On the learnability of erasing pattern languages in the query model

Steffen Lange[1] and Sandra Zilles[2]

[1] Deutsches Forschungszentrum für Künstliche Intelligenz,
Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany,
e-mail: lange@dfki.de
[2] Universität Kaiserslautern,
FB Informatik, Postfach 3049, 67653 Kaiserslautern, Germany,
e-mail: zilles@informatik.uni-kl.de

Abstract. A pattern is a finite string of constant and variable symbols. The erasing language generated by a pattern p is the set of all strings that can be obtained by substituting (possibly empty) strings of constant symbols for the variables in p.

The present paper deals with the problem of learning the erasing pattern languages and natural subclasses thereof within Angluin's model of learning with queries. The paper extends former studies along this line of research. It provides new results concerning the principal learning capabilities of query learners as well as the power and limitations of polynomial-time query learners.

In addition, the paper focusses on a quite natural extension of Angluin's original model. In this extended model, the query learner is allowed to query languages which are themselves not object of learning. Query learners of the latter type are often more powerful and more efficient than standard query learners. Moreover, when studying this new model in a more general context, interesting relations to Gold's model of language learning from only positive data have been elaborated.

1 Introduction

A pattern is a finite string of constant and variable symbols (cf. Angluin [2]). The erasing language generated by a pattern p is the set of all strings that can be obtained by substituting strings of constant symbols (including the empty one!) for the variables in p.[1] Thereby, each occurrence of a variable has to be substituted by the same string.

The erasing pattern languages have found a lot of attention within the past two decades both in the formal language theory community (see, e. g., Salomaa [15, 16], Jiang *et al.* [9]) and in the learning theory community (see, e. g.,

[1] The term 'erasing' is coined to distinguish these languages from those pattern languages originally defined in Angluin [2], where it is forbidden to replace variables by the empty string.

Shinohara [17], Erlebach *et al.* [6], Mitchell [12], Nessel and Lange [13], Reidenbach [14]). The learning scenarios studied include Gold's [7] model of learning in the limit and Angluin's [3] model of learning with queries. Besides that, interesting applications have been outlined. For example, learning algorithms for particular subclasses of erasing pattern languages have been used to solve problems in molecular biology (see Arikawa *et al.* [5]).

The present paper focusses on the learnability of the erasing pattern languages and natural subclasses thereof in Angluin's [3, 4] model of learning with queries. The paper extends the work of Nessel and Lange [13]; the first systematic study in this context.

In contrast to Gold's [7] model of learning in the limit, Angluin's [3] model deals with 'one-shot' learning. Here, a learning algorithm (henceforth called query learner) has the option to ask queries in order to receive information about an unknown language. The queries will truthfully be answered by an oracle. After asking at most finitely many queries, the learner is supposed to output its one and only hypothesis. This hypothesis is required to correctly describe the unknown language.

The present paper contains a couple of new results, which illustrate the power and limitations of query learners in the context of learning the class of all erasing pattern languages and natural subclasses thereof. Along the line of former studies, the capabilities of polynomial-time query learners (i. e. learners that are constrained to ask at most polynomially many queries before returning their hypothesis) are studied as well.

In addition, a problem is addressed that has mainly been ignored so far. The present paper provides the first systematic study concerning the strength of query learners that are – in contrast to standard query learners – allowed to query languages that are themselves not object of learning. As it turns out, these new learners often outperform standard learners, concerning their principal learning capability as well as their efficiency.

Moreover, the learning power of non-standard query learners is compared to the capabilities of Gold-style language learners. As a result of this comparison, quite interesting coincidences between Gold-style language learning and query learning – in the more general setting of learning indexable classes of recursive languages – have been observed. One of them allows for a new approach to the long-standing open question of whether or not the erasing pattern languages (over a finite alphabet with at least three constant symbols) are Gold-style learnable from only positive examples. To be more precise, the erasing pattern languages are learnable in the non-standard query model (using a particular type of queries, namely restricted superset queries), iff they are Gold-style learnable from only positive examples by a conservative learner (i. e. a learner that strictly avoids overgeneralized hypotheses).

Next, we summarize the disciplinary results on query learning of all erasing pattern languages or natural subclasses thereof.

Among the different types of queries investigated in the past (see, e. g., Angluin [3, 4]), we consider the following ones:

Membership queries. The input is a string w and the answer is 'yes' or 'no', respectively, depending on whether or not w belongs to the target language L.

Restricted subset queries. The input is a language L'. If $L' \subseteq L$, the answer is 'yes'. Otherwise, the answer is 'no'.

Restricted superset queries. The input is a language L'. If $L \subseteq L'$, the answer is 'yes'. Otherwise, the answer is 'no'.

In the original model of learning with queries (cf. Angluin [3]), the query learner is constrained to choose the input language L' exclusively from the class of languages to be learned. Our study involves a further approach, in which this constraint is weakened by allowing the learner to query languages that are themselves not object of learning.

The following table summarizes the results obtained and compares them to the previously known results. The focus is on the learnability of the class of all erasing pattern languages and the following subclasses thereof: the so-called regular, k-variable, and non-cross erasing pattern languages.[2] The items in the table have to be interpreted as follows. The item 'No' indicates that queries of the specified type are insufficient to learn the corresponding language class, while the item 'Yes' indicates that the corresponding class is learnable using queries of this type. The superscript † refers to results, which can be found or easily derived from results in Angluin [3], Matsumoto and Shinohara [11], and Nessel and Lange [13], respectively.

Type of queries	Type of erasing pattern languages						
	all	regular	1-variable	const.-free 1-variable	k-variable	const.-free k-variable	non-cross
membership	No†	Yes†	No	Yes	No	No	No
restr. subset	No	Yes	No	Yes	No	No	No
restr. superset	No†	Yes†	No†	No†	No†	No†	No†

If query learners are allowed to choose input languages that are themselves not object of learning, their learning capabilities change remarkably, particularly when the learner is allowed to ask restricted superset queries. It seems as if this type of queries is especially tailored to accumulate learning-relevant information about erasing pattern languages. Note that the superscript ‡ marks immediate outcomes of the table above.

Type of extra queries	Type of erasing pattern languages						
	all	regular	1-variable	const.-free 1-variable	k-variable	const.-free k-variable	non-cross
restr. subset	No	Yes‡	No	Yes‡	No	No	No
restr. superset	Open	Yes‡	Yes	Yes	Yes	Yes	Yes

Of particular interest is also the *complexity* of a successful query learner M, cf. Angluin [3]. M learns a class *polynomially*, if, for each target language L

[2] A pattern p is regular provided that p does not contain any variable more than once. Moreover, p is said to be a k-variable pattern, if it contains at most k variables, while it is said to be non-cross, if there are variables $x_1, \ldots, x_n$ and indices $e_1, \ldots, e_n$ such that $p = x_1^{e_1} \cdots x_n^{e_n}$.

in the class, the total number of queries to be asked by M in the worst-case is polynomial in the length of the minimal description for L. The table below summarizes the corresponding results. The first (second) row displays the types of queries (not) suitable for polynomial learning of a particular class; the third row marks open problems. Here *MemQ* (*SubQ*,*SupQ*) is short for membership (restricted subset, restricted superset) queries; the prefix x denotes extra queries. The superscript † refers to results by Nessel and Lange [13]. Note that the results on non-learnability are not displayed.

	Type of erasing pattern languages						
	all	regular	1-variable	const.-free 1-variable	k-variable	const.-free k-variable	non-cross
polynomially learnable		$\mathit{SupQ}^{\dagger}$	xSupQ	$\mathit{MemQ}, \mathit{SubQ}$ xSupQ	xSupQ, if $k = 2$	xSupQ, if $k = 2$	xSupQ
learnable, not polynomially		$\mathit{MemQ}^{\dagger}$ xSubQ					
open	xSupQ				xSupQ, if $k > 2$	xSupQ, if $k > 2$	

2 Preliminaries

In the following, Σ denotes a fixed finite alphabet, the set of *constant symbols*. Moreover, $\mathcal{X} = \{x_1, x_2, x_3, \ldots\}$ is a countable, infinite set of *variables*. To distinguish constant symbols from variables, it is assumed that Σ and $\mathcal{X}$ are disjoint. By Σ^* we refer to the set of all finite strings over Σ (*words*, for short), where ε denotes the empty string or empty word, respectively. A *pattern* is a non-empty string over $\Sigma \cup \mathcal{X}$.

Several special types of patterns are distinguished. Let p be a pattern. If $p \in \mathcal{X}^*$, then p is said to be a *constant-free* pattern. p is a *regular* pattern, if each variable in p occurs at most once. If p contains at most k variables, then p is a *k-variable* pattern. Moreover, p is said to be a *non-cross* pattern, if it is constant-free and there are some $n \geq 1$ and indices $e_1, \ldots, e_n \geq 1$ such that p equals $x_1^{e_1} \cdots x_n^{e_n}$.

For a pattern p, the *erasing pattern language* $L_\varepsilon(p)$ generated by p is the set of all words obtained by substituting all variables in p by strings in Σ^*. Thereby, each occurrence of a variable in p has to be replaced by the same word.

Below, we generally assume that the underlying alphabet Σ consists of *at least* three elements.[3] a, b, c always denote elements of Σ.

The erasing pattern languages and natural subclasses thereof will provide the target objects for learning. The formal learning model analyzed is called

[3] As results in Shinohara [17] and Nessel and Lange [13] impressively show, this assumption remarkably reduces the complexity of the proofs needed to establish learnability results in the context of learning the erasing pattern languages and subclasses thereof. However, some of the learnability results presented below may no longer remain valid, if this assumption is skipped. A detailed discussion of this issue is outside the scope of the paper on hand.

learning with queries, see Angluin [3, 4]. In this model, the learner has access to an oracle that truthfully answers queries of a specified kind. A *query learner* M is an algorithmic device that, depending on the reply on the queries previously made, either computes a new query or a hypothesis and halts. *M learns a target language L using a certain type of queries* provided that it eventually halts and that its one and only hypothesis correctly describes L. Furthermore, *M learns a target language class $\mathcal{C}$ using a certain type of queries*, if it learns every $L \in \mathcal{C}$ using queries of the specified type. As a rule, when learning a target class $\mathcal{C}$, M is not allowed to query languages not belonging to $\mathcal{C}$ (cf. Angluin [3]).

As in Angluin [3], the *complexity* of a query learner is measured by the total number of queries to be asked in the worst-case. The relevant parameter is the length of the minimal description for the target language.

Below, only indexable classes of erasing pattern languages are considered. Note that a class of recursive languages is said to be an *indexable class*, if there is an effective enumeration $(L_i)_{i\geq 0}$ of all and only the languages in that class that has uniformly decidable membership. Such an enumeration is called an *indexing*.

3 Strength and weakness of query learners

3.1 Learning in the original query model

We first present results related to Angluin's [3] original model. Here the learner is only allowed to query languages that are themselves object of learning.

The first result points to the general weakness of query learners when arbitrary erasing pattern languages have to be identified.

Theorem 1. *The class of all erasing pattern languages is (i) not learnable using membership queries, (ii) not learnable using restricted subset queries, and, (iii) not learnable using restricted superset queries.*

Proof. Assertions *(i)* and *(iii)* are results from Nessel and Lange [13].

To prove Assertion *(ii)*, assume that a query learner M identifies the class of all erasing pattern languages using restricted subset queries. Then it is possible to show, that M fails to identify either $L_\varepsilon(x_1^2)$ or all but finitely many of the languages $L_\varepsilon(x_1^2 x_2^z)$ for $z \geq 2$. □

The observed weakness has one origin: the query learners are only allowed to output one hypothesis, which has to be correct. To see this, consider the following relaxation of the learning model on hand. Suppose that a query learner M has the freedom to output in each learning step, after asking a query and receiving the corresponding answer, a hypothesis. Similarly to Gold's [7] model of learning in the limit, a query learner is now successful, if the sequence of its hypotheses stabilizes on a correct one. Accordingly, we say that M learns in the limit using queries.

Theorem 2. *The class of all erasing pattern languages is (i) learnable in the limit using membership queries, (ii) learnable in the limit using restricted subset queries, and, (iii) learnable in the limit using restricted superset queries.*

However, let us come back to the original learning model, in which the first hypothesis of the query learner has to be correct. As Theorem 1 shows, positive results can only be achieved, if the scope is limited to proper subclasses of the erasing pattern languages.

Suppose that a subclass of the erasing pattern languages is fixed. Naturally, one may ask whether – similarly to Theorems 1 and 2 – the learnability of this class does not depend on the type of queries actually considered. However, this is generally not the case as our next theorem shows.

Theorem 3. *Fix two different query types from the following ones: membership, restricted subset, and restricted superset queries. Then there is a class of erasing pattern languages, which is learnable using the first type of queries, but not learnable using the second type of queries.*

Proof. Scanning the first table above, the class of all erasing pattern languages generated by constant-free 1-variable patterns is learnable with membership or restricted subset queries, but not learnable with restricted superset queries.

Moreover, it is not hard to verify, that the class which contains $L_\varepsilon(a)$ and all languages $L_\varepsilon(ax_1^z)$, where z is a prime number, is learnable using restricted superset queries, but not learnable using membership queries and not learnable using restricted subset queries.

Next, the class containing $L_\varepsilon(x_1^2)$ and all languages $L_\varepsilon(x_1^2x_2^2x_3^z)$, $z \geq 2$, is learnable with membership queries, but not with restricted subset queries.

A class learnable with restricted subset queries, but not with membership queries can be constructed via diagonalization. For that purpose fix an effective enumeration $(M_i)_{i\geq 0}$ of all query learners using membership queries and posing each query at most once.[4] Let z_i denote the i-th prime number for all $i \geq 0$.

Given $i \geq 0$, let $L_{2i} = L_\varepsilon(x_1^{z_i}a)$. Moreover, simulate the learner M_i. If M_i queries a word $w \in \Sigma^*$, provide the answer 'yes' iff $w \in L_\varepsilon(x_1^{z_i}a)$; provide the answer 'no', otherwise. In case M_i never returns a hypothesis in this scenario, let $L_{2i+1} = L_{2i} = L_\varepsilon(x_1^{z_i}a)$. In case M_i returns a hypothesis, let l be the length of the longest word M_i has queried in the corresponding scenario. Then define $L_{2i+1} = L_\varepsilon(x_1^{z_i}ax_2^{z_l})$. Finally, let $\mathcal{C}$ consist of all languages L_i for $i \geq 0$.

Note that $(L_i)_{i\geq 0}$ is an indexing for $\mathcal{C}$; membership is decidable as follows: assume $w \in \Sigma^*$ and $j \geq 0$ are given. If $j = 2i$ for some $i \geq 0$, then $w \in L_j$ iff $w \in L_\varepsilon(x_1^{z_i}a)$. If $j = 2i+1$ for some $i \geq 0$ and $w \in L_{2i}$, then $w \in L_j$. If $j = 2i+1$ and $w \notin L_{2i}$, then let $A = \{l \geq 0 \mid w \in L_\varepsilon(x_1^{z_i}ax_2^{z_l})\}$. A is finite and can be computed from w and i. Simulate M_i as above in the definition of L_{2i+1}. If M_i does not return a hypothesis, then, since no query is posed twice, M_i queries a word of a length not in A. Thus there is no $l \in A$ with $L_j = L_\varepsilon(x_1^{z_i}ax_2^{z_l})$; in particular $w \notin L_j$. If M_i returns a hypothesis, one can determine the length l^* of the longest word M_i has queried. In this case $w \in L_j$ iff $l^* \in A$.

Next, we show that $\mathcal{C}$ is learnable using restricted subset queries. A learner M for $\mathcal{C}$ may first query the languages $L_\varepsilon(x_1^{z_0}a)$, $L_\varepsilon(x_1^{z_1}a)$, $L_\varepsilon(x_1^{z_2}a)$, ..., until the

[4] Note that any query learner can be normalized to pose each query at most once without affecting its learning capabilities.

answer 'yes' is received for the first time, say as a reply to the query $L_\varepsilon(x_1^{z_i}a) = L_{2i}$. Then M queries the language L_{2i+1}. In case the answer is 'yes', let M return the hypothesis L_{2i+1}. Otherwise, let M return the hypothesis L_{2i}. It is not hard to verify that M is a successful query learner for $\mathcal{C}$.

It remains to verify that $\mathcal{C}$ is not learnable using membership queries. Assume to the contrary, that $\mathcal{C}$ is learnable using membership queries, say by the learner M_i for some $i \geq 0$. Then M_i identifies the language $L_{2i} = L_\varepsilon(x_1^{z_i}a)$. In particular, if its queries are answered truthfully respecting L_{2i}, M_i must return a hypothesis correctly describing L_{2i} after finitely many queries. Let l be the length of the longest word M_i queries in the corresponding learning scenario. Then, by definition, $L_{2i+1} = L_\varepsilon(x_1^{z_i}ax_2^{z_l})$. Note that a word of length up to l belongs to L_{2i} iff it belongs to L_{2i+1}. Thus all queries in the learning scenario of M_i for L_{2i} are answered truthfully also for the language $L_{2i+1} \neq L_{2i}$. Since M_i correctly identifies L_{2i}, M_i fails to learn L_{2i+1}. This yields a contradiction. □

Next, we systematically investigate the learnability of some prominent subclasses of the erasing pattern languages in Angluin's [3] model.

Theorem 4. *The class of all regular erasing pattern languages is (i) learnable using membership queries, (ii) learnable using restricted subset queries, and, (iii) learnable using restricted superset queries.*

Proof. For a proof of Assertions *(i)* and *(iii)* see Nessel and Lange [13]. Adapting their ideas one can also prove *(ii)*. □

Theorem 5. *The class of all 1-variable erasing pattern languages is (i) not learnable using membership queries, (ii) not learnable using restricted subset queries, and, (iii) not learnable using restricted superset queries.*

Proof. *(i)* and *(iii)* are due to Nessel and Lange [13]. To verify *(ii)*, note that the class of all languages $L_\varepsilon(ax_1^zb)$, $z \geq 0$, is not learnable using restricted subset queries, even if it is allowed to query any 1-variable erasing pattern language. □

To prove Theorems 6 to 9, similar methods as above can be used. For the results concerning restricted superset queries, ideas from Nessel and Lange [13] can be exploited. Further details are omitted.

Theorem 6. *The class of all constant-free 1-variable erasing pattern languages is (i) learnable using membership queries, (ii) learnable using restricted subset queries, and, (iii) not learnable using restricted superset queries.*

Theorem 7. *The class of all k-variable erasing pattern languages is (i) not learnable using membership queries, (ii) not learnable using restricted subset queries, and, (iii) not learnable using restricted superset queries.*

Theorem 8. *The class of all constant-free k-variable erasing pattern languages is (i) not learnable using membership queries, (ii) not learnable using restricted subset queries, and, (iii) not learnable using restricted superset queries.*

Theorem 9. *The class of all non-cross erasing pattern languages is (i) not learnable using membership queries, (ii) not learnable using restricted subset queries, and, (iii) not learnable using restricted superset queries.*

3.2 Learning with extra queries

As it turns out, there are not so many natural subclasses of the erasing pattern languages that are learnable using restricted subset and restricted superset queries, respectively. But where does the observed weakness stem from? Does it result from the complexity of the considered language classes? The following investigations seem to prove that this is not the case. Instead it seems as if, at least in some cases, the query learners are simply not allowed to ask the 'appropriate' queries.

In the extended model, the query learner is not constrained to query just languages belonging to the target class. In a reasonable model, there has to be an *a priori* agreement of how to formulate the queries. For that purpose, we assume that the query languages are selected from an *a priori* fixed *indexable class of recursive languages.*

As we will see below, this may severely increase the learning power concerning natural subclasses of the erasing pattern languages. Still, if the class of all erasing pattern languages is considered, a benefit resulting from extra queries has not been verified yet.

Theorem 10. *The class of all erasing pattern languages is not learnable using extra restricted subset queries.*

It remains open whether or not the class of all erasing pattern languages is learnable using extra restricted superset queries. The relevance of this problem is discussed in the last section.

Extra restricted superset queries improve the power of query learners remarkably. Due to the space constraints the corresponding proof is omitted.

Theorem 11. *The classes of all regular, of all k-variable, and of all non-cross erasing pattern languages, respectively, are learnable using extra restricted superset queries.*

In contrast, extra restricted subset queries do not help for learning the natural subclasses of the erasing pattern languages considered. Note that there are still subclasses which are learnable using restricted subset queries if and only if the learner may ask extra languages. An example can be found in the demonstration of Theorem 3, third paragraph.

Theorem 12. *(i) The classes of all constant-free k-variable and of all non-cross erasing pattern languages, respectively, are not learnable using extra restricted subset queries.*
(ii) The classes of all regular and of all constant-free 1-variable erasing pattern languages, respectively, are learnable using extra restricted subset queries.

Proof. Assertion *(ii)* is an immediate consequence of Theorems 4 and 6. To prove Assertion *(i)*, note that the class consisting of $L_\varepsilon(x_1^2)$ and all languages $L_\varepsilon(x_1^2 x_2^z)$, $z \geq 2$, is not learnable with extra restricted subset queries, so the classes of all constant-free 2-variable and of all non-cross erasing pattern languages, respectively, aren't either. □

4 Efficiency of query learners

Having analyzed the learnability of natural subclasses of the class of all erasing pattern languages in the (extended) query model, we now turn our attention to the question, which of the learnable classes can even be learned efficiently, i. e. with polynomially many queries. In particular, it is of interest, whether or not the permission to query extra languages may speed up learning.

As it turns out, there are subclasses, which are not learnable in the original model, but even efficiently learnable with extra queries, see Theorem 13, Assertion *(iv)*. Thus, extra restricted superset queries may bring the maximal benefit imaginable. In contrast, extra restricted subset queries do not help to speed up learning of the prominent subclasses of the erasing pattern languages considered above.

Theorem 13. *(i) Polynomially many queries suffice to learn the class of all regular erasing pattern languages with restricted superset queries.*
(ii) Polynomially many queries suffice to learn the class of all constant-free 1-variable erasing pattern languages with membership queries.
(iii) Polynomially many queries suffice to learn the class of all constant-free 1-variable erasing pattern languages with restricted subset queries.
(iv) Polynomially many queries suffice to learn the classes of all regular, of all 1-variable, and of all non-cross erasing pattern languages, respectively, with extra restricted superset queries.

Proof. (i) is due to Nessel and Lange [13]. The proofs of *(ii)* and *(iii)* are omitted. Results by Nessel and Lange [13] help to verify Assertion *(iv)* for the case of regular erasing pattern languages. Details are omitted.

The more involved proof of Assertion *(iv)* for the case of non-cross erasing pattern languages is just sketched:

Assume that the target language L equals $L_\varepsilon(p)$ for some non-cross pattern $p = x_1^{e_1} \cdots x_n^{e_n}$. A query learner M successful for all non-cross erasing pattern languages may operate as follows.

1. M poses the query $\Sigma^* \setminus \{a\}$. If the answer is 'no', then M returns the hypothesis $L = L_\varepsilon(x_1)$ and stops; otherwise M acts as described in 2.
2. The queries $\{w \mid |w| \neq j\}$ for $j = 1, 2, \ldots$ help to determine the minimal exponent e in $\{e_1, \ldots, e_n\}$. Knowing e, M executes 3.
3. M poses the query $L_\varepsilon(x_1^e)$. If the answer is 'yes', then M returns the hypothesis $L = L_\varepsilon(x_1^e)$ and stops; otherwise M acts as described in 4.
4. The queries $(L_\varepsilon(x_1^e) \cap \{w \mid |w| \leq j\}) \cup \{w \mid |w| > j\}$ for $j = e, e{+}1, \ldots$ help to determine further candidates for elements in $\{e_1, \ldots, e_n\}$. Queries concerning special words in a selected class of (at most $e_1 + \cdots + e_n$) 2-variable erasing pattern languages help to exactly compute a next exponent e'. Knowing e', M executes 5.
5. The queries $\Sigma^* \setminus \{w\}$, for particular words $w \in \Sigma^*$ in order of growing length, help to determine in which order the exponents e and e' appear in p.

Afterwards, M executes (slightly modified versions of) steps 3 to 5 in order to find further exponents, until the correct structure of p is output.

All in all, this method is successful for all non-cross erasing pattern languages, but uses only polynomially many extra restricted superset queries. Instead of formalizing the details we try to illustrate the idea with an example.

Assume $\Sigma = \{a, b, c\}$ and the target language is $L_\varepsilon(x_1^4 x_2^2 x_3^8)$. Then the corresponding learning scenario can be described by the following table.

Step	*Query*	*Reply*	*Output of M*
1	$\Sigma^* \setminus \{a\}$	‘yes’	
2	$\{w \mid \lvert w\rvert \neq 1\}$	‘yes’	
	$\{w \mid \lvert w\rvert \neq 2\}$	‘no’	
	(* $e = 2$. *)		
3	$L_\varepsilon(x_1^2)$	‘no’	
	(* There is a second exponent e'. *)		
4	$(L_\varepsilon(x_1^2) \cap \{w \mid \lvert w\rvert \leq 2\}) \cup \{w \mid \lvert w\rvert > 2\}$	‘yes’	
	$\vdots$	$\vdots$	
	$(L_\varepsilon(x_1^2) \cap \{w \mid \lvert w\rvert \leq 5\}) \cup \{w \mid \lvert w\rvert > 5\}$	‘yes’	
	$(L_\varepsilon(x_1^2) \cap \{w \mid \lvert w\rvert \leq 6\}) \cup \{w \mid \lvert w\rvert > 6\}$	‘no’	
	(* $e' = 4$. *)		
5	$\Sigma^* \setminus \{a^2 b^4\}$	‘yes’	
	$\Sigma^* \setminus \{a^4 b^2\}$	‘no’	
	(* e' appears only before e in p. *)		
3	$L_\varepsilon(x_1^4 x_2^2)$	‘no’	
	(* There is a third exponent e''. *)		
4	$(L_\varepsilon(x_1^4 x_2^2) \cap \{w \mid \lvert w\rvert \leq 6\}) \cup \{w \mid \lvert w\rvert > 6\}$	‘yes’	
	$\vdots$	$\vdots$	
	$(L_\varepsilon(x_1^4 x_2^2) \cap \{w \mid \lvert w\rvert \leq 9\}) \cup \{w \mid \lvert w\rvert > 9\}$	‘yes’	
	$(L_\varepsilon(x_1^4 x_2^2) \cap \{w \mid \lvert w\rvert \leq 10\}) \cup \{w \mid \lvert w\rvert > 10\}$	‘no’	
	(* Candidates for e'' are 6 and 8. Interesting words are $a^6 b^4$, $a^2 b^8$, $a^8 b^2$. *)		
	$\Sigma^* \setminus \{a^6 b^4\}$	‘yes’	
	$\Sigma^* \setminus \{a^2 b^8\}$	‘no’	
	(* $e'' = 8$, e'' appears after e in p, step 5 is not necessary. *)		
3	$L_\varepsilon(x_1^4 x_2^2 x_3^8)$	‘yes’	hypothesis $L_\varepsilon(x_1^4 x_2^2 x_3^8)$

It remains to prove Assertion *(iv)* for 1-variable erasing pattern languages:

Assume the target language is $L = L_\varepsilon(p)$ for some 1-variable pattern p. Let v be the shortest word in L, $v = v_1 \cdots v_l$ for $v_1, \ldots, v_l \in \Sigma$. A query learner M successful for all 1-variable erasing pattern languages may operate as follows:

1. With the help of the queries $\Sigma^* \setminus \{a\}$ and $\Sigma^* \setminus \{b\}$ the learner M can find out, whether or not $L = L_\varepsilon(x_1)$. If yes, then M returns the hypothesis $L = L_\varepsilon(x_1)$ and stops; otherwise M acts as described in 2.

2. The queries $\{w \mid |w| \neq j\}$ for $j = 0, 1, 2, \ldots$ help to compute the length l of v. To compute v itself, the $|\Sigma|^l$ candidates for v are recursively split into two equally large sets V_1 and V_2; which of these sets is taken under consideration, in each splitting step only depends on the query $V_1 \cup \{w \mid |w| \neq l\}$. If v is computed, M goes on as in 3.
3. M poses the query $L_\varepsilon(v)$. On answer 'yes', M returns the hypothesis $L_\varepsilon(v)$ and stops. On answer 'no', M queries all the languages $L_\varepsilon(p_i)$, $1 \leq i \leq l+1$, where p_i is the pattern resulting from $x_1 v_1 x_2 v_2 \cdots x_l v_l x_{l+1}$, if the variable x_i is deleted. Thus M can detect exactly those positions in v, where the only variable has to occur (at least once). Knowing the positions of the variables, M goes on as in 4.
4. By posing the queries $\{v\} \cup \{w \mid |w| \geq l+j\}$ for $j = 1, 2, \ldots$, M finds out the number j^* of occurrences of the variable x_1 in p. Afterwards, special queries concerning the words of length $l + j^*$ help to find out the multiplicity of x_1 in the positions computed in 3. Finally, a hypothesis for $L_\varepsilon(p)$ is returned.

All in all, this method is successful for all 1-variable erasing pattern languages, but uses only polynomially many queries. Instead of formalizing the details we try to illustrate the idea with an example.

Assume $\Sigma = \{a, b, c\}$ and the target language is $L_\varepsilon(ax_1^3bx_1^2)$. Then the corresponding learning scenario can be described by the following table.

Step	*Query*	*Reply*	*Output of M*
1	$\Sigma^* \setminus \{a\}$	'yes'	
2	$\{w \mid \|w\| \neq 0\}$	'yes'	
	$\{w \mid \|w\| \neq 1\}$	'yes'	
	$\{w \mid \|w\| \neq 2\}$	'no'	
	(* $l = 2$. $v \in \{aa, ab, ac, ba, bb, bc, ca, cb, cc\}$. *)		
	$\{aa, ab, ac, ba\} \cup \{w \mid \|w\| \neq 2\}$	'yes'	
	$\{aa, ab\} \cup \{w \mid \|w\| \neq 2\}$	'yes'	
	$\{aa\} \cup \{w \mid \|w\| \neq 2\}$	'no'	
	(* $v = ab$. *)		
3	$L_\varepsilon(ab)$	'no'	
	$L_\varepsilon(ax_2bx_3)$	'yes'	
	$L_\varepsilon(x_1abx_3)$	'no'	
	$L_\varepsilon(x_1ax_2b)$	'no'	
	(* $p = ax_1^{e_1}bx_1^{e_2}$ for some $e_1, e_2 \geq 1$. *)		
4	$\{ab\} \cup \{w \mid \|w\| \geq 3\}$	'yes'	
	⋮	⋮	
	$\{ab\} \cup \{w \mid \|w\| \geq 7\}$	'yes'	
	$\{ab\} \cup \{w \mid \|w\| \geq 8\}$	'no'	
	(* $j^* = 5$. Test $a^2ba^4, a^3ba^3, a^4ba^2, a^5ba$. *)		
	$\Sigma^* \setminus \{a^2ba^4\}$	'yes'	
	$\Sigma^* \setminus \{a^3ba^3\}$	'yes'	
	$\Sigma^* \setminus \{a^4ba^2\}$	'no'	hypothesis $L_\varepsilon(ax_1^3bx_1^2)$

Further details are omitted. Note that a similar, slightly extended, method can be used to verify that polynomially many extra restricted superset queries suffice to learn the class of all 2-variable erasing pattern languages. □

Theorem 14. *Polynomially many queries do not suffice to learn the class of all regular erasing pattern languages with either membership queries, or restricted subset queries, or extra restricted subset queries.*

Proof. Note that, for any $n \geq 0$, there are at least $|\varSigma|^n$ distinct regular patterns, such that each pair of corresponding erasing pattern languages is disjoint. By a result in Angluin [3], given $n \geq 0$, any query learner identifying each of these $|\varSigma|^n$ erasing regular pattern languages using membership or restricted subset queries must make $|\varSigma|^n - 1$ queries in the worst case. Angluin's proof can be adopted for the case of learning with extra restricted subset queries. Concerning membership queries and restricted subset queries, Theorem 14 has also been verified by Nessel and Lange [13]. □

It remains open, whether or not, for any $k \geq 3$, the class of all k-variable erasing pattern languages, or at least the class of all constant-free k-variable erasing pattern languages, is learnable using polynomially many extra restricted superset queries. Until now, we have only been successful in showing that Theorem 13, Assertion *(iv)* generalizes to the case of learning the class of all 2-variable erasing pattern languages. The relevant details are omitted.

5 Connections to Gold-style learning

Comparing query learning to the standard models of Gold-style language learning from positive examples requires some more notions. These will be kept short, see, e. g., Gold [7], Angluin [1], and Zeugmann and Lange [18] for more details.

Let L be a language. Any infinite sequence $t = (w_j)_{j \geq 0}$ with $\{w_j \mid j \geq 0\} = L$ is called a *text* for L. For any $n \geq 0$, t_n denotes the initial segment $w_0, \ldots, w_n$ and t_n^+ the set $\{w_0, \ldots, w_n\}$.

Let $\mathcal{C}$ be an indexable class, let $\mathcal{H} = (L_i)_{i \geq 0}$ be a hypothesis space, and let $L \in \mathcal{C}$. An *inductive inference machine* (*IIM*) is an algorithmic device, that reads longer and longer initial segments of a text and, from time to time, outputs numbers as its hypotheses. An IIM M returning some i is construed to hypothesize the language L_i. Given a text t for L, M *identifies* L *from* t *with respect to* $\mathcal{H}$, if the sequence of hypotheses output by M, when fed t, stabilizes on a number i (i. e. past some point M always outputs the hypothesis i) with $L_i = L$. M *identifies* $\mathcal{C}$ *from text* with respect to $\mathcal{H}$, if it identifies every $L' \in \mathcal{C}$ from every corresponding text. We say that $\mathcal{C}$ can be *conservatively* identified with respect to $\mathcal{H}$ iff there is an IIM M that identifies $\mathcal{C}$ from text with respect to $\mathcal{H}$ and that performs exclusively justified mind changes, i. e. if M, on some text t, outputs hypotheses i and later i', then M must have seen some word $w \notin L_i$ before it outputs i'. In other words, M may only change its hypothesis when it has found hard evidence that it is wrong.

LimTxt (*ConsvTxt*) denotes the collection of all indexable classes $\mathcal{C}'$ for which there are an IIM M' and a hypothesis space $\mathcal{H}'$ such that M' (conservatively) identifies $\mathcal{C}'$ from text with respect to $\mathcal{H}'$. Note that $\mathit{ConsvTxt} \subset \mathit{LimTxt}$, cf. Zeugmann and Lange [18].

For the next theorem, let *xSupQ* denote the class of all indexable classes, which are learnable with extra restricted superset queries.

Theorem 15. *$\mathit{ConsvTxt} = \mathit{xSupQ} \subset \mathit{LimTxt}$.*

Proof. "$\mathit{ConsvTxt} \subseteq \mathit{xSupQ}$":

Fix $\mathcal{C} \in \mathit{ConsvTxt}$. Then there is an indexing $(L_i)_{i\geq 0}$ and a learner M, such that M *ConsvTxt*-identifies $\mathcal{C}$ with respect to $(L_i)_{i\geq 0}$. Obviously, if $L \in \mathcal{C}$ and t is a text for L, then M never returns an index i with $L \subset L_i$ on any segment of t.

Now the underlying indexable class used for the queries contains all languages in $(L_i)_{i\geq 0}$ and all languages $L_i \setminus \{w\}$ for $i \geq 0$ and $w \in \Sigma^*$. A learner M' identifying any $L \in \mathcal{C}$ with extra restricted superset queries may work as follows:

> M' poses queries $L_0, L_1, \ldots$, until the answer 'yes' is received for the first time, say upon the query L_k. (* Note that $L \subseteq L_k$. *)
>
> Let T be the set of all words $w \in \Sigma^*$, for which the query $L_k \setminus \{w\}$ is answered with 'no'. Note that $T = L$ and T is recursively enumerable in k. The latter guarantees that one can effectively enumerate a text t for L.
>
> M' executes step 0. In general, step n, $n \geq 0$, consists of the following instructions:
>
> > Determine $i := M(t_n)$. Pose the query L_i. If the answer is 'no', execute step $n+1$. Otherwise hypothesize i and stop. (* In the latter case, as M never returns an index of a proper superset of L, M' returns an index for L. *)

Further details are omitted.

"$\mathit{xSupQ} \subseteq \mathit{ConsvTxt}$":

Fix an indexable class $\mathcal{C} \in \mathit{xSupQ}$. Then there is an indexing $(L_i)_{i\geq 0}$ and a query learner M, such that M identifies $\mathcal{C}$ with extra restricted superset queries respecting $(L_i)_{i\geq 0}$. A new indexing $(L'_i)_{i\geq 0}$ is defined as follows:

- L'_0 is the empty language.
- If i is the canonical index of the finite set $\{i_1, \ldots, i_n\}$, then $L'_i = L_{i_1} \cap \cdots \cap L_{i_n}$.

A learner M' identifying $\mathcal{C}$ in the limit from text with respect to the hypothesis space $(L'_i)_{i\geq 0}$, given a text t, may work as follows.

> $M'(t_0) := 0$.
>
> To compute $M'(t_{n+1})$, the learner M' simulates a query learning scenario with M for n steps of computation. If M does not return a hypothesis in the n-th step, then $M'(t_{n+1}) := M'(t_n)$. Additionally, if M poses the query L_i in the n-th step, then M will receive the answer 'no', if $t_n^+ \cap \overline{L_i} \neq \emptyset$ (i. e. if $L_i \not\supseteq t_n^+$), and the answer 'yes', otherwise. If M returns a hypothesis i in the n-th step, then the hypothesis $M'(t_{n+1})$ is computed as follows:

- Let $L_{i_1^+}, \dots, L_{i_m^+}$ be the queries answered with 'yes' in the currently simulated scenario.
- Compute the canonical index i' of the set $\{i, i_1^+, \dots, i_m^+\}$.
- Return the hypothesis $M'(t_{n+1}) = i'$.

It is not hard to verify that M' learns $\mathcal{C}$ in the limit from text; the relevant details are omitted. Moreover, as we will see next, M' avoids overgeneralized hypotheses, that means, if t is a text for some $L \in \mathcal{C}$, $n \geq 0$, and $M'(t_n) = i'$, then $L'_{i'} \not\supset L$. Therefore, M' can easily be transformed into a learner M'' which identifies the class $\mathcal{C}$ conservatively in the limit from text.[5]

To prove that M' learns $\mathcal{C}$ in the limit from text without overgeneralizations, assume to the contrary, that there is an $L \in \mathcal{C}$, a text t for L, and an $n \geq 0$, such that the hypothesis $i' = M'(t_n)$ fulfills $L'_{i'} \supset L$. Then $i' \neq 0$. By definition of M', there must be a learning scenario S for M, in which

- M poses queries $L_{i_1^-}, \dots, L_{i_k^-}, L_{i_1^+}, \dots, L_{i_m^+}$ (in some particular order);
- the queries $L_{i_1^-}, \dots, L_{i_k^-}$ are answered with 'no';
- the queries $L_{i_1^+}, \dots, L_{i_m^+}$ are answered with 'yes';
- afterwards M returns the hypothesis i.

Hence i' is the canonical index of the set $\{i, i_1^+, \dots, i_m^+\}$. This implies $L'_{i'} = L_i \cap L_{i_1^+} \cap \dots \cap L_{i_m^+}$. So each of the languages $L_{i_1^+}, \dots, L_{i_m^+}$ is a superset of L. By definition of M', $L_{i_j^-} \not\supseteq t_n^+$ for $1 \leq j \leq k$. Therefore none of the languages $L_{i_1^-}, \dots, L_{i_k^-}$ are supersets of L. So the answers in the learning scenario S above are truthful respecting the language L. As M learns $\mathcal{C}$ with extra restricted superset queries, the hypothesis i must be correct for L, i. e. $L_i = L$. This yields $L'_i = L$ in contradiction to $L'_{i'} \supset L$.

So M' learns $\mathcal{C}$ in the limit from text without overgeneralizations, which finally implies $\mathcal{C} \in \mathit{Consv\,Txt}$.

"$\mathit{xSupQ} \subset \mathit{Lim\,Txt}$":

Finally, this is an immediate consequence of $\mathit{xSupQ} = \mathit{Consv\,Txt}$ and the fact $\mathit{Consv\,Txt}$ is a proper subset of $\mathit{Lim\,Txt}$. □

Theorem 15 is of relevance for the open question, whether or not the class of all erasing pattern languages is learnable in the limit from text, if the underlying alphabet consists of at least three symbols. Obviously, if this class is learnable with extra restricted superset queries, then the open question can be answered in the affirmative. Conversely, if it is not learnable with extra restricted superset queries, then it is not conservatively learnable in the limit from text. Of course the latter would not yet imply, that the open question can be answered in the negative. Still it would at least suggest that this is the case, since until now, there is no 'natural' class known that separates *Lim Txt* from *Consv Txt*.

[5] Note that a result by Zeugmann and Lange [18] states that any indexable class, which is learnable in the limit from text without overgeneralizations, belongs to *Consv Txt*.

References

1. D. Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45:117–135, 1980.
2. D. Angluin. Finding patterns common to a set of strings. *Journal of Computer and System Sciences*, 21:46–62, 1980.
3. D. Angluin. Queries and concept learning. *Machine Learning* 2:319–342, 1988.
4. D. Angluin. Queries revisited. *Proc. Int. Conf. on Algorithmic Learning Theory*, LNAI 2225, 12–31, Springer, 2001.
5. S. Arikawa, S. Miyano, A. Shinohara, S. Kuhara, Y. Mukouchi, T. Shinohara. A machine discovery from amino acid sequences by decision trees over regular patterns. *New Generation Computing*, 11:361–375, 1993.
6. T. Erlebach, P. Rossmanith, H. Stadtherr, A. Steger, T. Zeugmann. Learning one-variable pattern languages very efficiently on average, in parallel, and by asking questions *Proc. Int. Conf. on Algorithmic Learning Theory*, LNAI 1316, 260–276, Springer, 1997.
7. E. M. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.
8. J. E. Hopcroft, J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley Publishing Company, 1979.
9. T. Jiang, A. Salomaa, K. Salomaa, S. Yu. Decision problems for patterns. *Journal of Computer and System Sciences*, 50:53–63, 1995.
10. S. Lange, T. Zeugmann. Types of monotonic language learning and their characterization. *Proc. ACM Workshop on Computational Learning Theory*, 377–390. ACM Press, 1992.
11. S. Matsumoto, A. Shinohara. Learning pattern languages using queries. *Proc. European Conf. on Computational Learning Theory*, LNAI 1208, 185–197, Springer, 1997.
12. A. Mitchell. Learnability of a subclass of extended pattern languages. *Proc. ACM Workshop on Computational Learning Theory*, 64–71, ACM-Press, 1998.
13. J. Nessel, S. Lange. Learning erasing pattern languages with queries. *Proc. Int. Conf. on Algorithmic Learning Theory*, LNAI 1968, 86–100, Springer, 2000.
14. D. Reidenbach. A negative result on inductive inference of extended pattern languages. *Proc. Int. Conf. on Algorithmic Learning Theory*, LNAI 2533, 308–320, Springer, 2002.
15. A. Salomaa. Patterns (the formal language theory column). *EATCS Bulletin*, 54:46–62, 1994.
16. A. Salomaa. Return to patterns (the formal language theory column). *EATCS Bulletin*, 55:144–157, 1995.
17. T. Shinohara. Polynomial time inference of extended regular pattern languages. *Proc. RIMS Symposium on Software Science and Engineering*, LNCS 147, 115–127, Springer, 1983.
18. T. Zeugmann, S. Lange. A guided tour across the boundaries of learning recursive languages. *Algorithmic Learning for Knowledge-Based Systems*, LNAI 961, 190–258, Springer, 1995.