Model Learning Overview

Outline

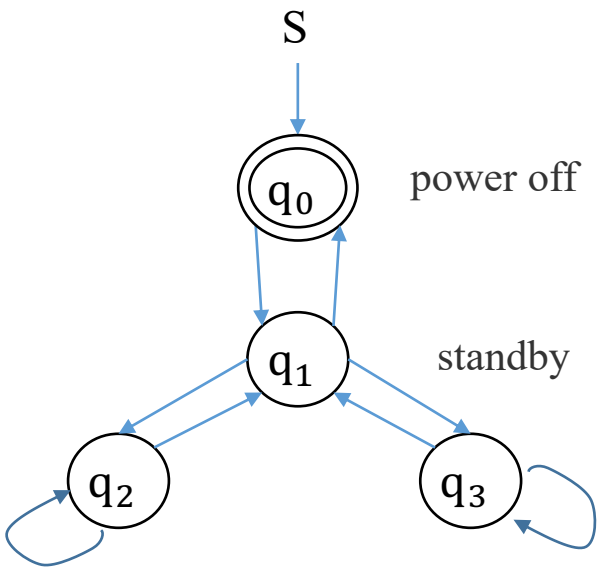
1. Introduction
2. Target Automata Type
3. Approach
4. Tools
5. Application
6. Challenge And Discussion

1. Introduction

- What ‘s model learning ?
- Why learning model ?
- How to learn model ?

Model Learning (Automata Learning)

- Model learning aims to **construct black-box state diagram** models of software and hardware systems **by providing inputs and observing outputs**.



Why learning model ?

- To understand the behavior of a component without having access to the code.
- Generate regression tests
- Complementary to model checking
- etc.

（We focus on *state diagrams*）

（We focus on *black-box technique*）

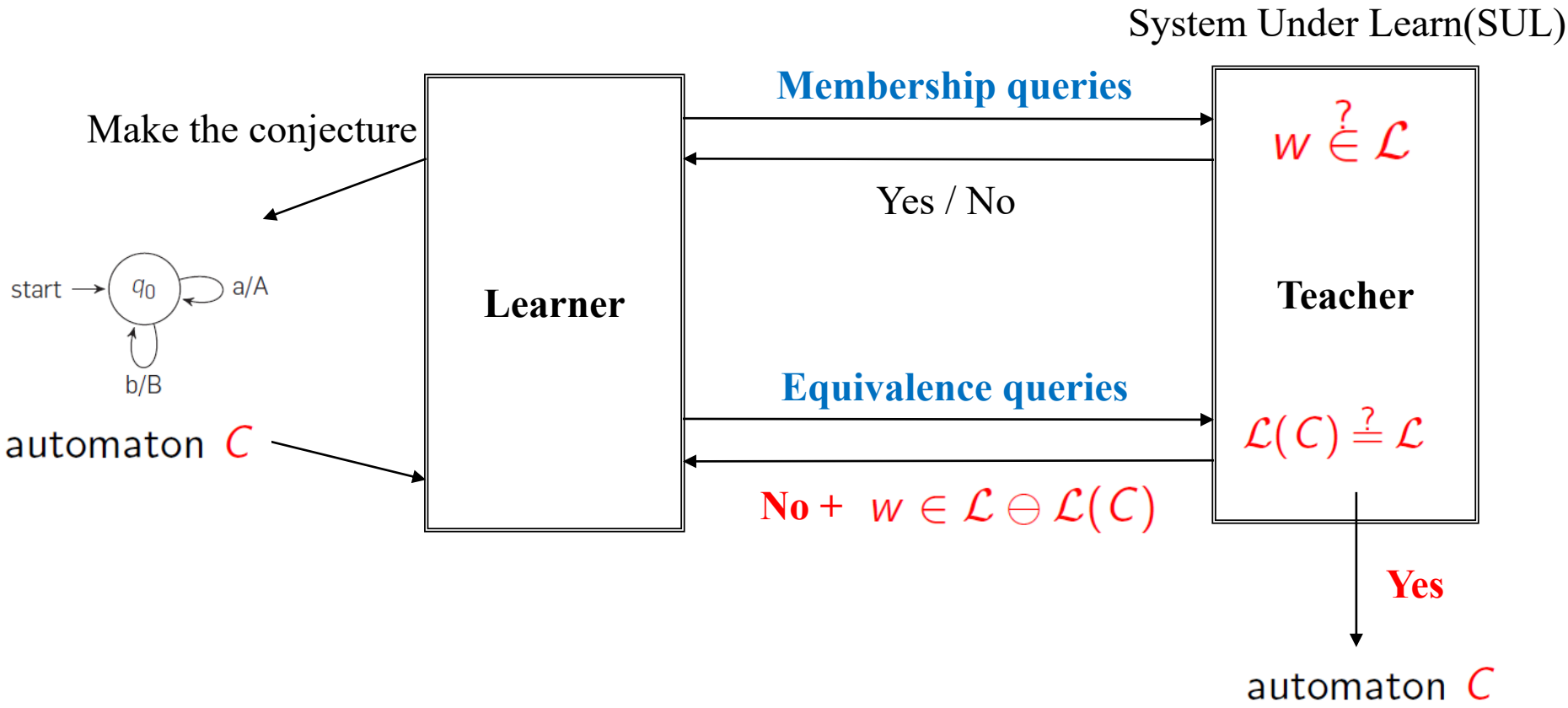
Model Checking

- Model Checker: given a mathematical model **M** and specification **φ**, automatically checks whether such specification holds for that model: $M \models \varphi$

Combining Model Learning

- Goal: to check a system satisfies a set of properties $\varphi_1, \ldots, \varphi_1$
- Learn **M** using **model learning**
- Check whether **M** satisfies all φ_i

Angluin-Style Exact Learning Framework



Model Learning Overview

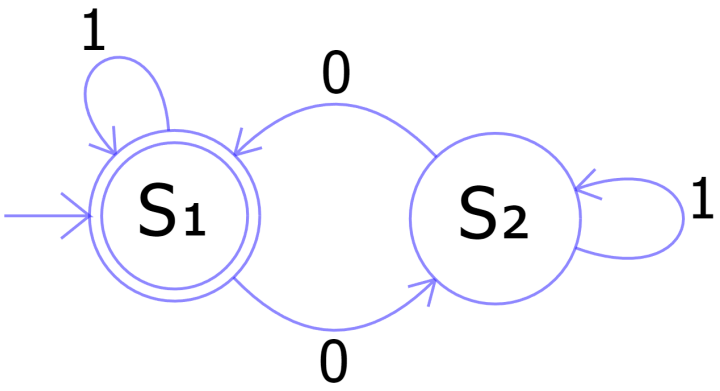
Outline

1. Introduction
2. **Target Automata Type**
3. Approach
4. Tools
5. Application
6. Challenge And Discussion

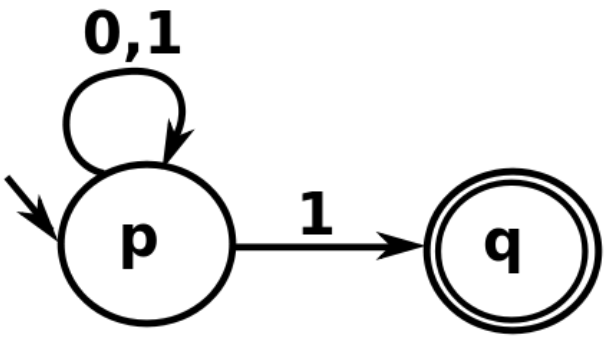
2. Target Automata Type

- Deterministic Finite Automata (DFAs)
- Nondeterministic Finite Automata (NFAs)
- Mealy/Moore Machine
- Register Automata (RAs)
- Büchi Automata (BAs)
- Nominal Automata
- Timed Automata
- Weighted Automata
- …

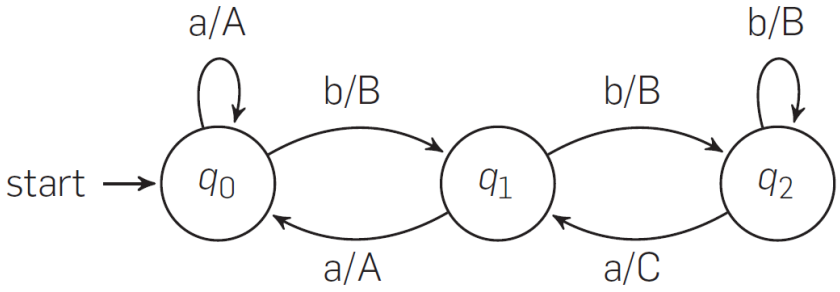
Examples of Different Automata Type



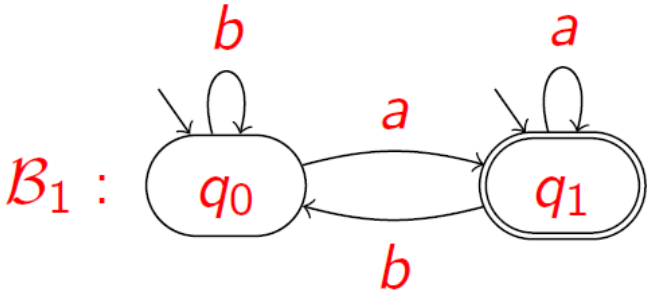
DFAs



NFAs



Mealy machine



$(ab)^\omega \in \mathcal{L}(\mathcal{B}_1)$

Model Learning Overview

Outline

1. Introduction
2. Target Automata Type
3. **Approach**
4. Tools
5. Application
6. Challenge And Discussion

Learning DFA

Goal: Given an unknown regular language L over an alphabet $\sum$, generating a DFA M that accepts L by queries.

DFA M is a tuple $(Q;\ \sum;\ I;\ \delta;\ F)$ where

- Q is a finite set of states
- $\sum$ is the set of alphabet
- $\mathbf{I \subseteq Q}$ is the set of initial states
- $\boldsymbol{\delta: Q \times \sum \rightarrow Q}$ is the transition relation
- $\mathbf{F \subseteq Q}$ is the set of accepting states

Right Congruence for DFA

For a DFA M, we define $x \sim_M y$ iff $\delta(q_0, x) = \delta(q_0, y)$

- The relation $\sim_M$ is an equivalence relation.
- Some states are irrelevant for the accepted language.
- $L(M)$ is the union of

Right Congruence for RE

For a language L, we define a relation $x \sim_L y$ such that for each $v \in \sum^*, xv \in L \leftrightarrow yv \in L$

- The relation $\sim_L$ is an equivalence relation.
- Some equivalence classes are irrelevant for L.
- L is the union of

Myhill-Nerode Theorem

The following statements are equivalent:

- L is a regular language on $\sum$
- there exists a right congruence relation over $\sum^*$ such that it has finitely many equivalent class, and L can be expressed as a union of some of the equivalences
- $\sim_L$ has finitely many equivalent classes

More over, for regular language, $|\sum^*/\sim_L|$ equals the number of states of the smallest DFA recognizing L.

Example

➢ the language consisting of binary representations of **numbers that can be divided by 3 is regular**.

Access String

For a given target (minimal) DFA M, we have :

- Access string: $M[x] := \delta(q_0, x)$
- we use the access string x to access the state $M[x]$
- in general, many access strings access the same state
- Distinguishing string: if $xv \notin L$ and $yv \in L$ or vice versa
- two access strings x, y access different states if such v exists

Approximation by Observation Table

- We maintain an observation table: $T: (S \cup S\sum) \rightarrow \{\mathsf{T}, \mathsf{F}\}^E$ where S is prefix closed
- T is closed and consistent
- **closed**: for $every\ w \in S\sum,\ there\ exist\ w' \in S,$ $row(w) = row(w')$.
- **consistent**: if $w_1, w_2 \in S, row(w_1) = row(w_2),$ $that\ row(w_1 \cdot \sum^*) = row(w_2 \cdot \sum^*)$.

Distinguishing string E

Access string S	OT_1	ε
	ε	T
$S\sum$	0	F
	1	F

T: $s \cdot e \in L$
F: $s \cdot e \notin L$

Angluin 's L* Algorithm

L* LEARNER

1 $S, E \leftarrow \{\epsilon\}$
2 **repeat**
3 **while** (S, E) is not closed or not consistent
4 **if** (S, E) is not closed
5 find $s_1 \in S$, $a \in A$ such that
$row(s_1 a) \neq row(s)$, for all $s \in S$
6 $S \leftarrow S \cup \{s_1 a\}$
7 **if** (S, E) is not consistent
8 find $s_1, s_2 \in S$, $a \in A$, and $e \in E$ such that
$row(s_1) = row(s_2)$ and $\mathcal{L}(s_1 ae) \neq \mathcal{L}(s_2 ae)$
9 $E \leftarrow E \cup \{ae\}$
10 Make the conjecture $M(S, E)$
11 **if** the Teacher replies **no**, with a counter-example t
12 $S \leftarrow S \cup \texttt{prefixes}(t)$
13 **until** the Teacher replies **yes** to the conjecture $M(S, E)$.
14 **return** $M(S, E)$

Figure 1. Angluin's algorithm for deterministic finite automata [3]

Example

Suppose the unknown regular set U is set of all strings over {0,1} with an even number of 0's and an even number of 1's.

OT_1	ε
ε	T
0	F
1	F

Initial observation table,
S = E = {ε}

The observation table is consistent, but not closed, since row(0) is distinct from row(ε)

OT_2	ε
ε	T
0	F
1	F
00	T
01	F

Augmented observation table,
S = {ε, 0}
E = {ε}

Example

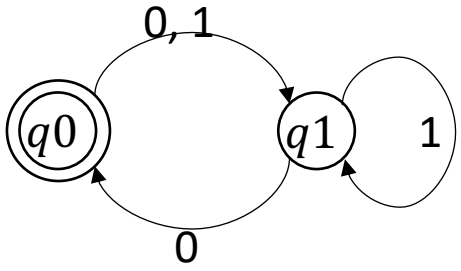
Suppose the unknown regular set U is set of all strings over {0,1} with an even number of 0's and an even number of 1's.

OT_2	ε
ε	T
0	F
1	F
00	T
01	F

Augmented observation table,
S = {ε, 0}
E = {ε}

The observation table is consistent and closed, so L* make a conjecture of the acceptor M_1

δ	0	1
$q0$	$q1$	$q1$
$q1$	$q0$	$q1$



M_1, the first conjecture of L*

Teacher said: No
Counterexample: 11
(it is in U but rejected by M_1)

Example

OT_3	ε
ε	T
0	F
1	F
11	T
00	T
01	F
10	F
110	F
111	F

Observation table,
S = {ε, 0, 1, 11}
E = {ε}

The observation table is closed, but not consistent, since row(0) = row(1) but row(00) ≠ row(10)

OT_4	ε	0
ε	T	F
0	F	T
1	F	F
11	T	F
00	T	F
01	F	F
10	F	F
110	F	T
111	F	F

Augmented observation table,
S = {ε, 0, 1, 11}
E = {ε, 0}

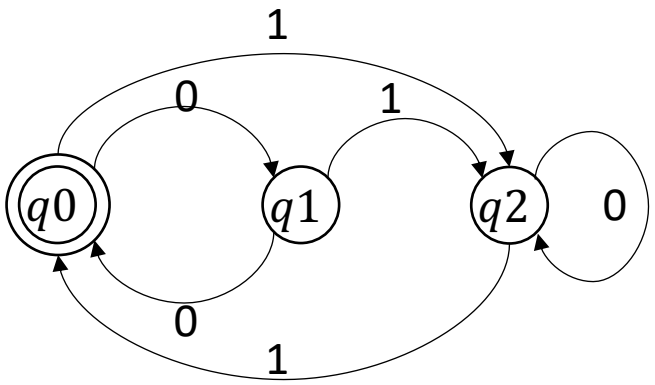
Example

OT_4	ε	0
ε	T	F
0	F	T
1	F	F
11	T	F
00	T	F
01	F	F
10	F	F
110	F	T
111	F	F

Observation table,
S = {ε, 0, 1, 11}
E = {ε, 0}

The observation table is consistent and closed, so L* make a conjecture of the acceptor M_2

δ	0	1
$q0$	$q1$	$q2$
$q1$	$q0$	$q2$
$q2$	$q2$	$q0$



M_2, the second conjecture of L*

Teacher said: No
Counterexample: 011
(it is not in U but accepted by M_2)

Example

OT_5	ε	0
ε	T	F
0	F	T
1	F	F
11	T	F
01	F	F
011	F	T
00	T	F
10	F	F
110	F	T
111	F	F
010	F	F
0110	T	F
0111	F	F

The observation table is closed, but not consistent, since row(1) = row(01) but row(11) ≠ row(011)

OT_6	ε	0	1
ε	T	F	F
0	F	T	F
1	F	F	T
11	T	F	F
01	F	F	F
011	F	T	F
00	T	F	F
10	F	F	F
110	F	T	F
111	F	F	T
010	F	F	T
0110	T	F	F
0111	F	F	F

$S = \{\varepsilon, 0, 1, 11, 01, 011\}$, $E = \{\varepsilon, 0\}$

$S = \{\varepsilon, 0, 1, 11, 01, 011\}$, $E = \{\varepsilon, 0, 1\}$

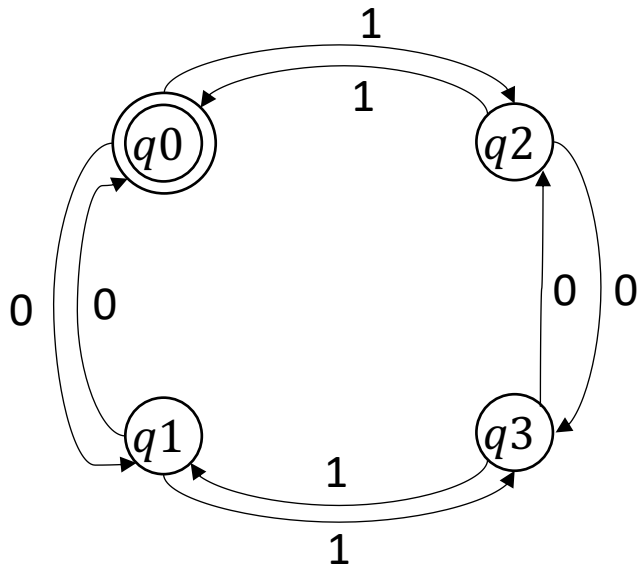
Example

OT_6	ε	0	1
ε	T	F	F
0	F	T	F
1	F	F	T
11	T	F	F
01	F	F	F
011	F	T	F
00	T	F	F
10	F	F	F
110	F	T	F
111	F	F	T
010	F	F	T
0110	T	F	F
0111	F	F	F

$S = \{\varepsilon, 0, 1, 11, 01, 011\}$, $E = \{\varepsilon, 0, 1\}$

The observation table is consistent and closed, so L* make a conjecture of the acceptor M_3

δ	0	1
$q0$	$q1$	$q2$
$q1$	$q0$	$q3$
$q2$	$q3$	$q0$
$q3$	$q2$	$q1$



M_3, the second conjecture of L*

Teacher said: Yes

Deterministic Finite Automata (DFAs)

	Algorithm	Publication
Angluins et al. 1987	Angluin's L*	Learning regular sets from queries and counterexamples
Rivest and Schapire 1993	R & S 's Algorithm	Inference of Finite Automata Using Homing Sequences
Kearns and Vazirani 1994	K & V 's Algorithm	An introduction to computational learning theory
Parekh et al. 1997	ID and IID	A polynomial time incremental algorithm for regular grammar inference
Denis et al. 2001	DeLeTe2	Learning regular languages using RFSAs
Bongard et al. 2005	Estimation-Exploration	Active Coevolutionary Learning of Deterministic Finite Automata
Isberner et al. 2014	The TTT Algorithm	The TTT Algorithm: A Redundancy-Free Approach to Active Automata Learning
Volpato et al. 2015	LearnLTS	Approximate Active Learning of Nondeterministic Input Output Transition Systems

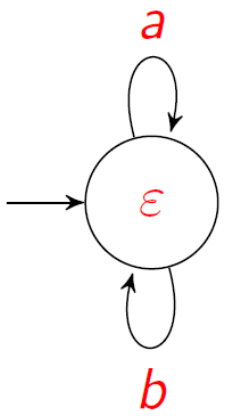
Example

Target language is

$L = \{ u \in \{a, b\}^+ \mid \text{the number of } b \text{ in } u \text{ is } 4n + 3 \}$

	ε
ε	**F**
a	**F**
b	**F**

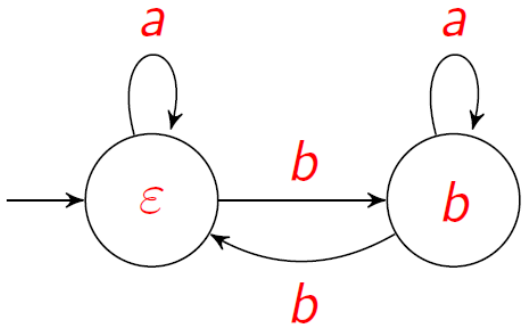
$\Rightarrow$



For a counterexample $bbab \in L$: we find a new experiment bab to distinguish ε and b

	ε	bab
ε	**F**	**F**
b	**F**	**T**
a	**F**	**F**
ba	**F**	**T**
bb	**F**	**F**

$\Rightarrow$

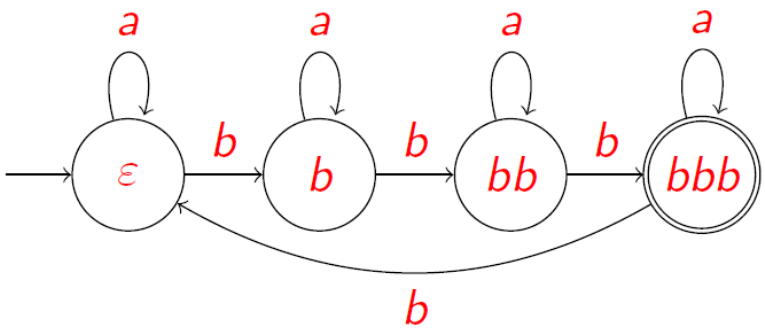


Example

We again receive $bbab$ as the counterexample and find ε and bb can be distinguished by ab

	ε	bab	ab
ε	**F**	**F**	**F**
b	**F**	**T**	**F**
bb	**F**	**F**	**T**
bbb	**T**	**F**	**F**
a	**F**	**F**	**F**
ba	**F**	**T**	**F**
bba	**F**	**F**	**T**
$bbba$	**T**	**F**	**F**
$bbbb$	**F**	**F**	**F**

$\Rightarrow$

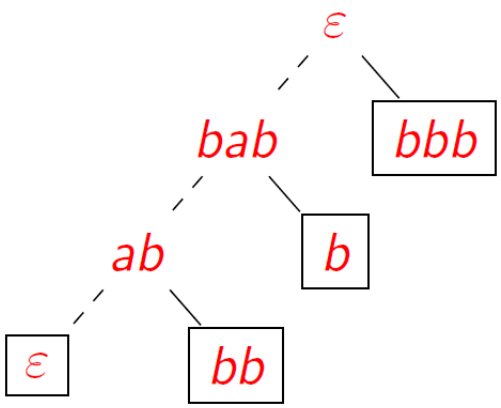


L* based on Classification Trees

Kearns & Vazirani'94

	ε	bab	ab
ε	**F**	**F**	**F**
b	**F**	**T**	**F**
bb	**F**	**F**	**T**
bbb	**T**	**F**	**F**
a	**F**	**F**	**F**
ba	**F**	**T**	**F**
bba	**F**	**F**	**T**
$bbba$	**T**	**F**	**F**
$bbbb$	**F**	**F**	**F**

$\Rightarrow$



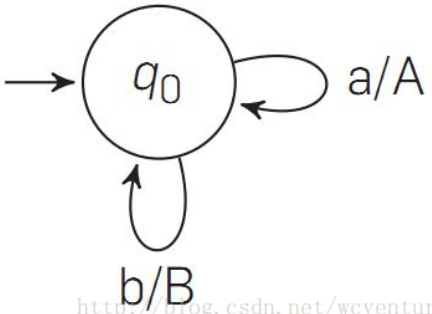
Moore Machine

	Algorithm	Publication
Georgios et al. 2016	MooreMI algorithm	Learning Moore Machines from Input-Output Traces
Moerman et al. 2017	Product L* Algorithm	Learning Product Automata

Mealy Machine

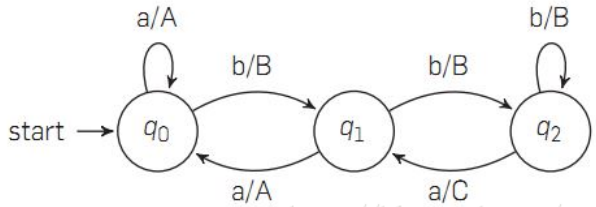
	Algorithm	Publication
Shahbaz et al. 2009	LM* and LM+ Algorithm	Inferring Mealy Machines
Aarts et al. 2010	IOA Algorithm	Learning I/O Automata
Steffen et al. 2011	DHC and LM* Algorithm	Introduction to Active Automata Learning from a Practical Perspective

$\mathcal{O}_1$	a	b
ϵ	A	B
a	A	B
b	A	B



$\mathcal{O}_2$	a	b
ϵ	A	B
b	A	B
bb	C	B
bba	A	B
a	A	B
ba	A	B
bbb	C	B
bbaa	A	B
bbab	C	B

$\mathcal{O}_3$	a	b	ba
ϵ	A	B	A
b	A	B	C
bb	C	B	C
bba	A	B	C
a	A	B	A
ba	A	B	A
bbb	C	B	C
bbaa	A	B	A
bbab	C	B	C



Nondeterministic Finite Automata (NFA)

	Algorithm	Publication
Oncina et al. 1992	RPNI Algorithm	Inferring Regular Languages in Polynomial Updated Time
Dupont et al. 1996	RPNI2 Algorithm	Incremental regular inference

Register Automata (RA)

	Algorithm	Publication
Howar et al. 2012	RAL Algorithm	Inferring Canonical Register Automata
Cassel et al. 2014	SL* Algorithm	Active learning for extended finite state machines
Aarts et al. 2015	A Mapper-Based Algorithm	Learning Register Automata with Fresh Value Generation

Büchi Automata

	Algorithm	Publication
Maler and Pnueli 1995	Lω Algorithm	On the learnability of infinitary regular sets
Farzan et al. 2008	L* based for Büchi automaton	Extending Automated Compositional Verification to the Full Class of Omega-Regular Languages
Angluin et al. 2014	FDFAs-based Algorithm	Learning Regular Omega Languages
Li et al. 2017	A Tree-based Algorithm for Büchi automaton	A Novel Learning Algorithm for Büchi Automata Based on Family of DFAs and Classification Trees

Other Automata Learning apporach

- **Event-Recording Automata**
  - Grinchtein et al. 2008. Learning of Event-Recording Automata.
- **deterministic finite cover automaton (DFCA)**
  - Ipate et al. 2012. Learning finite cover automata from queries.
- **Timed Automata**
  - Maier et al. 2014. Online passive learning of timed automata for cyber-physical production systems.
- **Weighted Automata**
  - Balle et al. 2015. Learning Weighted Automata.
- **Hybrid Automata**
  - Medhat et al. 2015. A framework for mining hybrid automata from input/output traces.
- **Visibly Pushdown Automata**
  - Isberner et al. 2015. Foundations of Active Automata Learning: An Algorithmic Perspective.
- **Symbolic Automata**
  - Drews et al. 2017. Learning Symbolic Automata.
- **Nominal Automata**
  - Moerman et al. 2017. Learning nominal automata.

Model Learning Overview

Outline

1. Introduction
2. Target Automata Type
3. Approach
4. **Tools**
5. Application
6. Challenge And Discussion

4. Tools

Open Source

- LearnLib https://learnlib.de/
- Libalf http://libalf.informatik.rwth-aachen.de/
- Tomte http://tomte.cs.ru.nl/
- ROLL http://iscasmc.ios.ac.cn/roll/doku.php
- Symbolicautomata https://github.com/lorisdanto/symbolicautomata

Not found

- Next Generation LearnLib (NGLL)
- RALT
- RALib

LearnLib

- LearnLib is a free and open source Java framework for automata learning.

Algorithm	Target model
Active learning algorithms	
ADT	Mealy
DHC	Mealy
Discrimination Tree	DFA、Mealy、VPDA
Kearns & Vazirani	DFA、Mealy
L*	DFA、Mealy
NL*	NFA
TTT	DFA、Mealy、VPDA
Passive learning algorithms	
RPNI	DFA、Mealy
RPNI – EDSM	DFA
RPNI – MDL	DFA

Libalf

Algorithm	Passive	Active	Target model
Angluin's L*		√	DFA
L* (adding counter-examples to columns)		√	DFA
Kearns / Vazirani		√	DFA
Rivest / Schapire		√	DFA
NL*		√	NFA
Regular positive negative inference (RPNI)	√		DFA
DeLeTe2	√		NFA
Biermann & Feldman's algorithm	√		NFA
Biermann & Feldman's algorithm (using SAT-solving)	√		DFA

Tomte

- This tool can only learn a restricted class of extended finite state machines (EFSM), i.e. Register Automata.

ROLL

- A Library of learning algorithms for ω-regular languages.
- It consists of two kinds of ω-regular learning algorithms:
  the learning algorithm for FDFAs
  the learning algorithm for Büchi automata

Symbolicautomata

- A symbolic automata library
- This efficient automata library allows you to represent large (or infinite) alphabets succinctly.

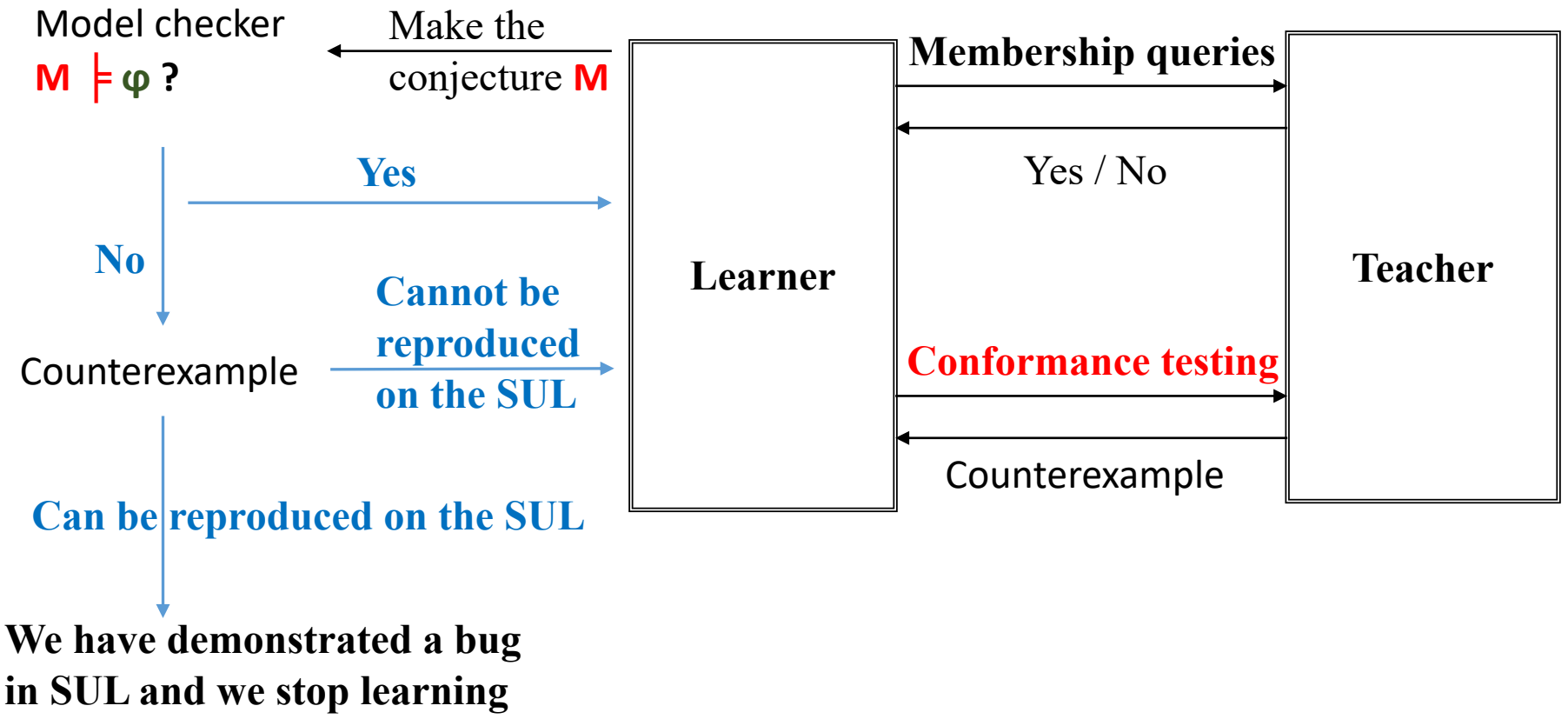
Model Learning Overview

Outline

1. Introduction
2. Target Automata Type
3. Approach
4. Tools
5. **Application**
6. Challenge And Discussion

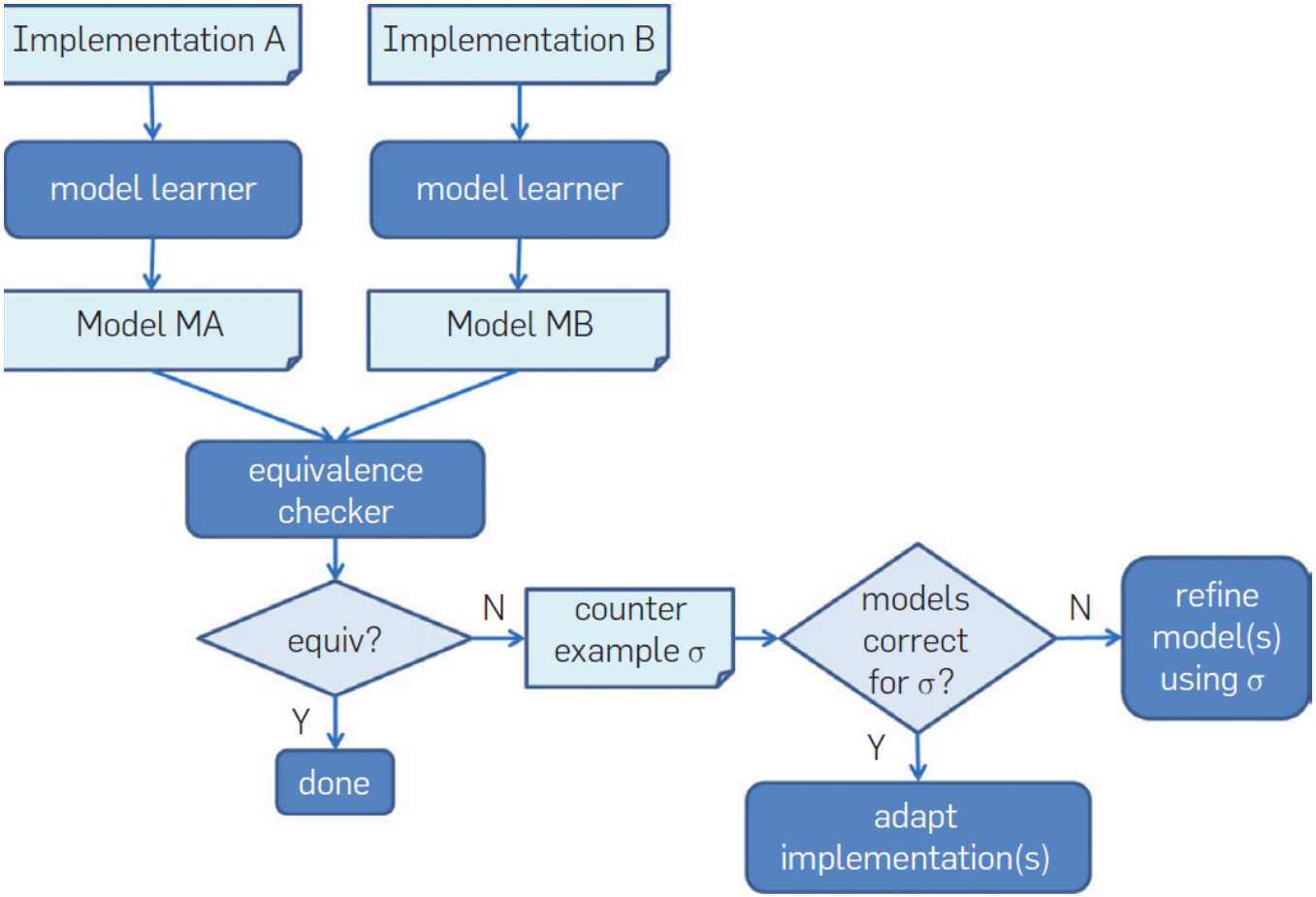
Related work	Approach	Tool	Application
Chalupar et al. 2014	L* for Mealy	LearnLib	SmartCard
Fiter et al. 2014	L* for Mealy	Tomte	TCP Network Protocol
Tijssen et al. 2014	L* for Mealy	LearnLib	SSH implementations
Shahbaz et al. 2014	LM+	RALT	testing black-box system
Xiao et al. 2014	L* (Lazy Alphabet)	TzuYu	Learn Stateful Typestates
Ruiter et al. 2015	L*	LearnLib	TLS Protocol state fuzzing
Smeenk et al. 2015	Lee & Yannakakis'	LearnLib	Embedded Control Software
Ipate et al. 2015	Ll		Testing
Meller et al. 2015	L*		Model Checking of UML Systems
He et al. 2015	L* based (MTBDD)		Verification of XXX
Fiter et al. 2016	L* for Mealy	LearnLib	TCP Implementations
Schuts et al. 2016	L* , TTT	LearnLib	Legacy Software
Chen et al. 2017	various L*	libalf	PAC verification & model synthesis
Weiss et al. 2017	L*		RNN
Aichernig et al. 2017	improved L* Mealy	LearnLib	Mutation Testing
Tappler et al. 2017	TTT	LearnLib	Testing IoT Communication

Model checking & Conformance testing



Legacy Software

Approach to compare Legacy component and refactored implementation.



Extracting Automata from RNN

Gail et al. use Angluin's L* algorithm to elicit an automaton from any type of recurrent neural network, using the network as the teacher.

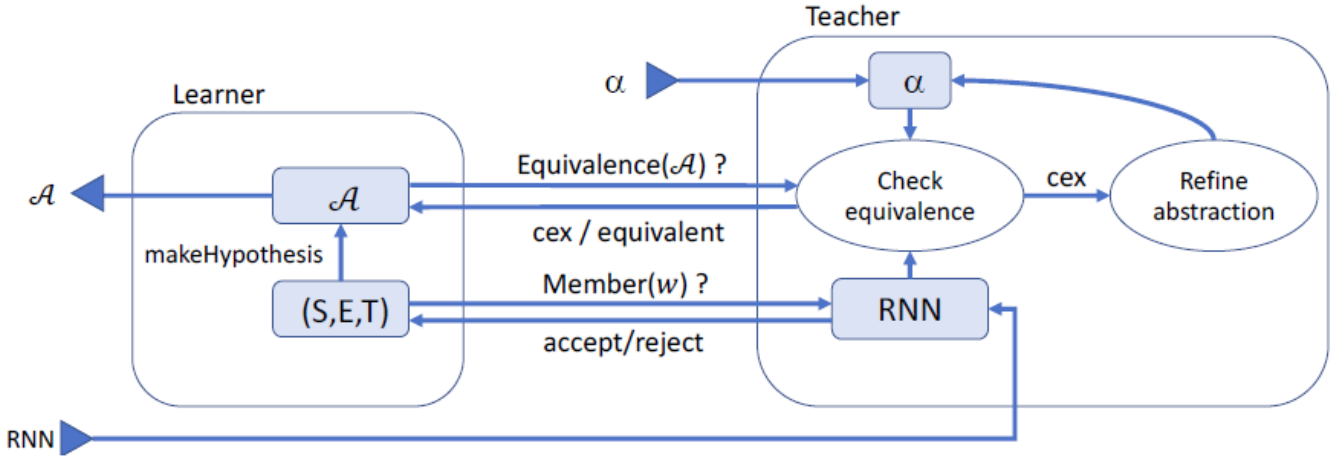


Fig. 4: An overview of our approach for learning with an RNN as a Teacher. The input is an RNN and an initial abstraction α, the output is a DFA $\mathcal{A}$ that maintains at least the observations made by the RNN under α.

Program Termination

Termination problem: we require that a terminating tool returns answers that are correct, but we donot necessarily require an answer.

- Trivial to build a tool: returns unknown simply
- Goal: keeping the unknown answers as low as possible
- Turing49: classical approach for proving termination
  - Termination argument search
  - Termination argument checking (easy)

Büchi automaton with rank Certificate (PLDI'18)

Model Learning Overview

Outline

1. Introduction
2. Target Automata Type
3. Approach
4. Tools
5. Application
6. **Challenge And Discussion**

Challenge And Discussion

- Equivalence query
- Beyond DFAs/Mealy machines
- Quality of models
- Predicates and operations on data
- Opening the box