4

On Induction of Morphology Grammars and its Role in Bootstrapping

Damir Ćavar, Joshua Herring, Toshikazu Ikuta, Paul Rodrigues, Giancarlo Schrementi

Different Alignment Based Learning (ABL) algorithms have been proposed for unsupervised grammar induction, e. g. Zaanen (2001) and Déjean (1998), in particular for the induction of syntactic rules. However, ABL seems to be better suited for the induction of morphological rules. In this paper we show how unsupervised hypothesis generation with ABL algorithms can be used to induce a lexicon and morphological rules for various types of languages, e. g. agglutinative or polysynthetic languages. The resulting morphological rules and structures are optimized with the use of conflicting constraints on the size and statistical properties of the grammars, i. e. Minimium Description Length and Minimum Relative Entropy together with Maximum Average Mutual Information. Further, we discuss how the resulting (optimal and regular) grammar can be used for lexical clustering/classification for the induction of syntactic (context free) rules.

4.1 Introduction

In previous approaches grammar induction algorithms consisted of three fundamental phases, see e. g. van Zaanen and Adriaans (2001), Zaanen (2001), Déjean (1998):

Proceedings of Formal Grammar 2004.
Paola Monachesi, Gerhard Jäger,
Gerald Penn and Shuly Wintner (eds.).

- Hypothesis generation
- Hypothesis selection
- Induction

From the computational perspective the main problems lie on the generational and selectional level. Both of these components try to reduce the set of hypotheses about the structure of selected natural language input to the smallest possible amount that provides the best coverage given a targeted formalism or description level. That is, it tries to maximize relevant and minimize irrelevant hypotheses based on the need to reduce computational complexity and errors in the final induction phase. Thus, the art of grammar induction is to find the equilibrium between the amount of hypotheses generated and the effort invested to select the *best* candidates.

In what follows, we will discuss the results from investigations into unsupervised grammar induction algorithms that make use of *string alignment* for hypothesis generation driven purely by previous experience, or, in other words, by the lexicon and the hypotheses generated at every step in the incremental consumption and induction procedure. ABL is such an approach, see for example Zaanen (2001). Its proponents have thus far hesitated to recognize ABL as an approach that is attractive from computational as well as a cognitive perspectives. ABL constrains the hypothesis space from the outset to the set of hypotheses that are motivated by previous experience/input or a preexisting grammar. Such constraining characteristics make ABL attractive from a cognitive point of view, both because the computational complexity is reduced on account of the reduced set of potential hypotheses, and also because the learning of new items, rules, or structural properties is related to a general learning strategy and previous experience only. The approaches that are based on a brute-force first order explosion of hypotheses with subsequent filtering of relevant or irrelevant structures are both memory intensive and require more computational effort.

The basic concepts in ABL go back to notions of *substitutability* and/or *complementarity*, as discussed in Harris (1955) and Harris (1961). The concept of *substitutability* is used in the central part of the induction procedure itself, the assumption being that substitutable elements (e. g. substrings, words, structures) are assumed to be of the same type (represented e. g. with the same symbol). The notion of "same type" is not uncontroversial. Its use in syntax as a test for membership in a particular "part-of-speech" category, for example, was rightly criticized in Pinker (1994) and Chomsky (1955). However, it remains a rather reliable constituent test, and is certainly reliable in the sense in which it is understood in this paper. By "substitutability," we understand not so much an instrument to identify consitutents of the same type but rather of a method of identifying constituents as such. The typing of constituents could be the result of independent components that use alignment information and other statistical properties.

Nevertheless, the ABL approach has disadvantages if not used properly. The size of the grammar can affect the runtime behavior of ABL systems, as can learning errors. In the following, we will describe our implementation and the use case, the problems and solutions for a grammar induction algorithm based on ABL.

While ABL is used in a slightly restricted way for hypothesis generation, we make use of different methods in the evaluation component to reduce the error rate and increase the performance of the algorithm, both with respect to the runtime behavior as well as the output quality. Interacting weighted constraints are used to increase the efficiency of the resulting grammar and eliminate irrelevant structural descriptions. In particular, the central constraints we use are:

- Maximum Average Mutual Information (MI), which requires that the mutual information between segments of a structural description and the complete structural descriptions acquired so far (the hypotheses space for the induction procedure), is maximized. Hypotheses that maximally contribute to the average mutual information are preferred.
- Minimum Relative Entropy (RE), which requires that the relative entropy for all resulting structural descriptions is minimized.
- Minimum Description Length (MDL), which requires that the size of the resulting grammar (including structural descriptions) is minimized.

All constraints are used for evaluation and selection of the best hypotheses by taking into account the properties of the resulting grammar and the structural descriptions it generates. All the constraints seem to be well-motivated from a cognitive perspective, assuming that the cognitive resources are limited with respect to e. g. memory capacity and processing time. Grammar acquisition is seen as a compression process that emerges under memory and processing time restrictions, i. e. compressing the language input as much as possible while maintaining online usability. The compression ratio is limited by the processing capacities and time constraints imposed by language use. Unlimited creativity is thus seen here to be a side effect of complementary constraints of memory driven compression (grammar induction) and time and processing driven usability. The algorithm is parametrized along these lines, allowing for fine grained restrictions of the runtime environment. This potentially allows us to test the impact of the different constraints on the resulting grammar.

One of the underlying research hypotheses here is also that a large amount of valuable syntactic information can be induced if information from other linguistic domains is used. That is, if large parts of the morphological, phonological or prosodic restrictions can be induced, assuming that these are to a great extent regular, we expect this information to be used as bootstraps to syntactic structure, assuming that this is mainly context free (or mildly context sensitive). In other words, the hypothesis here is that at least parts of context free grammar can be learned if regular grammars are used that describe or generate parts of natural language input. This is how we understand bootstrapping in grammar induction or natural language acquisition.

Thus, the design criteria for the algorithm presented here are computational, cognitive, and linguistic in nature, and though we assume that the algorithm can be used in virtually all linguistic domains (prosody, phonology, syntax), our concern here is mainly with the induction of morphological structure and any underlying rules that might (or might not) be used to describe such structure. The main research question is to what extent can we use this type of algorithm to induce morphological grammars that can then serve, together with the morphological terminals, as cues for syntactic structure and rules.

4.2 Specification of the algorithm

On the basis of the design criteria for the algorithm, as discussed above, a first implementation of the ABL-based induction algorithm was incremental. The algorithm was designed as an iterative procedure that consumes utterances, generates hypotheses for the morphological structure of every individual word in the utterance, and adds the most accurate hypotheses to the grammar and/or lexicon. The entire cycle of hypothesis generation, evaluation and induction is passed through for each word in the input. [1]

The variant of the ABL algorithm for morphology that we are using makes use of simple substring maching, described in further detail below. If a morpheme is found as a submorpheme in an input word, its edges are assumed to represent potential morpheme boundaries within that word. We apply the restriction that only words that occure as

[1] A Python implementation and more detailed information is available at `http://jones.ling.indiana.edu/ abugi/`.

independent morphemes are used for this alignment based hypothesis generation.

In order to reduce the computational complexity of the algorithm, we take all generated rules and lexical entries as final if there is more than one occurrence of a similar pattern in the grammar. There is no revision of the made hypotheses that enter the final hypothesis space and are part of the structural descriptions (SD) that serve as input to the induction procedure, generated wrong SDs being expected to become statistically insignificant as the incremental grammar induction process runs.

As mentioned, the input to the algorithm is a list of words that is processed incrementally. That is, we make use of word boundaries, presupposing an existing segmentation of the input. For the purpose here, and to generate better results, we focus on the generation of morphological segmentation for a given set of words. In principle, however, there is no reason why the same algorithm, with minor refinements, couldn't be used to segment a raw string of alphanumeric characters into usable units separated by word boundaries. Indeed, we believe that a defining feature of our work, and one which distinguishes it from previous work, e. g. Goldsmith (2001), is that there is a concentrated attempt to eliminate all built-in knowledge from the system. The algorithm starts with a clean state and uses only statistical, numerical, and string matching techniques in an effort to remain as close as possible to a cognitive model, with a central focus on unsupervised induction. The main goal of this strategy is to identify the algorithms that allow for induction of specific linguistic knowledge, and to identify the possibly necessary supervision for each algorithm type.

Thus we assume for the input:

- Alphabet: a non-empty set A of n symbols $\{s_1, s_2, \ldots s_n\}$
- Word: a word w a non-empty list of symbols $w = [s_1, s_2, \ldots s_n]$, with $s \in A$
- Corpus: a non-empty list C of words $C = [w_1, w_2, \ldots w_n]$

The output of the ABL hypothesis generation is a set of hypotheses for a given input word. A hypothesis is a tuple:

- $H =< w, f, g >$, with w the input word, f its frequency in C, and g a list of substrings that represent a linear list of morphemes i w, $g = [m_1, m_2, \ldots m_n]$

The hypotheses are collected in a hypotheses space. The hypothesis space is defined as a list of hypotheses:

- Hypotheses space: $S = [H_1, H_2, \ldots H_n]$

The algorithm does not make any assumptions about types of morphemes. There is no expectation of specific structure in the input, nor does it use notions like *stem*, *prefix*, or *suffix*. We assume only linear sequences. The fact that single morphemes exist as stems or suffixes is a side effect of their statistical properties (including Frequency and left and right Pointwise Mutual Information, a term that will be explained below) and alignment within the corpus, or rather within words.

There are no language specific rules built-in, such as what a morpheme must contain or how frequent it should be. All of this knowledge is learned, based on statistical analysis of prior experience. However, as discussed in the last section, at certain points in the learning procedure we lose the performance benefit of not relying on such rules to escape linguistic and statistical anomalies that might lead the program astray.

Each iteration of the incremental learning process consists of the following steps:

1. ABL Hypotheses Generation
2. Hypotheses Evaluation and Selection
3. Grammar Extension

In the ABL Hypotheses Generation, a given word in the utterance is checked against all the morphemes in the grammar. If an existing morpheme m aligns with the input word w, a hypothesis is generated suggesting a morphological boundary at the alignment positions:

$$w(speaks) + m(speak) = H[speak, s] \tag{4.1}$$

Another design criterion for the algorithm is complete language independence. It should be able to identify morphological structures of Indo-European type of languages, as well as agglutinative languages (e. g. Japanese and Turkish) and polysynthetic languages like some Bantu dialects or Native American languages like Lakhota. In order to guarantee this behavior, we extended the Alignment Based hypothesis generation with a pattern identifier that extracts patterns of character sequences of the types:

1. A – B – A
2. A – B – A – B
3. A – B – A – C

This component is realized with cascaded finite state transducers that are able to identify and return the substrings that correspond to the repeating sequences.[2]

[2]This addition might be understood to be a sort of *supervision* in the system.

All possible alignments for the existing grammar at the current state are collected and evaluated. The Hypothesis Evaluation and Selection step uses a set of different criteria to find the best hypotheses. The following evaluation criteria are used:

- Maximization of Pointwise Mutual Information between the morphemes
- Minimization of the Relative Entropy for the resulting grammar
- Minimization of the Description Length for the resulting grammar
- Minimization of the number of morphemes
- Maximization of the length of morphemes
- Maximization of the frequency of a morpheme boundary over all ABL Hypotheses

Each of these criteria is weighted relative to the others. While the different choices for the relative weights of the criteria were partially arbitrary in our experiments,[3] the choice of criteria is not.

The criteria are related to assumptions we make about cognitive aspects of language and grammar. Specifically, we assume that the properties of natural language grammars are constrained by limited memory resources resulting in a preference for smaller grammars which are maximally efficient in terms of run time, computational complexity and memory space consumption. We employ an interaction of MDL, Maximum MI and Minimum RE to reach an optimally- sized and efficient grammar. We relate efficiency to Information Theoretic notions of coding length, channel capacity and transmission time, as well as symbol replacement operations for the processing and generation of natural language utterances. Thus, indirectly the number of morphemes and their length is related to usability aspects, since the number of morphemes is related to the number of symbols used in induced rules, and thus to the number of replacement operations in processing and generation. Along these lines we group the above listed evaluation constraints into memory and usability oriented constraints.

The choice of evaluation criteria is also influenced by the expectation that languages will differ with respect to the importance of particular constraints at specific linguistic levels. The well-known correlation between richness of morphology and restrictiveness of word order as well

However, as shown in recent research on human cognitive abilities, and especially on the ability to identify patterns in the speech signal by very young infants Marcus et al. (1999), we can assume such an ability to be part of the general cognitive endowment, maybe not even language specific.

[3]Currently we are working on automatic adaption of these weights during the learning process. This is potentially the locus of a self-supervision strategy, as also pointed out by one reviewer.

as the quantitative correlation between the number of words per utterance and the number of morphemes per word is expected to be due to different weights on a set of constraints that natural languages are subject to.

In the following sections the components of the evaluation module are described in more detail.

4.2.1 Mutual Information (MI)

For the purpose of this experiment we use a variant of standard Mutual Information (MI), see Charniak (1996) and MacKay (2003) for some use cases. Information theory tells us that the presence of a given morpheme restricts the possibilities of the occurrence of morphemes to the left and right, thus lowering the amount of bits needed to store its neighbors. Thus we should be able to calculate the amount of bits needed by a morpheme to predict its right and left neighbors respectively. To calculate this, we have designed a variant of mutual information that is concerned with a single direction of information.

This is calculated in the following way. For every morpheme y that occurs to the right of x we sum the pointwise MI between x and y, but we relativize the pointwise MI by the probability that y follows x, given that x occurs. This then gives us the expectation of the amount of information that x tells us about which morpheme will be to its right. Note that $p(<x,y>)$ is the probability of the bigram $<x,y>$ occurring and is not equal to $p(<y,x>)$ which is the probability of the bigram $<y,x>$ occuring.

We calculate the MI on the right side of $x \in G$ by:

$$\sum_{y \in \{<x,Y>\}} p(<x,y>|x) lg \frac{p(<x,y>)}{p(x)p(y)} \tag{4.2}$$

and the MI on the left of $x \in G$ respectively by:

$$\sum_{y \in \{<Y,x>\}} p(<y,x>|x) lg \frac{p(<y,x>)}{p(y)p(x)} \tag{4.3}$$

One way we use this as a metric, is by summing up the left and right MI for each morpheme in a hypothesis. We then look for the hypothesis that results in the maximal value of this sum. The tendency for this to favor hypotheses with many morphemes is countered by our criterion of favoring hypotheses with fewer morphemes, a topic we will discuss in greater detail below.

Another way to use the left and right MI is in judging the quality of morpheme boundaries. In a good boundary, the morpheme on the left side should have high right MI and the morpheme on the right should have high left MI. Unfortunately, MI is not initially very reliable because of the low frequency of many words, and removing hypotheses with poor boundaries prevents the algorithm from bootstrapping itself as all boundaries are poor in the beginning. We are currently experimenting with phasing this in as MI is deemed more reliable in making these judgments.

4.2.2 Description Length (DL)

The principle of Minimum Description Length (MDL) as used in recent work on grammar induction and unsupervised language acquisition, e. g. Goldsmith (2001), Marcken (1996), and Grünwald (1998), explains the grammar induction process as an iterative minimization procedure of the grammar size, where the smaller grammar corresponds to the *best* grammar for the given data/corpus.

The description length metric, as we use it here, tells us how many bits of information would be required to store a word given a hypothesis of the morpheme boundaries, using our grammar. For each morpheme in the hypothesis that doesn't occur in the grammar we need to store the string representing the morpheme. For morphemes that do occur in our grammar we just need to store a pointer to that morpheme's entry in the grammar. We use a simplified calculation, taken from Goldsmith (2001), of the cost of storing a string that takes the number of bits of information required to store a letter of the alphabet and multiply it by the length of the string.

$$lg(length(A)) * len(morpheme) \tag{4.4}$$

We have two different methods of calculating the cost of the pointer. The first takes a cue from Morse code and gives a variable cost depending on the frequency of the morpheme that it is pointing to. So first we calculate the frequency rank of the morpheme being pointed to, (e. g. the most frequent has rank 1, the second rank 2, etc.). We then calculate:

$$floor(lg(freqrank) - 1) \tag{4.5}$$

to get a number of bits similar to the way Morse code assigns lengths to various letters.

The second is simpler and only calculates the entropy of the grammar of morphemes and uses this as the cost of all pointers to the grammar. The entropy equation is as follows:

$$\sum_{x \in G} p(x) lg \frac{1}{p(x)} \tag{4.6}$$

The second equation doesn't give variable pointer lengths, but it is preferred since it doesn't carry the heavy compuational burden of calculating the frequency rank.

We calculate the description length for each hypothesis individually by summing up the cost of each morpheme in the hypothesis. Those with low description lengths are favored. Note that we do not calculate the sizes of the grammars with and without any given hypothesis. The computational load of the algorithm is thus significantly reduced; by calculating only the relative increase for every suggested hypothesis and favoring the hypothesis with the smallest increase in overall description length, a large number of potentially wasteful computational steps are avoided. In subsequent versions of the algorithm the description length will be calculated on the basis of the resulting grammars after the induction step.

4.2.3 Relative Entropy (RE)

We use RE as a measure for the cost of adding a hypothesis to the existing grammar. We look for hypotheses that, when added to the grammar, will result in minimal divergence from the original.

We calculate RE as a variant of the Kullback-Leibler Divergence, see e. g. Charniak (1996) and MacKay (2003). Given grammar G_1, the grammar generated so far, and G_2, the grammar with the extension generated for the new input increment, $P(X)$ is the probability mass function (*pmf*) for grammar G_2, and $Q(X)$ the *pmf* for grammar G_1:

$$\sum_{x \in X} P(x) lg \frac{P(x)}{Q(x)} \tag{4.7}$$

Note that with every new iteration a new element can appear that is not part of G_1. Our variant of RE takes this into account by calculating the costs for such a new element x to be the point-wise entropy of this element in $P(X)$, summing up over all new elements:

$$\sum_{x \in X} P(x) lg \frac{1}{P(x)} \tag{4.8}$$

These two sums then form the RE between the original grammar and the new grammar with the addition of the hypothesis. Hypotheses with low RE are favored.

This metric behaves similarly to description length, discussed above, in that both calculate the distance between our original grammar and the grammar with the inclusion of the new hypothesis. The primary difference is that RE also takes into account how the probability mass function differs between the two grammars and that our variation punishes new morphemes based upon their frequency relative to the frequency of other morphemes. MDL does not consider frequency in this way, which is why we include RE as metric. We are currently investigating this to identify under what conditions they behave differently.

4.2.4 Further Metric

In addition to the mentioned metric, we take into account the following criteria:

- Frequency of Morpheme Boundaries
- Number of Morpheme Boundaries
- Length of Morphemes

The frequency of morpheme boundaries is given by the number of hypotheses that contain this boundary. The basic intuition is that the higher this number, i. e. the more alignments are found at a certain position within a word, the more likely this position represents a morpheme boundary. We favor hypotheses with high values for this criterion.

To prevent the algorithm from degenerating into a state where each letter is identified as a morpheme, we favor hypotheses with lower numbers of morpheme boundaries. For this same reason, we also take morpheme length into account, prefering hypotheses with longer morphemes (again, to avoid running into a situation where every letter in a word is taken to be morphologically significant).

4.2.5 Pre-grammar generation

With every successful evaluation of hypotheses a set of signatures for each morpheme is generated, similar to the approach suggested in Goldsmith (2001). An example signature is given in the following, where the symbol X represents the slot for the respective morpheme left of the arrow.

- `clock` → `[[['X', 's$'], 1], [['X'], 1]]`

The signature contains the possible realizations of the word *clock*, either with the suffix *s* or alone. Each possible realization contains the count of total occurencies of the respective word form.

With every evaluation result, the potential hypotheses are evaluated on the basis of the existing grammar by calculating the likelihood of the new potential signature, given the existing signatures in the grammar.

A hypothesis that fits into the general pattern found in the grammar is prefered.

Grammar generation is performed by replacement of all words with equal signatures with a symbol and merger of all signatures to one. The resulting grammar represents the basis for calculations of the description length. Further, the resulting signatures are used to derive type information for the respective morphemes, as described below.

4.3 Evaluation

We used two methods to evaluate the performance of the algorithm. The first analyzes the accuracy of the morphological rules produced by the algorithm after an increment of n words. The second looks at how accurately the algorithm parsed each word that it encountered as it progressed through the corpus.

The first analysis looks at each grammar rule generated by the algorithm and judges it on the correctness of the rule and the resulting parse. A grammar rule consists of a stem and the suffixes and prefixes that can be attached to it, similar to the signatures used in Goldsmith (2001). The grammar rule was then marked as to whether it consisted of legitimate suffixes and prefixes for that stem and also as to whether the stem of the rule was a true stem, as opposed to a stem plus another morpheme that wasn't identified by the algorithm. The number of rules that were correct in these two categories were then summed, and precision and recall figures were calculated for the trial. The trials described in the graph below were run on three increasingly large portions of the general ficiton section of the Brown Corpus. The first trial was run on one randomly chosen chapter, the second trial on two chapters, and the third trial run on three chapters. The graph in Figure 1 shows the harmonic average (F-score) of precision and recall.

The second analysis is conducted as the algorithm is running and examines each parse the algorihm produces. The algorithm's parses are compared with the correct morphological parse of the word using the following method to derive a numerical score for a particular parse. The first part of the score is the distance in characters between each morphological boundary in the two parses, with a score of one point for each character apart in the word. The second part is a penalty of two points for each morphological boundary that occurs in one parse and not the other. These scores were examined within a moving window of words that progressed through the corpus as the algorithm ran. The average scores of words in each such string of words were calculated as the window advanced. The purpose of these windows was to allow the

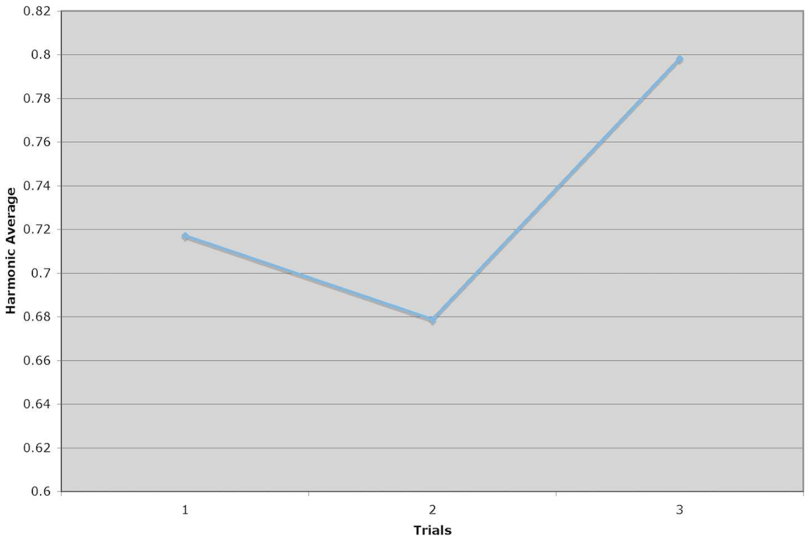

FIGURE 1 Precision and recall

performance of the algorithm to be judged at a given point without prior performance in the corpus affecting the analysis of the current window. The following chart shows how the average performance of the windows of analyzed words as the algorithm progresses through five randomly chosen chapters of general fiction in the Brown Corpus amounting to around 10,000 words. The window size for the chart in Figure 2 was set to 40 words.

We are currently performing detailed testing of the algorithm on Esperanto and Japanese corpora. The highly regular morphology of Esperanto should provide an interesting comparison against the fractured morphology of English. Likewise, the agglutinative nature of Japanese should provide a fertile test bed for morphological analysis.

The primary experiments conducted to date have been performed using the Brown Corpus of Standard American English, consisting of 1,156,329 words from American texts printed in 1961 organized into 59,503 utterances and compiled by W. N. Francis and H. Kucera at Brown University.[4]

[4]Additional experiments were done each for Classical Japanese and Esperanto. The Japanese experiment used a roman-character version of "Genji Monogatari" (The Tales of Genji), compiled by Prof. Eichi Shibuya of Takachiho University. Due to the highly regular (and pervasive) nature of the morphology, Esperanto provided an interesting frame for comparison. Tests were conducted on two corpora compiled

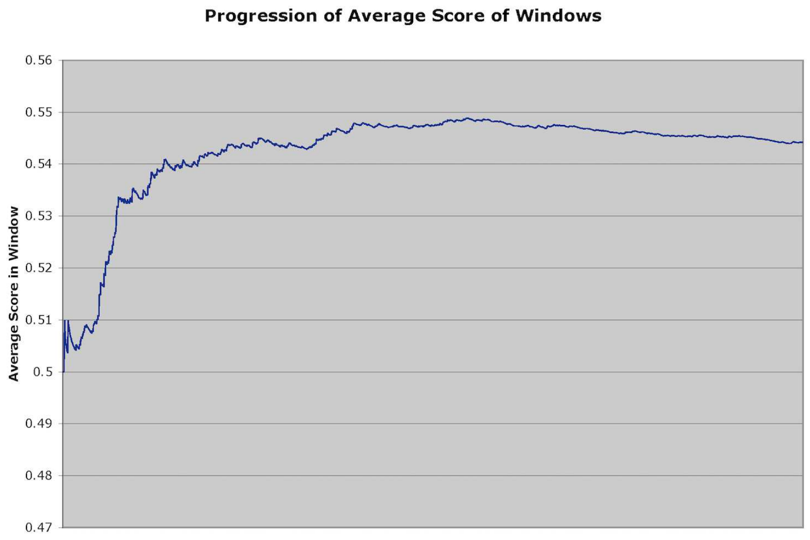

FIGURE 2 Average performance of the windows of analyzed words

4.4 Conclusion

The algorithm generates very good structures for the initial input, achieving, under certain settings, a precision of up to 100% (meaning that it returns a wordlist consisting entirely of "usable" words). Recall was significantly less accurate, but still respectable, scoring in the 60% range on the settings that reached 100% precision. It will have been noted in the graph provided above that the process is also quite stable and improves steadily (if slowly) over time, never falling even temporarily behind.

Our main focus in this project was to derive the necessary type information for words that can be used in the induction of syntactic structures. As discussed in Elghamry and Ćavar (2004), the type information can be used in a cue-based learning system to derive higher-level grammatical rules, up to the level of syntactic frames. The high levels of precision achived suggest that errors in the input will not be a barrier in this next step. Using the morphological information discovered here, it should be possible to induce word types based on their morphological signatures (in context). The main concern would be whether the algorithm generates results with high enough recall to provide a sufficient amount of information on which to base such an induction. The recall

from the Internet.

numbers achieved in our experiments strongly suggest that it does.

The weights of the system are not fixed and can be adjusted to increase recall, decreasing precision. This might be of relevance for other domains and applications of this approach.

Ongoing studies with different language types will help us in the development of the necessary self-supervision component, especially in the adaptation of the weights of the evaluation constraints during runtime. Given the post-evaluation component that evaluates the relevance of signatures for words, we are already able to predict that certain weights should be reduced, specifically those that are responsible for the generation of irrelevant hypotheses. More results will be available after detailed evaluation on data from agglutinative and synthetic or polysynthetic languages.

References

Charniak, Eugene. 1996. *Statistical language learning*. Cambridge, Mass.: MIT Press, 1st edn.

Chomsky, Noam. 1955. *The Logical Structure of Linguistic Theory*. Distributed by Indiana University Linguistics Club. Published in part by Plenum, 1975.

Déjean, Hervé. 1998. *Concepts et algorithmes pour la découverte des structures formelles des langues*. doctoral dissertation, Université de Caen Basse Normandie.

Elghamry, Khaled and Damir Ćavar. 2004. Bootstrapping cues for cue-based bootstrapping. Mscr. Indiana University.

Goldsmith, John. 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics* 27(2):153–198.

Grünwald, Peter. 1998. *The Minimum Description Length Principle and Reasoning under Uncertainty*. doctoral dissertation, Universiteit van Amsterdam.

Harris, Zellig S. 1955. From phonemes to morphemes. *Language* 31(2):190–222.

Harris, Zellig S. 1961. *Structural linguistics*. Chicago: University of Chicago Press. Published in 1951 under title: Methods in structural linguistics.

MacKay, David J. C. 2003. *Information Theory, Inference, and Learning Algorithms*. Cambridge: Cambridge University Press.

Marcken, Carl G. de. 1996. *Unsupervised Language Acquisition*. Phd dissertation, Massachusetts Institute of Technology.

Marcus, G. F., S. Vijayan, S. Bandi Rao, and P. M. Vishton. 1999. Rule-learning in seven-month-old infants. *Science* 283:77–80.

Pinker, Steven. 1994. *The language instinct*. New York, NY: W. Morrow and Co.

van Zaanen, Menno M. and Pieter Adriaans. 2001. Comparing two unsupervised grammar induction systems: Alignment-based learning vs. emile. Tech. Rep. TR2001.05, University of Leeds.

Zaanen, Menno M. van. 2001. *Bootstrapping Structure into Language: Alignment-Based Learning*. Doctoral dissertation, The University of Leeds.