Proc. the Fifth Genome Informatics Workshop (GIW'94), 39-48, Dec. 1994

Protein Motif Discovery from Positive Examples by Minimal Multiple Generalization over Regular Patterns

Hiroki Arimura Ryoichi Fujino Takeshi Shinohara* Setsuo Arikawa

*† Department of Artificial Intelligence
Kyushu Institute of Technology,
Iizuka 820, Japan

‡ Research Institute of
Fundamental Information Science
Kyushu University,
Fukuoka 812, Japan

Abstract

Recently, several attempts have been made at applying machine learning method to protein motif discovery, but most of these methods require negative examples in addition to positive examples. This paper proposes an efficient method for learning protein motif from positive examples. A regular pattern is a string consisting of constant symbols and mutually distinct variables, and represents the set of the constant strings obtained by substituting nonempty constant strings for variables. Regular patterns and their languages are called extended if empty substitutions are allowed. Our learning algorithm, called k-minimal multiple generalization (k-mmg), finds a minimally general collection of at most k regular patterns that explains all the positive examples. We have implemented this algorithm for subclasses for regular patterns and extended regular patterns where the number of variables are bounded by a small constant, and run experiments on protein data taken from GenBank and PIR databases. We incorporate three heuristics into these algorithms for controlling nondeterministic choices. The experiments show that the k-mmg algorithm can very quickly find a hypothesis on the computers in practice, and that the results of our system are comparable with the results of learning method from positive and negative data.

1 Introduction

A *motif* is a pattern common to the essential parts in proteins which share a function. It is one of the most important problems in Molecular Biology to capture a motif from amino acid sequences. Recently, several attempts have been made at applying machine learning method to protein motif discovery [2, 3, 9], but most of these methods require negative examples in addition to positive examples. For example, Arikawa et al. [2] reported a knowledge acquisition system for finding motifs from amino acid sequences based on Elementary Formal Systems [4]. Their system produced a set of regular patterns of high accuracy from randomly chosen positive and negative examples. This paper proposes an efficient method for learning protein motif from *positive* examples.

	A set S of positive examples
e_1	WLVNFIIVIMVFILFLVGLYLL
e_2	VALVTITLWFMAWTPYLVINCMGL
e_3	GFLAASALGVVMITAALAGIL
e_4	SKILGLFTLAIMIISCCGNGVVVYI
e_5	MTIKTSIMKILFIWMMAVFWT
e_6	IFYSIFVYYIPLFLICYSYWFIIAAVSA
e_7	GCGSLFGCVSIWSMCMIAFDRYNVIV

	A 3-mmg P of S
p_1	*F*M*LV*L
p_2	*FL*V*A*
p_3	*LF*M*V*

Table 1: An example of k-minimal multiple generalizations for $k = 3$, where '$*$' denotes a variable in a pattern. Regular patterns p_1, p_2, and p_3 in the 3-mmg P of S cover subsets $\{e_1, e_2\}$, $\{e_3, e_6\}$, and $\{e_4, e_5, e_7\}$ of the set S of positive examples, respectively.

A *regular pattern* [1, 11] is a string p of the form $w_0 x_1 w_1 x_2 \dots x_n w_n$ ($n \geq 0$) consisting of constant strings $w_0, w_1, \dots, w_n$ and mutually distinct variables $x_1, \dots, x_n$, and represents the language $L(p)$ of all the constant strings obtained by substituting nonempty constant strings for the variables. Patterns and their languages are called *extended* [12] if they allow empty substitutions. Our learning algorithm finds a minimally general collection $\{p_1, \dots, p_l\}$ of at most k regular patterns such that the union $L(p_1) \cup \cdots \cup L(p_l)$ of their languages covers all the positive examples. We call such a minimally general collection a *k-minimal multiple generalization* (k-mmg) of S over regular patterns. In Table 1, we show an example of the k-mmg's over extended regular pattern.

In the previous work [5], we developed a polynomial time algorithm, called k-mmg algorithm, for finding one of the k-mmg's over regular patterns of a given set of positive examples. Extending this result, we construct a polynomial time k-mmg algorithm for extended regular pattern languages. Based on these results, we have implemented the k-mmg algorithm for two subclasses, *m-variable regular patterns* and *m-variable extended regular patterns* by restricting the number of variables to a small constant $m \geq 0$. These k-mmg algorithms use a greedy search method to obtain more specific patterns and the order of search influences produced hypotheses. Therefore, we incorporate the following three heuristics into these algorithms for controlling the order of greedy search: *randomized approach, maximal covering approach*, and *using-negative approach.*

Then, we have run experiments on protein data of transmembrane domains and signal peptides drawn from PIR and GenBank [7, 10]. The selection of hypothesis space is one of the most important factors in applying a machine learning algorithm to practical problem. The classes of extended patterns and nonextended patterns have the same set of representations, but very different semantics because the former allows empty substitutions and the latter not [12]. For example, assume that a regular pattern p matches a string w. For every $n \geq 1$, if consecutive occurrences of n variables $x_1 x_2 \cdots x_n$ in p match some substring of w then the substring must be a string of length at least n for nonextended substitution, but can be string of any length for extended substitutions. Preliminary experiments revealed the weakness of nonextended regular patterns in learning protein motifs compared with extended regular patterns. Hence, we mainly consider k-mmg's over extended regular patterns in the experiments.

In experiments on transmembrane domains, the system produced some hypotheses of high accuracy for the data preprocessed with the hydropathy indices due to Kyte and Doolittle [8];

one of such hypotheses has the accuracy around 89% and 85% for positive and negative examples, respectively. These results of our system are comparable with the results of the learning system from positive and negative examples in [2]. The experiments also show that the k-mmg algorithm can very quickly find a hypothesis on the computers in practice. In the experiments, we also compared three control heuristics in accuracy of the produced hypothesis.

2 Minimal Multiple Generalizations

In this section, we give basic definitions and results on the framework of minimal multiple generalization according to Arimura et al. [5]. First, we introduce extended regular patterns [12]. For a set A, we denote by $\sharp A$ the number of elements of A. Let $\Sigma = \{a, b, A, B, \dots\}$ be a finite set of *constant symbols* and $X = \{x, y, z, x_1, x_2, \dots\}$ be a countable set of *variables* disjoint from Σ. We denote by Σ^* the set of all the finite strings over Σ and $\Sigma^+ = \Sigma^* - \{\varepsilon\}$, where ε is the *empty string.*

A *regular pattern* is a string consisting of constant symbols and variables in which any variable appears at most once. A *substitution* is a homomorphism θ from regular patterns to themselves such that $\theta(a) = a$ for any $a \in \Sigma$. The form $\{x_1 := p_1, \dots, x_n := p_n\}$ denotes the substitution that maps x_i to p_i and other symbol to itself. A regular pattern $p = w_0x_1w_1x_2 \cdots x_nw_n$ ($w_i \in \Sigma^*$) defines the *language* $L(p)$ as the set of all the constant strings obtained by substituting possibly empty constant strings to the variables $x_1, x_2, \dots, x_n$. In other words, $L(p)$ consists of all the strings containing substrings $w_0, w_1, \dots, w_n$ in this order. Note that we allow ε-substitutions, that is, ε may be substituted for a variable. A set L of constant strings is said to be an *extended regular pattern language* if $L = L(p)$ for some regular pattern p. Let P be a finite set of regular patterns. Then, the set P defines a *union* $L(P) = \bigcup_{p \in P} L(p)$ of extended regular pattern languages.

A regular pattern p is of *canonical form* if p contains no consecutive occurrences of variables, that is, $p = w_0x_1w_1 \cdots x_nw_n$, $w_i \in \Sigma^*$ for $i = 0, n$, and $w_j \in \Sigma^+$ for any $j = 1, \dots, n-1$. Any extended regular pattern language can be defined by a regular pattern of canonical form. We denote by $\mathcal{RP}_\varepsilon$ the class of all the regular patterns of canonical form consisting from symbols in $\Sigma \cup X$. Note that the class $\mathcal{RP}_\varepsilon$ contains ε. In what follows, we will deal with only regular patterns of canonical form and identify patterns obtained by renaming of variables from each other. We define a partial ordering $\preceq$ on $\mathcal{RP}_\varepsilon$: $p \preceq q$ iff $p = \theta(q)$ for some substitution θ. We write $p \prec q$ if $p \preceq q$ but $q \not\preceq p$. If $p \preceq q$, we say q is *more general than* p, p is *more specific than* q or q *subsumes* p.

Let D be any subset of $\mathcal{RP}_\varepsilon$. A set P of regular patterns is *reduced* if P contains no p, q such that $p \prec q$. Let k be a positive integer and D^k be the class of all the reduced collections of at most k regular patterns in D. Then, we define a partial ordering $\sqsubseteq$ on D^k: $P \sqsubseteq Q$ iff for any $p \in P$ there is some $q \in Q$ such that $p \prec q$. Note that $P \sqsubseteq Q$ implies $L(P) \subseteq L(Q)$ but the converse does not hold in general.

Definition. For a class D of patterns, a *k-minimal multiple generalization* (*k-mmg*, for short) of a set S of strings is a minimal element P in D^k with respect to $\sqsubseteq$ such that $S \subseteq L(P)$.

We say D^k has *the compactness with respect to containment* (*compactness*, for short) if for any $p \in D$ and any $Q \in D^k$, $L(p) \subseteq L(Q) \iff L(p) \subseteq L(q)$ for some $q \in Q$. If D^k has the compactness, then a k-mmg of a set S of strings defines a minimal language containing S

within unions of at most k extended regular pattern languages [5]. Under the assumption of compactness, the following theorem justifies the use of k-mmg in learning proteins in the sense of inductive inference from positive data [1, 11].

Theorem 1 (Arimura et al. [5]) *Assume that the class D^k has the compactness. If there is an algorithm that computes one of the k-mmg's of a finite set $S \subseteq \Sigma^*$ in time polynomial in the total size of strings in S, then the class of unions of at most k extended regular pattern languages in D is polynomial time inferable from positive data.*

3 Learning Algorithm

In this section, we develop a learning algorithm that produces a k-mmg of a given finite set of constant strings in polynomial time for every fixed $k \geq 1$ based on the framework in [5]. For the detail of the algorithm and the proofs of the theorems below, see [5] and [6].

We first review the general design scheme of k-mmg algorithm in [5]. Let D be a subclass of regular patterns. A *refinement operator* for D is a mapping ρ that maps a pattern $p \in D$ to a finite set $\rho(p)$ of refinements of p. For $P \subseteq D$, let $\rho(P) = \bigcup_{p \in P} \rho(p)$. We define $\rho^0(p) = \{p\}$ and $\rho^n(p) = \rho(\rho^{n-1}(p))$ for every $n \geq 1$. Let $\rho^+(p) = \bigcup_{n \geq 1} \rho^n(p)$. A refinement operator is said to be *complete* if $p \in \rho^+(q) \iff p \prec q$. A *one-step refinement* of p is a regular pattern q such that $q \prec p$ but there is no r satisfying $q \prec r \prec p$. We can easily observe that ρ is complete iff $\rho(p)$ contains all the one-step refinements of p for any p. A refinement operator is said to be *efficient* if given $p \in D$ the finite set $\rho(p) \subseteq D$ is polynomial time computable.

In Figure 1 to 3, we present an algorithm that computes k-mmg of a given finite set S for a class D of regular patterns. The algorithm searches sets of at most k regular patterns in D from general to specific by using a refinement operator ρ for D.

Theorem 2 (Arimura et al. [5]) *For any subclass D of regular patterns, if there is a complete and efficient refinement operator ρ for D, then MMG in Figure 1 computes one of the k-mmg's of a finite set S of strings in polynomial time in the total size of strings in S.*

Then, we extend the k-mmg algorithm for extended regular patterns. Let p be a canonical regular pattern. We denote by $v(p)$ the set of variables appearing in p. A substitution θ is said to be *extended basic* for p if θ satisfies one of the following conditions:

- $\theta = \{x := xay\}$, where $x \in v(p), y \notin v(p)$, $x \neq y$, and $a \in \Sigma$,
- $\theta = \{x := \varepsilon\}$, where $x \in v(p)$.

Then, we define a refinement operator $\rho(p) = \{\theta(p) \mid \theta \text{ is extended basic for } p\}$.

Theorem 3 *The refinement operator ρ is efficient and complete for extended regular patterns.*

Unfortunately, we could not currently show the compactness for the class $\mathcal{RP}^k_{\varepsilon}$. Hence, we restrict our attention to its subclass. For $m \geq 0$, a regular pattern is *m-variable* if it contains at most m distinct variables. We denote by $\mathcal{RP}_{\varepsilon,m}$ the class of m-variable regular patterns.

Theorem 4 *If $\sharp\Sigma > 2km$, then the class $\mathcal{RP}^k_{\varepsilon,m}$ of sets of at most k m-variable regular patterns has the compactness.*

```
MMG($k, S$)
    /* $k \geq 1$ and $S$ is the set of positive examples. */
    $P$ := Tighten($\{x\}, S$);
    $\Delta k := k$;
    while $\Delta k \geq 2$ and there exists some $p \in P$ that is
                $\Delta k$-divisible (*1) with respect to $\Delta S \overset{\mathrm{def}}{=} S - L(P - \{p\})$ (*2) do
        Choose such a divisible member $p$ in $P$ and the corresponding $\Delta S$;   (Choice 1)
        $\Delta P$ := Divide($p, \Delta k, \Delta S$);
        $\Delta P$ := Tighten($\Delta P, \Delta S$);
        $P := (P - \{p\}) \cup \Delta P$; and $\Delta k := k + \sharp P - 1$;
    endwhile
    Return $P$;
```

(*1) a member p in P is *Δk-divisible* with respect to ΔS if the set **Divide**($p, \Delta k, \Delta S$) exists.

(*2) ΔS is the set of positive examples subsumed only by p but not by other members in P.

Figure 1: Minimal multiple generalization algorithm

```
Divide($p, k, S$)
    /* $p$ is a pattern, $k \geq 2$ and $S$ is a set of positive examples */
    Compute the set $\rho(p)$ of one-step refinements;
    Choose a set $P$ of at most $k$ members in $\rho(p)$ that is          (Choice 2)
        reduced with respect to (*3) $S$;
    Return $P$;
```

(*3) a set P is *reduced with respect to S* if $S \subseteq L(P)$ but $S \not\subseteq L(P')$ for any proper subset $P' \subset P$.

Figure 2: Algorithm for dividing a member of ΔP into its refinements

```
Tighten($P, T$)
    /* $P$ is a set of patterns and $T$ is a set of positive examples */
    while for some $q \in P$, there is some $r$ in $\rho(q)$ such that
                $L(r) \supseteq \Delta T \overset{\mathrm{def}}{=} T - L(P - \{q\})$ (*4) do
        Choose such $q$ in $P$ and the corresponding $\Delta T$;          (Choice 3)
        Choose a refinement $r$ in $\rho(q)$ such that $\Delta T \subseteq L(r)$;   (Choice 4)
        $P := (P - \{q\}) \cup \{r\}$;
    endwhile
    Return $P$;
```

(*4) ΔT is the set of positive examples subsumed only by q but not by other members in P.

Figure 3: Algorithm for tightening a multiple generalization by refining patterns.

Now, we give a complete and efficient refinement operator ρ_m for m-variable extended regular patterns. A substitution θ is said to be *extended m-basic* for p if $\sharp v(\theta(p)) \leq m$ and θ satisfies one of the following conditions:

- $\theta = \{x := xay\}$, where $x \in v(p), y \notin v(p)$, $x \neq y$, and $a \in \Sigma$,
- $\theta = \{x := xa\}$ or $\theta = \{x := ax\}$, where $x \in v(p)$ and $a \in \Sigma$,
- $\theta = \{x := \varepsilon\}$, where $x \in v(p)$.

Then, we define $\rho_m(p) = \{\theta(p) \mid \theta$ is extended m-basic for $p\}$.

Theorem 5 *For every $m \geq 1$, the refinement operator ρ_m is efficient and complete for m-variable extended regular patterns.*

Corollary 6 *Let $k, m \geq 1$ be any positive integers, one of the k-mmg's over $\mathcal{RP}_{\varepsilon,m}$ of a finite set S of strings is computable in time $O(\sharp\Sigma \cdot k^3 m^k l^2 n)$, where l is the maximum length of strings in S and $n = \sharp S$.*

By Theorem 1, Theorem 4, and Corollary 6, the class $\mathcal{RP}^k_{\varepsilon,m}$ of unions is polynomial time inferable from positive data when more than $2km$ constant symbols are available. Unfortunately, the compactness may not be satisfied for hypotheses with a small alphabet such as hydropathy plotted data in Section 5. Nevertheless, this restriction on m is useful in avoiding overfitting of hypotheses and in reducing time complexity of the learning algorithm as we can see in Corollary 6 above.

4 Heuristics for Greedy Search

In this section we describe three heuristics used in our learning algorithm. The learning algorithm introduced in the previous section is a greedy algorithm in the sense that it does not backtrack in searching the hypothesis space from general to specific. On the other hand, the algorithm contains four kinds of nondeterministic choices:

(Choice 1) a pattern p in P to be divided at Line 5 in **MMG**,

(Choice 2) a subset P of $\rho(p)$ at Line 3 in **Divide**,

(Choice 3) a pattern q in P to be refined at Line 3 in **Tighten**, and

(Choice 4) a member r of the set $\rho(p)$ at Line 4 in **Tighten**.

At these choice points, the algorithm tests the candidates in some order, and chooses the first candidate satisfying the required condition. Whatever the order of testing candidates is, the algorithm will eventually find a correct k-mmg of given examples. However, different choices of alternatives yield different solutions. In particular, the accuracy of a produced hypothesis depends on the order.

For example, assume that we compute a 3-mmg of a given set $\{abc, abcba, cab, abab, cbbb, abb\}$. Starting from the same hypothesis $\{xaybz, xcybz\}$, if we first refine $xaybz$ then we get a 3-mmg $\{abxb, abcx, cxb\}$ while if we first refine $xcybz$ then we might get another 3-mmg $\{cbbb, abx, cab\}$. The latter solution contains constant strings $cbbb$ and cab. We call such constant strings in a solution *exceptions.*

For Choice 2 and Choice 4, we test the required subset P and the required pattern r in random order. For Choice 1 and Choice 3, respectively, we take three approaches in testing the patterns p and q:

- *Randomized approach*: the system tests the candidates in random order at Choice 1 and Choice 3.
- *Maximal covering approach*: To obtain hypotheses with less exceptions, the system tests patterns covering more positive examples earlier at Choice 1 and Choice 3.
- *Using-negative approach*: This heuristic uses some negative examples to guide the search. To obtain hypotheses with less negative errors, the system tests the patterns containing more negative examples earlier at Choice 1 and Choice 3.

5 Experimental Results

We have run experiments for the learning algorithm augmented with three types of heuristics: randomized, maximal covering, and using-negative approaches. We use the hypothesis space $\mathcal{RP}^k_{\varepsilon,m}$ and the refinement operator ρ_m, where the parameters k and m are appropriately chosen small integers from 3 to 10 and from 3 to 5, respectively. Programs are written in C++ and the tests are run on a Sun SparcStation-10.

Given a large set Pos of positive examples, the learning system randomly draws a small subset pos from Pos, where the number of examples in pos ranges from 20 to 90. For using-negative approach, the system also draws a small subset neg of a large set Neg of negative examples. Then it computes a hypothesis from pos (and neg in using-negative approach) according to the respective heuristics. To compare the effect of heuristics in learning, we used the score $(p\%, n\%)$ as the measure of accuracy, which denotes that the hypothesis covers $p\%$ of positive examples in Pos and excludes $n\%$ of negative examples in Neg.

5.1 Data

The experiments used the data from the following two sources.

(1) *Transmembrane domains*: From PIR database [10], we obtained 689 positive data and 19256 negative data. Since the data on transmembrane domains essentially consists of only positive examples, we randomly took as negative examples sequences of length exactly 30 from all the sequence registered in the database without overlap with transmembrane domains.

(2) *Signal peptide sequences*: From GenBank database [7], we obtained 1018 positive examples as the initial segments of amino acid sequences according to the indication, and 3158 negative examples as the initial segment of 30 amino acids. The leftmost amino acid is almost always ‘M’, and therefore is removed from all the positive and negative examples.

For each sources, we also prepared data preprocessed with hydropathy indices due to Kyte and Doolittle [8]. We transformed twenty symbols of amino acids to three symbols 0, 1, and 2 with the Table 2. This greatly reduced the search space [2].

5.2 Results on transmembrane domains

For transmembrane domains, the system produced some hypotheses with high accuracy. From randomly chosen 25 positive raw data, we obtained a hypothesis with accuracy $(70.4\%, 71.3\%)$ within $\mathcal{RP}^k_{\varepsilon,m}$ for $k = 5, m = 3$ in maximal covering approach (Table 3 (T1)). From randomly chosen 50 positive indexed data, we obtained a hypothesis with high accuracy $(89.1\%, 85.0\%)$ within $\mathcal{RP}^k_{\varepsilon,m}$ for $k = 5, m = 5$ in using-negative approach (Table 3 (T2)).

amino acid	hydropathy	symbol
R K D E N Q H	$-4.5 \sim -3.2$	0
P Y W S T G	$-1.6 \sim -0.4$	1
A M C F L V I	$+1.8 \sim +4.5$	2

Table 2: Hydropathy indices.

We have many hypotheses of high accuracy by other heuristics as well. The accuracy is comparable with the accuracy of some hypotheses found by the learning algorithm in [2]. It is surprising that the accuracy was achieved by a machine learning from only *positive* examples, while the system in [2] requires both positive and negative examples. It should be noted that our system run much faster than their system for the same hypothesis space. In most cases, the algorithm find a hypothesis within 1 second for $10 \sim 50$ positive examples.

(T1) raw data, M

patterns	$p\%$	$n\%$
*IL*I*	20.5	93.4
*L*GI*	17.1	93.1
*L*VI*	23.1	94.1
*LF*I*	18.8	95.0
*VL*A*	29.3	91.2
accuracy	70.4	71.3

(T2) hydropathy plotted data, U

patterns	$p\%$	$n\%$
*2*22*2222*22*	75.4	90.1
*221*2212*22*	57.3	92.3
*222212*21*121*22*	4.5	99.6
120*22*22*0*212122*	0.0	100.0
121201121212222221121212	0.0	100.0
accuracy	89.1	85.0

(S1) raw data, U

patterns	$p\%$	$n\%$
*F*AL*A	6.7	99.6
*FL*GV*	11.5	99.0
*I*FL*G	3.0	99.7
*LL*L*	64.0	65.7
K*F*LF*	1.4	99.5
accuracy	79.5	64.7

(S2) hydropathy plotted data, M

patterns	$p\%$	$n\%$
*1*2222*22*2*1	29.5	89.3
*222*1222*21*1	13.4	94.0
*222*2*22*2	42.0	81.6
0*22*2222*2*1*	30.6	87.8
02*22*00*222*2*	0.1	99.1
accuracy	83.3	64.9

Table 3: Some of the hypothesis with highest accuracy that our learning algorithm produced from positive examples of transmembrane domains (T1, T2) and signal peptides (S1, S2), where '$*$' denotes a variable. The second and the third columns of each table, respectively, show the percentages of the positive examples covered and negative examples excluded by the pattern in the first column. M and U denote "maximal covering" and "using-negative", respectively.

5.3 Results on signal peptide sequences

For signal peptide sequences, the accuracies of hypotheses produced by the system were not so high. From the randomly chosen 30 positive raw data, The best hypothesis we obtained has the accuracy $(79.5\%, 64.7\%)$ within $\mathcal{RP}^k_{\varepsilon,m}$ for $k = 5, m = 3$ in using-negative approach (Table

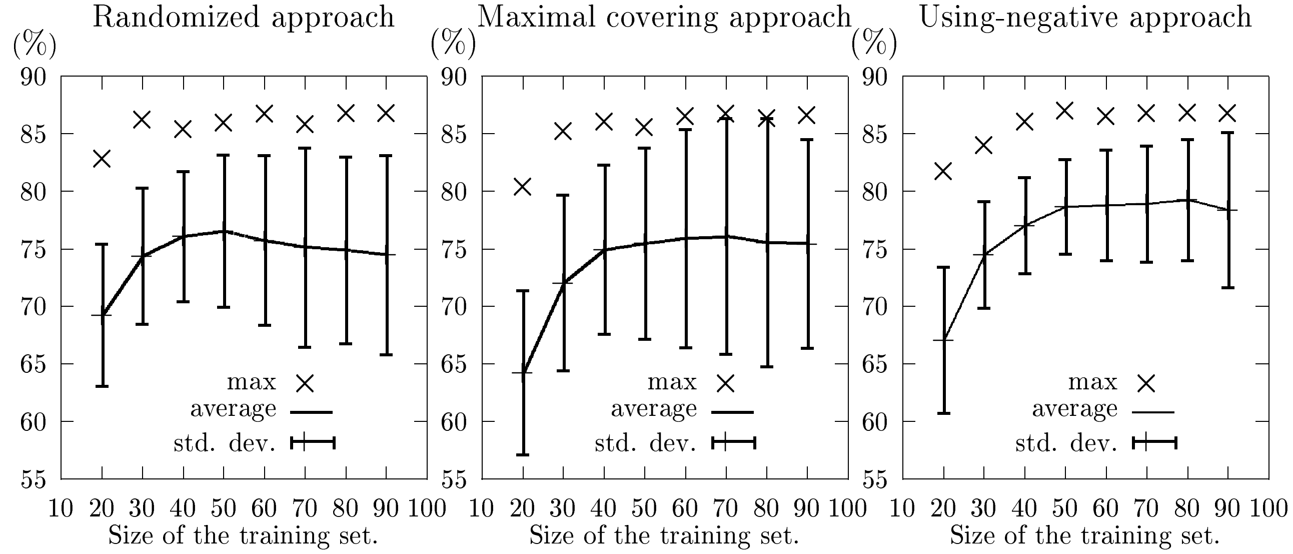

Figure 4: A comparison of accuracies of the solutions obtained by three heuristics on hydropathy plotted data of transmembrane domains. The hypothesis space is $\mathcal{RP}^k_{\varepsilon,m}$ for $k = 5, m = 5$. The X-axes and the Y-axes show the size of the training set and the total accuracy $\sqrt{p \cdot n}$ of a hypothesis, respectively, where p and n are positive and negative accuracies. Vertical bars show the variance of the accuracies.

3 (S1)). From the randomly chosen 60 positive indexed data, we obtained a hypothesis with accuracy $(83.3\%, 64.9\%)$ within $\mathcal{RP}^k_{\varepsilon,m}$ for $k = 5, m = 5$ in maximal covering approach (Table 3 (S2)).

5.4 Comparison among three heuristics

On hydropathy indexed transmembrane domain data, we run experiments to compare three heuristics introduced in Section 4. The hypothesis space is $\mathcal{RP}^k_{\varepsilon,m}$ for $k = 5, m = 5$. We summarize the results in Figure 4 above. From Table 3 and Figure 4, we observed the following advantages of the other two approaches compared to randomized approach:

- Maximal covering approach: This approach rarely generates exceptions, because this approach does not divide patterns covering a few positive examples. On the other hand, the randomized approach will generate exceptions by dividing already refined patterns.
- Using-negative approach: This approach produces stable solutions in accuracy because any pattern with high negative error will be divided and refined earlier. Therefore, the variance of accuracies obtained by this approach is smaller than those by others.

Nevertheless, we could not observe any significant differences of the highest and the average accuracies among three heuristics in the experiences.

6 Conclusion

In this paper, we implemented a polynomial time learning algorithm for unions of extended regular pattern languages from positive data, and showed that our method for learning protein motifs from positive data can efficiently work on computers in practice.

In the experiments of this paper, the results heavily depend on the selection of parameters k and $m \geq 1$ of the hypothesis space $\mathcal{RP}^k_{\varepsilon,m}$ because too large values cause overfitting as well as too small values cause overgeneralization to the examples. Thus, automatic selection of such a hypothesis space is a future problem.

Acknowledgements

We thank to the anonymous referees for their constructive criticism and careful suggestions which greatly improve the correctness and the readability of this paper. This work is partially supported by Grant-in-Aid for Scientific Research on Priority Areas, “Genome informatics” from the Ministry of Education, Science and Culture, Japan.

References

[1] D. Angluin. Finding patterns common to a set of strings. In *Proc. the 11th Annual Symposium on Theory of Computing*, pp. 130–141, 1979.

[2] S. Arikawa, S. Kuhara, S. Miyano, A. Shinohara, and T. Shinohara. A learning algorithm for elementary formal systems and its experiments on identification of transmembrane domains. In *Proc. the 25th Hawaii International Conference on System Sciences*, pp. 675–684, 1992.

[3] S. Arikawa, S. Miyano, A. Shinohara, S. Kuhara, Y. Mukouchi, and T. Shinohara. A machine discovery from amino acid sequences by decision trees over regular patterns. *New Generation Computing*, 11, pp. 361–375, 1993.

[4] S. Arikawa, T. Shinohara, and A. Yamamoto. Learning elementary formal systems. *Theoretical Computer Science*, 95, pp. 97–113, 1992.

[5] H. Arimura, T. Shinohara, S. Otsuki. Finding minimal generalizations for unions of pattern languages and its application to inductive inference from positive data. In *Proc. the 11th STACS*, LNCS 775, Springer-Verlag, pp. 649–660, 1994.

[6] R. Fujino. Learning unions of extended regular pattern languages from positive data and its application to discovering motifs in proteins. Master thesis, *Kyushu Inst. of Tech.*, 1994.

[7] GenBank. GenBank Release Notes. IntelliGenetics Inc., 1991.

[8] J. Kyte, R.F. Doolittle. In *J. Mol. Biol.*, 157, pp. 105-132, 1982.

[9] R. D. King, A. Srinivasan, S. Muggleton, C. Feng, R. A. Lewis, and M. J. E. Sternberg. Drug design using inductive logic programming. In *Proc. the 26th Hawaii International Conference on System Sciences*, pp. 646–655, 1993.

[10] PIR. Protein identification resource. National Biomedical Research Foundation, 1991.

[11] T. Shinohara. Polynomial time inference of pattern languages and its applications. In *Proc. the 7th IBM Symposium on Mathematical Foundations of Computer Science*, pp. 191–209, 1982.

[12] T. Shinohara. Polynomial time inference of extended regular pattern languages. In *RIMS Symposia on Software Science and Engineering*, LNCS 147, pp. 115–127, Springer-Verlag, 1982.