Learning Relational Patterns

Michael Geilke[1] and Sandra Zilles[2]

[1]Technische Universität Kaiserslautern

[2]University of Regina

October 05, 2011

Motivation (1)

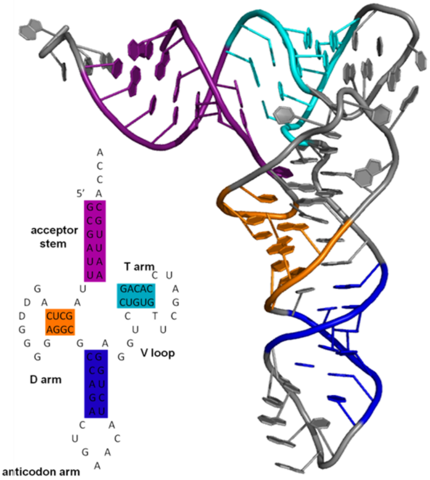
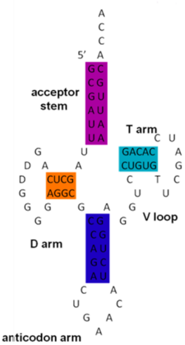


Figure: Source Wikipedia

Motivation (2)



RNA Sequence

GGGGAGGCGCCAGACUGAACAUCUG ...

UCGAUCCACAGAAUUGCACCAGCGA ...

AAGCAGGUUCCAGACUGCCCACCUG ...

GUUCUAAGGUCCAGACUUGGAUAUG ...

CCAGACUGAACAUCUGGACUCGAUU ...

⋮

Common pattern? $\rightsquigarrow$ formal language

Pattern Languages

- $\Sigma = \{a, b, ...\}$ be a finite set of **terminal symbols** with $|\Sigma| \geq 2$
- $X = \{x_1, x_2, ...\}$ be a countable set of **variables** such that $\Sigma \cap X = \emptyset$

Informal definition (Angluin)

A **pattern** *is any finite string over terminal symbols and variables. The* **language of a pattern** *p is the set of all words that result from substituting all variables in p by strings of terminal symbols.*

Example: $\Sigma = \{a, b, c\}$

$$p \quad = \quad (ab)^3 \quad x_1 \quad x_2 \quad b^2 c^4 \quad x_3 \quad b^3$$

Pattern Languages

- $\Sigma = \{a, b, ...\}$ be a finite set of **terminal symbols** with $|\Sigma| \geq 2$
- $X = \{x_1, x_2, ...\}$ be a countable set of **variables** such that $\Sigma \cap X = \emptyset$

Informal definition (Angluin)

A **pattern** *is any finite string over terminal symbols and variables.*
The **language of a pattern** *p is the set of all words that result from substituting all variables in p by strings of terminal symbols.*

Example: $\Sigma = \{a, b, c\}$

$$p \quad = \quad (ab)^3 \quad \mathbf{x_1} \quad \mathbf{x_2} \quad b^2 c^4 \quad \mathbf{x_3} \quad b^3$$

Pattern Languages

- $\Sigma = \{a, b, ...\}$ be a finite set of **terminal symbols** with $|\Sigma| \geq 2$
- $X = \{x_1, x_2, ...\}$ be a countable set of **variables** such that $\Sigma \cap X = \emptyset$

Informal definition (Angluin)

A **pattern** *is any finite string over terminal symbols and variables. The* **language of a pattern** *p is the set of all words that result from substituting all variables in p by strings of terminal symbols.*

Example: $\Sigma = \{a, b, c\}$

$$\theta(p) = \quad (ab)^3 \quad a^4 \quad ba \quad b^2c^4 \quad c^3 \quad b^3$$

Variants of Pattern Languages

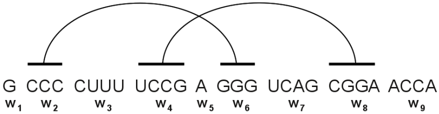
Bibliographic data entry system:

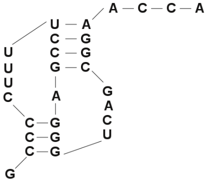
Author: x_1, Title: x_2, Year: x_3

- **erasing pattern languages:** variables can be substituted by ϵ
- **typed pattern languages:** each variable has exactly one type

$Y := \{t_1, t_2\}$, $X_{t_1} := \{x_1, x_2\}$, $X_{t_2} := \{x_3\}$
$L_{t_1} = \Sigma^+$,
$L_{t_2} = \{1900, \ldots, 2100\} \cup \{\epsilon\}$

Relational Pattern Languages (1)





$$x_1\ x_2\ x_3\ x_4\ x_5\ x_6\ x_7\ x_8\ x_9$$

If x_4 is substituted by $UCCG$, then x_8 has to be substituted by an element from $\{UGGA, CGGA, UGGU, CGGU\}$.

Relational Pattern Languages (2)

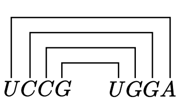
Solution: Introduce relations between variables into the pattern.

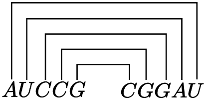
$$x_{(t,1)}\ x_{(t,2)}\ x_{(t,3)}\ x_{(t,4)}\ x_{(t,5)}\ x_{(t,6)}\ x_{(t,7)}\ x_{(t,8)}\ x_{(t,9)}$$

r

r

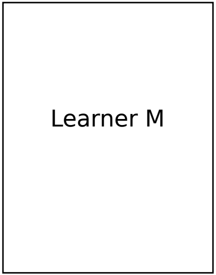
$A \quad - \quad U$
$C \quad - \quad G$
$G \quad - \quad U$





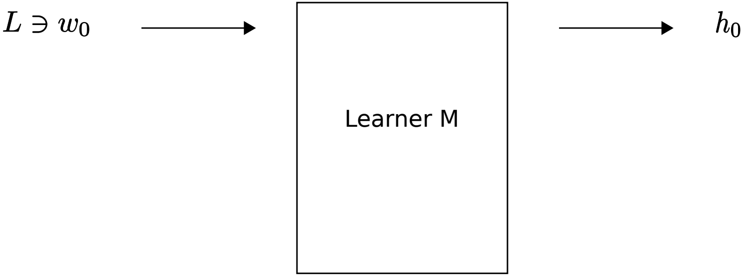
Learning from positive data

Let $\mathcal{L}$ be a set of languages, $L \in \mathcal{L}$ be the target language.



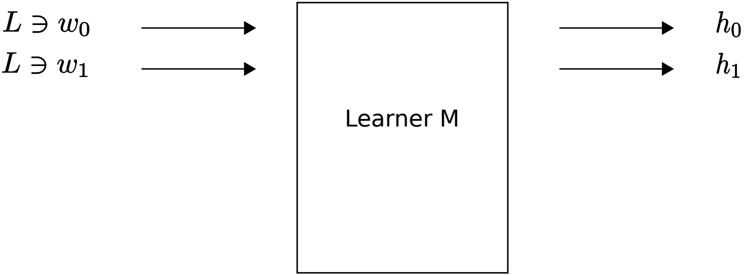
Learning from positive data

Let $\mathcal{L}$ be a set of languages, $L \in \mathcal{L}$ be the target language.



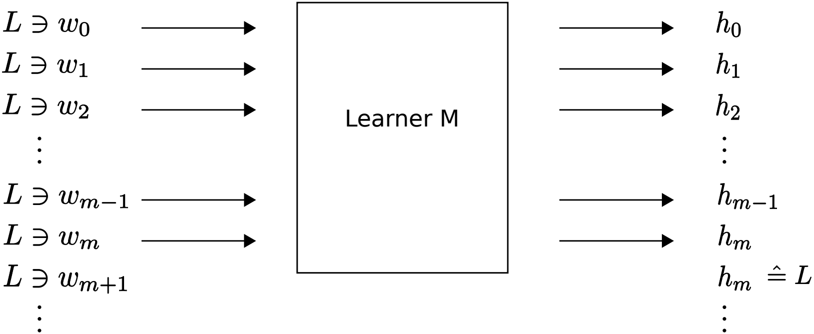
Learning from positive data

Let $\mathcal{L}$ be a set of languages, $L \in \mathcal{L}$ be the target language.



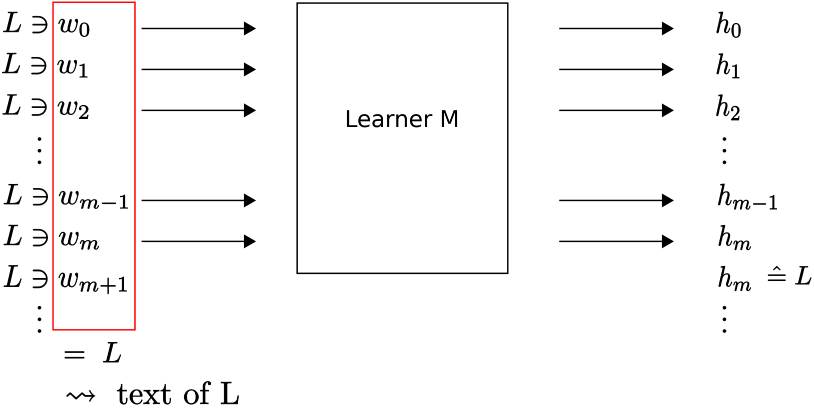
Learning from positive data

Let $\mathcal{L}$ be a set of languages, $L \in \mathcal{L}$ be the target language.



Learning from positive data

Let $\mathcal{L}$ be a set of languages, $L \in \mathcal{L}$ be the target language.



Learnability of Relational Pattern Languages

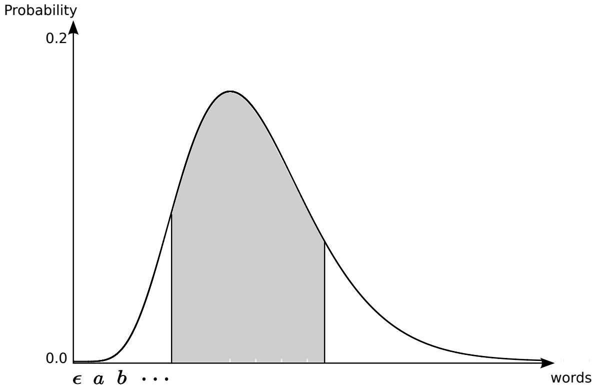
Assumption: There is a finite number of relations, all of which are decidable.

Non-erasing case:
The class of all relational pattern languages is learnable.

Erasing case:
There are classes of relational pattern languages that are not learnable.

Learning from Special Texts (1)

Not all texts occur with equal likelihood.

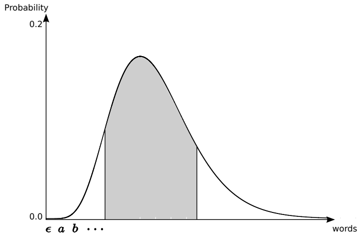


Author:	*Angluin*,	Title:	*Finding patterns* . . . ,	Year:	1980
Author:	*AAAAA*,	Title:	*ABC*,	Year:	2100

Learning from Special Texts (2)

New learning model:

- probability distributions on types
- texts can be generated by drawing substitutions according to the types' probability distributions
- learner only has to learn with a given confidence level



Positive result for relational pattern languages **with bounded arity**.

Membership Problem

The *membership problem* for a class of relational pattern languages is defined as

Given:	pattern p, word w
Question:	does p generate w?

Theorem

Let R be a finite set of recursive relations. Then the membership problem for a pattern p and word w is NP-hard.

But **efficient algorithms** for

- subclasses of patterns and subsets of words (e.g. words of restricted length)
- probabilistic algorithm for membership test for probabilistic relational patterns

Appendix

Relational Pattern Languages (3)

$L_1 := \{a^n b^n \mid n \geq 1\}$

Proposition

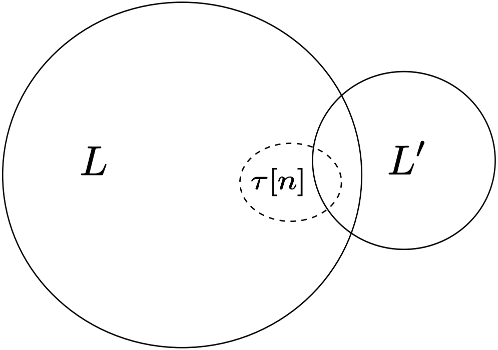
Let (p, T) be a typed pattern. If $L(p, T) = L_1$ then there is some $x \in X$ occurring in p such that $T(x)$ is not a regular language.

Proposition

There is a finite set R of relations such that R_1 contains only regular languages and $L_1 \in \mathcal{L}_{\Sigma,R}$.

Membership Problem (1)

In applications, the number of examples might be too low to identify the target language.



Therefore, make "reasonable" requirements on all hypotheses ever returned, e.g., consistency with observed data.

Membership Problem (2)

Testing for consistency:

The *membership problem* for a class of relational pattern languages is defined as

Given:	pattern p, word w
Question:	does p generate w?

Theorem

Let R be a finite set of recursive relations. Then the membership problem for a pattern p and word w is NP-hard.

Membership Problem (3)

Let $k, m \in \mathbb{N}$ be fixed and let all relations be decidable in polynomial-time.

Special case: efficient algorithms for

- p with at most k distinct variables, w arbitrary
- p with (number of variables that occur several times $\leq k$), w arbitrary
- p arbitrary, w with $|w| \leq m$

General case:

- probabilistic algorithm for membership test for probabilistic relational patterns