Prior Knowledge in Learning Finite Parameter Spaces

Dorota Głowacka, Louis Dorard, Alan Medlar and John Shawe-Taylor

Department of Computer Science, University College London

Abstract. We propose a new framework for computational analysis of language acquisition in a finite parameter space, for instance, in the "principles and parameters" approach to language. The *prior knowledge multi-armed bandit* algorithm abstracts the idea of a casino of slot machines in which a player has to play machines in order to find out how good they are, but where he has some prior knowledge that some machines are likely to have similar rates of reward. Each grammar is represented as an arm of a bandit machine with the mean-reward function drawn from a Gaussian Process specified by a covariance function between grammars. We test our algorithm on a ten-parameter space and show that the number of iterations required to identify the target grammar is much smaller than the number of all the possible grammars that the learner would have to explore if he was searching exhaustively the entire parameter space.

1 Introduction

A major aspect of linguistic theory is to provide an explanation as to how children, after being exposed to limited data, acquire the language of their environment. The rise of "principles and parameters" [4] provided a new context for the study of language acquisition. In this approach, a class of languages can be viewed as fixed by parametric variation of a finite number of variables. Thus, acquisition of language (grammar)[1] amounts to fixing correctly the parameters of the grammar that the child is trying to learn. The notion of finite parametrisation of grammar can be applied to syntax [4], phonological stress systems [6] or even lexical knowledge [9]. In this paper, we will concentrate on the analysis of the "principles and parameters" framework as applied to stress systems [6]. This choice has been prompted mainly by two considerations. First, stress systems can be studied in relative independence of other aspects of grammars, i.e. syntax or semantics. Second, the parameters of metrical theory exhibit intricate interactions that exceed in complexity the syntactic parameters.

Starting with Gold's seminal paper [8], most research on learnability concentrates on the issue of convergence in the limit. The learner receives a sequence of positive examples from the target language. After each example the learner either stays in the same state (does not change any of the parameters) or moves to

[1] In the remainder of this paper we will use the terms *language* and *grammar* interchangeably

a new state (changes its parameter setting). If, after a finite number of examples the learner converges to the target language and never changes his guess, then the target language has been identified in the limit. In the Triggering Learning Algorithm (TLA) [7] two additional constraints were added: the single-value constraint, i.e. the learner can change only one parameter value at a time, and the greediness constraint, i.e. if, after receiving an example he cannot recognise, the learner changes one parameter and now can accept the new data, the learner retains the new parameter setting. The TLA is an online learning algorithm that performs local hill climbing. This algorithm, however, is problematic as positive-only examples can lead to local maxima, i.e. an incorrect hypothesis from which the learner cannot move, thus rendering the parameter space under consideration unlearnable. In order to acquire the target grammar, the learner has to start from very specific points in the parameter space.

[12], [13] model the TLA as a Markov chain and modify the TLA by replacing the local single-step hill climbing procedure with a simple Random Walk Algorithm (RWA). RWA renders the learning process faster and always converges to the correct target language irrespective of the initialisation of the algorithm. [12], [13] tested the system tested on a very small three-parameter system that produced 8 grammars, where each grammar consisted of only up to 18 acceptable examples. Following [12], [13]'s observation that a RWA greatly improves the accuracy and speed of the learning process, we propose a new way for computational analysis of language acquisition within the "principles and parameters" setting. Our algorithm is set within the general framework of multi-armed bandit problems. The *prior knowledge multi-armed bandit* problem abstracts the idea of a casino of slot machines in which a player has to play machines in order to find out how good they are, but where he has some prior knowledge that some machines are likely to have similar rates of reward. Each grammar is represented as "an arm of a bandit machine" with the mean-reward function drawn from a Gaussian Process specified by a covariance function between grammars. We test our algorithm on the metrical stress ten-parameter space [6], which gives rise to 216 possible stress systems (as not all parameters are independent). The use of the algorithm, however, can be easily extended to much larger systems. We also compare the performance our algorithm to that of TLA and RWA and show that it "learns" the correct grammar faster than both TLA and RWA.

As emphasised by [7] and [12], [13], arriving at the correct parameter setting is only one aspect of the language acquisition problem. As noted by [3], an equally important point is how the space of possible grammars is "scattered" with respect to the primary language data. It is possible for two grammars to be so close to each other that it is almost impossible to separate them by psychologically realistic input data. This leads to the question of sample complexity [12], i.e. how many examples it will take to identify the target grammar. It is of not much use to the learner to be able to arrive at the correct target grammar within the limit if the time required to do so is exponentially long, which renders the learning process psychologically implausible. Thus, rather than concern ourselves with identifying the correct grammar, we will measure the number of

errors made by the learner in acquiring the correct grammar. We will give experimental evidence that the number of iterations required to reach a state where the learner makes virtually no mistakes is smaller than the number of grammars to be explored. We will also consider the impact of variations in data distribution and the presence of noise in the data on the performance of the algorithm.

The remainder of the paper is organised as follows: first we briefly describe the basic tenets of metrical stress theory followed by a short description of the multi-armed bandit problem, where we introduce in more detail our arm selection procedure. Next, we describe how the procedure can be applied to learning parametrised grammars. Finally, we present experimental results on learning the ten-parameter space addressing a set of specific questions.

2 Metrical Stress Parameters and the Learning Problem

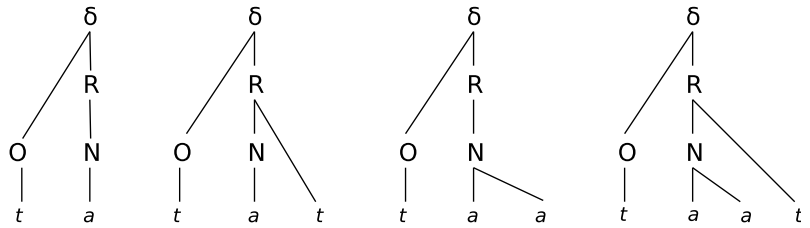
In this section, we describe the syllable structure and its relevance to stress assignment. Further, we present the stress parameters that we make reference to throughout the rest of this paper. Lastly, we discuss how the data is presented to and assessed by the learner in our system.

2.1 Syllable Structure and Stress Assignment

We assume that the input to the learning process are words. One of the principles shared by most theories of stress systems is that stress is sensitive to representations built on projections from syllable structure. In many languages, stress is sensitive to syllable weight, or quantity. Thus, we also assume the prior operation of rules that convert the speech signal into words and smaller word segments, such as syllables.

In general, syllables can be divided into two parts: an onset (O) and a rhyme (R). The onset consists of the consonant(s) before the syllable peak, which is usually a vowel. The rhyme consists of the vowel and the consonant(s) following it. The rhyme can be further divided into two parts: the nucleus (N), i.e. the vowel, and the coda (C), i.e. the consonant(s) following it. It is generally agreed that the onset plays no part in stress assignment. However, in quantity-sensitive languages, the structure of the rhyme plays an important role in stress assignment (see [5] for possible counter examples). Syllables that have only one element in the nucleus position and no coda are classified as light (Fig. 1a) and as such do not attract stress. Syllables with two elements in the nucleus position count as heavy (Fig. 1c, d) and attract stress, while syllables with one element in the nucleus and at least one element in the coda position can count as either light or heavy (depending on the setting of parameter 6 below) (Fig. 1b).

Furthermore, we also assume that various acoustic cues that indicate phonological stress are mapped into one of three degrees of stress. The three levels of stress are primary stress (marked as 2), secondary stress (marked as 1), and lack of stress (marked as 0). For the purpose of our analysis, we assume that every word must have a primary stress.



(a) /ta/ (b) /tat/ (c) /taa/ (d) /taat/

Fig. 1: Four examples of syllable with different rhyme structure (after [10]). δ signifies a syllable node; O, R, N and C represent the constituents of the syllable to which the segmental material is attached.

2.2 The Stress System Parameters

In metrical theory, stress patterns, and the corresponding differences between languages, are due to metrical structures built on the rhyme. The various possibilities of metrical structure construction can be expressed in terms of a series of binary parameters. In our analysis, we consider a 10-parameter model with the following parameters [6]:

- P1: The word-tree is strong on the left/right;
- P2: Feet are binary/unbounded;
- P3: Feet are built from left/right;
- P4: Feet are strong on the left/right;
- P5: Feet are quantity sensitive/insensitive;
- P6: Feet are quantity sensitive to the rhyme/nucleus;
- P7: A strong branch of the foot must/must not itself branch;
- P8: There is/is not an extrametrical syllable;
- P9: It is extrametrical on the left/right;
- P10: Feet are/are not non-iterative.

If all the parameters were independent, then we would have $2^{10} = 1024$ possible grammars. However, due to built-in dependencies, there only 216 distinct stress systems (see [6] for more details).

Let us consider the effect that different parameter settings can have on language structure. For example, P1 tells us where in the word the main stress should fall. If P1 is set to left, then the main stress will fall on the initial syllable, as in Latvian or Hungarian, if, however, we set P1 to right, then the main stress will fall on the final syllable, as in French or Farsi. In many languages, secondary stress can also be observed. In such languages, syllables are first grouped together into feet and every foot receives a stress. If a language has feet, a number of other parameters come into play. P2 allows feet to be at most binary or else unbounded. Selecting binary feet will give an alternating pattern of weak (with stress level 0) and strong (stress level 1 or 2) syllables. We must also set P3, which will trigger the direction of construction from left to right or

from right to left. Further, we must also set P4, which allows each foot to be left dominated or right dominated. For example, Maranungku, spoken in Australia, [10] has the following setting P1[left], P2[binary], P3[left], P4[left], which gives rise to the following alternating pattern of stresses: 201, 2010, etc. On the other hand, Warao, spoken in Venezuela, [10] has the following setting: P1[right], P2[binary], P3[right], P4[left], which results in the following stress pattern: 020, 01020, 10101020, 010101020.

2.3 Inclusion of Prior Knowledge

In [7] and [12], [13], the transition probabilities from one parameter setting state to another are calculated by counting the number of overlapping input data between each grammar corresponding to each parameter setting. We consider this to be an unrealistic model in that it is not clear how the learner would be able to assess this overlap without knowledge of the grammars and the sentence frequencies. We prefer to work with a weaker assumption of the prior knowledge that learners are equipped with, namely that learners are able to assess similarity of grammars by the *Hamming distance* between their parameter vectors. This accords with the expectation that the entries in the parameter vector control aspects of the production of sentences that involve varying levels of processing by the learner. Hence, our conjecture is that the parameter settings described above have cognitive correlates that enable the learner to compute the *Hamming distance* between the grammars. One of the questions addressed by the experiments in this paper (and answered in the affirmative) is whether this prior knowledge will be sufficient to enable subjects to learn to identify the correct grammar.

2.4 The Learning Algorithm

Our learning procedure is partly inspired by the TLA [7]. However, contrary to [7], we do not obey the single-value constraint or the greediness constraint. Our learning algorithm can be summarised as follows:

- Step 1 [Initialise]: Start at some random point in the finite space of possible parameter settings and specify a grammar hypothesis.
- Step 2 [Process input data]: Receive n number of positive example words from the target grammar (g_t). The words are drawn at random from a fixed probability distribution.
- Step 3 [Learnability on error detection]: Check if the currently hypothesised grammar (g_h) can generate the input data and receive a reward r ranging from 0 to 1. $r = 0$ corresponds to a situation, where none of the n words can be found in the hypothesis grammar, while if $r = 1$, all the n words are analysable in the currently hypothesised grammar. The reward function is calculated as follows:

$$r = \frac{\sum_{w_i \in g_h}^{n} pr_h(w_i)}{\sum_{w_i \in g_h}^{n} pr_h(w_i) + \sum_{w_i \notin g_h \cap w_i \in g_t}^{n} pr_t(w_i)} \tag{1}$$

where $pr(w_i)$ is the probability of the i^{th} word.
- Step 4 [Update] Update distributions over possible grammars based on reward.
- Step 5 [Grammar selection]: Stay in the current hypothesis state or 'jump' to a new parameter setting (the new grammar selection procedure is described in detail in Sec. 4). The newly hypothesised grammar does not necessarily have to allow the learner to analyse all or any of the n input examples.

The learning process is completed when after some iteration m, the learner ceases to make any errors, i.e. at virtually every iteration after iteration m the reward at step 3 is always $r = 1$, and the grammar selected at step 5 is always the target grammar.

3 The Multi-armed Bandit Problem

The multi-armed bandit problem is an analogy with a traditional slot machine, known as a one-armed bandit, but with multiple arms. In the classical bandit scenario, the player, after playing an arm selected from a finite number of arms, receives a reward. The player has no initial knowledge about the arms, and attempts to maximise the cumulative reward through repeated plays. It is assumed that the reward obtained when playing an arm i is a sample from an unknown distribution R_i with mean μ_i. The optimal playing strategy S^*, i.e. the strategy that yields maximum cumulative reward, consists in always playing an arm i^* such that $i^* = \text{argmax}_i \mu_i$. The expected cumulative reward of S^* at time t would then be $t\mu_{i^*}$. The performance of a strategy S is assessed by the analysis of its expected regret at time t, defined as the difference between the expected cumulative reward of S^* and S at time t.

A good strategy requires to optimally balance the learning of the distributions R_i and the exploitation of arms which have been learnt as having high expected rewards. Even if the number of arms is finite and smaller than the number of experiments allowed so that it is possible to explore all the arms a certain number of times, this only gives probabilistic information about the best performing arms. The multi-armed bandit problem is concerned with the design and analysis of algorithms that can trade exploration and exploitation to achieve only a small regret for a finite set of independent arms. In our prior knowledge multi-armed bandit problem, we are interested in learning with many fewer trials through exploiting knowledge about similarities between different arms or, in our case, grammars. We will encode this information in a covariance function, hence assuming a prior Gaussian Process over reward functions. We now describe our learning algorithm.

3.1 The Prior Knowledge Multi-armed Bandit Algorithm

We consider space $\mathcal{X}$, whose elements will be referred to as arms. κ denotes a kernel between elements of $\mathcal{X}$. The reward after playing arm $\mathbf{x} \in \mathcal{X}$ is given by

$f(\mathbf{x}) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2_{noise})$ and $f \sim \mathcal{GP}(\mathbf{0}, \kappa(\mathbf{x}, \mathbf{x}'))$ is chosen once and for all but is unknown. Arms played up to time t are $\mathbf{x}_1, \ldots, \mathbf{x}_t$ with rewards $y_1, \ldots, y_t$. The GP posterior at time t after seeing data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_t, y_t)$ has mean $\mu_t(\mathbf{x})$ with variance $\sigma^2_t(\mathbf{x})$.

Matrix C_t and vector $\mathbf{k}_t(\mathbf{x})$ are defined as follows:

$$(C_t)_{i,j} = \kappa(\mathbf{x}_i, \mathbf{x}_j) + \sigma^2_{noise}\delta_{i,j} \tag{2}$$

$$(\mathbf{k}_t(\mathbf{x}))_i = \kappa(\mathbf{x}, \mathbf{x}_i) \tag{3}$$

$\mu_t(\mathbf{x})$ and $\sigma^2_t(\mathbf{x})$ are then given by the following equations (see [14]):

$$\mu_t(\mathbf{x}) = \mathbf{k}_t(\mathbf{x})^T C_t^{-1} \mathbf{y}_t \tag{4}$$

$$\sigma^2_t(\mathbf{x}) = \kappa(\mathbf{x}, \mathbf{x}) - \mathbf{k}_t(\mathbf{x})^T C_t^{-1} \mathbf{k}_t(\mathbf{x}) \tag{5}$$

As noted by [15], if no assumption is made on the nature of the reward function it may be arbitrarily hard to find an optimal arm. In their work, they assume that the mean-reward μ_k of a newly played arm k is a sample of a fixed distribution, and they characterise the probability of playing near-optimal arms. Others such as [11] and [2] assume that there exists a mean-reward function f and make further assumptions on the regularity of f. In the work of [11], arms are indexed in a metric space and the mean-reward is a Lipschitz function in this space. In the model of [2], arms lie in a generic topological space and f has a finite number of global maxima around which the function is locally Hoelder.

We assume in our model that the reward of an arm $\mathbf{x}$ is determined by a function f applied at point $\mathbf{x}$ to which Gaussian noise is added. The variance of the noise corresponds to the variability of the reward when always playing the same arm. In order to cope with large numbers of arms, our assumption will be that the rewards of arms are correlated. Thus, playing an arm 'close' to $\mathbf{x}$ gives information on the expected gain of playing $\mathbf{x}$. This can be modelled with a Gaussian Process: by default, we take the mean of the Gaussian Process prior to be $\mathbf{0}$, and we can incorporate prior knowledge on how correlated the arms are in the covariance function between arms. The covariance function can be seen as a kernel function, and specifies how 'close' or 'similar' two given arms are. Hence, we assume in our model that f is a function drawn from a Gaussian Process (GP).

If arms are indexed in $\mathbb{R}^d$, for example, the covariance function can be chosen to be a Gaussian kernel $\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right)$, whose smoothness σ is adjusted to fit the characteristic length scale which is assumed for f. In our framework, we are not limited to problems where arms are indexed in $\mathbb{R}^d$, but we can also consider arms indexed by any type of structured data as long as a kernel can be defined between the data points. For instance, in the parametric grammar learning problem, each grammar can be associated with an arm, so that looking for the optimal arm corresponds to looking for the optimal grammar given a certain criterion which is incorporated into the reward function.

Arm selection. The algorithm plays a sequence of arms and aims at optimally balancing exploration and exploitation. For this, arms are selected iteratively according to a UCB-inspired formula (see [1] for more details on UCB):

$$\mathbf{x}_{t+1} = \text{argmax}_{\mathbf{x}\in\mathcal{X}}\left\{ f_t(\mathbf{x}) = \mu_t(\mathbf{x}) + B(t)\sigma_t(\mathbf{x})\right\} \tag{6}$$

This can be interpreted as active learning where we want to learn accurately in regions where the function looks good, while ignoring inaccurate predictions elsewhere. The $B(t)$ term balances exploration and exploitation: the bigger it gets, the more it favours points with high $\sigma_t(\mathbf{x})$ (exploration), while if $B(t) = 0$, the algorithm is greedy. In the original UCB formula, $B(t) \sim \sqrt{\log t}$.

Although this arm selection method seems quite natural, in some cases, finding the maximum of the "upper confidence function" f_t may prove costly, particularly as we would expect the function to become flatter as iterations proceed, as the algorithm aims to explore regions only as our uncertainty about their reward offsets the difference in our estimated reward. For this reason we will consider an alternative arm selection method based on the sampling of functions from the GP posterior.

3.2 Application to the Grammar Learning Problem

As suggested above, the problem of grammar learning can be considered as a many-armed bandit problem and the Gaussian Process approach can be used with a covariance function/kernel which applies to different grammars. Learning consists in looking for the 'best' grammar, i.e. the one that maximises the reward function.

Let us denote by $\mathcal{X}$ the set of parametrised grammars. In the 'principles and parameters' framework, a grammar $\mathbf{x}$ is a binary vector of length d, where d is the number of parameters under consideration. In our case, $d = 10$. We need to define a kernel $\kappa(\mathbf{x}, \mathbf{x}')$/covariance function between grammars. In our experiments, we consider a Gaussian kernel that takes the form:

$$\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\| \mathbf{x} - \mathbf{x}' \|^2}{2\sigma^2}\right) \tag{7}$$

where $\| \mathbf{x} - \mathbf{x}' \|^2$ is the *Hamming distance* between two grammars[2].

4 The Arm Selection Problem

The objective of the bandit algorithm is to minimise the regret over time, or in other words, to find as quickly as possible a good approximation of the maximum of f, $f(\mathbf{x}^*)$.

[2] Note that in this formulation, all parameters have equal effect on the produced data. As [6, p. 155, ft. 11] point out, theoretically, it is possible for a small change in the parameter setting to have large effects on the produced data and small changes to have small effects. This problem is not studied in [6]. However, if the parameters have a nonuniform effect on the output data, we can incorporate this information in the covariance function.

Let us define $f_t(\mathbf{x})$ by:

$$\begin{aligned} f_t(\mathbf{x}) &= \mu_t(\mathbf{x}) + B(t)\sigma_t(\mathbf{x}) \\ &= \mathbf{k}_t(\mathbf{x})^T C_t^{-1}\mathbf{y}_t + B(t)\sqrt{\kappa(\mathbf{x},\mathbf{x}) - \mathbf{k}_t(\mathbf{x})^T C_t^{-1}\mathbf{k}_t(\mathbf{x})} \end{aligned} \tag{8}$$

Our approximation of $f(\mathbf{x}^*)$ at time t is $f(\mathbf{x}_{t+1})$ where $\mathbf{x}_{t+1} = \text{argmax}_{\mathbf{x}\in\mathcal{X}}\{f_t(\mathbf{x})\}$.

Here we replace the problem of finding the maximum of the function f by iterations of a simpler problem, which is to maximise the function f_t whose form is known. At each iteration we learn new information which enables us to improve our approximation of $f(\mathbf{x}^*)$ over time. In the case where $\kappa(\mathbf{x},\mathbf{x}) = 1$ for all $\mathbf{x}$ (e.g. Gaussian kernel and cosine kernel), $f_t(\mathbf{x})$ can be written as a function of $\mathbf{k}_t(\mathbf{x}) = \mathbf{k}$:

$$f_t(\mathbf{x}) = g(\mathbf{k}) = \mathbf{k}^T C_t^{-1}\mathbf{y}_t + B(t)\sqrt{1 - \mathbf{k}^T C_t^{-1}\mathbf{k}} \tag{9}$$

4.1 Sampling Arm Selection

As we noted earlier, we would expect the upper confidence function to become flatter as iterations proceed, which would make it difficult to find its maximum. For this reason, we propose an alternative method for selecting the next arm rather than choosing the point that maximises the upper confidence function. Our strategy for selecting arms is to sample a function g from the posterior distribution and then select $\mathbf{x}_{t+1} = \text{argmax}_{\mathbf{x}\in\mathcal{X}} g(\mathbf{x})$. This implements sampling an arm with the probability that it is the maximum in the posterior distribution, and so implements a Bayesian approach to trading exploration and exploitation. We can interpolate between these methods by sampling a variable number K of functions $\mathbf{g}_1, \ldots, \mathbf{g}_K$ from the posterior and selecting $\mathbf{x}_{t+1} = \text{argmax}_{\substack{\mathbf{x}\in\mathcal{X} \\ 1\le k\le K}} \{g_k(\mathbf{x})\}$.

A naive sampling from the posterior distribution would require inverting a matrix indexed by the full grid. We avoid this by iteratively unveiling the posterior sample $g(\mathbf{x})$ only sampling points that are likely to lead to a maximal value. Since the function $g(\mathbf{x})$ is not expected to be flat, only a small number of samples should be required in practice. We would envisage selecting these samples by simple hill climbing heuristics that could work efficiently on the non-flat g, but in our experiments, we simply consider the enumeration of all 216 grammars.

Arm selection method:

1. Initialisation (t=1): $\mathbf{x}_1$ chosen randomly in $\mathcal{X}$. y_1 is the reward obtained when "playing" $\mathbf{x}_1$. The GP posterior after seeing the data $(\mathbf{x}_1, y_1)$ is sampled (f_1) to give iteration at time $t+1$ and $\mathbf{x}_2 = \text{argmax}\{f_t(\mathbf{x})\}$.
2. Iteration at time $t+1$: we have played $\mathbf{x}_1, \ldots, \mathbf{x}_t$ and have obtained rewards $y_1, \ldots, y_t$. The GP posterior is sampled to give f_t and $\mathbf{x}_{t+1} = \text{argmax}\{f_1(\mathbf{x})\}$.

We now describe the sampling method in detail, which returns the next selected arm $\mathbf{x}_{t+1}$. After seeing only the data $((\mathbf{x}_1, y_1)), \ldots, (\mathbf{x}_t, y_t)$, the posterior has mean $\mu_t^{(1)}$ and variance $\sigma_t^{(1)^2}$.

Algorithm 1 Sampling method

set $V = \{\}$ and $k = 1$
for $\mathbf{g}_i \in G$ **do**
  if $\mu_t^{(k)}(\mathbf{g}_i) + B(t)\sigma_t^{(k)}(\mathbf{g}_i) \geq \max(\{y_t\} \cup V)$ **then**
    sample from $\mathcal{N}(\mu_t^{(k)}(\mathbf{g}_i), {\sigma_t^{(k)}}^2(\mathbf{g}_i))$ and get v_k
    set $\mathbf{s}_k = \mathbf{g}_i$, $V = V \cup \{v_k\}$ and $k = k + 1$
    the GP posterior after seeing the data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_t, y_t)$ with observation noise σ^2_{noise} and $(\mathbf{s}_1, v_1), \ldots, (\mathbf{s}_k, v_k)$ without noise has mean $\mu_t^{(k+1)}$ and variance $\sigma_t^{(k+1)}$
  end if
end for
return $\mathbf{s}_j$, where $v_j = \max(v_1, \ldots, v_{k-1})$

5 Experiments

The aim of the experiments is to investigate the following issues:

1. The learning process is completed in fewer iterations than the number of grammars under consideration.
2. The learning process is successful irrespective of the initial grammar selected in Step 1 of the learning algorithm.
3. The learning takes place in an on-line fashion, i.e. the number of errors made as the learning process progresses is gradually being reduced until it reaches 0.
4. The algorithm is robust with respect to the presence of noise and the input data distribution.

5.1 The Data

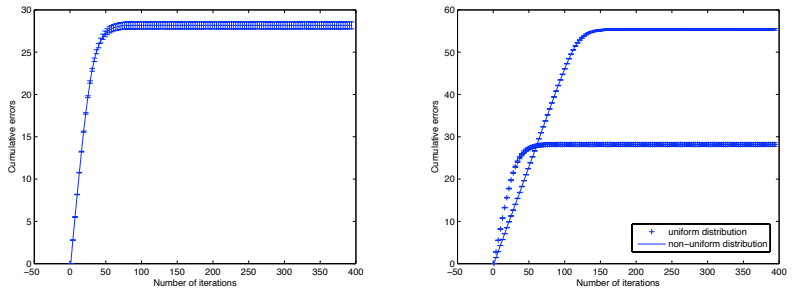
In our experiments, every input word consists of two parts: syllable representation followed by its stress assignment. In our analysis, we represent light syllables as L, syllables with a branching nucleus as N, and syllables with a branching rhyme as R. For example, a string of the form RLRL2010 represents a four-syllable word with primary stress on the initial syllable and secondary stress on the penultimate syllable. We consider words of up to a length of 7 syllables. This results from 2901 to 3279 words for each of the 216 possible grammars. Needless to say, a given word can belong to more than one grammar. The number of overlapping words between grammars ranges from 3 to 3189. The average number of overlapping words is 262.

5.2 The Experiment Design and General Results

All the experiments reported below are averaged over 600 runs with a random starting point. This random initialisation allowed us to study the influence of the initialisation step of the learning process. Each run consisted of 400 iterations

of the algorithm which we described in detail in Sec. 4.1. At each iteration the learner is presented with 5 words selected from the target grammar. The frequency with which each word is presented to the learner corresponds to the probability distribution of this word. Note that the learner is not allowed to use the particular words to inform his learning but only the average error of the currently hypothesised grammar on these words.

Below, we report results for the target grammar with the following parameter setting: P1[right], P2[binary], P3[right], P4[left], P5[QI], P6[rhyme], P7[no], P8[yes], P9[right], P10[yes], although similar results can be reported for the remaining 215 grammars. The data is drawn from a uniform distribution. As mentioned in the introduction, the error convergence to 0 corresponds to identifying the target grammar. As illustrated in Fig. 2a, the target grammar is identified within 30 - 50 iterations, irrespective of the initialisation step. Note that an exhaustive search would require "trying" all the 216 possible grammars, thus lengthening the learning process. The faster error convergence results from online nature of our learning algorithm and the incorporation of prior knowledge in our learning scenario. The algorithm is more efficient than one, where at each iteration a new grammar was selected completely at random, thus resulting in a larger number of errors and a slower convergence rate.



(a) Uniform distribution

(b) Uniform and non-uniform distribution

Fig. 2: Error convergence rate (with standard deviation) of the Prior Knowledge Multi-armed Bandit algorithm when the data is drawn from (a) a uniform distribution; (b) the probability distribution is correlated with the word length and compared to the uniform probability distribution.

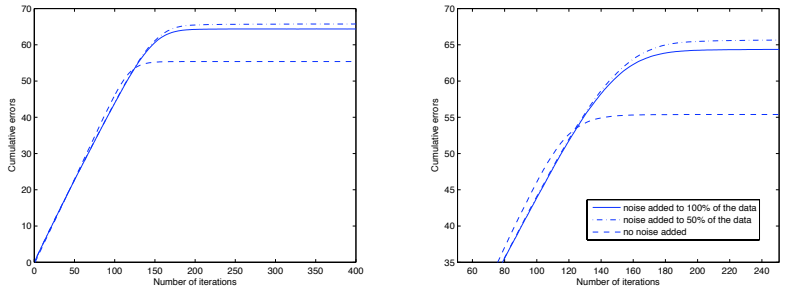
5.3 Varying the Probability Distributions

In the second set of experiments, we test the convergence time in two scenarios: (1) when the input data is presented to the learner from a uniform distribution, i.e. the probability to see every word is $1/n$, where n is the number of possible

words produced by a given grammar; (2) certain words are more likely to occur than others. In case (2), the probability distribution is correlated with the word-length, i.e. the shorter a given word is, the higher its probability of occurrence. As can be seen in Fig. 2b, varying the probability distribution affects the convergence rate. When the data is drawn from a non-uniform distribution, the convergence rate is slower, i.e. the target grammar is identified within 80 - 150 iterations, which is still lower than the number of all the 216 possible grammars.

5.4 The Impact of Noise

In the third set of experiments, we added noise (ω) to the input data, or, to be more precise to the reward function. Thus, the reward was $r + \omega$, where $\omega = randn * 0.05$. $randn$ is a random value drawn from a normal distribution with mean zero and standard deviation one. We tested the influence of noise when the noise was added to varying percentage of data ranging from 0% to 100%. Fig. 3 compares the error rate convergence for cases where noise was added to 0%, 50% and 100% of data, where the data was drawn from a non-uniform distribution. The algorithm performs best with no noise present. However, even with the addition of noise, the correct grammar is identified within 110 – 170 iterations.



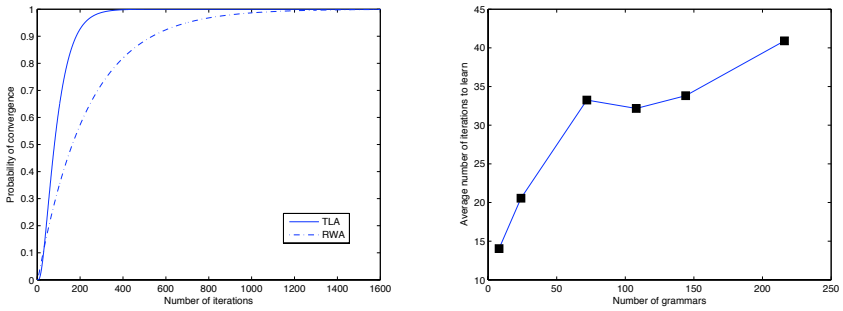
(a) (b)

Fig. 3: (a) Error convergence rates with noise added to 0%, 50% and 100% of data. (b) expansion of Figure 3a in the critical region.

5.5 Comparison with TLA and RWA

In the last set of experiments, we compared the performance of the 'prior knowledge multi-armed bandit" algorithm with that of TLA and RWA. Following [13], we implemented TLA as a Markov chain. Both in TLA and RWA, the words are drawn from a uniform distribution. The target grammar is the same as the one used in the previous experiments. As discussed earlier, it takes on average 30 - 50

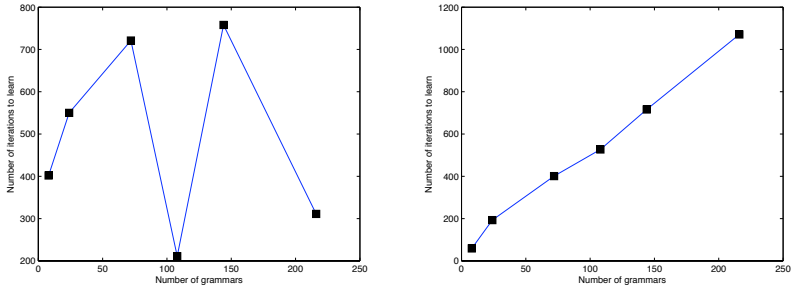
iterations to learn the correct grammar with the prior knowledge multi-arm bandit algorithm. As can be seen in Fig. 4a, it takes 311 iterations of TLA and 1071 iterations of RWA to learn the target grammar. It must be noted that at each iteration of our algorithm the learner is given a set of 5 words, while in the case of TLA and RWA the learner is given only one word at a time. However, even if we assume the worst-case scenario, where the learner needs 50 iterations of the prior knowledge multi-arm bandit algorithm to acquire the target grammar, we still require only 250 words to learn the language. Learning with TLA and RWA requires 311 and 1071 words, respectively. The prior knowledge multi-arm bandit algorithm converges faster than TLA and RWA in spite of the fact that TLA and RWA provide the learner with additional information of transition probabilities. Note that the prior knowledge multi-arm bandit algorithm does not take into account this type of extensional information.



(a) (b)

Fig. 4: (a) Probability of convergence of the Triggering Learning Algorithm (TLA) and the Random Walk Algorithm. (b) Average number of iterations required to learn the correct grammar with the prior knowledge multi-armed bandit algorithm as the size of the learning space increases.

[12] and[13] showed that in a three-parameter setting with 8 grammars, RWA converges faster than TLA. However, our experiments on a 10-parameter space show that the convergence rate of RWA is much slower than that of TLA. We further compared the convergence rate of the three algorithms as the size of the parameter space, and consequently the number of grammars, increases. We looked at a scenario, where the number of possible grammars was: 8, 24, 72, 108, 144 and 216. As can be seen in Figs. 4b and 5b, in the case of the prior knowledge multi-arm bandit algorithm and RWA, the complexity of the learning error is affected by the size of the learning space, i.e. the smaller the number of grammars the faster the learning process. The size of the learning space does not have the same effect on the TLA, i.e. there is no correlation between the number of parameters/grammars and the probability of convergence (Fig. 5a).



(a) (b)

Fig. 5: (a) Number of iterations required to learn the correct grammar with TLA as the size of the learning space increases. (b) Number of iterations required to learn the correct grammar with RWA as the size of the learning space increases.

6 Discussion and Future Directions

The problem of learning parametrised grammars can be approached from many different perspectives. In this paper, we concentrated on the problem of error convergence, i.e. how many examples it will take the learner to reach a stage where he can parse correctly all the incoming words. We have presented a new algorithm "prior knowledge multi-armed bandit" and have shown that the algorithm can successfully tackle the problem of sample complexity. The algorithm enables the learner to acquire the target language in an online fashion without the need to resort to searching the entire parameter space and without the danger of getting stuck in a local maximum. We have also shown that the learner can "discover" the parameter setting of the target grammar without direct access to the set difference of words belonging to the different grammars, but from the more cognitively realistic access to the *Hamming distance* between the grammars parameter vectors.

A number of directions for future research arise. As the number of parameters increases, so does the complexity of the learning process. It is worth investigating how the error convergence rate will change as the parameter space grows/decreases. Further, we also need to conduct a more extensive empirical analysis of the impact of noise and data distribution on the convergence rate, i.e. how increasing the level of noise or a very unfavourable data distribution will affect the learning process.

Another possible direction is the derivation of a language change model from the current language acquisition model as well as a language acquisition model where the learner is exposed to data coming from different languages or dialects. The procedure discussed in this article concentrates on modelling the language acquisition process of a single child. Needless to say, a language change model would require scaling up the present model to an entire population.

References

1. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine Learning (47), 235–256 (2002)
2. Bubeck, S., Munos, R., Stoltz, G., Szepesvari, S.: Online optimization in x-armed bandits. In: Proceedings of NIPS (2008)
3. Chomsky, N.: Aspects of the Theory of Syntax. MIT Press, Cambridge MA (1965)
4. Chomsky, N.: Lectures on government and binding. Foris, Dordrecht (1981)
5. Davis, S. M.: Syllable onsets as a factor in stress rules. Phonology (5), 1–19 (1988)
6. Dresher, B. E., Kaye, J. D.: A computational learning model for metrical phonology. Cognition 34, 137-195 (1990)
7. Gibson, T., Wexler, K.: Triggers. Linguistic Inguiry 25(4), 407–474 (1994)
8. Gold, E. M.: Language identification in the limit. Information and Control 10(4), 407–454 (1967)
9. Hale, K., Keyser, J.: On argument structure and the lexical expression of syntactic relations. In: Hale, K., Keyser, J. (eds.): The view from building 20. pp. 53–110. MIT Press, Cambridge MA (1993)
10. Hayes, B.: Metrical Stress Theory: Principles and Case Studies. The University of Chicago Press, Chicago (1995)
11. Kleinberg, R.,Slivkins, A. Upfal, E.: Multi-Armed Bandits in Metric Spaces. In: Proceedings of STOC (2008)
12. Niyogi, P., Berwick, R. C.: A language learning model for finite parameter spaces. Cognition 61, 161–193 (1996)
13. Niyogi, P.: The Computational Nature of Language Learning and Evolution. MIT Press, Cambridge MA (2006)
14. Rasmussen, C. E., Williams, C. K. I.: Gaussian Processes for Machine Learning. MIT Press, Cambridge MA (2006)
15. Wang, Y., Audibert, J., Munos, R.: Algorithms for infinitely many-armed bandits. In: Proceedings of NIPS (2008)