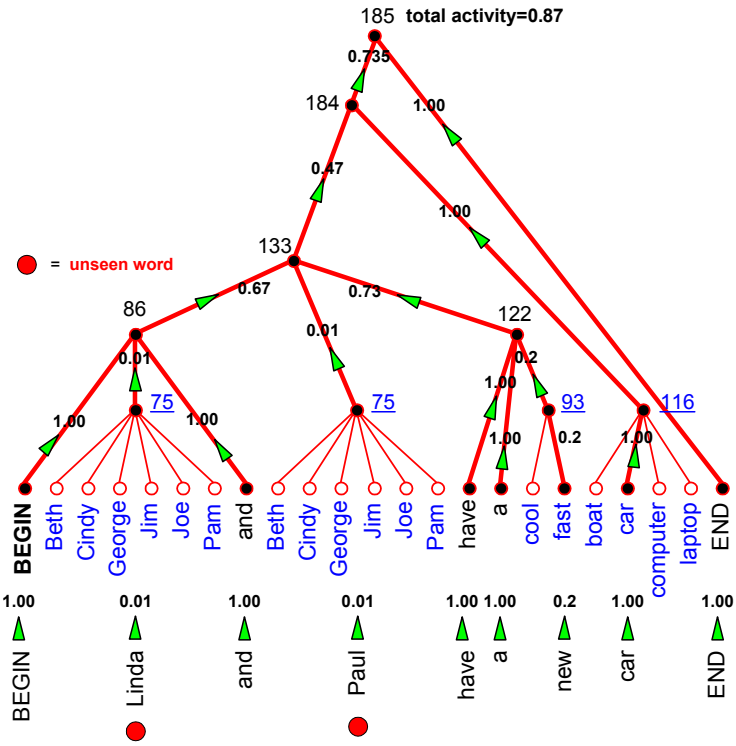

Fig. 4.9: An active pattern responding to the partially novel input Linda and Paul have a new car. Leaf activation, which is proportional to the mutual information between input words and various members of the equivalence classes, is propagated upwards by taking the average at each junction.

winner; if two choices received the same top score, the answer was "don't know". The algorithm's performance is plotted in Figure 4.10 (right) against the size of the CHILDES training set. Over the course of training, the proportion of questions that received a definite answer grew (red bars), while the proportion of correct answers remained around 60% (blue curve); compare this to the 45% precision with 20% recall achieved by a straightforward bi-gram benchmark.[4]

sentence	*choice 1*	*choice 2*	*choice 3*
The pilot __ look worried.	isn't	doesn't	don't
She asked me __ at once.	come	to come	coming
The tickets have been paid for, so you __ not worry.	may	dare	need
We've gone slightly __ course.	of	off	from

Tab. 4.9: Sample questions from a multiple-choice test used in ESL instruction in Göteborg, Sweden. A score $< 50\%$ in this 100-question test (available online) is considered pre-intermediate, $50 - 70\%$ intermediate, and a score $> 70\%$ advanced.

		ADIOS		bi-gram	
benchmark	*#item*	*correct*	*answered*	*correct*	*answered*
Linebarger, Schwartz and Saffran, 1983	25	65%	65%	42%	92%
Lawrence, Giles and Fong, 2000	70	59%	73%	38%	63%
Allen and Seidenberg, 1999	10	83%	60%	40%	50%
Martin and Miller, 2002	10	75%	80%	67%	60%
Goteborg/ESL	100	58%	57%	45%	20%
	215	61%	64%	43%	46%

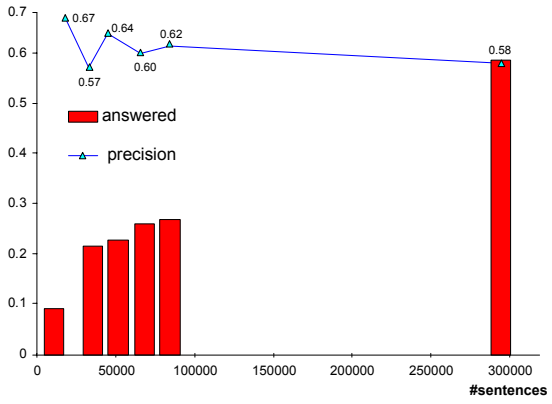

Fig. 4.10: Left: the results of several grammaticality tests reported in the literature. *Right:* a summary of the performance of ADIOS in the Göteborg ESL test, plotted against the number of sentences (paths) scanned during training (red bars: recall; blue rectangles: precision).

[4] Chance performance in this test is 33%. We note that the corpus used here was too small to train an n-gram model for $n > 2$; thus, my algorithm effectively overcomes the problem of sparse data by putting the available data to a better use.

4.1.2.5 Segmentation

To demonstrate the ability of MEX to find patterns in texts, I applied it to an artificial problem of text segmentation. I started by transforming the text of *Alice in the Wonderland* [13] as follows: each paragraph of the book was represented as a long string of letters, omitting spaces among words and any other punctuation symbols, and downcasing capital letters. The challenge was to retrieve the correct words. The text has 809 paragraphs. They were all loaded onto the graph spanned by 26 vertices corresponding to the 26 letters of the alphabet. MEX was then applied to the graph, searching it along 809 search paths.

I started out by setting $\eta = 0.8$ and $\alpha = 0.001$ and running MEX iteratively until it ceased discovering any further motifs. The last set of motifs acquired for each path I call 'root-motifs' because they include within their tree structures all the previously extracted patterns. I then analyzed the results as shown in Table4.10. I compared the text of the first paragraph (the input to the process) to the outcome of the iterative application of MEX at the level of $\alpha = 0.001$.[5] The results of this analysis can be fed back to MEX with higher α values, leading to even better performance.

Let us define an error measure for estimating the significance of such results. Referring to the display of Figure 4.10, note that a space stands for an edge connecting two vertices on the search-path. In Table 4.10A, the vertices are the 26 letters, thus we see a single space between every two letters. Once motifs are observed and turned into vertices, as seen in the following Table 4.10B, the number of spaces (edges visited on the search-path) decreases. Some of them are proper spaces among words, while the others are 'wrong', i.e., spaces that do not actually occur between proper words, as is the case when single letters, or fractions of words, fail to be fused into

[5] Applying the same procedure to a text of 1 million random digits of π we find no motifs at this level of α.

words. Thus, I define the error

$$E_S = \frac{\text{number of wrong spaces in final text}}{\text{number of letters in original text}} \tag{4.2}$$

where the denominator is equal to the number of spaces in the initial text (as in Figure 4.10A).

Table 4.10B shows that there are still quite a few word fractions that were not fused into correct words. This can be remedied by reapplying MEX to the $\alpha = 0.001$ result using a higher value of α. By iterating this procedure, the results can be improved. Measuring E_S, I found the values 0.10 ± 0.02 for $\alpha = 0.01$, decreasing to 0.08 ± 0.03 for $\alpha = 0.1$ and 0.07 ± 0.03 for $\alpha = 0.5$. In other words, the algorithm is able to fuse well over 90% of the spaces in the input. The large values of α may sound alarming; however, they make sense, as will be explained below.

Why does MEX work? MEX is designed to find patterns such that different paths converge onto them (at the beginning of the pattern) and diverge from them (at the end). The D_R and D_L criteria seek in both cases a divergence, once moving to the right and once to the left. Clearly, the motifs themselves are over-represented within the given local context where they are found. However, over-representation is not the criterion for their extraction. To illustrate this point, I present the statistics of the first words in the text of *Alice* in Figure 4.11. Clearly 'alice' is over-represented, but so is also 'alicew' and 'alicewas' and 'was' etc. It is the divergence occurring at the end of 'alice' that does the job of separating it from 'w' or from 'was'. Thus, after 'alice', the 397 paths that contain this word separate into M=24 directions, corresponding to almost all the letters of the alphabet, with 48 paths continuing to 'w'.

This brings to mind a different formulation for the significance of the end (and, by analogy, the beginning) of a pattern. Consider the following two contrasting hypotheses: random continuation [H_1': $D_R(e_i; e_j) = 1/M$], where M is the degree of

(A)

```
a l i c e w a s b e g i n n i n g t o g e t v e r y t i r e d o f s
i t t i n g b y h e r s i s t e r o n t h e b a n k a n d o f h a v
i n g n o t h i n g t o d o o n c e o r t w i c e s h e h a d p e e
p e d i n t o t h e b o o k h e r s i s t e r w a s r e a d i n g b
u t i t h a d n o p i c t u r e s o r c o n v e r s a t i o n s i n
i t a n d w h a t i s t h e u s e o f a b o o k t h o u g h t a l i
c e w i t h o u t p i c t u r e s o r c o n v e r s a t i o n
```

(B)

```
alice was begin n ing toget very t i re do f sitting b y hersister
onthe b an k and of ha v ing no thing to do on c eortw i ce shehad
p ee p ed in tothe b ook hersister was reading but it hadno p i c
t u re s or conversation s in it and what is the us e of a b ook t
hought alice with out p i c t u re s or conversation
```

(C)

```
alice was beginning toget very tiredof sitting b y hersister onthe
b an k and of ha v ing no thing to do on c eortw i ce shehad p ee
p ed in tothe b ook hersister was reading but it hadno picture s or
conversation s in it and what is the us e of a b ook thought alice
with out picture s or conversation
```

(D)

```
alice was beginning toget very tiredof sitting b y hersister onthe
bank and of ha v ing nothing to do on c eortw i ce shehad p ee p ed
in tothe b ook hersister was reading but it hadno picture s or conv
ersation s in it and what is the us e of a b ook thoughtalice with
out picture s or conversation
```

(E)

```
alicewas beginning toget very tiredof sitting by hersister onthe ba
nk and of having nothing to do onceortwice shehad peep ed intothe b
ook hersister was reading but it hadno picture s or conversation s
in it and what is theuseof ab ook thoughtalice without picture s or
conversation
```

Tab. 4.10: The first paragraph of Alice in Wonderland. *(A)* representation of letters only. *(B)* Results of applying MEX to the search-path representing this paragraph, with $\alpha = 0.001$. Motifs are being grouped together. *(C)* Recursive application with $\alpha = 0.01$ to the same path. Further recursive applications shown here are *(D)* $\alpha = 0.1$ and *(E)* $\alpha = 0.5$.

L	a	l	i	c	e	w	a	s
a	8770							
l	1046	4704						
i	468	912	7486					
c	397	401	637	2382				
e	397	397	488	703	13545			
w	48	48	51	66	579	2671		
a	21	21	21	23	192	624	8770	
s	17	17	17	19	142	377	964	6492
b	2	2	2	2	5	10	14	63
e	2	2	2	2	4	6	9	24
g	2	2	2	2	4	5	5	10

P_R	a	l	i	c	e	w	a	s
a	0.08							
l	0.12	0.043						
i	0.45	0.19	0.069					
c	0.85	0.44	0.085	0.022				
e	1	0.99	0.77	0.3	0.12			
w	0.12	0.12	0.1	0.094	0.043	0.024		
a	0.44	0.44	0.41	0.35	0.33	0.23	0.08	
s	0.81	0.81	0.81	0.83	0.74	0.6	0.11	0.059
b	0.12	0.12	0.12	0.11	0.035	0.027	0.015	0.0097
e	1	1	1	1	0.8	0.6	0.64	0.38
g	1	1	1	1	1	0.83	0.56	0.42

M	a	l	i	c	e	w	a	s
a	26							
l	20	26						
i	9	23	25					
c	1	24	23	15				
e	24	21	23	27	27			
w	5	11	18	16	24	25		
a	3	11	8	10	24	24	26	
s	10	8	7	12	23	22	26	25
b	1	1	1	1	2	5	5	18
e	1	1	1	1	1	2	5	13
g	1	1	1	1	1	1	2	7

Fig. 4.11: Values of L (number of paths), P_R (right moving probabilities) and M (number of possible divergence directions) for paths starting with the letters specified on top along directions specified on the left. This example is taken from the first few words in the text of *Alice in the Wonderland*. Note the natural boundaries occurring at the end of the highlighted motif 'alice'.

divergence observed at e_{j-1}, and coherent continuation [H_0: $D_R(e_i; e_j) \geq \eta$].[6] The decision whether to accept H_1' instead of H_0 is made if the value of the probability distribution for H_1' is larger than that of H_0. The significance parameter α is thus determined in each case as the left-handed tail of the probability distribution defining H_0. The statistics of Figure 4.11 fits very well the H_1' hypothesis of random continuation, with significance that is much lower than the $\alpha = 0.001$ limit set at the starting stage of MEX. Iterated application of MEX changes the picture. After each iteration, the newly found patterns, or motifs, are added as vertices to the graph. But, at the same time, the connectivity of the graph becomes sparse, i.e., the number of options for divergence decreases. In this case, the comparison of H_1' with H_0 calls for increasing values of α. Computationally, it is expensive to compare H_0 with H_1' for each endpoint or beginning of a candidate pattern. I prefer therefore to use the simpler approach of setting an overall α value, and proceed by increasing it gradually.

4.1.2.6 Entropy reduction

Using the extracted motifs, together with their appearance probability in the text, one can ask for the results of the Shannon game: guessing the next letter after being given a string of letters in some text. The relevant uncertainty is represented by the entropy rate of the problem. For a given set of motifs, or words, the entropy rate per letter is $\mathcal{H}_w/\bar{s}$, where $\mathcal{H}_w$ is the entropy calculated for the known words, and $\bar{s}$ is the average word length.

The *Alice* exercise has been studied by Redlich [83] using an algorithm based on reducing entropy rate per letter. He was able to reduce it from 4.16 bits for the initial state (Figure 4.10), to 2.35 bits. The MEX algorithm also causes entropy reduction. It surpasses Redlich's results if, after reaching the $\alpha = 0.5$ level of MEX, it is allowed to make another step, in which no significance constraints are imposed (that is, every

[6] Yet another possible criterion is to distinguish randomness from order by evaluating the entropy of all possible next steps. The decreases and increases of entropy are analogous to those of M in Figure 4.11.

drop in D_R or D_L is accepted). This reduces the entropy per letter down to the level of 2.2 bits. All these values are still higher than 2.17 bits, the answer one gets if one considers the original words of the *Alice* text as the appropriate elements. Running MEX with $\eta = 1$, one gets an even better entropy reduction, reaching a final value of 1.96 bits, i.e., surpassing the compression possible by using the actual words.

An additional exercise I carried out was to run MEX on a scrambled version of *Alice*. The text has been scrambled at the level of words, and then turned into a continuous string of letters, as in the previous experiment. The scrambled text is guaranteed to have an entropy per letter of 2.17 bits, whereas the unscrambled text has a lower entropy, whose correct value is unknown (it is affected, e.g., by collocations). In this case, I found MEX to lead to a decrease in entropy from 2.81 for $\eta = 0.8$ and $\alpha = 0.001$ through 2.64 for $\alpha = 0.1$, down to to 2.43 bits at the final stage. This leads to two observations: first, MEX application reduces entropy but does not reach the lowest value possible, and, second, the collocations in the unscrambled text are responsible for the further reduction by 0.23 bits (from 2.43 to 2.2).

4.1.2.7 Data Compression

After allowing MEX to iterate, one ends up with a graph that represents the original text in terms of root-patterns that are, in turn, describable as trees of patterns, whose leaves are the letters of the text. One may ask whether this MEX-graph version of the text is a compressed representation of the initial data. The answer is affirmative. I measured the compression ratio by evaluating the decrease in the physical computer memory required to represent the data. In the *Alice* exercise I reached, for $\eta = 0.8$ stopping at $\alpha = 0.5$, a compression ratio of 0.378. This is slightly better than the compression ratio 0.382 obtained by the ZIP algorithm [64] applied to the original *Alice* text. The compression ratio of 0.378 was the optimal result I have obtained with MEX. Continuing with $\eta = 0.8$ beyond $\alpha = 0.5$ leads to an increase in the compression

ratio. Running with $\eta = 1$ leads to slightly higher results (best compression ratio 0.385).

It should be emphasized that the MEX algorithm has not been optimized for compression, or for entropy reduction. Yet it has achieved impressive results on these two fronts, just because it has succeeded in picking out important structures in the text.

4.1.2.8 Language modeling

A good language model, capable of reliably predicting the next word from its predecessors, would be a key component in many natural language processing applications, such as speech recognition, optical character recognition, machine translation, spelling correction, etc. The two main approaches to language modeling are statistical and grammar-based. Statistical methods aim to determine the likelihood of a word sequence by making inference from word order data that are available in the form of a training corpus. The main disadvantages of the purely statistical approach are the extremely large quantity of data needed to train realistically complex models (i.e., n-grams for $n \geq 3$), and the low rate of return in terms of performance improvement on the investment in model complexity. As Goodman [37, p.48] summarized the situation in a 2001 review, "Practical word error rate reductions are hopeless."

In contrast to the statistical approach, pure grammar-based methods attempt to capture the structure of word sequences using a set of syntactic rules — typically, a hand-coded Context Free Grammar. Here, the problems stem from the extreme difficulty of creating grammars that have both good precision (i.e., produce acceptable sentences), and good recall (i.e., accept sentences known to be grammatical). Quoting again from [37, p.42], "...in practice, a grammar broad coverage enough to parse almost all grammatical sentences is broad coverage enough to parse many ungrammatical sentences as well, reducing the information it provides." In addition, grammars are brittle: people often deviate from the rules of syntax, especially in spoken language.

Combining statistical and structural techniques may be a promising way out of this predicament. In particular, research into the use of Stochastic Context-Free Grammars [56] is leading to a renewed progress in language modeling. Statistical and syntactic cues can be combined "on the fly" by having the model estimate the n-gram statistics simultaneously with the probabilities of possible left-to-right parse trees of

the input sentence (e.g., [16] demonstrated an 11% performance improvement relative to a baseline trigram model, while [14] achieved an even larger 24% improvement). In the present study, I evaluated the grammars acquired by a previously reported algorithm (ADIOS, [90, 91]), by first constructing probabilistic language models based on those grammars, then measuring the perplexity[7] and the coverage of the models.

Language models are evaluated on the basis of their ability to predict the next word in a sequence, using sentences taken from a previously unseen text. Models that result in a high average word probability — that is, low perplexity — are considered superior. As mentioned above, standard statistical language models, such as those based on estimated n-gram probabilities, are problematic, for two reasons: (1) probability estimates for rare or unseen events are unreliable, and (2) low-n models fail to capture long-range dependencies between words. The experiments described below show that the grammar learned by ADIOS can be used to construct a simple, yet effective *Structured Statistical Language Model* [37, 15]. Because the patterns learned by ADIOS generalize well, and because they capture long range dependencies, the resulting SSLM achieves an improvement in perplexity on the ATIS corpora over state-of-the-art models, despite requiring much less training data.

Consider an l-word sentence $W = w_1 \ldots w_l$, and the parse trees induced over it by the learned grammar; note that a parse tree may be complete ($T_{1,l}$, spanning the entire sentence), or partial ($T_{i,j}$; $1 \leq i < j \leq l$). My goal is to assign the probability $p(w_{k+1}|T_{1,k})$ to w_{k+1} which is the next word in the prefix $(w_1, \ldots, w_k)$ and the probabilities $p(w_{k+1}|T_{k-n,k})$ to each one of the partial derivations $T_{k-n,k}$, where $1 \leq n < k$, $1 \leq k \leq l$. $p(w_{k+1}|T_{k,k-n})$ is determined by first matching the given structure $T_{k,k-n}$ to the current input $(w_1, \ldots, w_k)$ and then identifying the BRANCHING LEVEL, all the possible words that can populate the next word slot w_{k+1}. The probabilities are estimated ac-

[7] Perplexity measures the degree of uncertainly about the next word, averaged over a set of sentence prefixes used to test the model [37]. The lower the perplexity, the better the model. We define the test set perplexity in the usual way [60], as $perp = 2^{-\frac{1}{n}\sum_{w \in S} \log_2 p(w)}$, where n is the number of words in the test set S.

Algorithm 6 The parser.

for all learners $z = 1 : Z$ **do** {Z *is the number of learners*}
  for all sentences $s = 1 : N$ in the test corpus **do** {N *is the number of sentences*}
    for all k-prefixes, $k = 2 : L - 1$ **do** {L *is the number of words in sentence* s}
      IDENTIFY the m paths in the graph of learner z that span entirely or partially the k-prefix of the sentence;
      for $a = 1 : m$ **do**
        IDENTIFY the matched parts $T_{k-1,k-n}(a)$ of the path that span the input; {*the number* n *of matching words defines the level of history dependence*}
        LOCATE next $l(n)$ possible words in position k, $w_{k,n}^{d}(a)$, $d = 1 : l(n)$, given $T_{k-1,k-n}(a)$;
        ESTIMATE their probabilities p; {*based on the branching factor*}
      end for
      for $n = 2 : k$ **do**
        for $d = 1 : l(n)$ **do**
          ESTIMATE $p_n(w_k^d)$ from equation 4.3;
        end for
        ESTIMATE $P(w_k^d)$ from equation 4.4; {*the predicted word probabilities for position* k}
      end for
    end for
  end for
  AVERAGE among learners and NORMALIZE;
end for

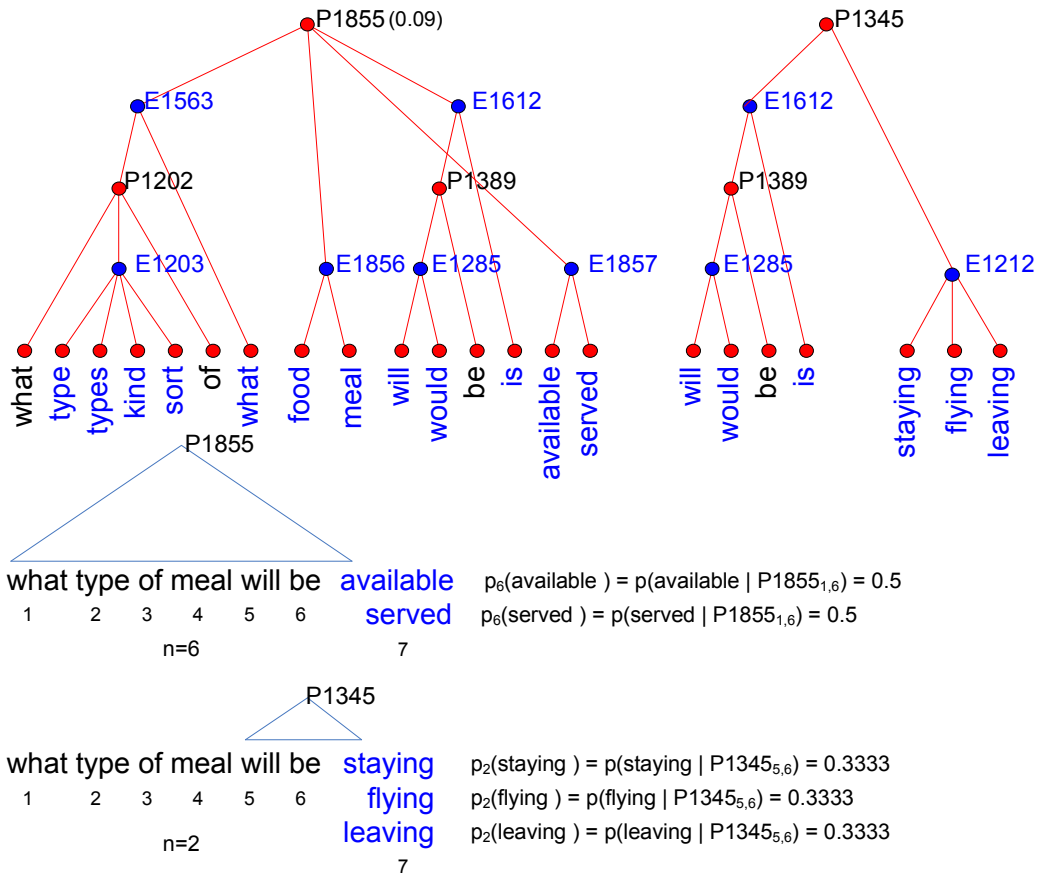

Fig. 4.12: An illustration of the parser's search for matches between patterns (context free structures) and the input sentence ("what type of meal will be. . . "), in constructing a Structured Statistical Language Model from the ADIOS grammar. Once a match is established, the parser identifies the possible next words and estimates their probabilities from the same corpus used by the ADIOS algorithm in learning the grammar. The number n of matching words defines the variable *level of dependence* of the predicted word on its history. In this simplified example there are two pattern #1855 and #1345 which match the input with $n = 6$ and $n = 2$.

cordingly. In Figure 4.12, left, for example, the branching level at w_7 (the next location in the sequence of terminals) is 2, and thus $p(\text{available}|P1855_{1,6}) = p(\text{served}|P1855_{1,6})1/2$. Because there may be several trees of the same length, I label them as $T_{k-1,k-n}(a)$ for

$a = 1, \ldots, m$. We calculate these probabilities iteratively. At each step, a simple deterministic parser analyzes all the relevant root patterns (including those that span only parts of the prefix), seeking matching parts (see Figure 4.12). The number n of matching words defines the variable *level of history dependence* of the predicted word (in standard n-gram language models n is fixed). We estimate the probability $p(w_{k+1}|w_{k-n} \ldots w_k)$ of a word given its sentence prefix as:

$$p_n(w_{k+1}) \doteq p(w_{k+1}|w_{k-n} \ldots w_k) = \sum_{a=1:m} p(w_{k+1}|T_{k,k-n}(a))p(T_k, k-n)(a)) \tag{4.3}$$

where $p(T_{k,k-n})$ is the probability of finding the pattern tree $T_{k,k-n}$ in the corpus, and m is the number of structures that span $(k, k-n)$. For each word in the test set, the parser provides the values of the n-gram probability functions $p_n(w_k)$. The final probability is estimated by linear smoothing:

$$P(w_k|w_{k-n} \ldots w_{k-1}) = c_1 p_1(w_k) + c_2 p_2(w_k) + \ldots + c_{k-1} p_{k-1}(w_k) \tag{4.4}$$

where $\sum_i c_i = 1$; I set $c_i = i / \sum_{j=1:k-1} j$.

Because the ADIOS algorithm is greedy (the best available pattern in each iteration is irreversibly committed to), the syntax it acquires depends on the order of sentences in the training set. This is expected to affect the learning of a complex CFG, especially if it contains many loops. To mitigate this dependence, I train multiple learners on different order-permuted versions of the corpus [91]. When multiple learners are used, I average the probabilities they feed into the language model for every predicted word at every location, then normalize.

The experiments described next involved the Air Travel Information System (ATIS) data [47]. I used the ATIS2 and ATIS3 corpora, with lexicons of 1212 and 1596 words,

respectively.[8] The multiple-learner ADIOS algorithm was first applied to the ATIS training corpus, extracting the syntactic patterns. We then weighted the elements of each learner's grammar probabilistically, using the sentences of the training set, by applying the parser to successive prefixes of each of the sentences in that set. The results were used to assign probabilities to the relevant pattern elements (terminals and equivalence class members). The final probability is the average of the weighted probability and the predicted probability.[9]

Coverage. As the inference process proceeds, the learned grammar is expected to cover more and more of the test set. This, however, may also lead to over-generalization. The trade-off that arises in this issue has been examined by [90, 91]; here I focus on the coverage — the proportion of accepted test sentences — attained by my model (a sentence is accepted if it can be generated by the acquired grammar). As Figure 4.13, right, shows, the coverage of ADIOS as applied to ATIS2 and ATIS3 grows with the number of learners, asymptoting around 50% and 30%

respectively (the coverage can reach 70% for a modified model as explained in the figure legend; such a modified model was *not* used in the perplexity study reported below). Although these results may seem modest, it is important to note that they have been obtained by training the model *solely* on the ATIS corpora themselves.

Perplexity. The first experiment involved the ATIS2 training corpus, which consists of $13,044$ utterances ($130,773$ tokens). We used all but 400 of the utterances for training, and the rest for testing. The second experiment involved the ATIS3 corpus, which contains $7,362$ utterances ($83,867$ tokens), and the test set that accompanies it, which contains 981 utterances ($12,029$ tokens).

The best perplexity scores achieved by my model is 11.5 for ATIS2, and 13.5 for

[8] The ATIS corpora were used because of the availability of perplexity benchmark data and the possibility of training ADIOS on the very same texts for which perplexity is then computed. The larger WSJ corpus, whose lexicon is also much larger, is typically simplified before it can be used effectively for language modeling [85].

[9] A more complicated and time-consuming, but probably better, way of producing a Probabilistic Context Free Grammar (PCFG) out of the ADIOS rules would have been to use the Inside-Outside Algorithm [66].

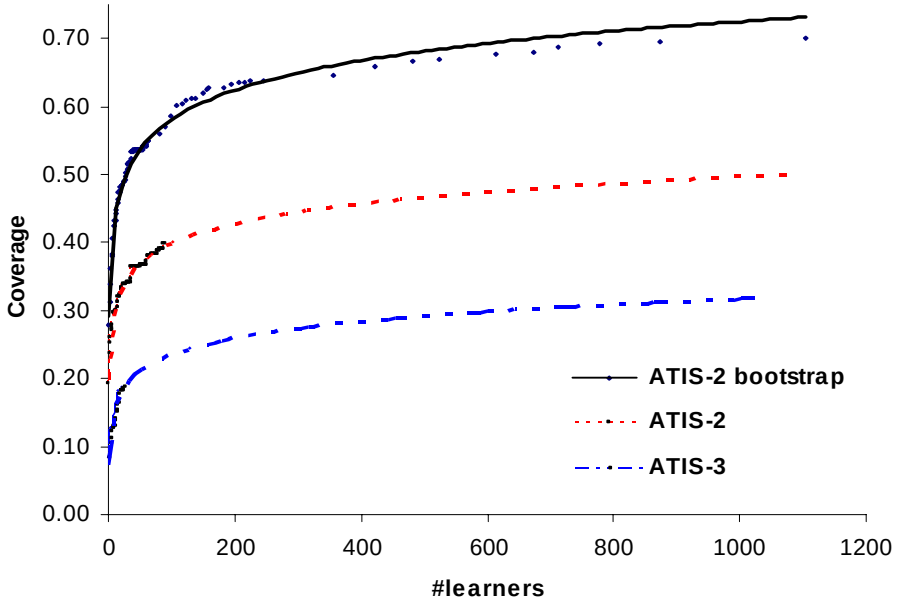

Fig. 4.13: the coverage of the acquired grammar as a function of the number of learners on the ATIS2 and ATIS3 corpora. The perplexity results on these corpora, reported in Table 4.11 below, were obtained with 30 learners each; the coverage of the corresponding grammars is shown by the middle and bottom curves. The top curve, intended as a demonstration of a technique that can be used to improve coverage, was obtained with a modified version of the ADIOS algorithm, which differs from the one described in [90, 91] in two respects. First, to facilitate learning, I provided the system ahead of time with 11 equivalence classes of nouns, such as the names of cities and days of the week. Second, if two candidate patterns received the same significance score in the inference stage, the algorithm chose the more productive one. With these two modifications, the system reaches 70% coverage of the ATIS2 set after training on only 12,000 sentences.

ATIS3. These figures compare favorably to those of other published results, summarized in Table 4.11. It is especially interesting to compare also the model size for each case: whereas the state of the art LM systems use thousands of parameters, my SSLM has only as many as required by the (usually small) size of the acquired grammar, which, moreover, can be learned from very small corpora.

In his review, Goodman [37, p.44] hypothesizes that much of the benefit of using a structured language model (SLM) stems from the same source as that of statistical

ATIS2

method	*# of parameters*	*perplexity*	*ref.*
ADIOS SSLM	under 5000	**11.5**	
Trigram Kneser-Ney backoff smooth.	1.E+05	14	[63]
Probabilistic finite automata (PFA) Inference (ALERGIA) combined with a trigram model	1.E+05	20	[59]
Probabilistic finite automata (PFA) Inference (ALERGIA)	1.E+05	42	[59]

ATIS3

method	*# of parameters*	*perplexity*	*ref.*
ADIOS SSLM	under 5000	**13.5**	
SSLM-trained on Wall Street Journal combined with a trigram model	1.E+05	15.8	[15]
SSLM based on the NLPwin parser combined with a trigram model	1.E+05	15.9	[15]
SSLM-trained on the ATIS corpus combined with a trigram model	1.E+05	15.9	[15]
trigram	4.E+04	16.9	[15]
SSLM based on the NLPwin parser	1.E+05	17.2	[15]
SSLM-trained on the Wall Street Journal	1.E+05	17.7	[15]
SSLM-trained on the ATIS corpus	1.E+05	17.8	[15]

Tab. 4.11: The perplexity of the ADIOS SSLM, compared with some results from the literature [59, 63, 15]. Note that my SSLM uses for training *only* the data provided for that purpose in the ATIS corpora themselves. Although my model requires that only the three parameters of the ADIOS algorithm be specified in advance, I have stated the approximate overall number of patterns of all learners as the counterpart to the number of parameters in the other methods.

techniques, such as skipping, clustering, or larger n-grams. In relation to the latter, he points out that significant improvements can be obtained by moving from trigrams to five-grams. Interestingly, the model I describe achieves an average level of context dependence of about 5.

In the standard approaches, the inference of n-gram models for $n > 3$ is severely hampered by the extremely large number of parameters they require. My structural stochastic language model (SSLM) avoids this problem by capitalizing on an efficient learning algorithm, ADIOS, capable of acquiring a compact representation (a grammar) from small amounts of data. The ADIOS algorithm is capable of inferring context-free patterns (rules), accompanied by equivalence classes (in the language

modeling terminology, clusters), from raw, untagged corpora. It is perhaps not surprising that subsequent use of these structures results in a good language model: Goodman [37, p.23] notes that automatically learned clusters outperform speech tags (when there is enough data), and that modeling each sentence type (what I call root pattern) separately improves performance.

The present approach achieves the equivalent of a relatively long-range n-gram dependence ($n \approx 3.8$ to $n \approx 5.0$) with a model that can be trained on very little data. In the present work, I used about $12,000$ training sentences in the ATIS2 experiment. In the ATIS3 experiment, only about $7,000$ training sentences were used (which was probably not quite enough, judging from the relatively low performance of our my in this experiment).

I conclude that the ADIOS SSLM achieves performance comparable to or exceeding that of other recent models that use syntactic information, such as [76, 15, 85, 60], without relying on a hand-parsed corpus. This makes my approach applicable in principle to any body of language data, in any language. In addition, this performance is achieved without any optimization efforts on my part. We attribute this to the relatively high quality of the grammar inferred by ADIOS, which attains better coverage than other comparable efforts, such as that of [76]. Further improvements should be possible if the acquired grammar is weighted using the Inside-Outside algorithm, yielding a better PCFG (in the present version, all the pattern weights are equal). Better smoothing should also lead to improved performance.

4.2 Bioinformatics

4.2.1 Proteomics

4.2.1.1 Classification of enzymes

This study evaluated the ability of root patterns found by ADIOS to support functional classification of proteins (enzymes). The function of an enzyme is specified by an Enzyme Commission (EC) name. The name corresponds to an EC number, which is of the form: n1:n2:n3:n4. In this experiment, we concentrated on the oxidoreductases family (EC 1.x.x.x). Protein sequences and their EC number annotations were extracted from the SwissProt database Release 40.0; sequences with double annotations were removed. First, ADIOS was loaded with all the 6751 proteins of the oxidoreductases family. Each path in the initial graph thus corresponded to a sequence of amino acids (20 symbols).

The training stage consisted of the two-stage action described in section 4.1.2.5. In the first stage ($\eta = 0.9$, $\alpha = 0.01$), the algorithm identified $10,200$ motifs (words). In the second stage ($\eta = 1.0$, $\alpha = 0.01$) after removing those letters that were not associated with one of the identified motifs, it extracted additional 938 patterns. Classification was tested on level 2 (EC 1.x, 16 classes) and on level 3 (EC 1.x.x, 54 classes). Proteins were represented as vectors of ADIOS root patterns. A linear SVM classifier (SVM-Light package, available online at http://svmlight.joachims.org/) was trained on each class separately, taking the proteins of the class as positive examples, and the rest as negative examples. 75% of the examples were used for training and the remainder for testing. Performance was measured as Q = (TP + TN)/(TP + TN + FP + FN), where TP, TN, FP and FN are, respectively, the number of true positive, true negative, false positive, and false negative outcomes. Table 4.12 presents the performance of the ADIOS algorithm on level 2 alongside the performance of the SVM-PRot system [11]; Table 4.13 presents the performance on level 3. The ADIOS

performance matched the performance of the SVM-PRot system, even though the latter uses a representation composed of features such as hydrophobicity, normalized Van der Waals volume, polarity, polarizability, charge, surface tension, secondary structure and solvent accessibility, while I use solely the structure found by my algorithm in the amino acid sequence data (see Figure4.14). The average recall/precision on level 2 was $71 \pm 13\%$ and $90 \pm 9\%$, respectively, while recall/precision on level 3 was $70 \pm 26\%$ and $93 \pm 23\%$, indicating that the ADIOS representation can accurately discriminate the enzyme's low-level functionality.

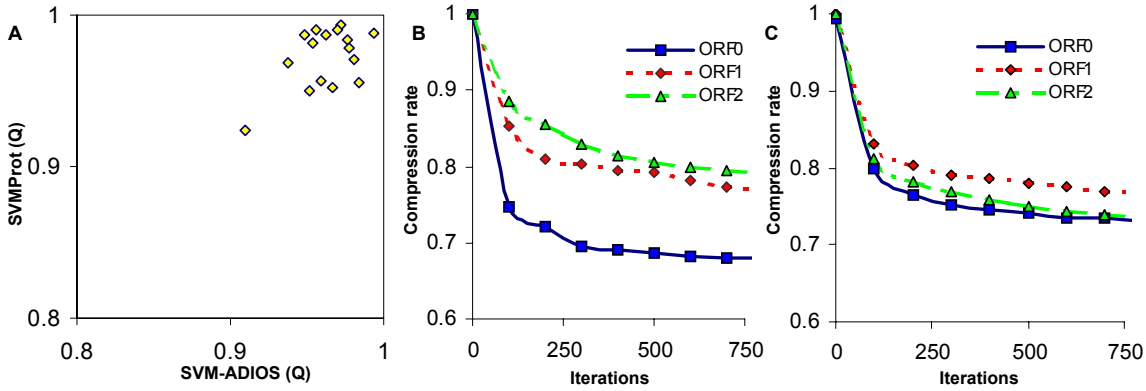

Fig. 4.14: (A), Functional protein classification (16 Enzyme Classes (EC), level 2). This plot compares the Q-performance of a linear SVM based on ADIOS root-patterns with that of the SVM-Prot package; for definitions of the task, see [11]. Note that we match the performance of SVM-PRot using solely the structure found by ADIOS in protein sequences. (B,C), Syntax found by ADIOS can be used to distinguish between coding and non-coding regions of the genome. The correctness of the codon representation of a DNA sequence depends on knowing the starting point of the gene's Open Reading Frame (ORF). Here we compare three corpora generated from the same sequence data read in the original ORF (ORF0) or shifted ones (ORF1 and ORF2). The 64 codons (nucleotide triplets) served as the original units of the graph. In each case we compare the total description length of the final ADIOS graph with that of the original graph, i.e., the compression achieved by ADIOS. The corpora consisted of the first exons of $4,777$ *C. Elegans* embryonic genes (B), and of the first 500 bases of the same genes, including introns (C). The difference between the compression ratios achieved for the different ORFs in the two figures demonstrates that the ADIOS procedure allows us to distinguish the correct reading frame when it is functionally significant.

Tab. 4.12: The performance of the ADIOS algorithm versus the SVM-Prot system on level 2.

Class	TP	FN	FP	TN	ADIOS Q	recall	precision	SVM-Prot Q
1.1	333	89	64	1201	0.91	0.79	0.84	0.92
1.2	110	49	26	1502	0.96	0.69	0.81	0.99
1.3	62	36	14	968	0.95	0.63	0.82	0.98
1.4	33	23	9	556	0.95	0.59	0.79	0.99
1.5	20	19	8	384	0.94	0.51	0.71	0.97
1.6	198	23	25	1441	0.97	0.90	0.89	0.99
1.7	23	13	2	365	0.96	0.64	0.92	0.99
1.8	51	21	3	717	0.97	0.71	0.94	0.99
1.9	117	21	4	1376	0.98	0.85	0.97	0.96
1.10	16	13	0	292	0.96	0.55	1.00	0.96
1.11	61	16	3	772	0.98	0.79	0.95	0.98
1.13	16	15	2	315	0.95	0.52	0.89	0.95
1.14	106	41	13	1462	0.97	0.72	0.89	0.95
1.15	54	4	0	582	0.99	0.93	1.00	0.99
1.17	22	6	0	285	0.98	0.79	1.00	0.97
1.18	32	10	1	424	0.98	0.76	0.97	0.98

4.2.2 Genomics

4.2.2.1 Coding DNA Regions

The ADIOS algorithm also provides a useful tool for identifying Open Reading Frames (ORF) and coding regions in DNA sequences, based on comparing the description length of the representation before and after learning. The description length of the ADIOS representation consists of two parts: the graph (vertices and paths) and the identified patterns. The *compression ratio* of the description length can be quantified by evaluating the decrease in the physical memory it occupies (in bits). I have calculated the compression at several points along the curves of the ATIS-CFG recall/precision graph (Figure 4.1). Figure refrp-compression shows the correlation between the recall/precision levels (ordinate) and the compression rate (abscissa). It can be seen that ADIOS recall level strongly depends on (increases with) the compression level, but the precision level only weakly depends on the latter. The compression ratio characteristic is particularly useful when comparing the performance of ADIOS on different data for which the target "grammars" are not available. The ORF problem

Tab. 4.13: The performance of the ADIOS algorithm on level 3.

class	TP	FN	FP	TN	Q	Recall	Precision
1.1.1	331	67	48	1241	0.93	0.83	0.87
1.1.3	4	4	0	80	0.95	0.50	1.00
1.1.99	6	8	0	147	0.95	0.43	1.00
1.10.2	8	8	1	166	0.95	0.50	0.89
1.10.3	6	3	0	95	0.97	0.67	1.00
1.10.99	3	0	0	30	1.00	1.00	1.00
1.11.1	62	15	4	771	0.98	0.81	0.94
1.12.99	6	0	0	65	1.00	1.00	1.00
1.13.11	15	12	0	277	0.96	0.56	1.00
1.13.12	0	3	0	30	0.91	0.00	0.00
1.14.11	8	3	0	117	0.98	0.73	1.00
1.14.12	4	1	0	55	0.98	0.80	1.00
1.14.13	14	11	1	251	0.96	0.56	0.93
1.14.14	48	9	0	572	0.99	0.84	1.00
1.14.15	8	1	0	95	0.99	0.89	1.00
1.14.16	6	0	0	67	1.00	1.00	1.00
1.14.18	1	2	0	35	0.95	0.33	1.00
1.14.19	6	0	0	65	1.00	1.00	1.00
1.14.99	15	3	0	180	0.98	0.83	1.00
1.15.1	53	5	2	580	0.99	0.91	0.96
1.16.1	2	3	0	52	0.95	0.40	1.00
1.17.4	21	7	1	281	0.97	0.75	0.95
1.18.1	7	4	0	117	0.97	0.64	1.00
1.18.6	25	5	0	307	0.99	0.83	1.00
1.2.1	95	29	4	1236	0.98	0.77	0.96
1.2.3	3	0	0	32	1.00	1.00	1.00
1.2.4	10	6	0	165	0.97	0.63	1.00
1.2.7	2	6	0	82	0.93	0.25	1.00
1.2.99	2	5	0	72	0.94	0.29	1.00
1.21.3	3	0	0	32	1.00	1.00	1.00
1.3.1	29	8	1	369	0.98	0.78	0.97
1.3.3	23	11	0	347	0.97	0.68	1.00
1.3.5	4	0	0	45	1.00	1.00	1.00
1.3.7	0	3	0	37	0.93	0.00	0.00
1.3.99	13	5	1	181	0.97	0.72	0.93
1.4.1	15	5	0	207	0.98	0.75	1.00
1.4.3	10	12	0	222	0.95	0.45	1.00
1.4.4	2	1	0	35	0.97	0.67	1.00
1.4.99	6	1	0	77	0.99	0.86	1.00
1.5.1	18	12	1	299	0.96	0.60	0.95
1.5.3	0	3	0	37	0.93	0.00	0.00
1.5.99	2	1	0	37	0.98	0.67	1.00
1.6.1	4	1	0	52	0.98	0.80	1.00
1.6.2	5	0	0	50	1.00	1.00	1.00
1.6.5	162	5	8	1512	0.99	0.97	0.95
1.6.99	24	19	8	429	0.94	0.56	0.75

Tab. 4.14: (continued): The performance of the ADIOS algorithm on level 3.

class	**TP**	**FN**	**FP**	**TN**	**Q**	**Recall**	**Precision**
1.7.1	10	4	0	145	0.97	0.71	1.00
1.7.2	5	0	0	55	1.00	1.00	1.00
1.7.3	3	2	0	50	0.96	0.60	1.00
1.7.99	5	5	0	107	0.96	0.50	1.00
1.8.1	30	4	0	345	0.99	0.88	1.00
1.8.4	30	4	0	342	0.99	0.88	1.00
1.9.3	110	28	6	1374	0.98	0.80	0.95
1.97.1	3	0	0	32	1.00	1.00	1.00

Tab. 4.15: Some of the specific ADIOS patterns appear in specific Enzyme Classes.

Enzyme Class	Pattern
1.1.1	WSG {VNVAGV, RT}
1.1.1	GKVIKCKAA VL
1.1.1	ALVTG {AGK, ST, AAQ, AS, SR, SK, TS, NK} GIG
1.1.1	ANQNGAIWKLDLG LDA
1.1.1	AAY {GEV, SSVL, STV, SSV} {MN,AQA}
1.1.1	LTNKNV IFVAGLGGIGLDTS
1.2.1	IF IDG EH GTTGLQI
1.2.1	VSV IDNLVKGA GQAIQN
1.4.3	TG {FQ,GI} YGL
1.6.5	TD {RVL, LKSLI} AY
1.6.5	IAL {TSL, ME, PT} HT
1.8.1	FT {EL, VLPM, HL} YP
1.8.4	EVR {SAHG,SNA,KNA,RAA,SKL,RFA,KYD} DS
1.8.4	{NR,TT} QG
1.11.1	VKFHWKPTCGVK {SM, CL}
1.11.1	{QE,QP} WWPAD
1.11.1	{AI,AP} KFPDFIHTQKR
1.11.1	FDHER IPERVVHARG
1.11.1	GIPASYR HM GFGSHT
1.11.1	VS LDKARRLLWPIKQKYG
1.15.1	FW {VVN,LVN,MP} WD
1.18.6	{IPL,CIG} VHGGQGC MFV

is a typical example of such an analysis.

Following its iterative application, ADIOS compresses the initial graph to a final graph plus a forest of distilled root-patterns. Although the latter can generate a much larger corpus than the original one, the description length of the ADIOS representation is diminished. The recall level of ADIOS increases with compression. Applying ADIOS to the coding region of the *C. Elegans* genome (Figure 4.14B), one may conclude that the syntax of ORF0 (the correct open reading frame) is to be preferred. Moreover, it becomes possible to distinguish between coding and non-coding regions (Figure 4.14B vs. 4.14C), because for the latter different ORFs lead to similar compression levels.

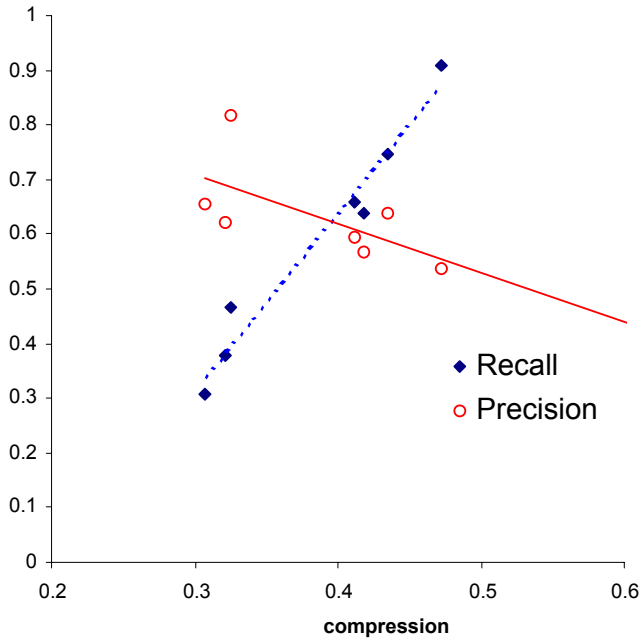

Fig. 4.15: Correlation between the recall/precision levels (ordinate, blue and red respectively), versus compression rate (abscissa), obtained for the ATIS-CFG problem

4.2.2.2 Non-coding DNA Regions

Regulation of gene expression is mediated mainly through specific interactions of transcription factor (TF) proteins with DNA promoter elements. TF binding sites are short (6-20 bases) and imprecise. Binding sites typically comprise a minority of the nucleotides within a promoter region. They are embedded within sequence that is assumed to be non-functional with respect to transcription. Identifying genuine binding sites is a challenging task as the physical extent of a promoter is rarely well defined, and within this ill-defined region we are seeking sparsely distributed, short and imprecise sequence motifs.

The advances in genome research, including whole genome sequencing and mRNA expression monitoring has allowed the development of computational methods for binding site prediction. Among the most popular and powerful methods for regulatory motif detection is Gibbs sampling. In this method, motifs that are overrepresented in the data may be found. However, because regulatory motifs are very short, and, in contrast, the regulatory portion of the genome is very long (e.g., 6,000,000 base-pairs in yeast, and much longer in mammals), and because the size of gene regulatory networks is relatively small (typically tens of genes), most regulatory motifs are not expected to be overrepresented on a genome-wide scale. The task of motif identification is thus often first tackled by grouping together relatively small sets of genes (tens or hundreds) that are likely to be co-regulated, followed by motif searching only within such groups (Tavazoie *et al.* 1999, Brazma *et al.* 1998, Wolsberg *et al.* 1999). While such efforts were proved to be successful, they are not always applicable (e.g. due to insufficient means to group co-regulated genes) and identification of potentially co-regulated genes is largely inaccurate. Thus, there is a clear need to develop motif detection algorithms that will operate directly on the entire genome level with no a priori knowledge about the identity of co-regulated genes. Because such motifs cannot be overrepresented at a genome-wide scale, a different property, and a

suitable probabilistic formalism, should be sought.

While position-specific weight matrices (Stormo 2000) usually ignore dependencies between nucleotide positions in regulatory motifs, such dependencies are known to occur (Bulyk *et al.* 2002, Benos *et al.* 2002). Statistical models that account for such dependencies include hidden Markov models, and Bayesian networks (Durbin *et al.* 1998). Yet even sophisticated models of this kind have relatively low values of sensitivity and specifity when required to represent the known binding sites (Barash *et al.* 2003).

A statistical model that would identify improbable consecutive chains of interdependencies between adjacent nucleotide positions is expected to identify motifs as statistically significant on a genome-wide scale even without significant over representation. In that sense, the MEX algorithm seems to be appropriate for the job. MEX has been run on the entire set of yeast promoters and identified thousands of highly scoring motifs. Then in a collaboration with the research group of Dr. Yitzhak Pilpel from the Weizmann Institute of Science, the expression coherence method (Pilpel *et al.* 2001, Sudarsanam *et al.* 2002, Lapidot and Pilpel 2003) was used in order to check which of the identified motifs exerts a statistically significant effect on the expression profiles of the genes down-stream from them. The expression analysis showed an enormous enrichment of highly-scoring motifs among the set of predictions of the MEX algorithm. It also identified potential biological conditions in which they act.

4.2.2.2.1 Results I In the yeast problem, the MEX algorithm was applied to 4800 promoters. Each promoter sequence, of length up to 1000bp, is considered as a data-path on the graph. After all information is loaded onto the graph, I used all 4800 sequences as trial-paths in order to extract the motifs. Obviously the latter are determined by probabilistic arguments driven by sequence data. Some of these motifs, in particular those of very high occurrence (in the thousands) may be completely

α	$\#\mathrm{s}-\mathrm{motifs}$	#sites	#accounted	#included
0.2	5842	433	284	462
0.3	8823	433	341	672
0.4	12587	433	366	962
0.5	17844	433	405	1303

Tab. 4.16: Comparison with AlignACE results

unrelated to regulatory functions. Other may indeed fit cis regulatory binding sites. cis regulatory motifs are often represented in a PSSM notation, or through some other underlying probability model which assigns a large class of binding sites to one specific transcription factor. The interesting question is how many of my sequence motifs (to be referred to henceforth as s-motifs) are indeed binding sites.

4.2.2.2.2 Regulatory Motifs - Comparison with Known Data As a first test of the relevance of the MEX s-motifs, they can be compared to the results of the AlignACE website of the Church lab, http://atlas.med.harvard.edu/cgi-bin/compareace_motifs.pl, using their data of 26 known cis regulatory motifs. The number of binding sites per motif range from 12 to 119 sites (an average of 29 sites per motif). There are altogether 433 different binding sites. In Table 4.16 I list the number of known binding sites that are describable by my s-motifs. This number depends on the choice of the parameter α. As seen here, for $\alpha = 0.5$, one can account for 405 out of 433 i.e. 94% of the binding sites. The fourth column specifies the number of binding sites that were accounted for in my list of s-motifs. The fifth column specifies how many of my s-motifs are included within the available list of 433 binding sites.

It is interesting to note that the numbers in the fourth and fifth column of this table grow proportionally to the number of observed sequence-motifs. The latter grows of course with α. This implies that there is no good reason to limit oneself to small α values. Another interesting observation is that the number of s-motifs per observed regulatory motif grows with α from $462/26 = 18$ to $1303/26 = 50$. If one

considers these numbers as indicative of the truth, it is possible to estimate the total number of cis regulatory motifs in the yeast, by dividing the second column by these ratios. The results are quite stable for all α, varying from 329 to 356.

To check this method further, the same 26 cis regulatory motifs and their 756 different sites were used (there are 433 different ones among them, as mentioned above). The algorithm then identified all the s-motifs present in each site, and thus may regard the site as a vector in a (high-dimensional) s-motif space. Now we are in a position to perform cross-validation tests on each one of the classes of binding sites, i.e., each one of the 26 cis regulatory motifs. This has been done by using 80% of the instances as labeled prototypes and evaluating the model on the remaining 20% of the instances. Classification was performed by using an Euclidean K=1 nearest-neighbor algorithm. The results are specificity=1 for all α values, and sensitivity growing from 0.62 at $\alpha = 0.2$ to 0.77 at $\alpha = 0.5$. These classification results are very satisfactory, in comparison with similar analysis by other methods (see section 3.2 of Barash $et\ al.$ 2003). They are even more impressive given the simple underlying procedure.

4.2.2.2.3 MEX and Multiple Alignment To explain better the MEX method for motif extraction, I describe one example in Figure 4.16. Shown here are 12 different subsequences taken from promoters of 12 different genes. All have in common one motif, AAAACGCGAA, identified by the MEX algorithm. This figure is presented as an alignment matrix. It came about by first identifying AAAACGCGAA as a motif, following the procedure outlined above and sketched in Figure 4.16. This motif is a variation on MCB (AAACGCGTAA), the main $S.\ cerevisiae$ cell cycle control regulator, governing the transition from G1 to S phase (Tavazoie $et\ al.$ 1999). Since the data-path identity is being kept on the graph constructed by all paths, it is a simple exercise to regenerate all subsequences that have a certain motif in common. Note that the relative occurrence of this motif is not high, it occurs in 12 out of all 4800 promoters. Yet its

```
YOL149W              ATATGTCGAAAACGCGAAGCAAGAAAGAA
YPL241C              TGTCATACAAAACGCGAAGAATCTGAAAT
YPL179W              TTCGAGGTAAAACGCGAAGTTCGTAGAGA
YBL061C_YBL060W      AGCAAGTCAAAACGCGAAGTGAAACGTGG
YBR073W              TAAAAAATAAAACGCGAAGAGCTAAAAAA
YBR088C              GACGCACCAAAACGCGAAACGCGTAACTT
YLL022C_YLL021W      AATTGGAAAAAACGCGAAACTTCAGTGCA
YLR372W              GAGAAAAAAAAACGCGAAATTTTTCTTCC
YNL283C_YNL282W      AACAGGAAAAAACGCGAAATGTCCGTAAC
YOL090W              AAAAATAGAAAACGCGAAAACTTGTCATT
YDR081C_YDR082W      CAAAATTCAAAACGCGAACTAGTCACGAT
YPL145C_YPL144W      CAGCTAGTAAAACGCGAACGGAATGAGTT
```

Fig. 4.16: The appearance of one motif, AAAACGCGAA, in 12 promoters, i.e. 12 data-paths of MEX, which is a natural multiple alignment tool.

discovery is quite straightforward. It obeys my criteria because of the variance of prefixes and suffixes of this motif. Thus we realize that MEX can serve as a very useful tool in performing multiple alignment: just load the strings of interest as data-paths, extract sequence motifs, and check single or multiple alignments of any order in the data.

4.2.2.2.4 Results II: Expression Coherence Test In order to check which of the motifs extracted by MEX are likely to function as regulatory elements in yeast, we have characterized each of them using the expression coherence (EC) method (Pilpel *et al.* 2001, Sudarsanam *et al.* 2002, Lapidot and Pilpel 2003). The EC score of a motif that appears in the promoters of N genes is defined as the fraction of gene pairs, (i, j) in the set such that the Euclidean distance between their mean- and variance-normalized expression profiles, D_{ij}, falls below a threshold, D, divided by the total number of gene pairs in the set, $0.5(N-1)N$. The value D is determined as a distance that random gene pairs have a probability p of scoring below. The EC score

may range between 0 and 1 and is higher for sets of genes that cluster in one or a few tight clusters. Additionally, a sampling-based means exists for the assessment of the statistical significance, in terms of p-value, of EC scores, given the gene set size N (Lapidot and Pilpel 2003).

The EC scores and p-values were calculated for each of 6708 sequence-motifs in each of 40 conditions in which whole-genome mRNA expression of yeast was monitored using DNA chips and microarrays (http://arep.med.harvard.edu/cgi-bin/ExpressDByeast/E The 6708 s-motifs were chosen from the $\alpha = 0.5$ set, constrained by length larger than 6 and occurrence rate between 5 and 30. To account for the testing of multiple hypotheses and to control for the amount of false positives, the false discovery rate (FDR) theorem was used (Benjamini and Hochberg 1995) to determine the p-value cutoff below which motifs are guaranteed to be statistically significant at a specified false discovery rate. Setting this rate at 10%, revealed that 20% of the 6708 sequence-motifs have a significant EC score in at least one of the conditions. For comparison, in an equally-sized set of random motifs, with same motif-length distribution, only 0.6% of the motifs pass this FDR threshold. Figure 4.17 shows a comparison of the ranges of p-values obtained when computing EC scores for the motifs derived by MEX, relative to the p-values obtained for the randomly generated motif set. As clearly seen, relative to random motifs, MEX provides an enrichment in highly scoring motifs and an under-representation of low-scoring motifs.

Expression analysis of genes that contain regulatory motifs in their promoters allows not only to select potentially functional motifs, but also to decipher their semantics. A comprehensive semantic characterization of a regulatory motif would amount to describing the condition in which it acts, and its regulatory effects, e.g. increase in expression during a particular stress, or peaking of expression profile, during a particular phase of the cell cycle. Figure 4.18 shows such semantic annotation of two of the high-scoring sequence-motifs generated by MEX. These motifs govern opposite re-

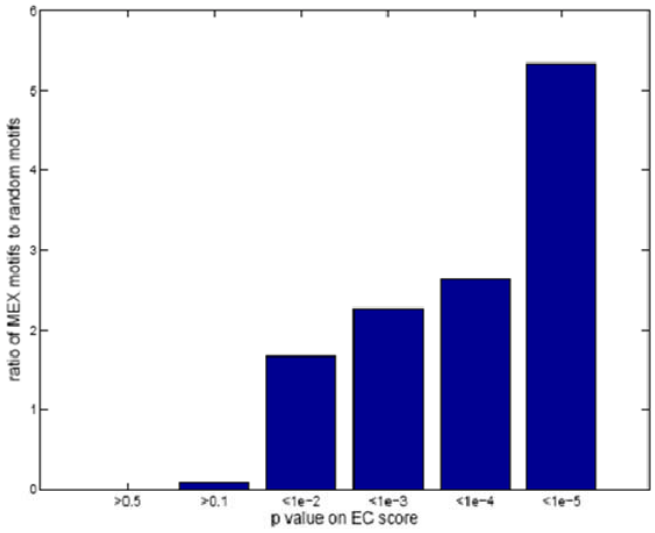

Fig. 4.17: The s-motifs extracted by MEX are enriched with significant motifs, as judged by the p-values on their EC scores, computed over 40 different expression conditions (http://arep.med.harvard.edu/cgi-bin/ExpressDByeast/EXDStart). EC scores and p-values were computed for the set of motifs extracted by MEX, as well as for a random set of motifs with equal length distribution. For both data sets, the fraction of motifs falling in each p-value range was computed. The graph shows the ratio between the fraction of MEX motifs in each bin and the fraction of random motifs. It is apparent that MEX motifs are enriched with highly scoring motifs, whereas low-scoring motifs are under represented.

sponses to hypo-osmotic pressure. Manual analysis on individual motifs can be done through the expression coherence site http://longitude.weizmann.ac.il/services.html.

I have outlined here a combined semantic and syntactic approach for the discovery and analysis of promoter regulatory motifs in *S.cerevisiae*. This approach has identified most of the known regulatory motifs, and discovered many new, rigorously prioritized motifs. It is clearly demonstrated that a combination of both levels of analysis performed here is crucial for efficient and meaningful motif discovery. The underlying principle should prove useful in motif derivation efforts also in other contexts. For instance in efforts for deriving protein-coding sequence motifs one might consider characterizing motifs according to the local (e.g. secondary) 3D structure of the amino acids that constitute the motif.

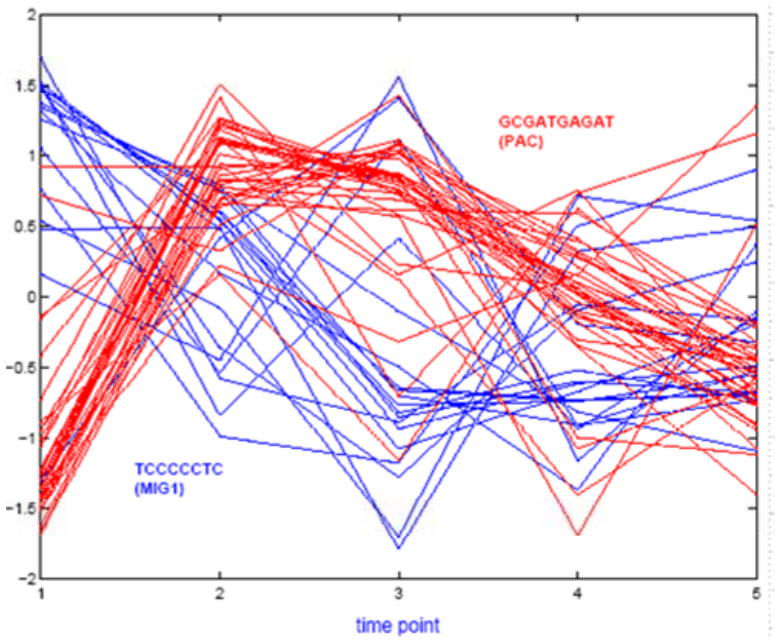

Fig. 4.18: A semantic characterization of two of the motifs extracted by MEX. MEX has successfully identified two motifs governing opposite responses to hypo-osmotic stress. As shown by the graph, all the genes containing the s-motif `GCGTATGAGAT` (corresponding to the PAC motif) in their promoters (plotted in red) behave similarly in response to hypo-osmotic stress (EC=0.45, p-value $<$1*e-5), whereas all the genes containing the s-motif `TCCCCCTC` (corresponding to the binding site of the MIG1 TF) in their promoters (plotted in blue), behave similar to each other (EC=0.20, p-value 4*e-5), yet different from the first group of genes. This illustrates the strength of MEX in identifying sequence motifs corresponding to known s. cerevisiae regulatory motifs based on promoter sequence alone. The expression data for the analysis was taken from Gasch *et al.* .

One of the crucial features of MEX, and the entire ADIOS approach, is that it is truly unsupervised. The common motif finding algorithms, in particular those based on over-representation of motifs, such as Gibbs sampler (Hughes *et al.* 2000), all require that promoters will be clustered first (e.g. based on expression or functional classifications). This is absolutely essential for such methods, as functional regulatory motifs are not over-represented on a genome-wide scale and thus are only detectable by methods such as Gibbs sampling if applied to meaningful clusters.

The application of the MEX algorithm studied here is a first level of feature extraction from biological sequences. Higher level patterns may be extracted by repeated application of MEX after the observed sequence-motifs are incorporated as nodes in the MEX graph. Moreover, the full extent of the ADIOS approach (Solan *et al.* 2003) may lead in the future to revealing higher syntactic structures in the data.

4.3 Computational Complexity

I conducted several experiments based on the TA1 grammar to estimate the computational complexity of ADIOS. I found four variables that have major effects: the total number of words in a given corpus, the average sentence length, the size of the initial lexicon and the value of the context window parameter L. For each of these, I conducted an experiment that exclusively manipulated the variable in question, while measuring the time until convergence. The results, plotted in Figure 4.19, reveal the following dependencies: the training time grows linearly with the size of the corpus, and logarithmically with the average sentence length. It shows inverse power dependence both on respect the lexicon size and on the value of L. Overall, the computational complexity of ADIOS according to this empirical estimate is $O\left(nlog\left(l\right)/\left(L^{\lambda}N^{\gamma}\right)\right)$, where n is the total number of words in the corpus, l is the average sentence length, L is the value of context window parameter, and N is the lexicon size. The conclusion from this experiment is that ADIOS is easily scalable to larger corpora; this is consistent with the actual tests described elsewhere in this thesis.

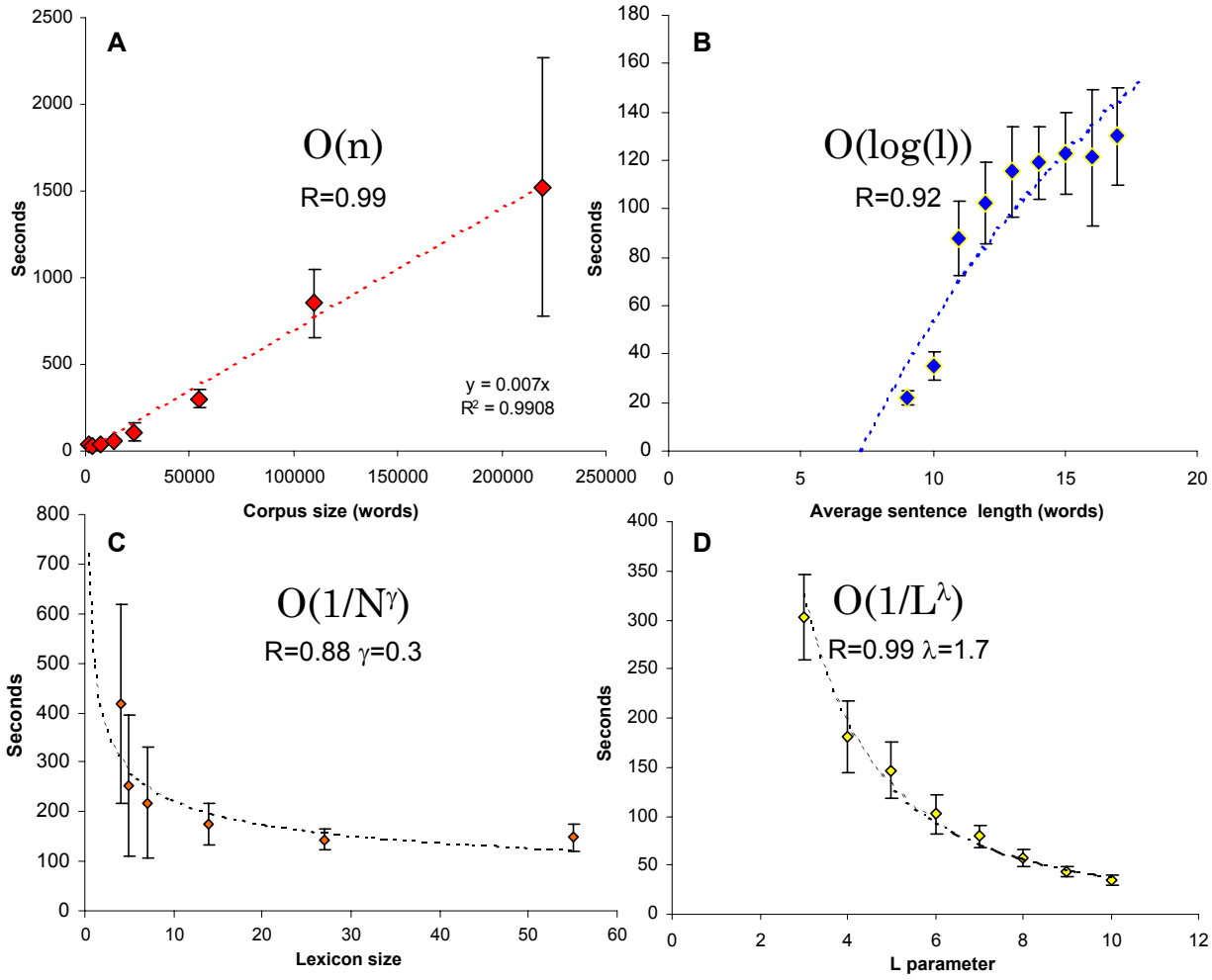

Fig. 4.19: Four experiments estimating the computational complexity of ADIOS by measuring the training time until convergence (ordinate) on the TA1 grammar versus: (A) The total number of words in the corpus; (B) The average sentence length (the experiment manipulated the average sentence length without increasing the total number of words in the corpus); (C) The initial lexicon size; (D) The value of the context parameter L.

5. DISCUSSION

In this chapter I distinguish the characteristics of the ADIOS approach from other grammar induction methods, represented by works such as [100, 66, 79, 96, 26, 39, 10, 25, 52, 95, 99, 21, 61, 48, 3, 70, 86, 62, 2]. I start by reviewing in Section 5.1.1 methods that rely on structurally annotated corpora, and methods that rely on raw, untagged data. In subsections 5.1.1.3 to 5.1.1.5, I describe in detail the different approaches to grammar induction, such as local grammar versus global grammar, Markov models and probabilistic treebank-based. In Section 5.1.2, I describe related linguistic approaches and distinguish between approaches that are motivated mainly by linguistic and psychological considerations (Cognitive and Construction grammars), and those motivated computationally (Local and Tree Adjoining grammars). The final section 5.2 discusses the prospects and the challenges of the ADIOS algorithm.

5.1 Distinguishing characteristics of the ADIOS' approach

The ADIOS algorithm differs from other methods of grammar induction in the data it requires and in the representations it builds, as well as in its algorithmic approach. Most existing approaches require corpora tagged with part-of-speech (POS) information [21]. The very few exceptions are not known to scale up [100], or effectively try to bootstrap to a POS representation (*and* run into the scaling-up problem) by committing ahead of time to learning a Categorial Grammar [48, 3]. The extraction of grammatical primitives in published methods may rely on collocation frequencies [100], or on global criteria such as the likelihood of the entire corpus given the grammar [66, 96, 26, 21, 48].

In comparison, ADIOS carries out its inferences locally, in the context provided by the current search path, alleviating the credit assignment problem in learning, and making productive use of learned structures safer. Furthermore, ADIOS works with raw text or transcribed speech, and makes no prior assumptions about the struc-

tures it seeks. At the same time, the patterns and equivalence classes it learns can be translated in a straightforward manner into the form of context-sensitive rewriting rules. These representations are both expressive enough to support extensive generativity, and, in principle, restrictive enough to capture many of the structure-sensitive aspects of syntax [80] documented by linguists; examples include long-range agreement (Figure 4.4) and tough movement (Figure 4.5).

It is instructive to consider ADIOS in the context of the problem of language acquisition, which has long been a daunting challenge for cognitive scientists [19, 28, 21]. Because a completely bias-free unsupervised learning is impossible [19, 77], the real issue in language acquisition is to determine the model constraints. In the approach developed here, the constraints are defined algorithmically, in the form of a method for detecting units (patterns) that are hierarchically structured and are supported by context-sensitive statistical evidence. When considered as a model of language acquisition, ADIOS is clearly incomplete, as it currently relies on syntactic regularities and leaves out conceptual knowledge and grounding of speech acts in the external events. Nevertheless, this approach is compatible with a range of findings concerning language acquisition, such as the use of statistical cues [87, 35] and the importance of pattern-like constructions [30, 33, 12]. Moreover, it performs well in a wide variety of situations that require unsupervised learning of structural information from untagged data. In grammar induction from large-scale raw corpora, my method achieves precision and recall performance unrivaled by any other unsupervised algorithm. It exhibits good performance in grammaticality judgment tests (including standard tests routinely taken by students of English as second language), and replicates the behavior of human subjects in certain psycholinguistic tests of artificial language acquisition. Finally, the very same algorithmic approach is also proving effective in other settings where knowledge discovery from sequential data is called for, such as bioinformatics.

5.1.1 computational approaches to grammar induction

5.1.1.1 Learning from tagged data.

Methods that rely on structurally annotated corpora [10, 88] are less relevant in the present context because of my interest in truly unsupervised learning. The most popular family of approaches to grammar acquisition requires corpora tagged with part-of-speech (POS) information. As a representative of this line of work, I mention the classical approach of [96], which learns structure (the topology of a Hidden Markov Model, or the productions of a Stochastic Context Free Grammar) by iteratively maximizing the probability of the current approximation to the target grammar, given the data. Perhaps because it is global in that all the data contribute to the figure of merit at each iteration (hence more susceptible to the credit assignment problem), and because of the overly general nature of the POS categories, this method has "difficulties with large-scale natural language applications" [96]. More recently, Clark attempted to learn a grammar from tagged text in two stages, first using local distributional cues and filtering spurious non-terminals using a mutual information criterion, then reducing the resulting maximum likelihood grammar greedily to achieve a minimum description length (MDL) representation [21, 20]. Interestingly, Clark too doubts the appropriateness of POS tags to support grammatical inference, arguing that "a lot of syntax depends on the idiosyncratic properties of particular words" [21], p.36; cf. [24]. I note that unlike the methods just mentioned ADIOS is *local* in the sense that its inferences only apply to the current pattern candidate; furthermore, instead of general-scope rules stated in terms of parts of speech, it seeks context-specific patterns.

5.1.1.2 Learning from untagged data.

Very few grammar induction methods work (as ADIOS does) with raw, untagged data. One of these is the unsupervised structure learning algorithm developed by Wolff

between 1970 and 1985 [100]. The unsupervised structure learning algorithm developed by Wolff stands out in that it does not need the corpus to be tagged. In a 1988 book chapter describing his system [100], Wolff offers an excellent survey of earlier attempts at unsupervised learning of language, and of much relevant behavioral data. His representations consist of SYN (syntagmatic), PAR (paradigmatic) and M (terminal) elements. Although the ADIOS patterns and equivalence classes can be seen as analogous to the first two of these, Wolff's learning criterion is much simpler than that of ADIOS: in each iteration, the most frequent pair of contiguous SYN elements are joined together. His system, however, had a unique provision for countering the usual propensity of unsupervised algorithms for overgeneralization: PAR elements that did not admit free substitution among all their members in some context were rebuilt in a context-specific manner. Unfortunately, it appears that Wolff's system has not been tested on large corpora of real natural language. Recently, [48] described a pilot implementation of an unsupervised method for learning a variant of Lambek's categorial grammar from raw data (transcribed speech), by iteratively maximizing the total "disorder" (inverse of interpretability) of a corpus of utterances.

5.1.1.3 Local Grammar and Markov models.

In capturing the regularities inherent in multiple criss-crossing paths through a corpus, ADIOS superficially resembles finite-state Local Grammars [39] and Variable Order Markov (VOM) models [41, 70] that aim to produce a minimum-entropy finite-state encoding of a corpus. There are, however, crucial differences, as explained below. The ADIOS pattern significance criteria involve conditional probabilities of the form $P(e_n|e_1, e_2, e_3, \ldots, e_{n-1})$, which does bring to mind an n'th-order Markov chain, with the (variable) n corresponding roughly to the length of the sentences we deal with. The VOM approach starts out by postulating a maximum-n VOM structure, which is then fitted to the data. The maximum VOM order n, which effectively deter-

mines the size of the window under consideration, is in practice much smaller than in the present approach, because of computational complexity limitations of the VOM algorithms. The final parameters of the VOM are set by a maximum likelihood condition, fitting the model to the training data.

The ADIOS philosophy differs from the VOM approach in several key respects. *First*, rather than fitting a model to the data, I use the data to construct a (recursively structured) graph. Thus, ADIOS naturally addresses the inference of the graph's structure, a task that is more difficult than the estimation of parameters for a given configuration.[1] *Second*, because ADIOS works from the bottom up in a data-driven fashion, it is not hindered by complexity issues, and can be used on huge graphs, with very large windows sizes. *Third*, ADIOS transcends the idea of VOM structure, in the following sense. Consider a set of patterns of the form $b_1[c_1]b_2[c_2]b_3$, etc. The equivalence classes $[\cdot]$ may include vertices of the graph (both words and word patterns turned into nodes), wild cards (i.e., any node), as well as ambivalent cards (any node or no node). This means that the terminal-level length of the string represented by a pattern does not have to be of a fixed length. This goes conceptually beyond the variable order Markov structure: $b_2[c_2]b_3$ do not have to appear in a Markov chain of a finite order $||b_2|| + ||c_2|| + ||b_3||$ because the size of $[c_2]$ is ill-defined, as explained above. *Fourth*, as illustrated in Figure 4.4, ADIOS incorporates both context-sensitive substitution and recursion, and hence is more powerful even than the eMOTIFs of [51], which allow equivalence classes and wild cards.

5.1.1.4 Global grammar

Stolcke and Omohundro [96] learn structure (the topology of a Hidden Markov Model, or the productions of a Stochastic Context Free Grammar), by iteratively maximiz-

[1] I note in passing that although ADIOS attains per-character entropy comparable to that of the best probabilistic FSA while using a fraction of the number of parameters, the goal of entropy reduction *per se* is at odds with some of the key principles of cognitive utility: a good representation makes redundancy explicit and puts it to use, rather than reducing it [8].

ing the probability of the current approximation to the target grammar, given the data. In contrast to this approach, which is global in that all the data contribute to the figure of merit at each iteration, ADIOS is local in the sense that its inferences only apply to the current bundle candidate. Another important difference is that instead of general-scope rules stated in terms of parts of speech, ADIOS seeks context-specific patterns. Perhaps because of its globality and unrestricted-scope rules, Stolcke and Omohundro's method has "difficulties with large-scale natural language applications" [96]. Similar conclusions are reached by Clark, who observes that POS tags are not enough to learn syntax from ("a lot of syntax depends on the idiosyncratic properties of particular words." [21], p.36). Clark's own algorithm [20] had attempted to learn a grammar from tagged text, by starting with local distributional cues, then filtering spurious non-terminals using a mutual information criterion (namely, requiring high MI between pattern prefix and suffix). In the final stage, his algorithm clustered the results to achieve a minimum description length (MDL) representation, by starting with maximum likelihood grammar, then greedily selecting the candidate for abstraction that would maximally reduce the description length. In its greedy approach to optimization (but not in its local search for good patterns or its ability to deal with untagged data), the present approach resembles Clark's.

5.1.1.5 Probabilistic treebank-based learning.

Bod, whose algorithm learns by gathering information about corpus probabilities of potentially complex trees, observes that "[. . .] the knowledge of a speaker-hearer cannot be understood as a grammar, but as a statistical ensemble of language experiences that changes slightly every time a new utterance is perceived or produced. The regularities we observe in language may be viewed as emergent phenomena, but they cannot be summarized into a consistent non-redundant system that unequivocally

defines the structures of new utterances." ([10], p.145). Consequently, his memory- or analogy-based language model is not a typical example of unsupervised learning through redundancy reduction; I mention it here mainly because of the parallels between the data representation it employs (Stochastic Tree-Substitution Grammar [88]) and some of the formalisms discussed later, in section 5.1.2.

5.1.2 Related linguistic approaches

5.1.2.1 Cognitive Grammar.

The main methodological tenets of ADIOS — populating the lexicon with "units" of varying complexity and degree of entrenchment, and using cognition-general mechanisms for learning and representation — are very much in the spirit of the foundations of Cognitive Grammar laid down by Langacker [65]. At the same time, whereas the cognitive grammarians typically attempt to hand-craft structures that would reflect the logic of language as they perceive it, ADIOS discovers the primitives of grammar empirically rather than accept them by fiat.

5.1.2.2 Construction Grammar.

Similarities also exist between ADIOS and the various Construction Grammars [33, 23] (albeit the latter are all hand-crafted). A construction grammar consists of elements that differ in their complexity and in the degree to which they are specified: an idiom such as "big deal" is a fully specified, immutable construction, whereas the expression "the X, the Y" (as in "the more, the better"; cf. [58]) is a partially specified template. The patterns learned by ADIOS likewise vary along the dimensions of complexity and specificity (not every pattern has an equivalence class, for example). Moreover, there are reasons to believe that these patterns capture much of the semantics of the sentences from which they are abstracted, just as constructions are designed to serve as vehicles for expressing the conceptual/semantic content of

intended messages in a form compatible with the structural constraints that apply to language. A proper evaluation of this claim must wait for the emergence of a semantic theory capable of dealing with all the complexities of natural language — something that current formal theories [81] cannot do. In the meanwhile, I concur with Jackendoff's position: "[. . .] we must explicitly deny that conceptual structures [. . .] *mean* anything. Rather, we want to say that they *are* meaning: they do exactly the things meaning is supposed to do, such as support inference and judgment." ([54], p.306).

5.1.2.3 Tree Adjoining Grammar.

In capturing the regularities inherent in multiple criss-crossing paths through a corpus, ADIOS closely resembles the finite-state Local Grammar approach of Gross [39].[2] Note, however, that pattern-based representations have counterparts for each of the two composition operations, substitution and adjoining, that characterize a Tree Adjoining Grammar, or TAG, developed by Joshi and others [55]. Specifically, both substitution and adjoining are subsumed in the relationships that hold among ADIOS patterns, such as the membership of one pattern in another (cf. section 3.2). Consider a pattern $\mathcal{P}_i$ and its equivalence class $\mathcal{E}(\mathcal{P}_i)$; any other pattern $\mathcal{P}_j \in \mathcal{E}(\mathcal{P}_i)$ can be seen as substitutable in $\mathcal{P}_i$. Likewise, if $\mathcal{P}_j \in \mathcal{E}(\mathcal{P}_i)$, $\mathcal{P}_k \in \mathcal{E}(\mathcal{P}_i)$ and $\mathcal{P}_k \in \mathcal{E}(\mathcal{P}_j)$, then the pattern $\mathcal{P}_j$ can be seen as adjoinable to $\mathcal{P}_i$. Because of this correspondence between the TAG operations and the ADIOS patterns, I believe that the latter represent regularities that are best described by Mildly Context-Sensitive Language formalism [55]. Moreover, because the ADIOS patterns are learned from data, they already incorporate the constraints on substitution and adjoining that in the original TAG framework must be specified manually.

[2] There are also interesting parallels here to the Variable Order Markov (VOM) models of symbolic sequence data [70].

5.1.3 Representation.

Expressive power. Because the ADIOS patterns and equivalence classes can be translated in a straightforward manner into the form of (generally context-sensitive) rewriting rules, the representations acquired by the method described in this thesis are, in principle, powerful enough to capture most of the structure-sensitive aspects of syntax documented by linguists [80].[3] *Openness.* At the same time, because ADIOS makes no prior assumptions about the structures it seeks, it is more general than algorithms that seek to learn particular classes of grammars, such as that of [48] or [3].

5.1.4 Psychological and linguistic evidence

Recent advances in understanding the psychological role of representations based on what I call patterns and equivalence classes focus on the use of statistical cues such as conditional probabilities [87, 42, 36] and on the importance of exemplars [50] and constructions [30, 5] in children's language acquisition [9, 97, 57, 12]. Converging evidence for the centrality of pattern-like structures is provided by corpus-based studies of *prefabs* — sequences, continuous or discontinuous, of words that appear to be prefabricated, that is, stored and retrieved as a whole, rather than being subject to syntactic processing [102, 101]. About 55% of words in both spoken and written English are parts of prefabs [29]; it is no wonder, therefore, that mastery of a language depends on getting right prefabs such as "pull _'s leg" or "take it to the bank," (which is not the same as "carry it to the bank").

Similar ideas concerning the ubiquity in syntax of structural peculiarities hith-

[3] In particular, my representations transcend VOM structures (and *a fortiori* finite-state grammars). The VOM approach starts by postulating a maximum-order structure, which is then fitted to the data, usually by maximizing the likelihood of the training corpus. In comparison, rather than fitting a preset model to the data, we use the data to construct a system of recursively structured representations. Consider an ADIOS pattern of the form $p_i = p_j[c_k]p_l$ and note that its equivalence class c_k may include vertices of the graph (both terminals and patterns turned into nodes) and wild cards (any node, or no node). Thus, the terminal-level length of the strings represented by the pattern $||p_i|| = ||p_j|| + ||c_k|| + ||p_l||$, does not have to be fixed and is, indeed, ill-defined prior to learning.

erto marginalized as "exceptions" are now being voiced by linguists [24, 54] and, in particular, by typologists who compare the nature of constructions and the composition of the construction lexica across different languages [23]. The idea of populating the lexicon with "units" of varying complexity and degree of entrenchment also fits the spirit of the foundations of Cognitive Grammar [65]. It should be stressed that whereas the Cognitive grammarians typically face the chore of hand-crafting structures that would reflect the logic of language as they perceive it, ADIOS discovers the units empirically and autonomously. The same is true also for the comparison between ADIOS and the various Construction Grammars [30, 32, 58, 33, 23].

5.2 Prospects and challenges

I have compared the present approach to unsupervised learning of sequence structure (which yields good results when applied to raw corpora of language such as transcribed children-oriented speech [93]) to some recent work in computational linguistics and in grammar theory. The representations learned by the ADIOS algorithm are truly emergent from the (unannotated) corpus data, whereas those found in published works on cognitive and construction grammars and on TAGs are hand-tailored. Thus, the present results complement and extend both the computational and the more linguistically oriented research into cognitive/construction grammar.

To further the cause of an integrated understanding of language, a crucial challenge must be met: a viable approach to the evaluation of performance of an unsupervised language learner must be developed, allowing testing both (1) neutral with respect to the linguistic dogma, and (2) cognizant of the plethora of phenomena documented by linguists over the course of the past half century.

Unsupervised grammar induction algorithms that work from raw data are in principle difficult to test, because any "gold standard" to which the acquired representation can be compared (such as the Penn Treebank [73]) invariably reflects its design-

ers' preconceptions about language, which may not be valid, and which usually are controversial among linguists themselves [20]. As Wolff observes, a child "...must generalize from the sample to the language without overgeneralizing into the area of utterances which are not in the language. *What makes the problem tricky is that both kinds of generalization, by definition, have zero frequency in the child's experience.*" ([100], p.183, italics in the original). Instead of shifting the onus of explanation onto some unspecified evolutionary processes (which is what the innate grammar hypothesis amounts to), I suggest that a system such as ADIOS should be tested by monitoring its acceptance of massive amounts of human-generated data, and at the same time by getting human subjects to evaluate sentences generated by the system (note that this makes psycholinguistics a crucial component in the entire undertaking).

Such a purely empirical approach to the evaluation problem would waste the many valuable insights into the regularities of language accrued by the linguists over decades. Although some empiricists would consider this a fair price for quarantining what they perceive as a runaway theory that got out of touch with psychological and computational reality, I believe that searching for a middle way is a better idea, and that the middle way can be found, if the linguists can be persuaded to try and present their main findings in a theory-neutral manner. From recent reviews of syntax that do attempt to reach out to non-linguists (e.g., [80]), it appears that the core issues on which every designer of a language acquisition system should be focusing are dependencies (such as co-reference) and constraints on dependencies (such as island constraints), especially as seen in a typological (cross-linguistic) perspective [23].

6. BIBLIOGRAPHY

BIBLIOGRAPHY

[1] *Language*. Harcourt, Brace Co, New York, NY, 1921.

[2] P. Adriaans and M. van Zaanen. Computational grammar induction for linguists. *Grammars*, 7:57–68, 2004.

[3] P. Adriaans and M. Vervoort. The EMILE 4.1 grammar induction toolbox. In P. Adriaans, H. Fernau, and M. van Zaanen, editors, *Grammatical Inference: Algorithms and Applications: 6th International Colloquium: ICGI 2002*, volume 2484 of *Lecture Notes in Computer Science*, pages 293–295. Springer-Verlag, Heidelberg, 2002.

[4] J. Allen and M. S. Seidenberg. Grammaticality judgment and aphasia: A connectionist account. In B. MacWhinney, editor, *Emergence of Language*. Lawrence Earlbaum Associates, Hillsdale, NJ, 1999.

[5] C. F. Baker, C. J. Fillmore, and J. B. Lowe. The Berkeley FrameNet project. In *Proceedings of the COLING-ACL*, volume 1, pages 86–90. Montreal, Canada, 1998.

[6] H. B. Barlow. Sensory mechanisms, the reduction of redundancy, and intelligence. In *The mechanisation of thought processes*, pages 535–539. H.M.S.O., London, 1959.

[7] H. B. Barlow. What is the computational goal of the neocortex? In C. Koch and J. L. Davis, editors, *Large-scale neuronal theories of the brain*, chapter 1, pages 1–22. MIT Press, Cambridge, MA, 1994.

[8] H. B. Barlow. Redundancy reduction revisited. *Network: Computation in Neural Systems*, 12:241–253, 2001.

[9] E. Bates and J. C. Goodman. On the emergence of grammar from the lexicon. In B. MacWhinney, editor, *Emergence of Language*, pages 29–79. Lawrence Earlbaum Associates, Hillsdale, NJ, 1999.

[10] R. Bod. *Beyond grammar: an experience-based theory of language*. CSLI Publications, Stanford, US, 1998.

[11] C. Z. Cai, L. Y. Han, Z. L. Ji, X. Chen, and Y. Z. Chen. SVM-Prot: Web-based Support Vector Machine software for functional classification of a protein from its primary sequence. *Nucleic Acids Research*, 31:3692–3697, 2003.

[12] T. Cameron-Faulkner, E. Lieven, and M. Tomasello. A construction-based analysis of child directed speech. *Cognitive Science*, 27:843–874, 2003.

[13] Lewis Carroll. *Alice in Wonderland*, volume 11. 1991.

[14] E. Charniak. Immediate-head parsing for language models. In *Proc. ACL '01*, pages 116–123, 2001.

[15] C. Chelba. Portability of syntactic structure for language modeling. In *Proc. of the Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 544a–544d. Salt Lake City, UT, 2001.

[16] C. Chelba and F. Jelinek. Exploiting syntactic structure for language modeling. In C. Boitet and P. Whitelock, editors, *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics*, pages 225–231, San Francisco, CA, 1998. Morgan Kaufmann.

[17] N. Chipere. Individual differences in syntactic skill. *Working Papers in English and Applied Linguistics*, 4:1–32, 1997.

[18] N. Chipere. Native speaker variations in syntactic competence: implications for first language teaching. *Language Awareness*, 10:107–124, 2001.

[19] N. Chomsky. *Knowledge of language: its nature, origin, and use*. Praeger, New York, 1986.

[20] A. Clark. Unsupervised induction of Stochastic Context-Free Grammars using distributional clustering. In *Proceedings of CoNLL 2001*, Toulouse, 2001.

[21] A. Clark. *Unsupervised Language Acquisition: Theory and Practice*. PhD thesis, COGS, University of Sussex, 2001.

[22] H. H. Clark and E. V. Clark. *Psychology and language: an introduction to psycholinguistics*. Harcourt Brace Jovanovich, New York, 1977.

[23] W. Croft. *Radical Construction Grammar: syntactic theory in typological perspective*. Oxford University Press, Oxford, 2001.

[24] P. W. Culicover. *Syntactic nuts: hard cases, syntactic theory, and language acquisition*. Oxford University Press, Oxford, 1999.

[25] W. Daelemans. Toward an exemplar-based computational model for cognitive grammar. In J. van der Auwera, F. Durieux, and L. Lejeune, editors, *English as a human language*, pages 73–82. LINCOM Europa, Munchen, 1998.

[26] C. G. de Marcken. *Unsupervised Language Acquisition*. PhD thesis, MIT, 1996.

[27] S. Edelman. Constraining the neural representation of the visual world. *Trends in Cognitive Sciences*, 6:125–131, 2002.

[28] J. L. Elman, E. A. Bates, M. H. Johnson, A. Karmiloff-Smith, D. Parisi, and K. Plunkett. *Rethinking innateness: A connectionist perspective on development*. MIT Press, Cambridge, MA, 1996.

[29] B. Erman and B. Warren. The idiom principle and the open-choice principle. *Text*, 20:29–62, 2000.

[30] C. J. Fillmore. Syntactic intrusion and the notion of grammatical construction. *Berkeley Linguistic Society*, 11:73–86, 1985.

[31] S. Geman. Minimum Description Length priors for object recognition. In *Challenging the frontiers of knowledge using statistical science (Proc. JSM'96)*, 1996.

[32] A. E. Goldberg. *Constructions: A construction grammar approach to argument structure*. University of Chicago Press, Chicago, 1995.

[33] A. E. Goldberg. Constructions: a new theoretical approach to language. *Trends in Cognitive Sciences*, 7:219–224, 2003.

[34] R. L. Goldstone. Unitization during category learning. *Journal of Experimental Psychology: Human Perception and Performance*, 26:86–112, 2000.

[35] R. L. Gómez. Variability and detection of invariant structure. *Psychological Science*, 13:431–436, 2002.

[36] R. L. Gómez and L. Gerken. Infant artificial language learning and language acquisition. *Trends in Cognitive Science*, 6:178–186, 2002.

[37] J. T. Goodman. A bit of progress in language modeling: Extended version. Technical Report MSR-TR-2001-72, Microsoft Research, 2001.

[38] P. Grimes. *Data from Ethnologue: Languages of the World (14th Edition)*. SIL International, 2001.

[39] M. Gross. The construction of local grammars. In E. Roche and Y. Schabès, editors, *Finite-State Language Processing*, pages 329–354. MIT Press, Cambridge, MA, 1997.

[40] E. Gurman-Bard, D. Robertson, and A. Sorace. Magnitude estimation of linguistic acceptability. *Language*, 72:32–68, 1996.

[41] I. Guyon and F. Pereira. Design of a linguistic postprocessor using Variable Memory Length Markov Models. In *Proc. 3rd Intl. Conf. on Document Analysis and Recogition*, pages 454–457. Montreal, Canada, 1995.

[42] C. L. Harris. Psycholinguistic studies of entrenchment. In J. Koenig, editor, *Conceptual Structures, Language and Discourse*, volume 2. CSLI, Berkeley, CA, 1998.

[43] Z. S. Harris. *Methods in structural linguistics*. University of Chicago Press, Chicago, IL, 1951.

[44] Z. S. Harris. Distributional structure. *Word*, 10:140–162, 1954.

[45] Z. S. Harris. *Language and information*. Columbia University Press, New York, 1988.

[46] Z. S. Harris. *A theory of language and information*. Clarendon Press, Oxford, 1991.

[47] C. T. Hemphill, J. J. Godfrey, and G. R. Doddington. The ATIS spoken language systems pilot corpus. In *Proceedings of a workshop on Speech and natural language*, pages 96–101. Morgan Kaufmann, San Francisco, CA, 1990.

[48] P. J. Henrichsen. GraSp: Grammar learning from unlabeled speech corpora. In *Proceedings of CoNLL-2002*, pages 22–28. 2002.

[49] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, MA, 1979.

[50] P. J. Hopper. Emergent grammar. In M. Tomasello, editor, *The new psychology of language*, pages 155–175. Erlbaum, Mahwah, NJ, 1998.

[51] J. Y. Huang and D. L. Brutlag. The eMOTIF database. *Nucleic Acids Research*, 29:202–204, 2001.

[52] S. Hunston and G. Francis. *Pattern grammar: a corpus-driven approach to the lexical grammar of English*. John Benjamins, Amsterdam, 2000.

[53] J. Hutchens, M. Alder, and Y. Attikiouzel. Natural language grammatical inference. Technical Report HT94-03, University of Western Australia, 1994.

[54] R. Jackendoff. *Foundations of language*. Oxford University Press, Oxford, 2002.

[55] A. Joshi and Y. Schabes. Tree-Adjoining Grammars. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 69–124. Springer, Berlin, 1997.

[56] D. Jurafsky, C. Wooters, J. Segal, A. Stolcke, E. Fosler, G. Tajchman, and N. Morgan. Using a Stochastic Context-Free Grammar as a language model for speech recognition. In *Proc. Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 189–192, Detroit, MI, 1995.

[57] K. Karmiloff and A. Karmiloff-Smith. *Pathways to language*. Harvard University Press, Cambridge, MA, 2001.

[58] P. Kay and C. J. Fillmore. Grammatical constructions and linguistic generalizations: the What's X Doing Y? construction. *Language*, 75:1–33, 1999.

[59] C. Kermorvant, C. de la Higuera, and P. Dupont. Improving probabilistic automata learning with additional knowledge. In *Lecture Notes in Computer Science*, volume 3138. Springer, Germany, 2004.

[60] C. Kermorvant, C. de la Higuera, and P. Dupont. Learning typed automata from automatically labeled data. *Journal électronique d'intelligence artificielle*, 6(45), 2004.

[61] D. Klein and C. D. Manning. Natural language grammar induction using a constituent-context model. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 35–42. MIT Press, Cambridge, MA, 2002.

[62] D. Klein and C. D. Manning. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting of the ACL*, pages 478–485. 2004.

[63] R. Kneser and H. Ney. Improved backing-off for N-gram language modeling. In *Proc. Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 181–184, 1995.

[64] A. Lampel and J. Ziv. A universal algorithm for sequential data compression. *IEEE Trans. Inform. Theory*, 23:337–343, 1977.

[65] R. W. Langacker. *Foundations of cognitive grammar*, volume I: theoretical prerequisites. Stanford University Press, Stanford, CA, 1987.

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

[67] S. Lawrence, C. L. Giles, and S. Fong. Natural language grammatical inference with recurrent neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 12:126–140, 2000.

[68] M. C. Linebarger, M. Schwartz, and E. Saffran. Sensitivity to grammatical structure in so-called agrammatic aphasics. *Cognition*, 13:361–392, 1983.

[69] M. C. MacDonald. Distributional information in language comprehension, production, and acquisition: Three puzzles and a moral. In B. MacWhinney, editor, *Emergence of Language*, pages 177–196. Lawrence Earlbaum Associates, Hillsdale, NJ, 1999.

[70] M. Mächler and P. Bühlmann. Variable Length Markov Chains: Methodology, computing and software. Seminar for Statistics Report 104, ETH Zürich, 2002.

[71] B. MacWhinney and C. Snow. The Child Language Exchange System. *Journal of Computational Lingustics*, 12:271–296, 1985.

[72] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, 1999.

[73] M. Marcus. The penn treebank: A revised corpus design for extracting predicate-argument structure. In *Proceedings of the ARPA Human Language Technology Workshop*, Princeton, NJ, 1994. Morgan-Kaufman.

[74] E. Markman. *Categorization and naming in children*. MIT Press, Cambridge, MA, 1989.

[75] R. C. Martin and M. D. Miller. Sentence comprehension deficits: Independence and interaction of syntax, semantics, and working memory. In A. Hillis, editor, *Handbook of Adult Language Disorders: Integrating Cognitive Neuropsychology, Neurology, and Rehabilitation*. Psychology Press, New York, 2002.

[76] M. McCandless and J. Glass. Empirical acquisition of word and phrase classes in the ATIS domain. In *Proc. EuroSpeech'93*, pages 981–984. 1993.

[77] M. A. Nowak, N. L. Komarova, and P. Niyogi. Evolution of universal grammar. *Science*, 291:114–119, 2001.

[78] F. Pereira. Formal grammar and information theory: Together again? *Philosophical Transactions of the Royal Society*, 358(1769):1239–1253, 2000.

[79] F. Pereira and Y. Schabès. Inside-outside reestimation from partially bracketed corpora. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 128–135, 1992.

[80] C. Phillips. Syntax. In L. Nadel, editor, *Encyclopedia of Cognitive Science*, volume 4, pages 319–329. Macmillan, London, 2003.

[81] P. M. Pietroski. The character of natural language semantics. In A. Barber, editor, *Epistemology of Language*. Oxford University Press, Oxford, UK, 2003. to appear.

[82] S. Pinker and P. Bloom. Natural language and natural selection. *Behavioral and Brain Sciences*, 13:707–784, 1990.

[83] N. Redlich. Redundancy reduction as a strategy for unsupervised learning. *Neural Computation*, 5:289–304, 1993.

[84] P. Resnik, M. B. Olsen, and M. Diab. The Bible as a parallel corpus: annotating the 'Book of 2000 Tongues'. *Computers and the Humanities*, 33:129–153, 1999.

[85] B. Roark. Probabilistic top-down parsing and language modeling. *Computational Linguistics*, 27:249–276, 2001.

[86] A. Roberts and E. Atwell. Unsupervised grammar inference systems for natural language. Technical Report 2002.20, School of Computing, University of Leeds, 2002.

[87] J. R. Saffran, R. N. Aslin, and E. L. Newport. Statistical learning by 8-month-old infants. *Science*, 274:1926–1928, 1996.

[88] R. Scha, R. Bod, and K. Sima'an. A memory-based model of syntactic analysis: data-oriented parsing. *J. of Experimental and Theoretical Artificial Intelligence*, 11:409–440, 1999.

[89] C. T. Schütze. *The empirical base of linguistics: grammaticality judgments and linguistic methodology*. University of Chicago Press, Chicago, IL, 1996.

[90] Z. Solan, D. Horn, E. Ruppin, and S. Edelman. Unsupervised context sensitive language acquisition from a large corpus. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press.

[91] Z. Solan, D. Horn, E. Ruppin, and S. Edelman. Unsupervised learning of natural languages. *Proceedings of the National Academy of Science*, 102, 2005.

[92] Z. Solan, D. Horn, E. Ruppin, S. Edelman, M. Lapidot, S. Kaplan, and Y. Pilpel. Motif extraction from promoter regions of s. cerevisiae, 2004. unpublished manuscript.

[93] Z. Solan, E. Ruppin, D. Horn, and S. Edelman. Automatic acquisition and efficient representation of syntactic structures. In S. Thrun, editor, *Advances in Neural Information Processing*, volume 15, Cambridge, MA, 2003. MIT Press.

[94] Z. Solan, E. Ruppin, D. Horn, and S. Edelman. Unsupervised efficient learning and representation of language structure. In R. Alterman and D. Kirsh, editors, *Proc. 25th Conference of the Cognitive Science Society*, Hillsdale, NJ, 2003. Erlbaum.

[95] L. Steels. The emergence of grammar in communicating autonomous robotic agents. In W. Horn, editor, *Proceedings of ECAI 2000*, pages 764–769, Amsterdam, 2000. IOS Publishing.

[96] A. Stolcke and S. Omohundro. Inducing probabilistic grammars by Bayesian model merging. In R. C. Carrasco and J. Oncina, editors, *Grammatical Inference and Applications*, pages 106–118. Springer, 1994.

[97] M. Tomasello. The item-based nature of children's early syntactic development. *Trends in Cognitive Science*, 4:156–163, 2000.

[98] M. van Zaanen. ABL: Alignment-Based Learning. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 961–967. 2000.

[99] M. van Zaanen and P. Adriaans. Comparing two unsupervised grammar induction systems: Alignment-based learning vs. EMILE. Report 05, School of Computing, Leeds University, 2001.

[100] J. G. Wolff. Learning syntax and meanings through optimization and distributional analysis. In Y. Levy, I. M. Schlesinger, and M. D. S. Braine, editors, *Categories and Processes in Language Acquisition*, pages 179–215. Lawrence Erlbaum, Hillsdale, NJ, 1988.

[101] A. Wray. *Formulaic language and the lexicon*. Cambridge University Press, Cambridge, UK, 2002.

[102] A. Wray and M. R. Perkins. The functions of formulaic language: an integrated model. *Language and communication*, 20:1–28, 2000.

7. APPENDIX - THE ADIOS SITE