An experimental comparison of active learning algorithms on fixed data sets

Du Hyun Cho - 362801 - du.hyun.cho@rwth-aachen.de

March 2025

1 Introduction

In this paper we compare the performance of active automata learning algorithm. For this goal we use specific data sets when we construct automaton. Since regular languages or for example finite automaton are quite complex to be (passive) learned through finite samples, we use active learning instead of passive learning in this paper [Loeding, 2018].

2 Preliminaries

2.1 Basic Notation

In this section we define basic concepts which are used throughout the thesis.

- I^* is the set of all words, e.g. $aaa, abb, bbca, \ldots \in I^*$,
- A subset $L \subseteq I^*$ of words is called a (formal) language.
- usw.. other notations could be added while i write in chapter 3,4
-

2.2 Learning

: We use for automaton learning L^* algorithm. It is an active automaton learning algorithm [Angluin, 1987]. Automaton learning is used for, for example construction of a model from an implementation (e.g. implementation of protocol such like TCP). We need for the learning of DFA "membership query" and "equivalence query" [Aarts et al., 2014]. For these two queries we need to know what a learner and teacher are. In this subsection we describe about who they are and what is their role for learning automaton.
Assume, L is an unknown language and learner wants to construct automaton for L. So what is the role of learner? The learner is given a finite sample of a language L: examples of words in L and examples of words not in L. For example, the learner can query the language L. First, learner can ask "Is $w \in L$"? for words w that the learner can choose. We call this a "Membership query". And

then, for a DFA A, the learner can ask if whether $L = L(A)$. We call this a "Equivalence query". This A is often called a hypothesis.
So what is about a teacher? Above we see membership query and equivalence query. Now we consider a model that is referred to as "minimally adequate teacher" (MAT) model, with the following two types of queries [Bollig et al., 2009]. Assume that w (The Learner can choose) is a word as input and there is a membership query "Is the input $w \in L$?" Then the teacher should answer this question with "yes" or "no". And there is equivalence query "Does an Automaton accept L?" that the learner can choose. Then the teacher can answer with "yes" or "no". The answer "no" means that A is not correct on w for a word w that the teacher can choose. We call this w "counter example". A counterexample is a string in the symmetric difference of the correct language and the guessed language.

2.3 L* Learning

The learning algorithm L^* is described as correctly learning the regular set from the minimally adequate Teacher in polynomial time in the minimum number of DFA states for the set and the longest length of any counter example provided by the teacher [Bollig et al., 2009, Vaandrager, 2017]. Now we have a look the algorithm L^*.

Algorithm 1 The pseudo-code for L^* is given in Algorithm 1 [Angluin, 1987].

initialization: an observation table $B = (R, E, f)$ with $R{:=}\{\epsilon\}$ and $E{:=}\{\epsilon\}$
repeat
    while (R, E) is not closed or not consistent **do**
        if (R, E) is not closed **then**
            find $r_1 \in R$, $a \in \Sigma^*$ s.t. $row(r_1 a) \neq row(r)$, for all $r \in R$
            $R{:=}R \cup \{r_1 a\}$
        end if
        if (R, E) is not consistent **then**
            find $r_1, r_2 \in R, a \in A$, and $e \in E$ such that $row(r_1) = row(r_1)$ and $L(r_1 ae) \neq L(r_2 ae)$
            $E{:=}E \cup \{ae\}$
        end if
        Make the conjecture $M(R, E)$
        if The teacher answers with no, with a counter-example c **then**
            $R{:=}R \cup prefixes(c)$
        end if
        Until the teacher answers with yes to the conjecture $M(R, E)$.
return $M(R, E)$

To understand this algorithm, we need to know what exactly are observation tables, escaping words and closed observation tables. First, an observation table $B = (R, E, f)$ consists of

- a set $R \subseteq \Sigma^*$ (for the representatives),
- a set $E \subseteq \Sigma^*$ (for the experiments),
- and a function $f\colon (R \cup R \cdot \Sigma) \times E \longrightarrow O$ [Kasprzik, 2010].

To understand these symbols, here is an example.

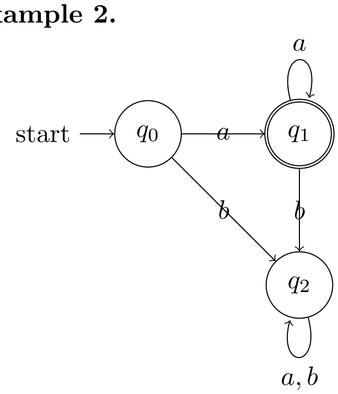
Example 1.

	ϵ	a
ϵ	0	1
a	1	1
ab	0	0
b	0	0
bb	0	0

In the first column ϵ, a, ab are the set R and the second and third column ϵ and a are the set E. And we can fill in 0 or 1 by a given automaton. 0 means "reject" and 1 means "accept". If $re \in L$, then $f(r, e) = 1$. Otherwise, $f(r, e) = 0$ [Kasprzik, 2010]. For this example, let be $L := \{w \mid w$ contains only the letter a$\}$. Then choose $r = \epsilon \in R$ and $e = a \in E$. $re = \epsilon a = a \in L$. So $f(r, e) = 1$. And we choose another r and e. If $r = bb$ and $e = a$, then $re = bba \notin L$, because w contains the letter b. In this way, we can interpret the observation table.

Automaton for this observation table is as following:

Example 2.



As above automaton, we can find the accepted words: for example, a, aa, aaa, ... The first row and the first column is ϵ. Since ϵ is not accepted by this automaton, there should be 0. And the first row, the second column is ϵa. The state q_0 can reach the state q_1 with the word a. So this word a is accepted by the automaton, therefore there should be 1. In this way, we can make an observation table. Thus, B is an observation table for L if $f(w) = 1 \Leftrightarrow w \in L$.

And when is a word escaping? For this we should know first about reduced observation table.

Definition 2.1 (reduced observation table)**.** An observation table $B = (R, E, f)$ is called *reduced* if $\forall\ u, v \in R$ with $u \neq v$ there is $w \in E$ with $f(uw) \neq f(vw)$.

For example, here is an observation table.

Example 3.

	ϵ	a
ϵ	1	0
a	0	1
b	0	0
aa	1	0
ab	0	0
ba	0	0
bb	1	0

Assume the word w is a and $u = a$, $v = b$. Then uw is aa and vw is ba. Have a look at $f(aa)$ and $f(ba)$. $f(aa)$ is 1 and $f(ba)$ is 0. But we have to check for every u and v as well. That means, we should check for $u = \epsilon$, $v = a$ and $u = \epsilon$, $v = b$. $f(\epsilon)$ is 1 and $f(b)$ is 0, so the condition is satisfied. And $f()$ is 1 and $f(a)$ is 0, satisfied as well. Since this condition is satisfied, we can call this table as reduced. Now we go back to definition of escaping word.

Definition 2.2 (escaping word)**.** Assume an observation table B for L is reduced. A word $u \in \Sigma^*$ is called as *escaping* if $\forall\ v \in R$ there is $w \in E$ with $uw \in L \Leftrightarrow vw \notin L$.

Example 4.

	ϵ	b
ϵ	1	0
a	0	0
b	0	1
aa	1	0
ab	0	0

Assume a given language is $L \subseteq \{a, b\}^*$ with $w \in L$ iff the number of a in w is even and the number of b in w is even. For example, $aa, bb, aabb, bbaa \in L$. Here we choose the word $u = b$. The word v is all representatives ϵ and a, and w is the experiment word in column ϵ or b. Now we check if whether b is escaping or not. Let be first $v = a$ and $w = b$. Then $uw = bb \in L$ but $vw = ab \notin L$. But we have to check one more for case $v = \epsilon$. Then $uw = bb \in L$ but $vw = b \notin L$. (Because the number of a and b should be even.) So the condition for escaping is satisfied, then the word b is escaping. As above L^* algorithm, if a word escaping, we can make a new observation table [Bollig et al., 2009, Vaandrager, 2017].

Example 5.

	ϵ	b
ϵ	1	0
a	0	0
b	0	1
aa	1	0
ab	0	0
ba	0	0
bb	1	0

Since the word b is escaping word, we add this word b in representative column.

Definition 2.3 (Closed table)**.** If there are no words escaping from R, then the table is *closed*.

Example 6.

	ϵ	a	b
ϵ	1	0	0
a	0	1	0
b	0	0	1
ab	0	0	0
aa	1	0	0
ba	0	0	0
bb	1	0	0
aba	0	0	1
abb	0	1	0

If we check for all representatives, there is not any escaping word. Now we can call this table as *closed table*.

3 Experiment

with learnlib implement using various dataset simulation

4 Analyse

compare outputs and performance in various perspective.

5 Conclusion

Describe limitation..

References

[Aarts et al., 2014] Aarts, F., Kuppens, H., Tretmans, J., Vaandrager, F., and Verwer, S. (2014). Improving active mealy machine learning for protocol conformance testing. *Machine learning*, 96(1-2):189–224.

[Angluin, 1987] Angluin, D. (1987). Learning regular sets from queries and counterexamples. *Information and computation*, 75(2):87–106.

[Bollig et al., 2009] Bollig, B., Habermehl, P., Kern, C., and Leucker, M. (2009). Angluin-style learning of nfa. In *Twenty-First International Joint Conference on Artificial Intelligence*.

[Kasprzik, 2010] Kasprzik, A. (2010). Learning residual finite-state automata using observation tables. *arXiv preprint arXiv:1008.1663*.

[Loeding, 2018] Loeding, C. (2018). Mitschrift: Algorithmisches lernen. Accessed: 2018–05-07.

[Vaandrager, 2017] Vaandrager, F. (2017). Model learning. *Communications of the ACM*, 60(2):86–95.