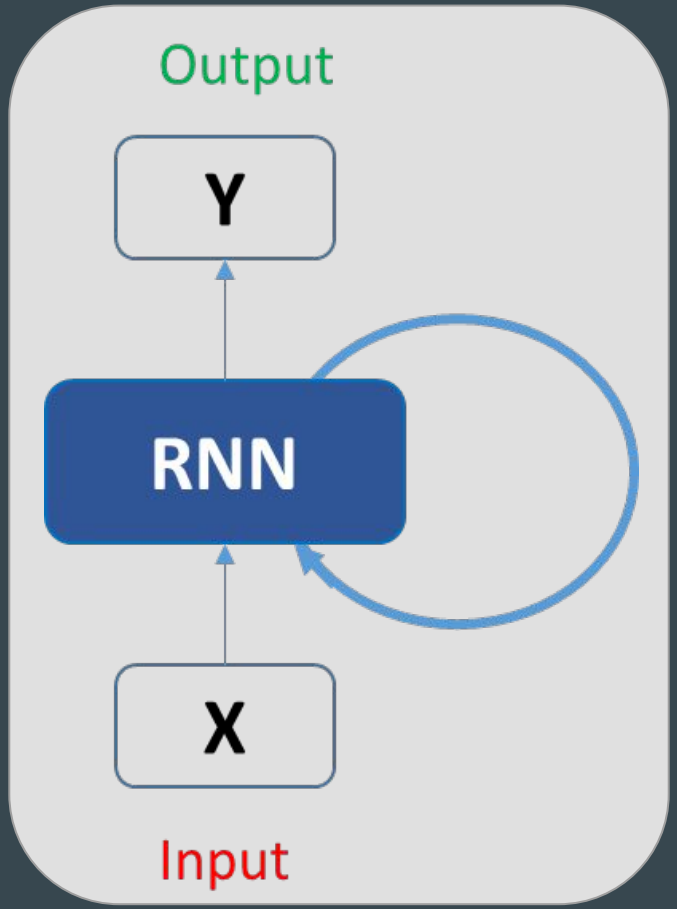
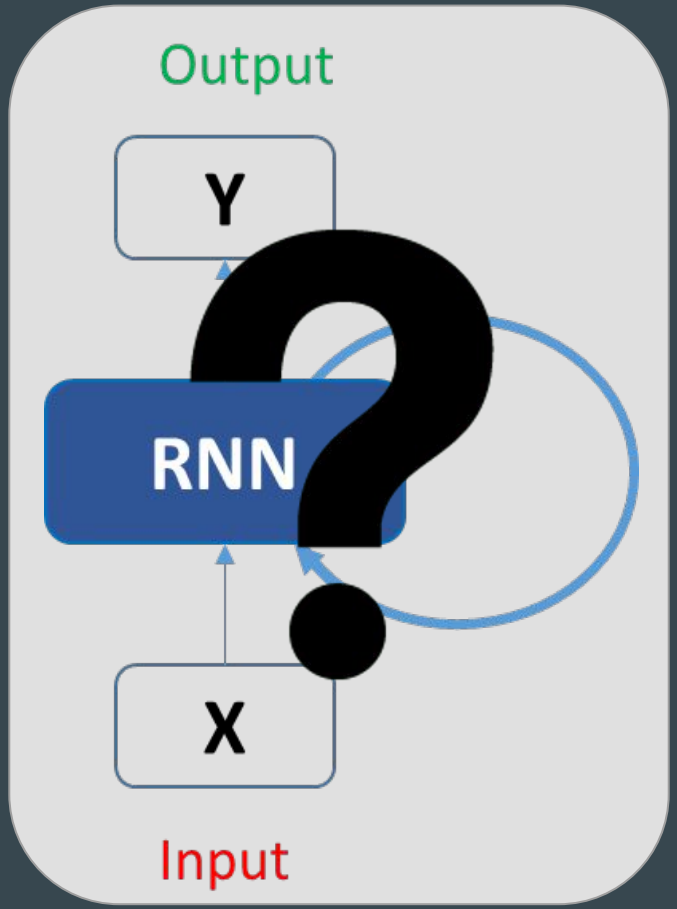
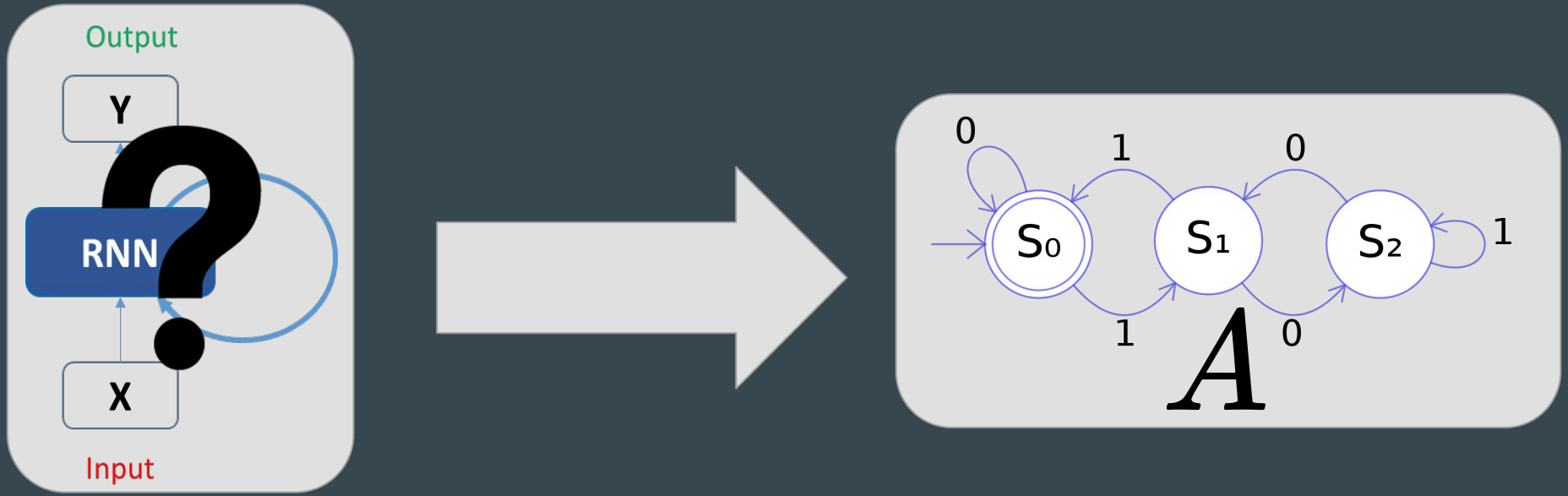
Extracting Automata from Recurrent Neural Networks Using Queries and Counterexamples

...

article de Gail Weiss, Yoav Goldberg et Eran Yahav (2017) présenté par Guillaume Ollier


Output
Y
RNN
X
Input


Output
Y
RNN
X
Input


Output
Y
RNN
X
Input
0
1
0
1
S0
S1
S2
1
1
0
A

$$w \in \Sigma?$$

L’algorithme L*


RNN
Oracle
L*
A

Oracle

- Les requêtes d'**appartenance**

- Les requête d'**équivalence**

Abstraction d'un Réseau

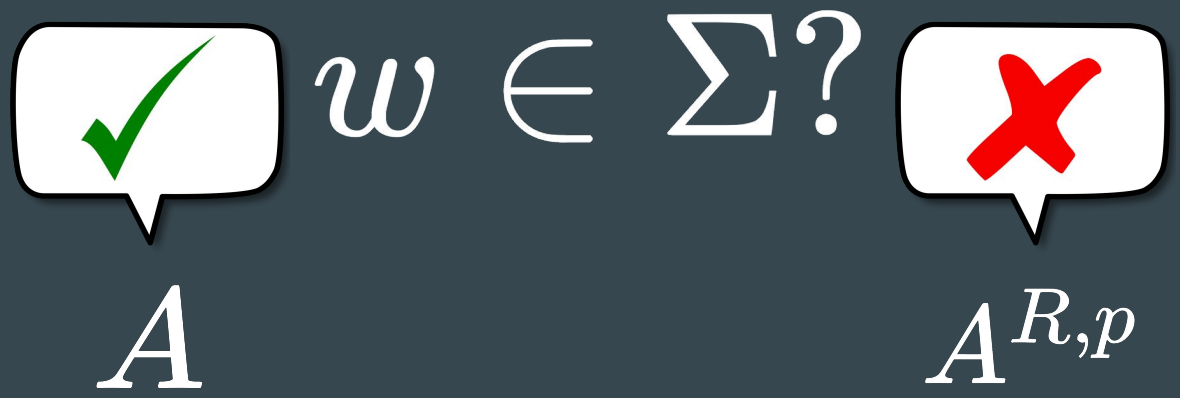
Le partitionnement du RNN

Soit S l'ensemble des états de notre RNN:

$$p : S \to \mathbb{N}$$

Equivalence Queries

Vue d'ensemble



Nous trouvons un exemple pour lequel les 2 automates sont en désaccord

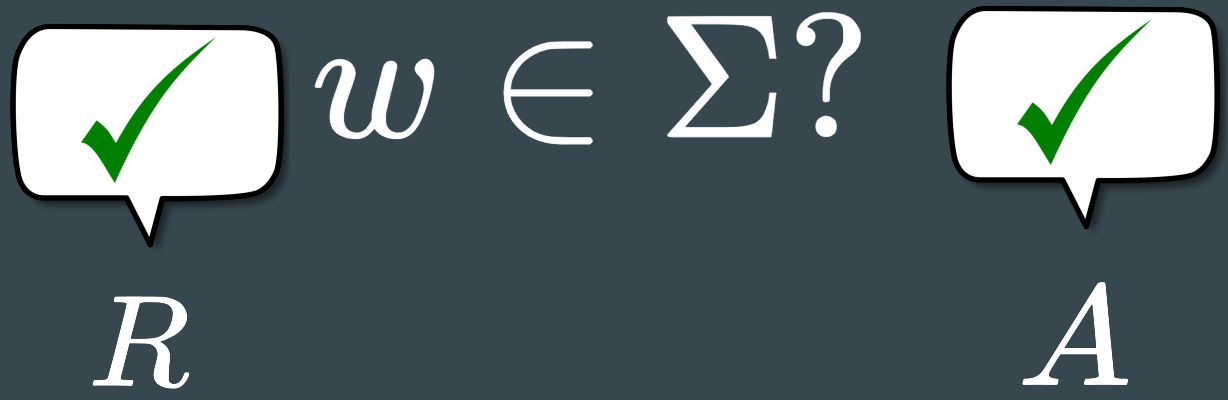
Vue d'ensemble

$w \in \Sigma?$

R

On récupere le vrai résultat dans le RNN

Vue d'ensemble

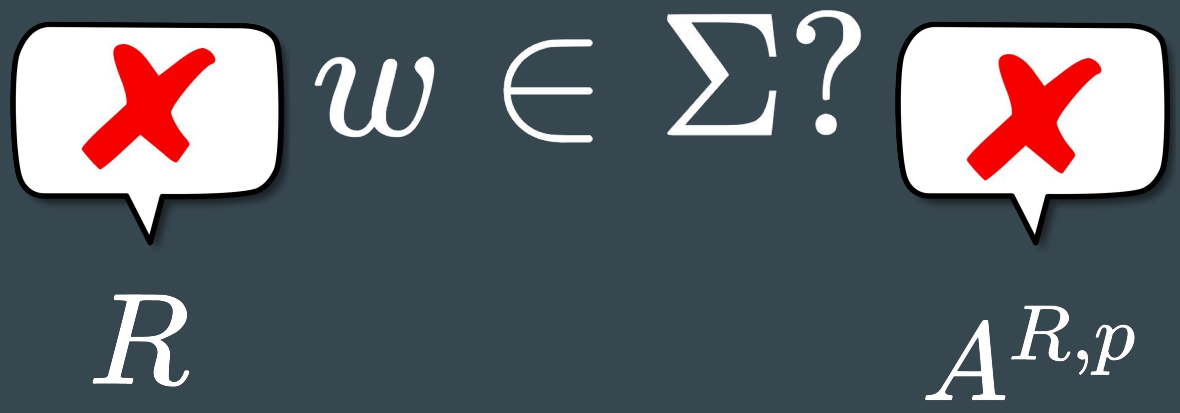


Soit le RNN donne le même résultat que le DFA extrait par L*...

Vue d'ensemble

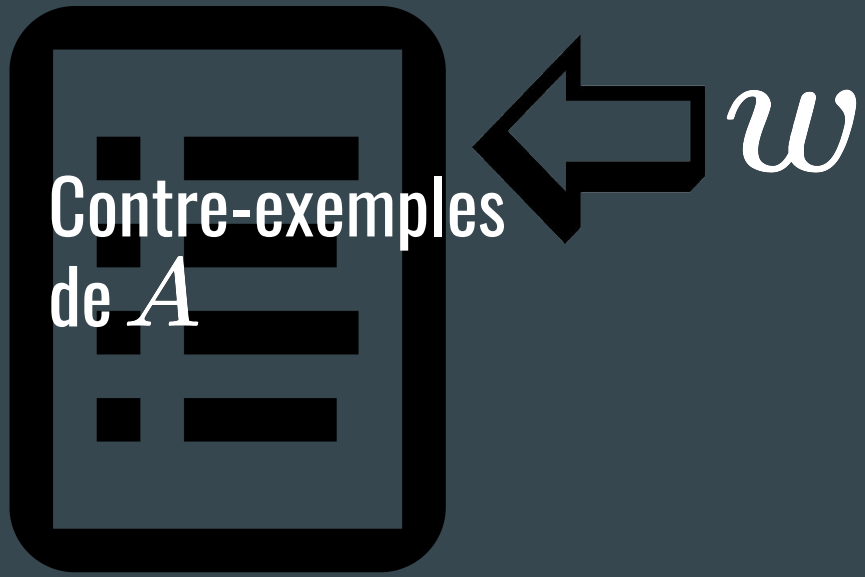
... dans ce cas, nous réaffinons la fonction p
et donc l'abstraction

Vue d'ensemble



Soit le RNN donne le même résultat que le DFA d'abstraction du réseau...

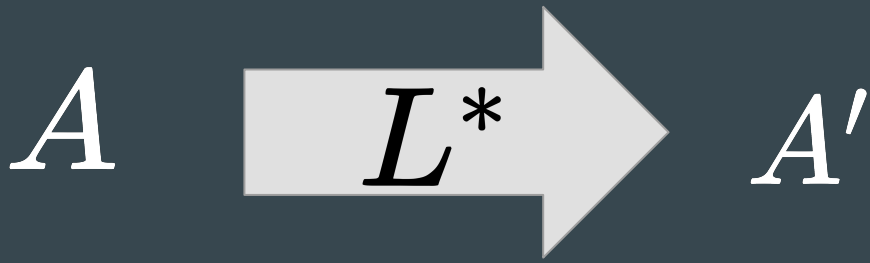
Vue d'ensemble



A

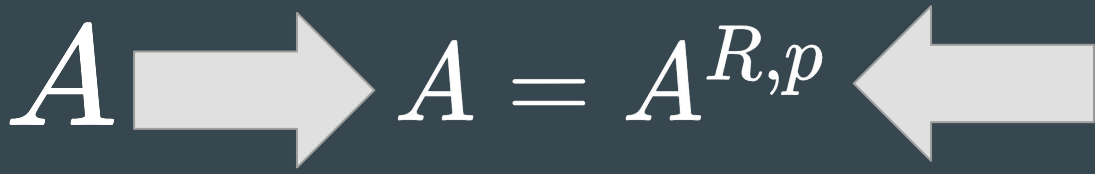
... dans ce cas, nous ajoutons ce mot en contre-exemple de A ...

Vue d'ensemble



... puis nous utilisons L* pour mettre à jour A

Vue d'ensemble



On continue jusqu'à ce que les 2 DFA convergent.

Propriété 1

Chacun des états de A peut se justifier par une entrée du réseau .

Propriété 2

Nous réaffinons p seulement lorsque l'on prouve que celui-ci représente mal R on a donc:

Chacun des réaffinement de p peut se justifier par une entrée du réseau .

Exploration des 2 automates en parallèle

Abstract classification conflicts

Lorsqu'un état d'acceptation de A et associé avec un état de rejet $A^{R,p}$ et vice-versa .

Clustering conflicts

Lorsqu'un état de $A^{R,p}$ est associé à plusieurs états de A.

Classification conflicts

Lorsque la classification de chaque état de R rencontré lors de l'extraction de $A^{R,p}$ n'est pas identique aux états de A que l'on atteint.

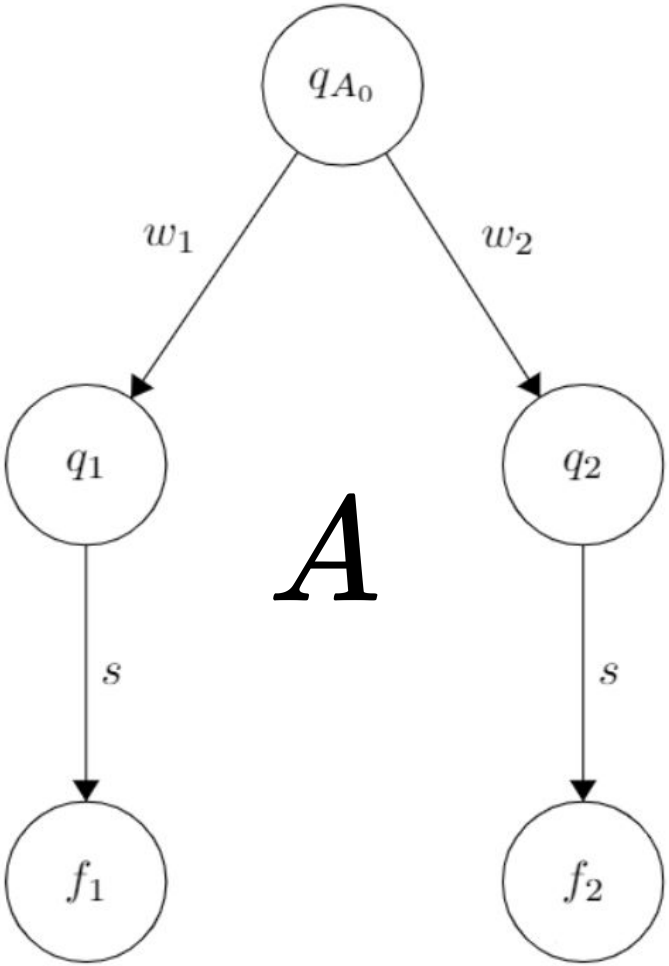
Clustering conflicts

Classification conflicts

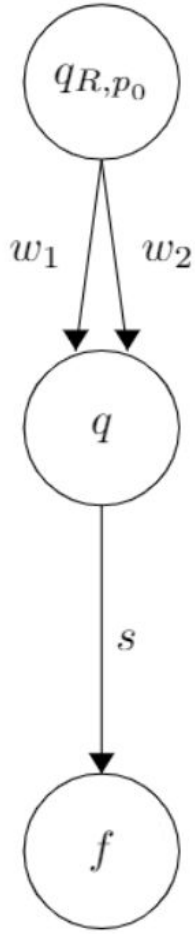
Résolution des conflits

Classification conflicts

On trouve un chemin $w \in \Sigma^*$ pendant l'extraction de $A^{R,p}$ pour lequel la classification de A et R sont différente. On le renvoie en contre-exemple de A



A



$A^{R,p}$

Résultats

Balanced Parentheses Language

$\sum$ ={a-z,(,)}

$(A[a-z]^*)^n([a-z]^*B)^n$

((((()))))

(((a))z)

(((((((((a()z)))))))))

Balanced Parentheses Language

~44600 features -1000 secondes d’entrainement

36% positifs

parent max: 11

2-layer GRU et 2-layer LSTM

(couche caché de taille 50)

Network	Train Set Accuracy		Max Nest. Depth	
	Abstr	RS	Abstr	RS
GRU	99.98	87.12	8	2
LSTM	99.98	94.19	8	3

Table 3. Counterexamples generated during extraction of automata from a GRU network trained on BP.

Refinement Based		Brute Force	
example	Time (s)	example	Time (s)
))	1.1	))	0.4
(())	1.2	(()i)ma	32.6
((()))	2.1		
(((())))	3.1		
((((()))))	3.8		
((((()))))	4.4		
(((((())))))	6.6		
((((((()))))))	9.2		
(((((((v)))))))	10.7		
(((((((a()z)))))))	8.3		

Conclusion

Annexe

Formalisation d’un RNN

Il s’agit d’une fonction qui prend en paramètre:

$g_R(h, x)$ un vecteur d'état $h_t \in (S_R = \mathbb{R}^{d_s})$ et x_{t+1}

un vecteur d’entrée et qui renvoit h_{t+1} en sortie

Formalisation d'un RNN

La fonction est appliquée récursivement sur une séquence $x_1, x_2, \ldots, x_n$. On applique donc dans l'ordre:

$$
\begin{aligned}
g_R(h_0, x_1) &= h_1, \\
g_R(h_1, x_2) &= h_2, \\
&\ldots, \\
g_R(h_{n-1}, x_n) &= h_n
\end{aligned}
$$

Formalisation d'un RNN

Pour les RNN accepteurs, on définit également une fonction $f_R : S_R \to \mathbb{B}$

qui renvoit $VRAI$ si et seulement si on est dans un état d'acceptation

Fonction de réaffinement

$$ref : p, h, H \rightarrow p'$$

p une partition,

h un état de R,

H un ensemble d'état de R ($H \subseteq S \backslash \{h\}$),

retourne une nouvelle partition p'

Fonction de réaffinement

$$\forall h_1 \in H : p'(h_1) \neq p'(h)$$

h doit avoir une partition différente des états de H

Fonction de réaffinement

$$\forall h_1, h_2 : p(h_1) \neq p(h_2) \Rightarrow p'(h_1) \neq p'(h_2)$$

On ne peut pas fusionner des partitions