The VC-Dimension of Subclasses of Pattern Languages*

Andrew Mitchell[1], Tobias Scheffer[2], Arun Sharma[1], and Frank Stephan[3]

[1] University of New South Wales, School of Computer Science and Engineering, Sydney, 2052 NSW, Australia, andrewm,arun@cse.unsw.edu.au

[2] Otto-von-Guericke-Universität Magdeburg, FIN/IWS, Universitätsplatz 2, 39106 Magdeburg, Germany, scheffer@iws.cs.uni-magdeburg.de

[3] Mathematisches Institut, Universität Heidelberg, Im Neuenheimer Feld 294, 69120 Heidelberg, Germany, fstephan@math.uni-heidelberg.de

Abstract. This paper derives the Vapnik Chervonenkis dimension of several natural subclasses of pattern languages. For classes with unbounded VC-dimension, an attempt is made to quantify the "rate of growth" of VC-dimension for these classes. This is achieved by computing, for each n, size of the "smallest" witness set of n elements that is shattered by the class. The paper considers both erasing (empty substitutions allowed) and nonerasing (empty substitutions not allowed) pattern languages. For erasing pattern languages, optimal bounds for this size — within polynomial order — are derived for the case of 1 variable occurrence and unary alphabet, for the case where the number of variable occurrences is bounded by a constant, and the general case of all pattern languages. The extent to which these results hold for nonerasing pattern languages is also investigated. Some results that shed light on efficient learning of subclasses of pattern languages are also given.

1 Introduction

The simple and intuitive notion of pattern languages was formally introduced by Angluin [1] and has been studied extensively, both in the context of formal language theory and computational learning theory. We give a brief overview of the work on learnability of pattern languages to provide a context for the results in this paper. We refer the reader to Salomaa [20, 21] for a review of the work on pattern languages in formal language theory.

In the present paper, we consider both kinds of pattern languages: *erasing* (when empty substitutions are allowed) and *nonerasing* (when empty substitutions are *not* allowed). Angluin [2] showed that the class of nonerasing pattern languages is identifiable in the limit from only positive data in Gold's model [8]. Since its introduction, pattern languages and their variants have been a subject of intense study in identification in the limit framework (for a review, see Shinohara and Arikawa [24]). Learnability of the class of erasing pattern languages was

* Algorithmic Learning Theory (ALT-99), Tokyo, 1999. © Springer-Verlag

first considered by Shinohara [23] in the identification in the limit framework. This class turns out to be very complex and it is still open whether for finite alphabet of size > 1, the class of erasing pattern languages can be identified in the limit from only positive data[1].

Since the class of nonerasing pattern languages is identifiable in the limit from only positive data, a natural question is if there is any gain to be had if negative data is also present. Lange and Zeugmann [14] observed that in the presence of both positive and negative data, the class of nonerasing pattern languages is identifiable with 0 mind changes; that is, there is a learner that after looking at a sufficient number of positive and negative examples comes up with the correct pattern for the language. This restricted "one-shot" version of identification in the limit is referred to as *finite identification*.

Since finite identification is a batch model, finite learning from both positive and negative data may be viewed as an idealized version of Valiant's [25] PAC model. In this paper, we show that even the VC-dimension of patterns with one single variable and an alphabet size of 1 is unbounded. This implies that even this restricted class of pattern languages is not learnable in Valiant's sense, even if we omit all polynomial time constraints from Valiant's definition of learning. This result (which holds for both the nonerasing and erasing cases) may appear to be at odds with the observation of Lange and Zeugmann [14] that the class of nonerasing pattern languages can be finitely learned from both positive and negative data. The apparent discrepancy between the two results is due to a subtle difference on the manner in which the two models treat data. In finite identification, the learner has the luxury of waiting for a finite, but unbounded, sample of positive and negative examples before making its conjecture. On the other hand, the learner in the PAC model is required to perform on any fixed sample of an "adequate" size. So, clearly the conditions in the PAC setting with respect to data presentation are more strict.

Since this restricted class and several other subclasses of pattern languages considered in this paper have unbounded VC-dimension, we make an attempt to quantify the rate of growth of VC-dimension for these classes. This is achieved by computing, for each n, size of the "smallest" witness set of n elements that is shattered by the class. The motivation for computing such a *Vapnik Chervonenkis Witness Size* is as follows. Although classes with unbounded VC-dimension are not PAC-learnable in general, they may become learnable under certain constraints on the distribution. An often used constraint is that the distribution favors short strings. Therefore, an interesting question is: How large is the VC-dimension if only strings of a certain length are considered? Mathematically, it is perhaps more elegant to pose the question: What is the least length m such that n strings of size up to m are shattered? We refer to this value of m as the *Vapnik Chervonenkis Dimension Witness Size* for n and express it as a function of n, $vcdws(n)$. Hence, higher the growth rate of the function $vcdws$,

[1] See Mitchell [17] where a subclass of erasing pattern languages is shown to be identifiable in the limit from only positive data. This paper also shows that the class of erasing pattern languages is learnable if the alphabet size is 1 or ∞.

smaller is the "local" VC-dimension for strings up to a fixed length and "easier" it may be to learn the class under a suitably constrained variant of the PAC model.

Although the VC-dimension of 1-variable pattern languages is unbounded, we note that if at least one positive example is present, the VC-dimension of nonerasing pattern languages becomes bounded (this can be formally expressed in the terminology of version-spaces). Unfortunately, this does not help in the case of erasing pattern languages, as we are able to show that the VC-dimension of 1-variable erasing pattern languages is unbounded even in the presence of a positive example.

In k-variable patterns, the bound k is on the number of distinct variables in the pattern and not on the total number of occurrences of all variables. We also consider the case where the number of occurrences of all the variables in a pattern is bounded. We show that the VC-dimension of the class of languages generated by patterns with at most 1 variable occurrence is 2. For variable occurrence count ≥ 2, the VC dimension turns out to be unbounded provided the alphabet size is at least 2 and the pattern has at least two distinct variables. We also consider the case where the only requirement is that each variable occur exactly n times in the pattern (so, there is neither any bound on the number of distinct variables nor any bound on the total number of variable occurrences). We show that the VC dimension of languages generated by patterns in which each variable occurs exactly once is unbounded. We note that this result also holds for any general n.

Having established several VC-dimension results, we turn our attention to issues involved in *efficient* learning of pattern language subclasses. One problem with efficient learning of pattern languages is the *NP*-completeness of the membership decision [1].[2] This NP-completeness result already implies that pattern languages cannot be learned polynomially in Valiant's sense when the hypotheses are patterns (because Valiant requires that, for a given instance, the output of the hypothesis must be computable in polynomial time). Schapire [22] strengthened this result by showing that pattern languages cannot be polynomially PAC-learned independent of the representation chosen for the hypotheses. Computing the output of a hypothesis cannot be done in polynomial time using *any* coding scheme which is powerful enough for learning pattern languages. Also, Ko, Marron and Tzeng [13] have shown that the problem of finding any pattern consistent with a set of positive and negative examples is *NP*-hard. Marron and Ko [16] considered necessary and sufficient conditions on a finite positive initial sample that allows exact identification of a target k-variable pattern from the initial sample and from polynomially many membership queries. Later, Marron [15] considered the exact learnability of k-variable patterns with polynomially

[2] Angluin [1] showed that the class of nonerasing pattern languages is not learnable with polynomially many queries if only equivalence, membership, and subset queries are allowed and as long as any hypothesis space with the same expressive power as the class being learned is considered. However, she gave an algorithm for exactly learning the class with a polynomial number of superset queries.

many membership queries, but where the initial sample consists of only a single positive example.

In the PAC setting, Kearns and Pitt [11] showed that k-variable pattern languages can be PAC-learned under product distributions[3] from polynomially many strings. At first blush, their result appears to contradict our claim that k-variable patterns have an unbounded VC-dimension. A closer look at their result reveals that they assume an upper bound on the length of substitution strings — which essentially bounds the VC-dimension of the class. When the substitutions of all variables are governed by independent and identical distributions, then k-variable pattern languages can (under a mild additional distributional assumption) even be learned linearly in the length of the target pattern and singly exponentially in k [19].

In this paper we show that in the case of nonerasing pattern languages, the first positive example string contains enough information to bound the necessary sample size without any assumptions on the underlying distribution. (This result holds even for infinite alphabets.) Unfortunately, as already noted this result does not translate to the case of k-variable erasing pattern languages, as even in the presence of a positive example, the VC-dimension of single-variable erasing pattern languages is unbounded.

We finally consider some results in the framework of agnostic learning [9, 12]. Here no prior knowledge about the actual concept class being learned is assumed. The learner is required to approximate the observed distribution on classified instances almost as well as possible in the given hypothesis language (with high probability) in polynomial time. Agnostic learning may be viewed as the branch of learning theory closest to practical applications. Unfortunately, not even conjunctive concepts [12] and half-spaces [10] are agnostically learnable. Shallow decision trees, however, have been shown to be agnostically learnable [3, 6].

2 Preliminaries

The symbol ϵ denotes the empty string. Let s be a string, word, or pattern. Then the *length* of s, denoted $|s|$, is the number of symbols in s. A pattern σ is a string over elements from the basic language Σ and variables from a variable alphabet; we use lower case Latin letters for elements of Σ and upper case Latin letters for variables. The number of variables is the number of distinct variable symbols occurring in a pattern, the number of occurrences of variables is the total number of occurrences of variable symbols in a pattern. An erasing pattern language contains all words x generated by the pattern in the sense that every variable occurrence A is substituted within the whole word by the same string $\alpha_A \in \Sigma^*$, a non-erasing pattern language contains the words where

[3] More precisely, they require the positive examples in the sample to be generated according to a product distribution, but allow any arbitrary distribution for the negative examples.

the variables are substituted by non-empty strings only. Thus, in non-erasing pattern languages every word is at least as large as the pattern generating it.

For example, if $\sigma = aAbbBabAba$, then the length is 10, the number of variables 2 and the number of variable occurrences is 3. In an erasing pattern language, σ generates the words *abbabba* (by $A = \epsilon$ and $B = \epsilon$) and *abbaabba* (by $A = \epsilon$ and $B = a$) which it does not generate in the nonerasing case. In both cases, erasing and nonerasing, σ generates the word *aabbabaababa* (by $A = a$ and $B = aba$). This allows us to define subclasses of pattern languages generated by a pattern with up to k variables or up to l variable occurrences.

Quantifying Unbounded VC-Dimension. The VC-dimension of a class $\mathcal{L}$ of languages is the size of the largest set of words S such that $\mathcal{L}$ shatters S. The VC-dimension of a class $\mathcal{L}$ is unbounded iff, for every n, there are n words $x_1, x_2, \ldots, x_n$ such that $\mathcal{L}$ shatters them. As motivated in the introduction, we introduce the function

$$vcdws(n) = \min\{\max\{|x_1|, |x_2|, \ldots, |x_n|\} : \mathcal{L} \text{ shatters } \{x_1, x_2, \ldots, x_n\}\}$$

where $vcdws$ stands for *Vapnik Chervonenkis Dimension Witness Size* and returns the size of the smallest witness for the fact that the VC-dimension is at least n. Determining $vcdws$ for several natural classes is one of the main results of the present work.

Version Spaces. Given a set of languages $\mathcal{L}$ and a set of positive and negative example strings S, the version space $VS(\mathcal{L}, S)$ consists of all languages in $\mathcal{L}$ that generate all the positive but none of the negative strings in S. It follows from Theorem 2.1 of Blumer *et al.* [5] that $\mathcal{L}$ can be PAC-learned with the sample S and a finite number of additional examples if $VS(\mathcal{L}, S)$ has a finite VC-dimension. In Section 3 we will show that, for certain classes, the VC dimension of the version space remains infinite after a sample S has been read while in Section 5 we show that, for other classes, the VC-dimension of the version space can turn from infinite to finite when the first positive example arrives.

3 VC-dimension of erasing pattern languages

Our first result shows that even the very restrictive class of 1-variable erasing pattern languages over the unary alphabet has an unbounded VC-dimension. This special case is the only one for which the exact value of $vcdws$ is known.

Theorem 1. *The VC-dimension of the class of erasing* 1*-variable pattern languages is unbounded. If the size of the alphabet is* 1*, then one can determine the exact size of the smallest witness by the formula* $vcdws(n) = p_2 \cdot p_3 \cdot \ldots \cdot p_{n-1}$ *where* p_m *is the* m*-th prime number* ($p_1 = 2, p_2 = 3, p_3 = 5, p_4 = 7, \ldots$).

Proof: Let $\Sigma = \{a\}$. For the direction $vcdws(n) \leq p_2 \cdot p_3 \cdot \ldots \cdot p_{n-1}$, let $x_k = a^{m_k}$ where $m_k = p_1 \cdot p_2 \cdot \ldots \cdot p_{n-1}/p_k$, that is, m_k is the product of the first $n-1$ primes except the k-th one. Let $x_n = \epsilon$. For every subset $E \subseteq \{x_1, x_2, \ldots, x_n\}$, let p_E be the product of those p_k where $k < n$ and $x_k \notin E$ and where $p_E = 1$ if $E \subseteq \{x_n\}$. Now the patterns A^{p_E} and $a^{p_E}A^{p_E}$ generate a word a^m with $m > 0$ iff p_E divides m. Furthermore, the word ϵ is generated by A^{p_E} but not by $a^{p_E}A^{p_E}$. So the language generated by A^{p_E} contains exactly the $x_k \in E$ in the case $x_n \in E$; the language generated by $a^{p_E}A^{p_E}$ contains exactly the $x_k \in E$ in the case $x_n \notin E$. Since the longest x_k is x_1 whose length is $p_2 \cdot p_3 \cdot \ldots \cdot p_{n-1}$ one has that $vcdws(n) \leq p_2 \cdot p_3 \cdot \ldots \cdot p_{n-1}$.

For the converse direction assume that the erasing 1-variable pattern languages shatter $E = \{a^{m_1}, a^{m_2}, \ldots, a^{m_n}\}$ where $m_1 < m_2 < \ldots < m_n$. Let $a^{d_k}A^{e_k}$ generate all elements in E except a^{m_k}. Now one has, for $k' \in \{2, 3, \ldots, n\} - \{k\}$, that $a^{m_1} = a^{d_k + c_1 \cdot e_k}$ and $a^{m_{k'}} = a^{d_k + c_{k'} \cdot e_k}$, so $m_{k'} - m_1 = (c_{k'} - c_1)e_k$. On the other hand, $m_k - m_1$ is not a multiple of e_k and $m_k - m_1$ has a prime factor q_k which does not divide any difference $m_{k'} - m_1$. It follows that for every difference $m_k - m_1$ there is one prime number q_k dividing all other differences $m_{k'} - m_1$ and therefore, any product of $n-2$ of such different prime numbers must divide some difference $m_k - m_1$. The product $q_2 \cdot q_3 \cdot \ldots \cdot q_n / q_k$ is a lower bound for m_k and, for the k with the smallest number q_k, m_k is at least the product of the primes $p_2 \cdot p_3 \cdot \ldots \cdot p_{n-1}$. So $vcdws(n) \geq p_2 \cdot p_3 \cdot \ldots \cdot p_{n-1}$. ∎

As noted, this is the only case where $vcdws$ has been determined exactly. It will be shown that more variables enable smaller values for $vcdws(n)$ while it is unknown whether larger alphabets give smaller values for $vcdws(n)$ in the case of erasing 1-variable pattern languages. The above proof even shows the following: Given any positive example w, there is still no bound on the VC-dimension of the version space of the class with respect to the example set $\{w\}$. Since the given w takes the place of x_n in the proof above, one now gets the upper bound $|w| + p_2 \cdot p_3 \cdot \ldots \cdot p_{n-1}$ instead of $p_2 \cdot p_3 \cdot \ldots \cdot p_{n-1}$ and uses for x_k the words wa^{m_k} with $m_k = p_1 \cdot p_2 \cdot \ldots \cdot p_{n-1}/p_k$ in the case $k < n$ and $x_k = w$ in the case $k = n$.

Theorem 2. *For any positive example w, the VC-dimension of the version space of the class of all erasing 1-variable pattern languages is unbounded and $vcdws(n) \leq |w| + p_2 \cdot p_3 \cdot \ldots \cdot p_n$.*

However, if two positive examples are present, then in some rare cases it may be possible to bound the number of patterns. For example, let the alphabet be $\Sigma = \{a, b\}$. Then if both strings a and b are in the language and are presented as positive examples, it is immediate that the only pattern language that satisfies this case is Σ^*.

An alternative to limiting the number of variables is limiting the number of variable occurrences. If the bound is 1, then the class is quite restrictive and has VC-dimension 2. However, as soon as the bound becomes 2, one has unbounded VC-dimension.

Theorem 3. *The VC-dimension of the class of erasing pattern languages generated by patterns with at most 1 variable occurrence is 2.*

One can generalize the result and show that every 1-variable erasing pattern language with up to k occurrences of this variable has bounded Vapnik Chervonenkis dimension. This is no longer true for 2-variable erasing pattern languages with up to 2 occurrences.

Theorem 4. *The VC-dimension of the class of all erasing pattern languages generated by patterns with at most 2 variable occurrences is unbounded. Furthermore, $vcdws(n) \leq (3n+2) \cdot 2^n$.*

Proof: For each k, let x_k be the concatenation of all strings $a^n b\sigma b a^n$ where $\sigma \in \{a,b\}^n$ and the k-th character of σ is a b. Now, for every subset E of $\{x_1, x_2, \ldots, x_n\}$ let σ_E be a strings of length n such that $\sigma_E(k) = a$ if $x_k \notin E$ and $\sigma_E(k) = b$ if $x_k \in E$. Now the language generated by $Ab\sigma_E bB$ contains x_k iff $b\sigma_E b$ is a substring of x_k iff the k-th character in σ_E is a b which by definition is equivalent to $x_k \in E$. So, the set $\{x_1, x_2, \ldots, x_n\}$ is shattered. ∎

The corresponding theorem holds also for nonerasing pattern languages since A and B take at least the string a^n or even something longer. A natural question is whether the lower bound is also exponential in n. The next theorem answers this question affirmatively.

Theorem 5. *For given k, the class of all pattern languages with up to k variable occurrences satisfies $vcdws(n) \geq 2^{(n-1)/(2k+1)}$.*

Proof: Given $x_1, x_2, \ldots, x_n$, one needs 2^{n-1} patterns which contain x_1 and shatter $x_2, x_3, \ldots, x_n$. Let $m = |x_1|$ which is a lower bound for the size if $x_1, x_2, \ldots, x_n$ are the shortest words shattered by the considered class. A pattern generating x_1 has $h \leq k$ variable occurrences and, for the l-th variable occurrence in this word, one has a beginning entry $a_l \leq |x_1| \leq m$, and the length b_l of the variable in x_1. Knowing x_1, each pattern generating x_1 has a unique description with the given parameters. So one gets the upper bound $1 + m^2 + m^4 + \ldots + m^{2k} \leq m^{2k+1}$ for the number of patterns generating x_1 and has $m^{2k+1} \geq 2^{n-1}$, that is, $m \geq 2^{(n-1)/(2k+1)}$. ∎

The next theorem is about the general case of the class of all erasing pattern languages. Strict lower bounds are $vcdws(n) \geq \log(n)/\log(|\Sigma|)$ for the case of alphabet size 2 or more and $vcdws(n) \geq n-1$ for the case of alphabet size 1. These lower bounds are given by the size of the largest string within a set of n strings. These straightforward bounds are modulo a linear factor optimal for the unary and binary alphabets.

Theorem 6. *For arbitrary erasing pattern languages:*
(a) If $\Sigma = \{a\}$ then $vcdws(n) \leq 2n$.
(b) If $\Sigma = \{a,b\}$ then $vcdws(n) \leq 4 + 2 \cdot \log(n)$.

Proof (a) In this case $\Sigma = \{a\}$. Let $x_1 = a^{n+1}, x_2 = a^{n+2}, \ldots, x_n = a^{n+n}$ and E be a subset of $x_1, x_2, \ldots, x_n$. Now let σ_E be the concatenation of those A_k^{n+k} with $x_k \in E$: Taking $A_k = \{a\}$ and all other variables to ϵ, the word generated by σ_E is x_k. If at least two variables are not empty or one takes a string strictly longer than 1 then the overall length is at least $2n+2$ and the word generated outside $\{x_1, x_2, \ldots, x_n\}$. So, σ_E generates exactly those words x_k which are in E and the erasing pattern languages shatter $\{x_1, x_2, \ldots, x_n\}$.

Proof (b) In this case $\Sigma = \{a, b\}$. Let m be the first integer such that the set U_m contains at least n strings where U_m consists of all words $w \in \{a, b\}^m$ such that a, b occur similar often in w, the first character of w is a and aa, bb, ab and ba are subwords of w. One can show that $m \leq 4 + 2 \cdot \log(n)$. Let $x_1, x_2, \ldots, x_n$ be different words in U_m. Now, for every k, let σ_k be the pattern obtained from x_k by replacing a by A_k and b by B_k, and let σ_E be the concatenation of those σ_k where $x_k \in E$. Since every variable occurs in σ_E $m/2$ times, one has that either one variable is assigned to some fixed $u \in \{a, b\}^2$ and the generated word is $u^{m/2}$ or there are two variables such that one of them is assigned to a and the other one to b. Since $x_k \neq u^{m/2}$ — the word $u^{m/2}$ does not contain all subwords aa, ab, ba, bb — and since $x_k \notin \{a^{m/2}b^{m/2}, b^{m/2}a^{m/2}\}$ and since the first character of x_k is a one can conclude that $A_l = a$ and $B_l = b$ for some l. Now the word generated by σ_E is x_l and $x_l = x_k$ only for $l = k$. Thus σ_E generates exactly those x_k with $x_k \in E$. ∎

The trivial lower bound is constant 1 for infinite alphabet Σ. But it is impossible to have constant upper bound for the size of the smallest witness, indeed there is, for every k, an n with $vcwds(n) > k$.

4 VC-dimension of nonerasing pattern languages

Many of the theorems for erasing pattern languages can be adapted to the case of nonerasing pattern languages. In many theorems the upper bound increases by a sublinear factor (measured in the size of the previous value of the function $vcdws$). Furthermore one gets the following lower bound:

Theorem 7. *For any class of nonerasing pattern languages,* $vcdws(n) \geq \frac{n-1}{2 \cdot \log(n+2)}$.

Proof: Given $x_1, x_2, \ldots, x_n$, one needs 2^{n-1} patterns which contain x_1 and shatter $x_2, x_3, \ldots, x_n$. Let $m = |x_1|$ which is a lower bound for the size if $x_1, x_2, \ldots, x_n$ are the shortest words shattered by the considered class. A pattern generating x_1 can be described as follows: To each position one assigns either the value 0 if this position is covered by a constant from the pattern generating it, the value h if it is the first character of some occurrence of the h-th variable $(h \leq m)$ and $m+1$ if it is some subsequent character of the occurrence of some variable. Together with x_1 itself, this string either describes uniquely the pattern generating x_1 or is invalid if, for example, some variable occurring twice has at each occurrence a different length. So one gets at most $(m+2)^m$ patterns which generate x_1. It follows that $(m+2)^m \geq 2^{n-1}$. This condition only holds if $m \geq \frac{n-1}{2 \cdot \log(n+2)}$. ∎

Furthermore, all lower bounds on *vcdws* carry over from erasing pattern languages to nonerasing pattern languages. For upper bounds, the following bounds can be obtained by adapting the corresponding results for erasing pattern languages. In these three cases, the upper bounds are only slightly larger than those for the erasing pattern languages, but in the general case with an alphabet of size 2 or more, the above lower bound is $\frac{n-1}{2 \cdot \log(n+2)}$ for nonerasing pattern languages while $4 + 2 \cdot \log(n)$ is an upper bound for the erasing pattern languages.

Theorem 8. *For 1-variable pattern languages, $vcdws(n) \leq p_1 \cdot p_2 \cdot \ldots \cdot p_n$, where $p_1, p_2, \ldots, p_n$ are the first n prime numbers.*
For patterns with exactly two variable occurrences, $vcdws(n) \leq (3n+2) \cdot 2^n$.
If the alphabet size is 1, then $vcdws(n) \leq \frac{1}{2} \cdot (3n^2 + 5n)$ for the class of all nonerasing pattern languages.

Note that in the case the alphabet size is two, one can — using the well-known fact that nonerasing pattern languages shatter the set of the $x_k = a^{k-1}ba^{n-k}$ — obtain that $vcdws(n) \leq n$.

5 Learning k-variable nonerasing patterns

Having established several VC-dimension results in the previous section, we now present some PAC-learnability results. The fact that the VC dimension of k-variable pattern languages is infinite suggests that this class is not learnable. However, we will show that the version space becomes finite after the first positive example has been seen.

Theorem 9. *Let ε and δ be given. Let L be a k-variable pattern language and D be an arbitrary distribution on Σ^*. Let S be an initial set of positive sentences of size at least one and let $l_{min} = \min\{|w| \mid w \in S\}$. Regarding the version space, we can claim that $|VS(k\text{-variable pattern languages}, S)| \leq (l_{min} + k)^{l_{min}}$. Let h be any pattern consistent with a sample of size at most $m \geq \frac{l_{min}}{\varepsilon} \log \frac{1}{\delta \cdot \log(l_{\min}+k)}$. Then $P(Err_{L,D}[h] > \varepsilon) \leq \delta$.*

An exhaustive learner can find a consistent hypothesis (if one exists) after enumerating all possible $(l_{min}+k)^{l_{min}}$ patterns. In order to decide whether a pattern is consistent with a sample the learner has to check if $x \in L(h)$ for each example x and pattern h. While this problem is NP-complete for general patterns, it can be solved polynomially for any fixed number of variables k.

Theorem 10. *Given a sample S of positive and negative strings, a consistent nonerasing k-variable pattern h can be found in $O((l_{min} + k)^{l_{min}} \cdot \max\{|x| \mid x \in S\}^k)$ – that is, learning is – as in the case of PAC – polynomial in parameters $\frac{1}{\delta}$ and $\frac{1}{\epsilon}$ but depends exponentially on the parameters l_{min} and k.*

An algorithm that learns a k-variable pattern still has a run time which grows exponentially in l_{min}. Under an additional assumption on D and on the length of substitution strings, they become efficiently learnable for fixed k [11].

Patterns with k variable occurrences. If we restrict the patterns to have at most k occurrences of any variables, they become even more easily learnable. The number of k-occurrence patterns which are consistent with an initial example x is at most as large as the number of k-variable patterns – that is, the logarithm of the hypothesis space size is polynomial which makes the required sample size polynomial, too. However, the learner can find a consistent hypothesis much more quickly.

Theorem 11. *Pattern languages with up to k occurrences of variables can be learned from a sample in $O(l_{min}^{2k-2} \cdot k^k)$ – that is, polynomially for fixed k.*

The idea of the proof is that up to k substrings in the shortest example can be substituted by variables. Hence, we only need to try all "start and end positions" of the variables and to enumerate all possible identifications of some variables.

6 Length-bounded pattern languages

In this section we show that length-bounded pattern languages are efficiently learnable – even in the agnostic framework. Due to lack of space our treatment here is informal.

We assume the alphabet size to be *finite*. There are $(|\Sigma| + k + 1)^k$ patterns of length at most k ($|\Sigma|$ constants, up to k variables, and an empty symbol). It follows immediately from Theorem 1 of [9] that $P(Err_{L,D}[h^*] < Err_{L,D}[h] - \varepsilon) \leq \delta$ when h minimizes the empirical error, $h^* = \inf_{h \in H}\{Err_{L,D}[h]\}$ is the truly best approximation of L in H, and the sample size is at least $m \geq \frac{1}{\varepsilon^2} \log \frac{(|\Sigma|+k)^k}{\delta}$. In other words, by returning the hypothesis with the least empirical error a learner returns (with high probability) a hypothesis which is almost as good as the best possible hypothesis in the language. Hence, length bounded pattern languages are agnostically learnable. In order to find h^*, a learner can enumerate the hypothesis space in $O((|\Sigma| + k + 1)^k)$.

The union of length bounded patterns is the power set of the set of length bounded patterns – hence, this hypothesis space can be bounded to at most $2^{(|\Sigma|+k+1)^k}$. This implies that the sample complexity is $m \geq \frac{(|\Sigma|+k+1)^k}{\delta} \log \frac{1}{\delta \log(|\Sigma|+k+1)}$ (that is polynomial in $|\Sigma|, \frac{1}{\varepsilon}$, and $\frac{1}{\delta}$) but we still need to find an algorithm which finds a consistent hypothesis in polynomial time – together this proves that this class is polynomially learnable [4]. A greedy coverage algorithm which subsequently generates patterns which cover at least one positive and no negative example can be guaranteed to find a consistent hypothesis (if one exists) in $O((|\Sigma| + k)^k \cdot m^+)$ where m^+ is the number of positive examples (that is, polynomially for a fixed k).

In order to learn unions of length bounded pattern languages agnostically we would have to construct a polynomial algorithm which finds an empirical error minimizing hypothesis. Note that this is much more difficult: The greedy algorithm will find *a* consistent hypothesis – if one exists. It may occur that every positive instance is covered by a distinct pattern. In the agnostic framework, we would have to find *the* hypothesis which minimizes the observed error. An enumerative algorithm, however, would have a time complexity of $O(2^{(|\Sigma|+k)^k})$.

7 Conclusion

We studied the VC-dimension of several subclasses of pattern languages. We showed that even single variable pattern languages have an unbounded VC-dimension. For this and several other classes with unbounded VC-dimension we furthermore quantified the VC-dimension witness size, thus characterizing just how quickly the VC-dimension grows. We showed that the VC-dimension of the class of single variable pattern languages which are consistent with a positive example is unbounded; by contrast, the class of pattern languages with k variable occurrences which are consistent with a positive example is finite. Hence, after the first positive example has been read, the sample size which is necessary or good generalization can be quantified. This result does seem to vindicate recent attempts by Reischuk and Zeugmann [18] (see also [7]) to study feasible average case learnability of single variable pattern languages by placing reasonable restrictions on the class of distributions.

Acknowledgment

We would like to express our gratitude to the reviewers for several helpful comments that have improved the exposition of the paper. Andrew Mitchell is supported by an Australian Postgraduate Award. Tobias Scheffer is supported by grants WY20/1-1 and WY20/1-2 of the German Research Council (DFG), and an Ernst-von-Siemens fellowship. Arun Sharma is supported by the Australian Research Council Grants A49600456 and A49803051. Frank Stephan is supported by the German Research Council (DFG) grant Am 60/9-2.

References

1. D. Angluin. Finding patterns common to a set of strings. *Journal of Computer and System Sciences*, 21:46–62, 1980.
2. D. Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45:117–135, 1980.
3. P. Auer, R. C. Holte, and W. Maass. Theory and applications of agnostic PAC-learning with small decision trees. In *Proceedings of the 12th International Conference on Machine Learning*, pages 21–29. Morgan Kaufmann, 1995.
4. A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Occam's razor. *Information Processing Letters*, 24:377–380, 1987.
5. A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.
6. D. Dobkin, D. Gunopoulos, and S. Kasif. Computing optimal shallow decision trees. In *Proceedings of the International Workshop on Mathematics in Artificial Intelligence*, 1996.

7. T. Erlebach, P. Rossmanith, H. Stadtherr, A. Steger, and T. Zeugmann. Learning one-variable pattern languages very efficiently on average, in parallel, and by asking queries. In Ming Li and Akira Maruoka, editors, *Algorithmic Learning Theory: Eighth International Workshop (ALT '97)*, volume 1316 of *Lecture Notes in Artificial Intelligence*, pages 260–276, 1997.
8. E. M. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.
9. D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100(1):78–150, 1992.
10. K. Hoeffgen, H. Simon, and K. van Horn. Robust trainability of single neurons. Preprint, 1993.
11. M. Kearns and L. Pitt. A polynomial-time algorithm for learning k-variable pattern languages. In R. Rivest, D. Haussler, and M. Warmuth, editors, *Proceedings of the Second Annual Workshop on Computational Learning Theory*. Morgan Kaufmann, 1989.
12. M. Kearns, R. Schapire, and L. Sellie. Towards efficient agnostic learning. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 341–352. ACM Press, 1992.
13. K.-I Ko, A. Marron, and W.-G. Tseng. Learning string patterns and tree patterns from examples. In *Machine Learning: Proceedings of the Seventh International Conference*, pages 384–391, 1990.
14. S. Lange and T. Zeugmann. Monotonic versus non-monotonic language learning. In G. Brewka, K. Jantke, and P. H. Schmitt, editors, *Proceedings of the Second International Workshop on Nonmonotonic and Inductive Logic*, volume 659 of *Lecture Notes in Artificial Intelligence*, pages 254–269. Springer-Verlag, 1993.
15. A. Marron. Learning pattern languages from a single initial example and from queries. In D Haussler and L. Pitt, editors, *Proceedings of the First Annual Workshop on Computational Learning Theory*, pages 345–358. Morgan Kaufmann, 1988.
16. A. Marron and K. Ko. Identification of pattern languages from examples and queries. *Information and Computation*, 74(2), 1987.
17. A. Mitchell. Learnability of a subclass of extended pattern languages. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*. ACM Press, 1998.
18. R. Reischuk and T. Zeugmann. Learning one-variable pattern languages in linear average time. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pages 198–208. ACM Press, 1998.
19. P. Rossmanith and T. Zeugmann. Learning k-variable pattern languages efficiently stochastically finite on average from positive data. In *Proc. 4th International Colloquium on Grammatical Inference (ICGI-98)*, LNAI 1433, pages 13–24. Springer, 1998.
20. A. Salomma. Patterns (The Formal Language Theory Column). *EATCS Bulletin*, 54:46–62, 1994.
21. A. Salomma. Return to patterns (The Formal Language Theory Column). *EATCS Bulletin*, 55:144–157, 1994.
22. R.E. Schapire. Pattern languages are not learnable. In M. Fulk and J. Case, editors, *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 122–129. Morgan Kaufmann, 1990.
23. T. Shinohara. Polynomial time inference of extended regular pattern languages. In *RIMS Symposia on Software Science and Engineering, Kyoto, Japan*, volume 147 of *Lecture Notes in Computer Science*, pages 115–127. Springer-Verlag, 1982.

24. T. Shinohara and A. Arikawa. Pattern inference. In Klaus P. Jantke and Steffen Lange, editors, *Algorithmic Learning for Knowledge-Based Systems*, volume 961 of *Lecture Notes in Artificial Intelligence*, pages 259–291. Springer-Verlag, 1995.
25. L. Valiant. A theory of the learnable. *Communications of the ACM*, 27:1134–1142, 1984.