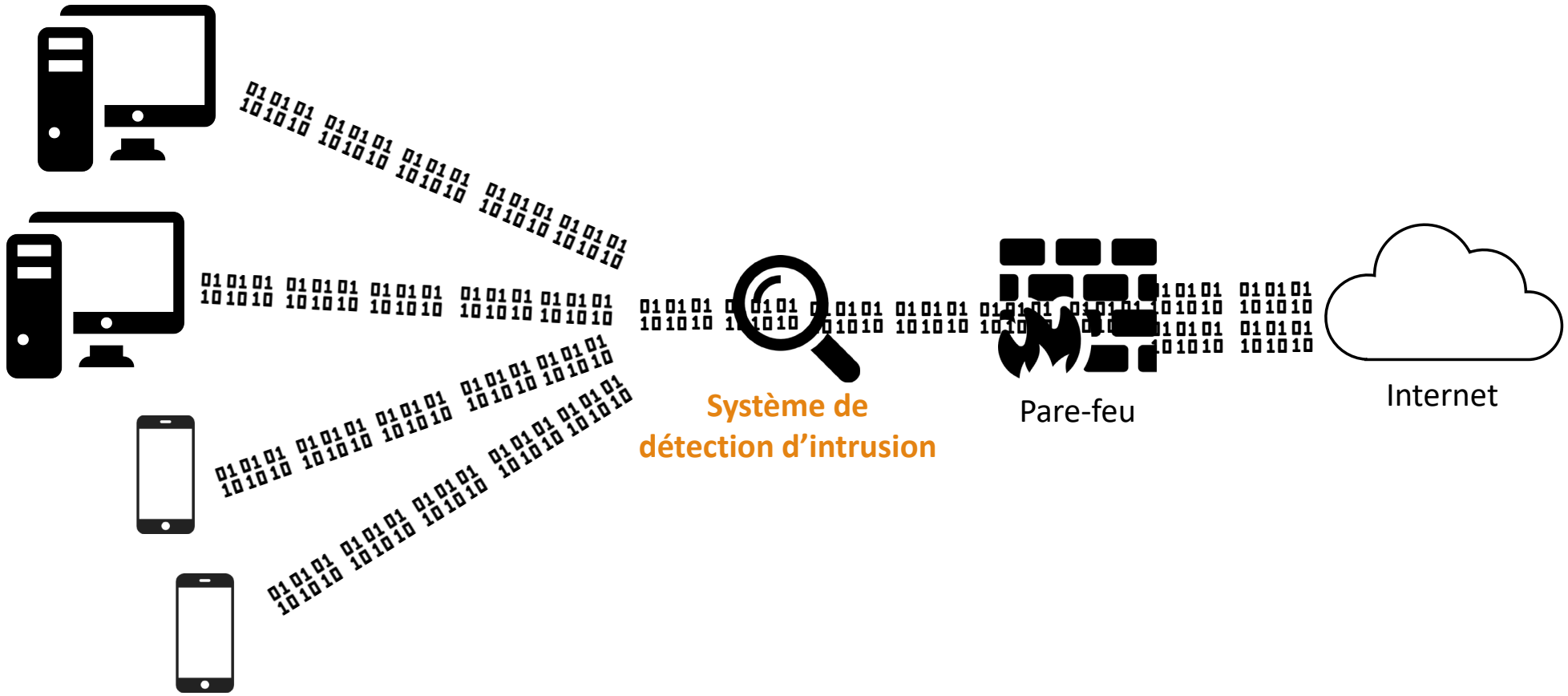
Utilisation des **R**ecurrent **N**eural **N**etworks pour la détection d'intrusion dans le réseau

PRÉSENTATION

Encadrants : Romain Laborde & Arnaud Oglaza
(IRIT – Équipe SIERA)

- Contexte
- Réseau de Neurones
- Protocole expérimental
- Résultats

Qu’est-ce qu’un système de détection d’intrusion ?



Fonctionnement

Système de détection d'intrusion

Méthodes d'analyse

Signature	Anomalie

Schéma d'attaque connue détecté ?

Oui	Non

⚠ Activités anormales

Activités normales ?

Comment améliorer la détection de schémas d'attaque connue ?

➡ **Systèmes d'apprentissage automatique**(1)

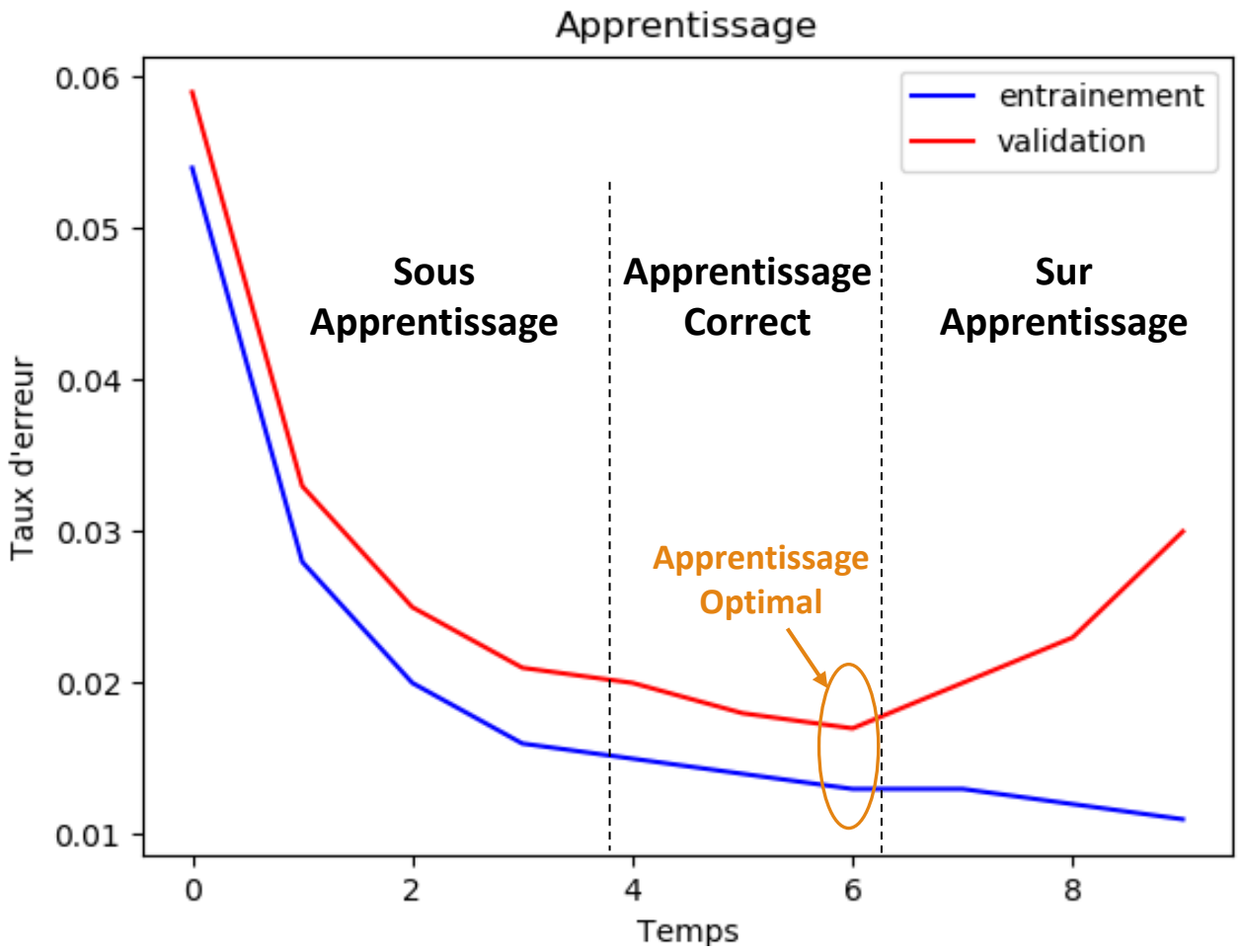
(1) "Applying long short-term memory recurrent neural networks to intrusion detection" - Ralf C. Staudemeyer

Jeu de données entrainement *(494 021 connexions)*

Données *(41 valeurs)*	Résultats
(0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00, ... (0,tcp,hostnames,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,225,11,1.00,1.00,0.00,0.00,0.05,0.07,0.00,255,11,0.04,0.07,0.00,0.00,1.00,1.00,0.00,0.00,	normal.) ... neptune.)

Jeu de données validation *(311 029 connexions)*

Données *(41 valeurs)*	Résultats
(0,icmp,ecr_i,SF,508,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,31,31,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,1.00,0.00,0.00,0.00,0.00,0.00, ... (0,udp,private,SF,105,147,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,254,1.00,0.01,0.01,0.00,0.00,0.00,0.00,0.00,	smurf.) ... normal.)



JEUX DE DONNÉES DARPA – KDD CUP'99

Somme pondérée:

$$z = \sum_{i=1}^{n} (x_i * \omega_i)$$

Fonction sigmoïde:

$$y = f(z) = \frac{1}{1 + e^{-z}}$$

APPRENTISSAGE

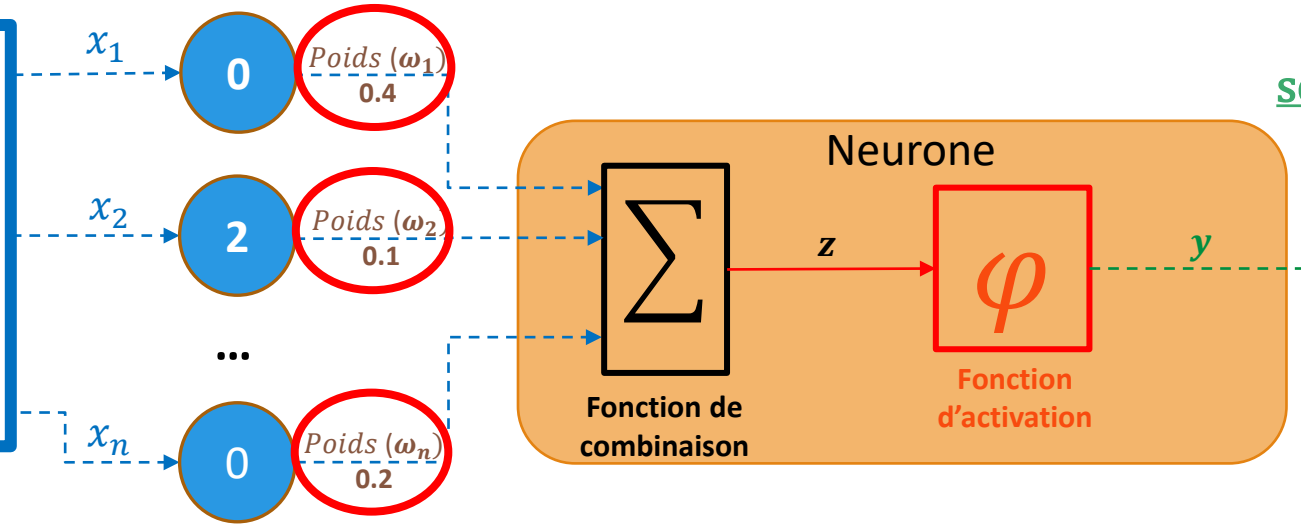
SI sortie observée ≠ sortie attendue **ALORS**
calculer_gradient(erreur)
modifier(poids)

(0,udp,private,SF,105
,146,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,1,1,0.0
0,0.00,0.00,0.00,1.0
0,0.00,0.00,255,254,
1.00,0.01,0.00,0.00,
0.00,0.00,0.00,0.00)

Données

(normal.)

Résultat



sortie observée

[0.560
0.824
0.228
0.015
0.549]

≠

[***1.000***
0.000
0.000
0.000
0.000]

sortie attendue

Type de données	Numéro
Normal	0
Probe	1
DoS	2
U2R	3
R2L	4

Donnée importante	0,9	Poids fort
Donnée secondaire	0,2	Poids faible
Constante	0	Poids nul

Somme pondérée:

$$z = \sum_{i=1}^{n} (x_i * \omega_i)$$

Fonction sigmoïde:

$$y = f(z) = \frac{1}{1 + e^{-z}}$$

APPRENTISSAGE

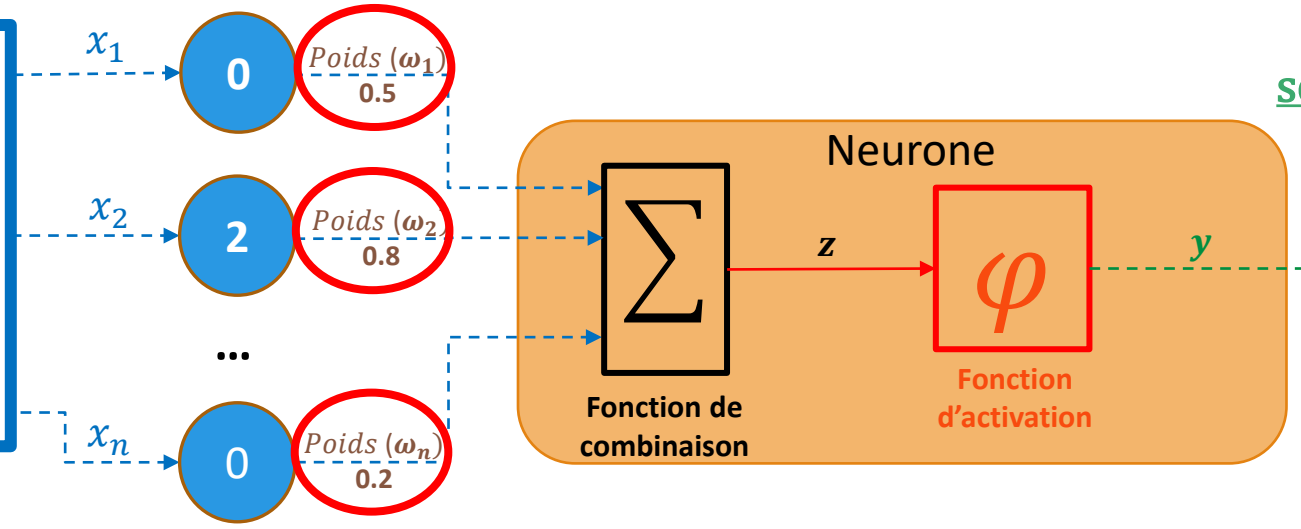
SI sortie observée ≠ sortie attendue **ALORS**
calculer_gradient(erreur)
modifier(poids)

(0,udp,private,SF,105
,146,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,1,1,0.0
0,0.00,0.00,0.00,1.0
0,0.00,0.00,255,254,
1.00,0.01,0.00,0.00,
0.00,0.00,0.00,0.00)

Données

(normal.)

Résultat



sortie observée

[*0.723*
0.132
0.128
0.215
0.001]

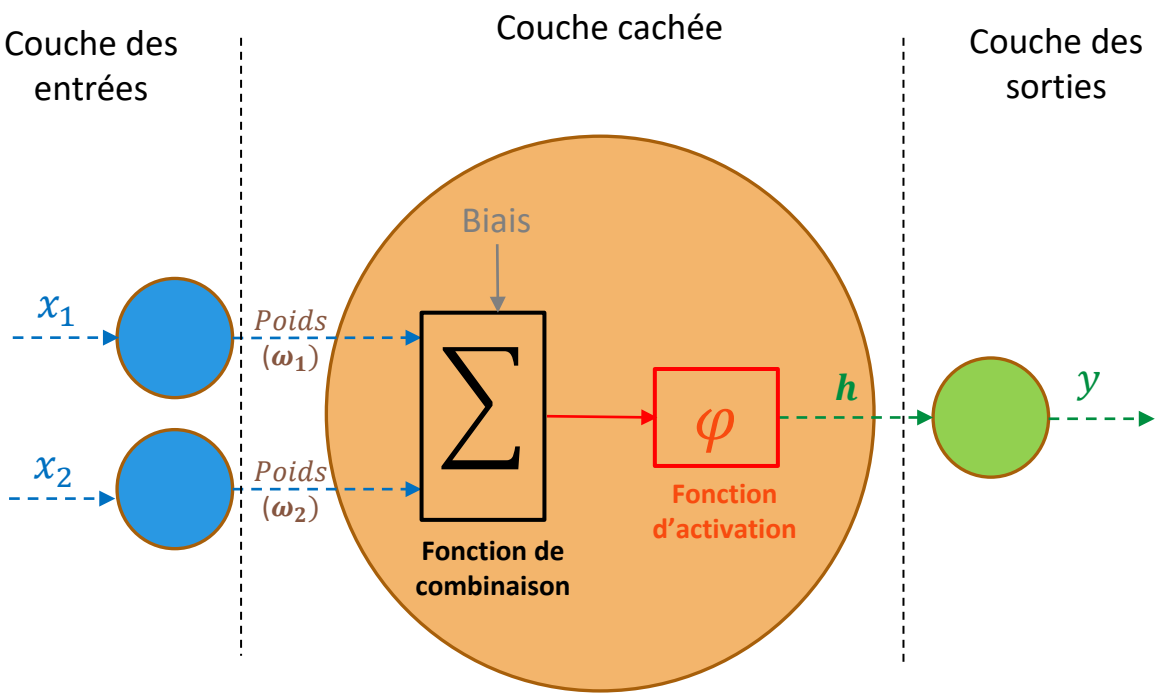
=

[*1.000*
0.000
0.000
0.000
0.000]

sortie attendue

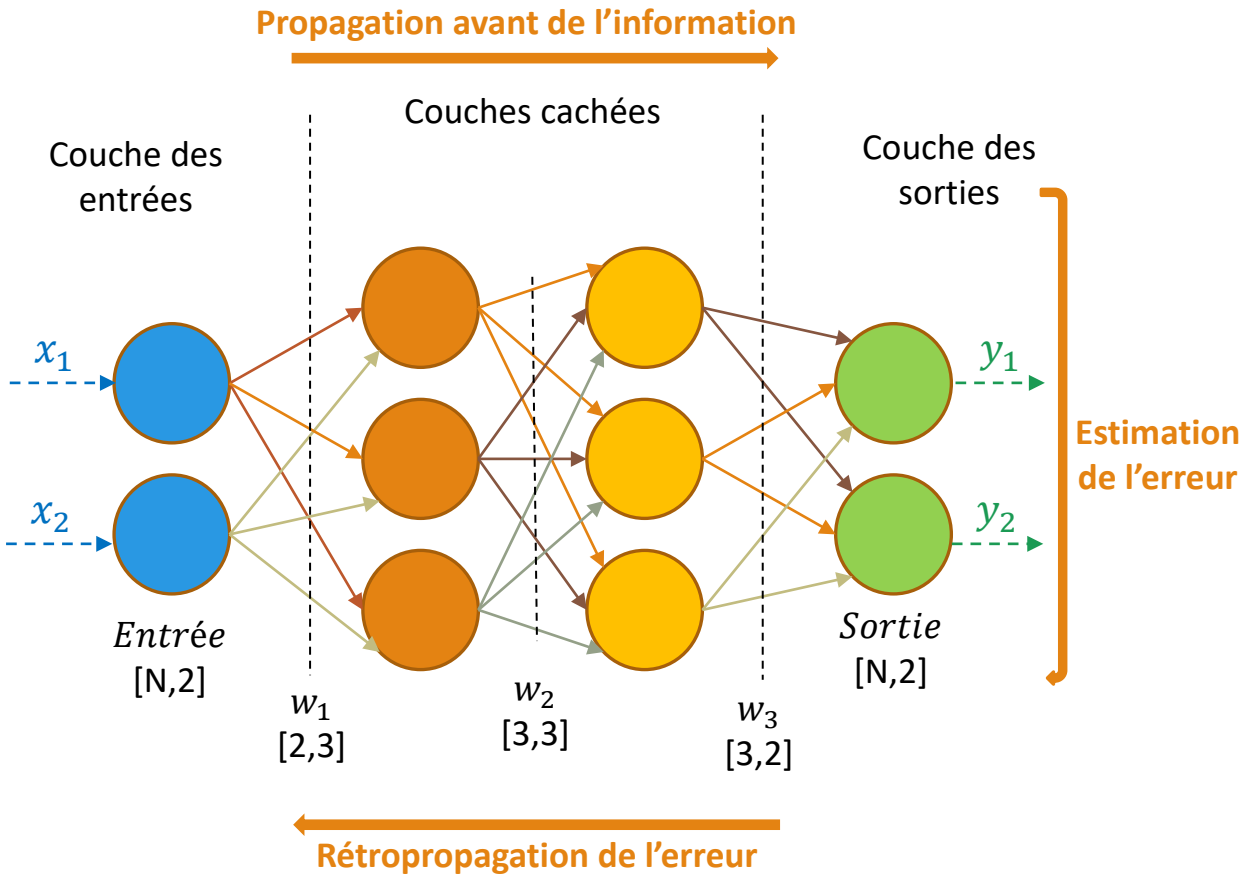
Type de données	Numéro
Normal	0
Probe	1
DoS	2
U2R	3
R2L	4

Donnée importante	0,9	Poids fort
Donnée secondaire	0,2	Poids faible
Constante	0	Poids nul



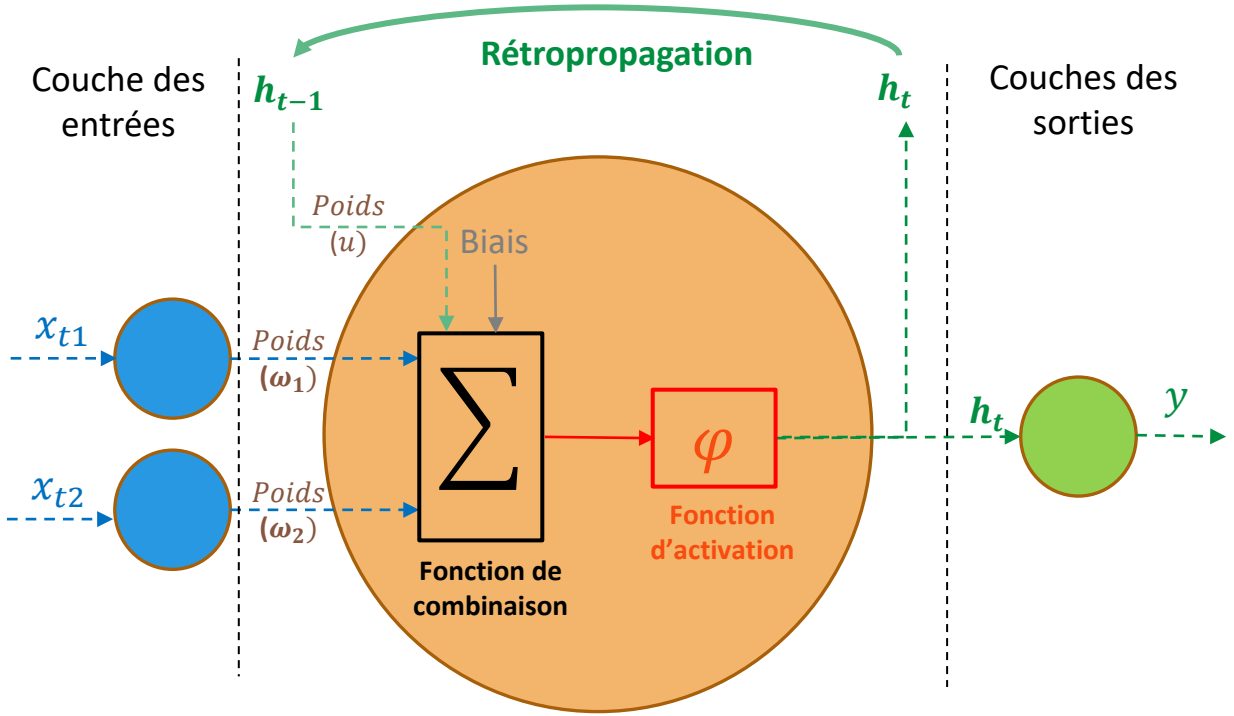
$$h = f\left(\sum_{i=1}^{n} x_i * \omega_i + biais\right)$$

NEURONE À PROPAGATION AVANT



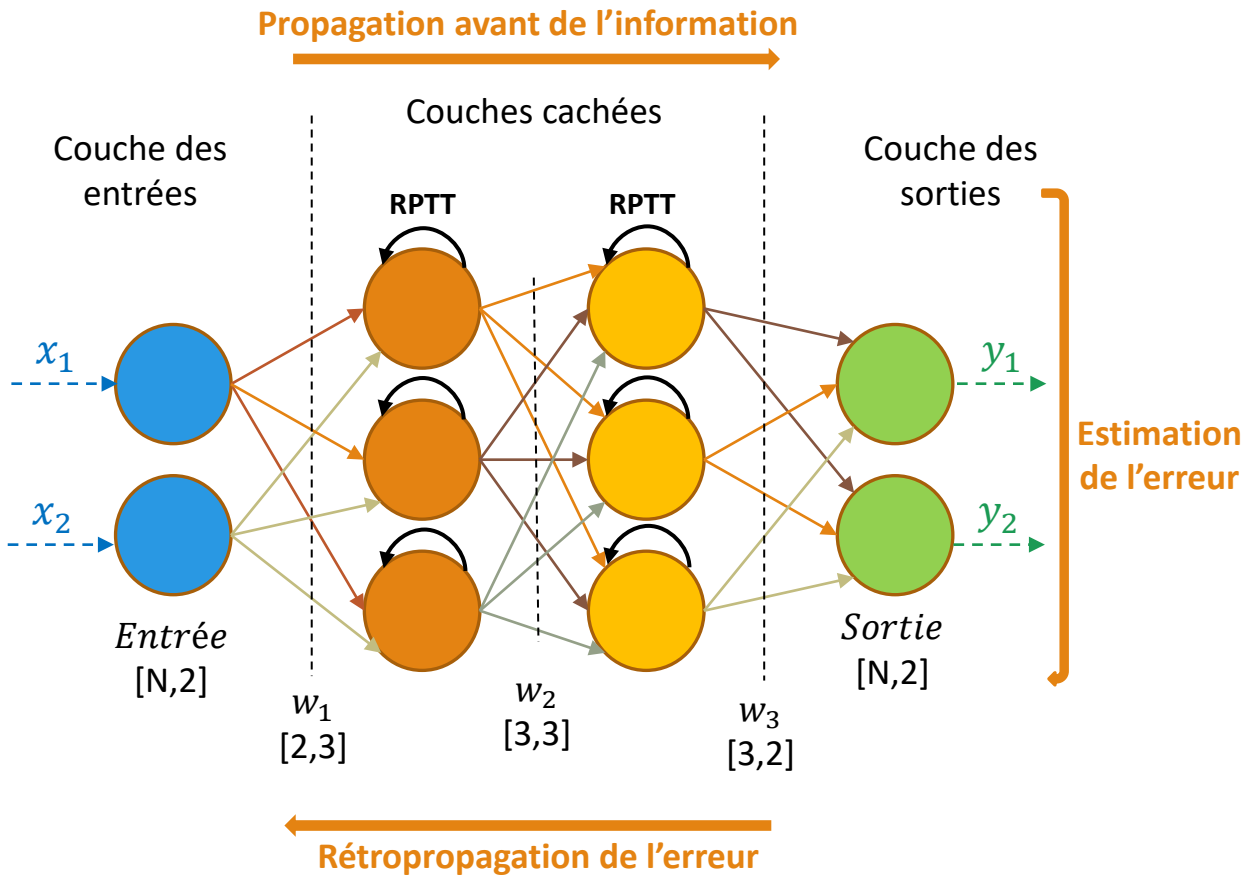
RÉSEAU DE NEURONES À PROPAGATION AVANT

(Pas de mémoire)



$$h_t = f\left(\sum_{i=1}^{n} x_{ti} * \omega_i + h_{t-1} * u + biais\right)$$

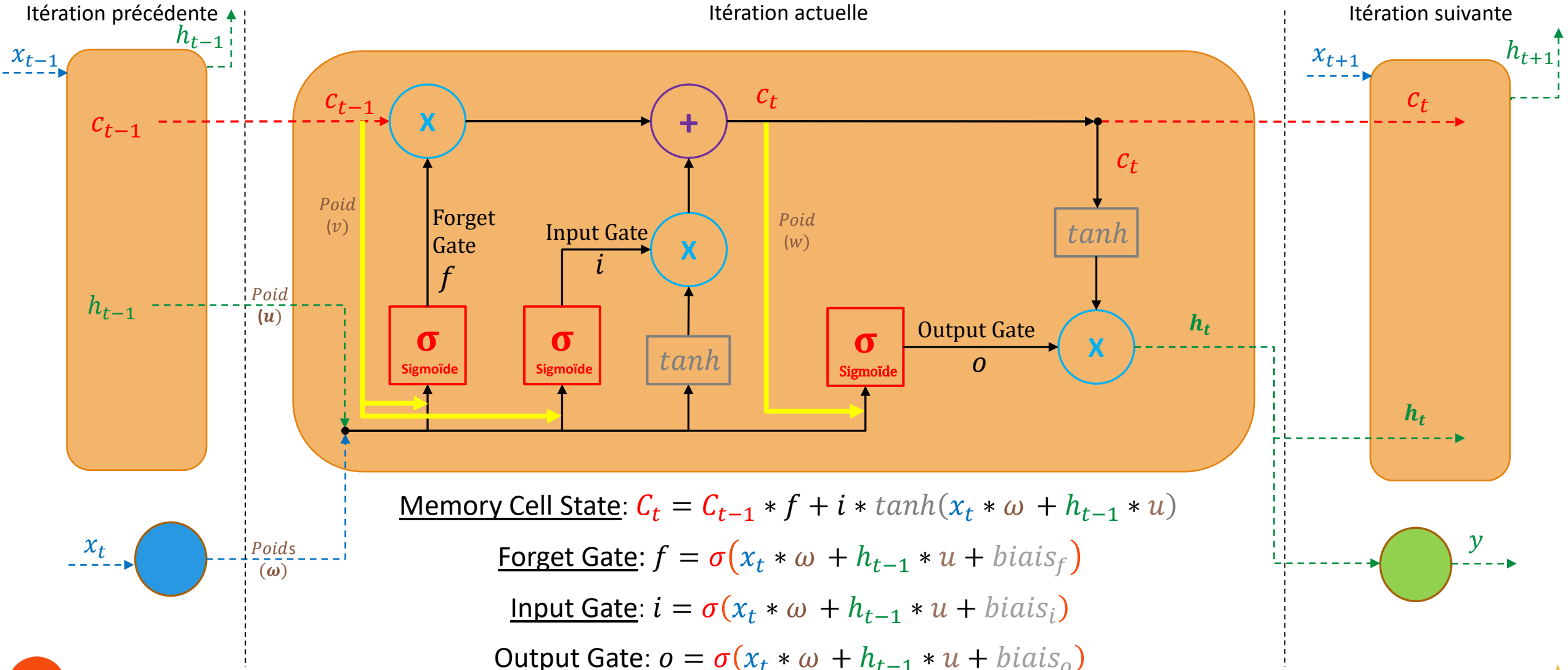
NEURONE DE RÉSEAU RÉCURRENT



Comment avoir de la mémoire long terme?

RÉSEAU DE NEURONES RÉCURRENTS

(Mémoire court terme)



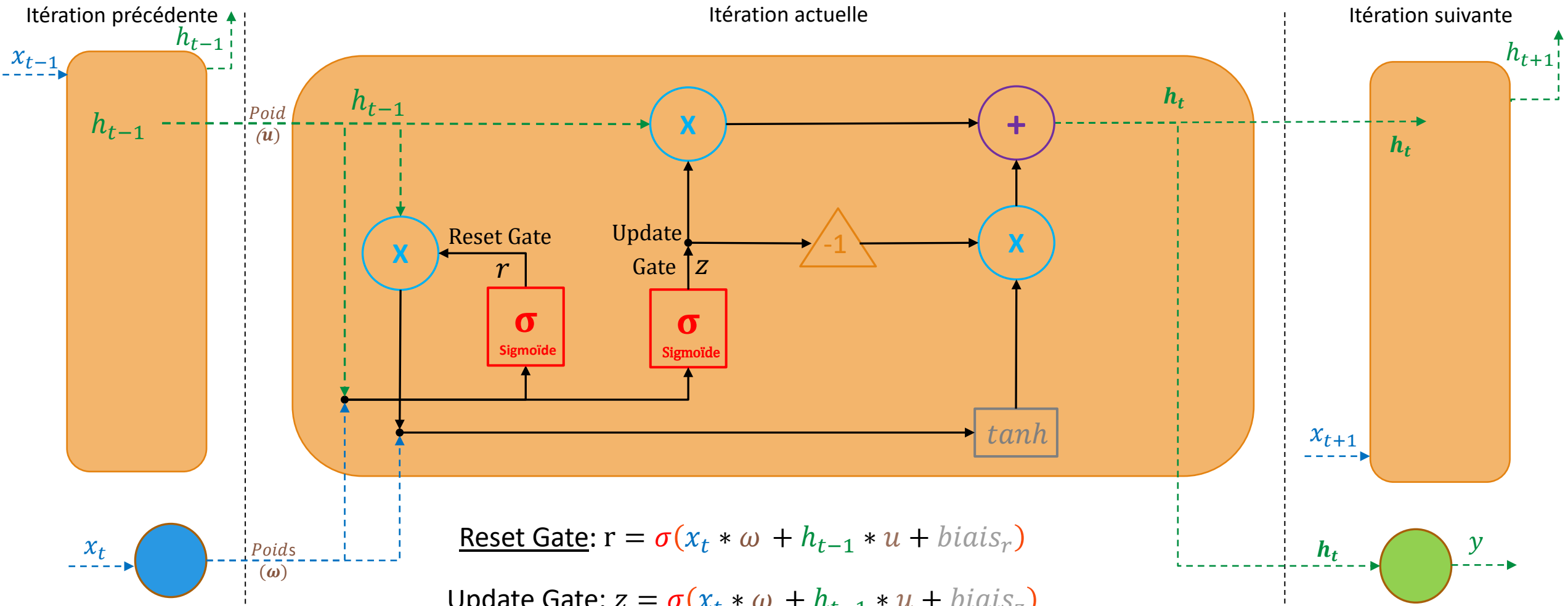
Memory Cell State: $C_t = C_{t-1} * f + i * tanh(x_t * \omega + h_{t-1} * u)$

Forget Gate: $f = \sigma(x_t * \omega + h_{t-1} * u + biais_f)$

Input Gate: $i = \sigma(x_t * \omega + h_{t-1} * u + biais_i)$

Output Gate: $o = \sigma(x_t * \omega + h_{t-1} * u + biais_o)$

Hidden state: $h_t = tanh(C_t) * o$



Reset Gate: $r = \sigma(x_t * \omega + h_{t-1} * u + biais_r)$

Update Gate: $z = \sigma(x_t * \omega + h_{t-1} * u + biais_z)$

Hidden state: $h_t = (1 - z) * h_{t-1} + z * \sigma(x_t * \omega + u(r * h_{t-1})) + biais_h$

Paramètres d'entrée : X *(8 éléments)*

Index	0	1	2	3	4	5	6	7
0	http	181	0	0	5450	0	0	9
1	http	239	0	0	486	0	0	19
2	http	235	0	0	1337	0	0	29
3	http	219	0	0	1337	0	0	39

Paramètres de sortie : Y *(5 éléments)*

Type de données	Numéro
Normal	0
Probe	1
DoS	2
U2R	3
R2L	4

Encodage *Standard Scaler*

Index	0	1	2	3	4	5	6	7
0	-1,67319	-0,00287	-0,28286	-0,25209	0,13866	-0,04413	-0,00978	-1,69431
1	-1,67319	-0,00281	-0,28286	-0,25209	-0,01157	-0,04413	-0,00978	-1,60001
2	-1,67319	-0,00282	-0,28286	-0,25209	0,01417	-0,04413	-0,00978	-1,50570
3	-1,67319	-0,00284	-0,28286	-0,25209	0,01417	-0,04413	-0,00978	-1,41140

Encodage One Hot

Numéro	0	1	2	3	4
0	**1**	0	0	0	0
1	0	**1**	0	0	0
2	0	0	**1**	0	0
3	0	0	0	**1**	0
4	0	0	0	0	**1**

STANDARDISATION DES DONNÉES

Pseudo-code sous Keras

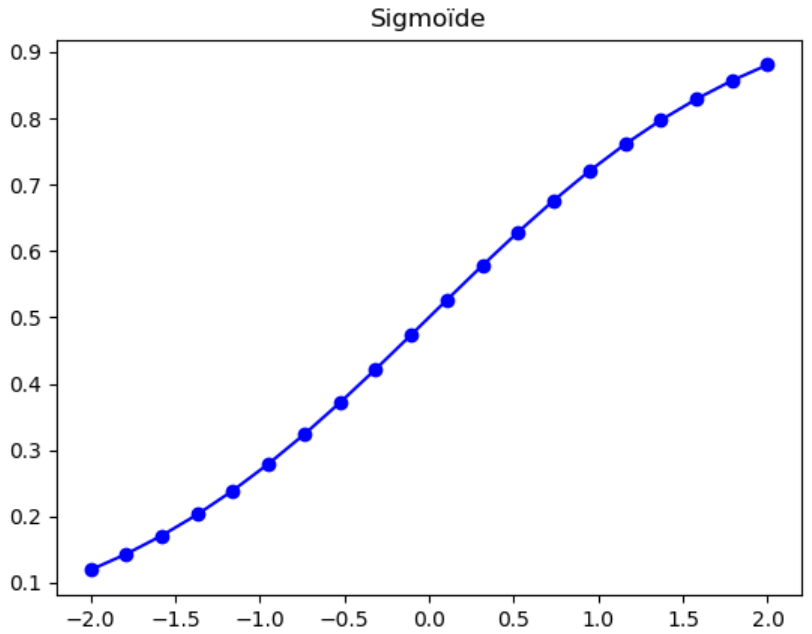
```
modele = Sequentiel()

modele.ajouter(LSTM(cellule_LSTM=128,
                    forme=(param_entree=8, echantillon=494021)))
modele.ajouter(Perte(20%))

modele.ajouter(Liason(param_sortie=5,
                      fonction_activation='sigmoide'))

modele.compiler(taux_perte=erreur_quadratique_moyenne,
                taux_apprentissage=0.001,
                mesure_efficacite=['précision'])

modele.entrainement(x=entree_entrainement, y=resultat_entrainement,
                    nb_iteration=20,
                    validation=(x=entree_test, y=resultat_test))
```

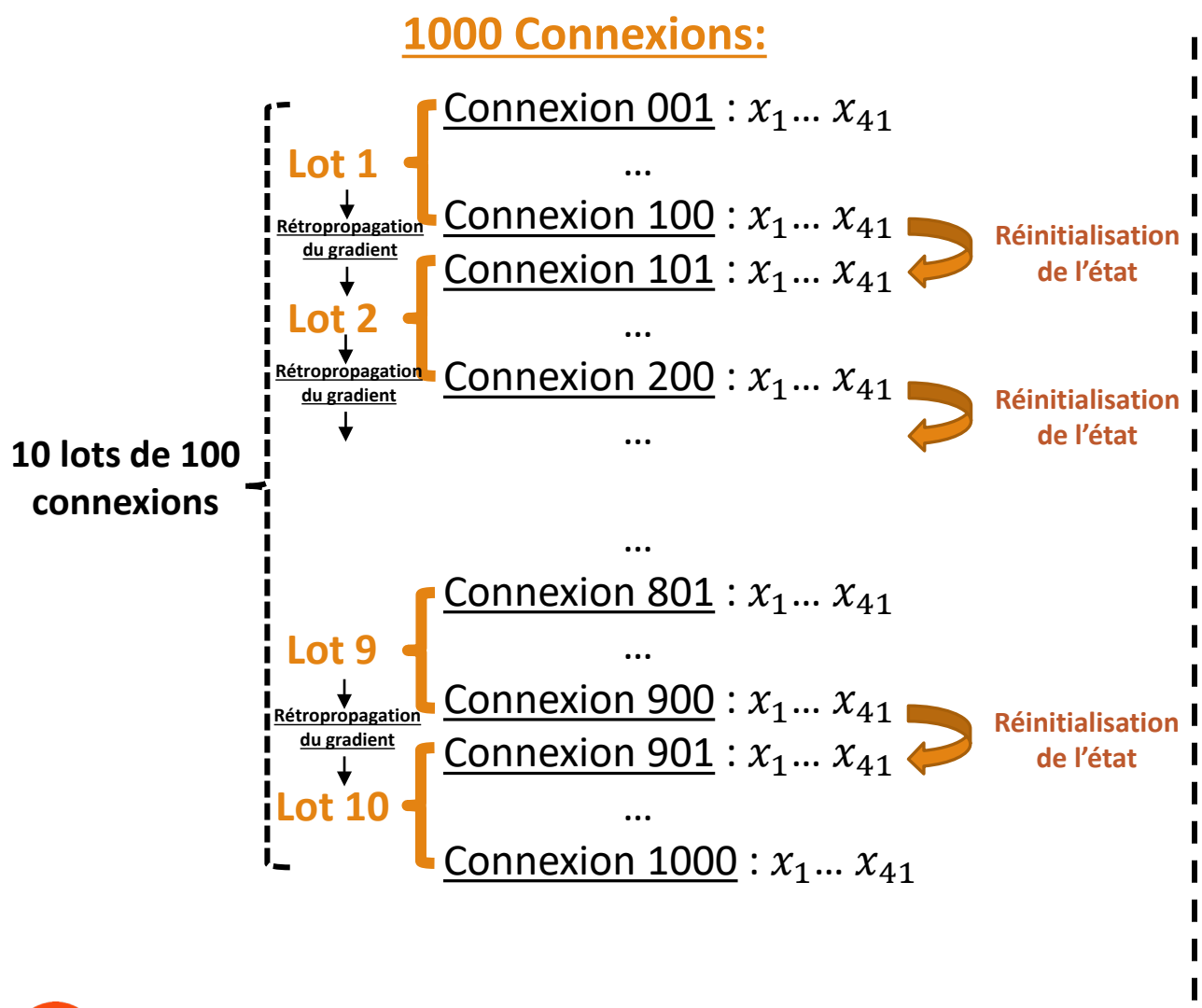


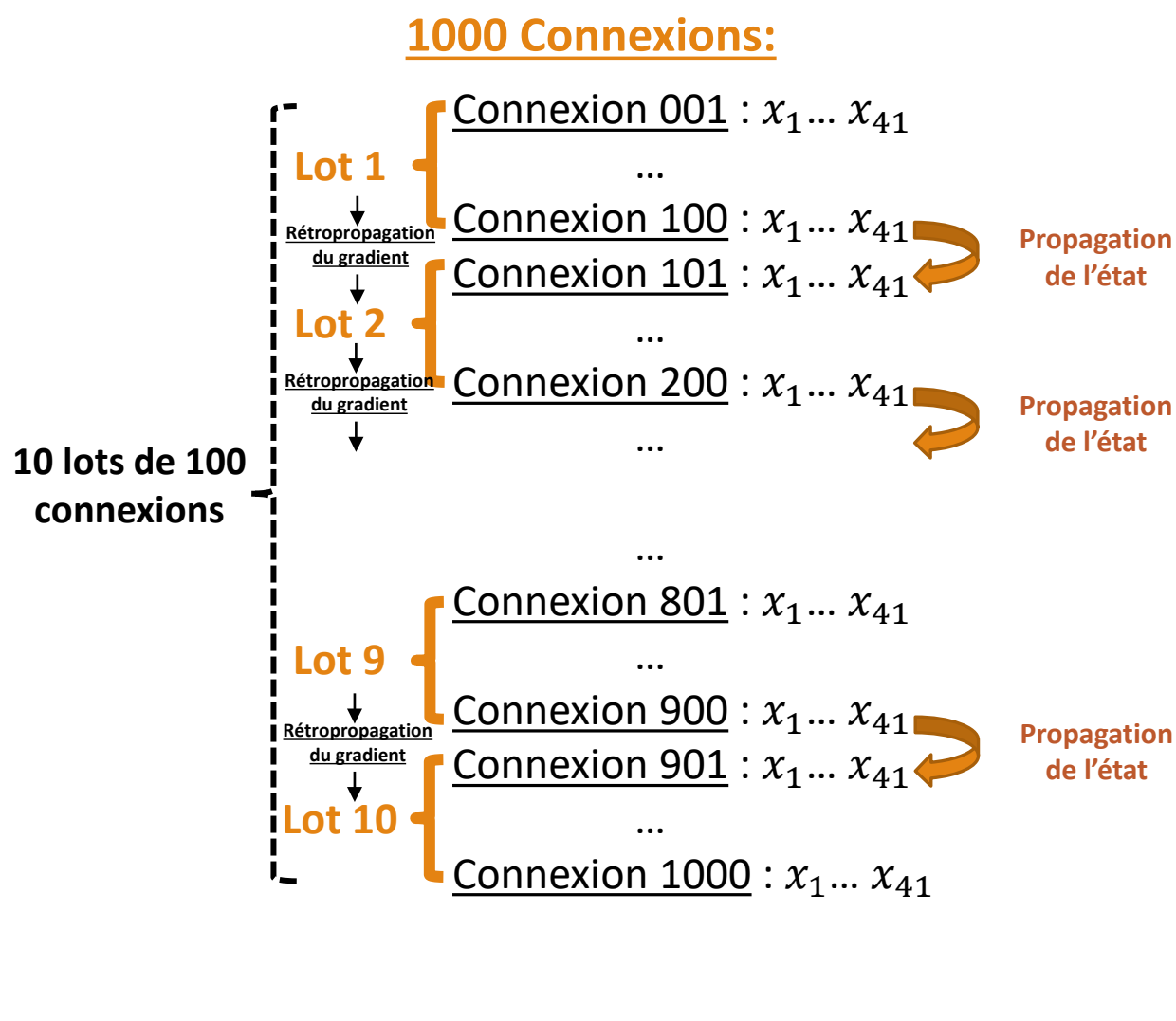
Fonction sigmoïde: $\sigma(x) = \frac{1}{1+e^{-x}}$

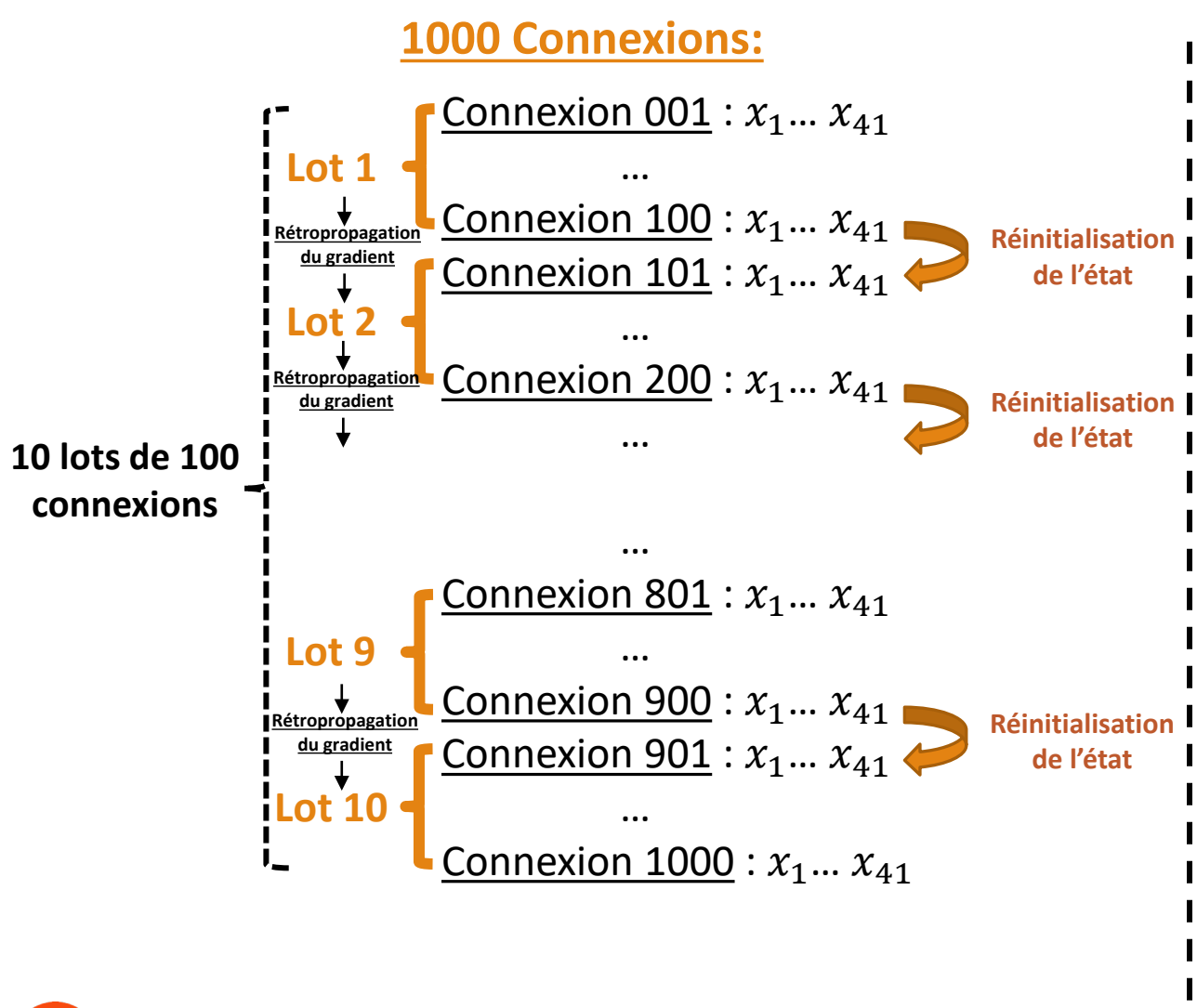
Erreur Quadratique Moyenne:

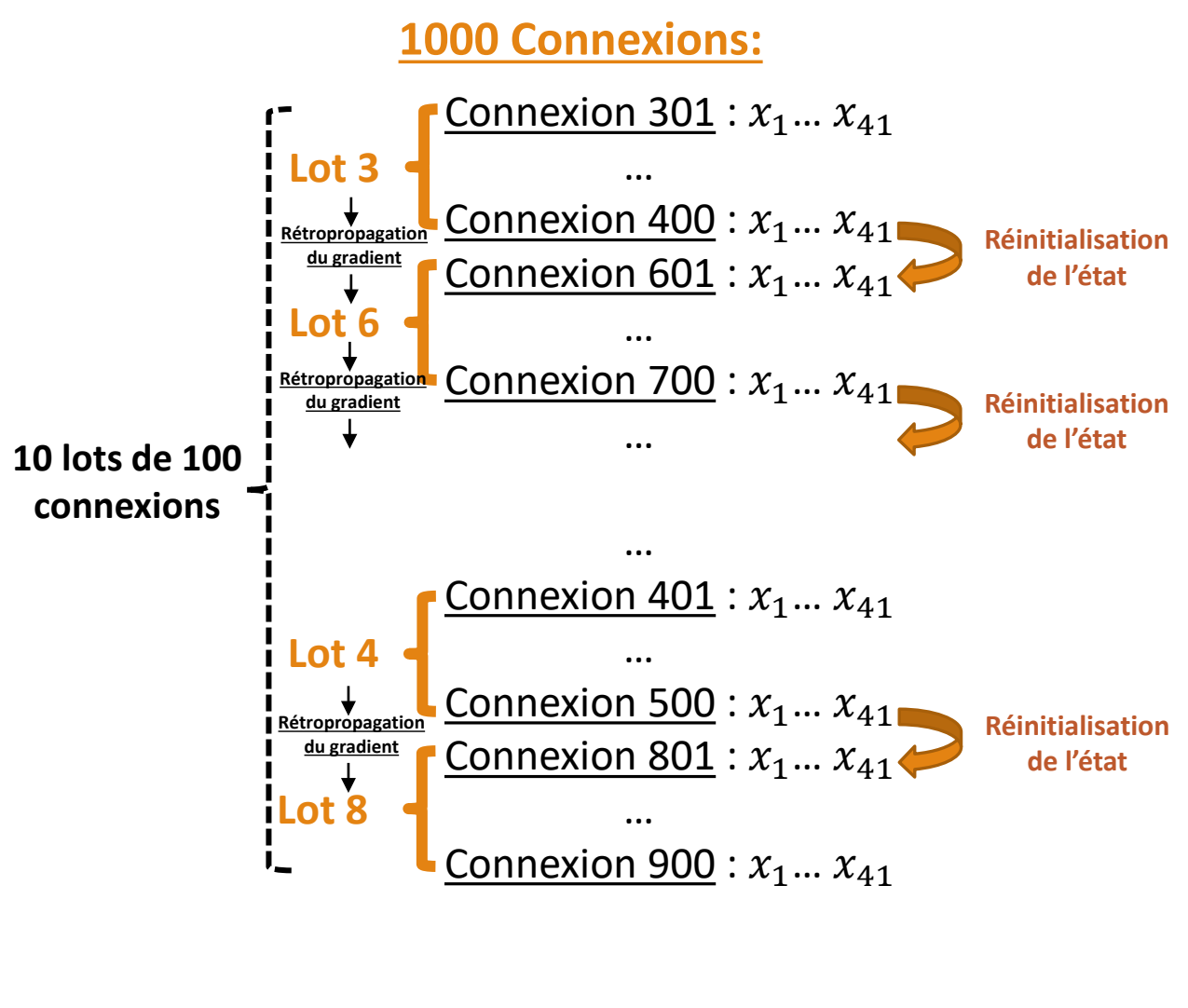
$$Mean\ Squared\ Error(x) = Biais(x)^2 + Variance(x)$$

CRÉATION D'UN MODÈLE

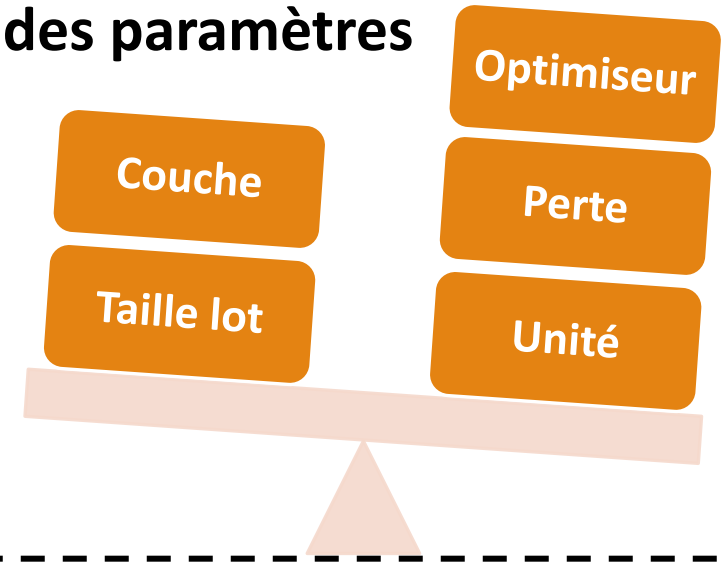

1000 Connexions:
10 lots de 100 connexions
Lot 1
Rétropropagation du gradient
Lot 2
Rétropropagation du gradient
Lot 9
Rétropropagation du gradient
Lot 10
Connexion 001 : $x_1 \dots x_{41}$
Connexion 100 : $x_1 \dots x_{41}$
Connexion 101 : $x_1 \dots x_{41}$
Connexion 200 : $x_1 \dots x_{41}$
Connexion 801 : $x_1 \dots x_{41}$
Connexion 900 : $x_1 \dots x_{41}$
Connexion 901 : $x_1 \dots x_{41}$
Connexion 1000 : $x_1 \dots x_{41}$
Réinitialisation de l'état
Réinitialisation de l'état
Réinitialisation de l'état
SANS ÉTAT

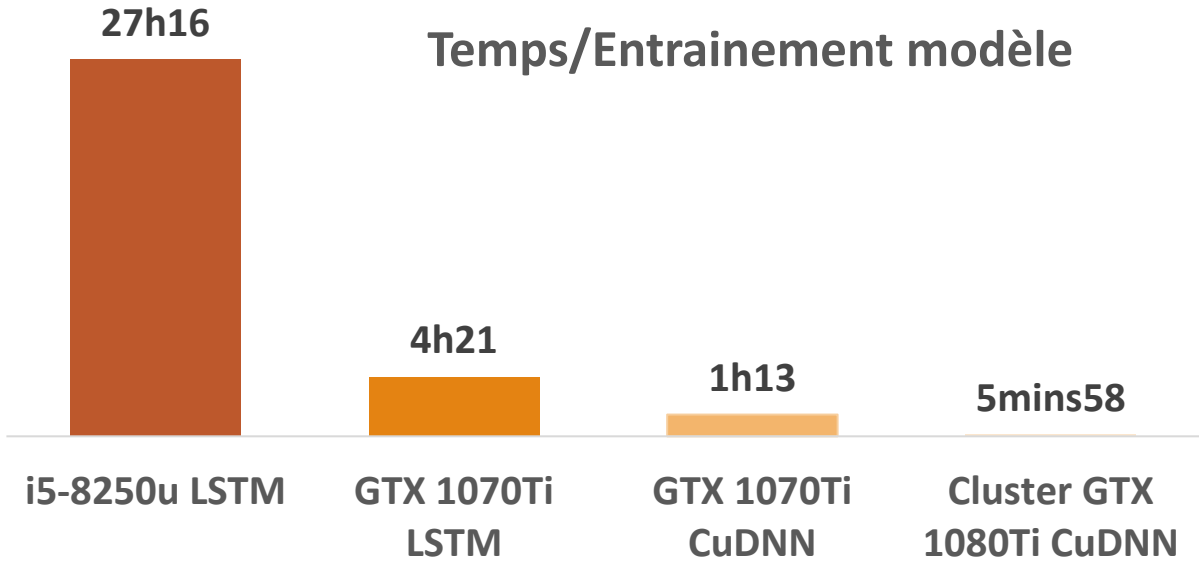

1000 Connexions:
10 lots de 100 connexions
Lot 1
Rétropropagation du gradient
Lot 2
Rétropropagation du gradient
Lot 9
Rétropropagation du gradient
Lot 10
Connexion 001 : $x_1 \dots x_{41}$
Connexion 100 : $x_1 \dots x_{41}$
Connexion 101 : $x_1 \dots x_{41}$
Connexion 200 : $x_1 \dots x_{41}$
Connexion 801 : $x_1 \dots x_{41}$
Connexion 900 : $x_1 \dots x_{41}$
Connexion 901 : $x_1 \dots x_{41}$
Connexion 1000 : $x_1 \dots x_{41}$
Propagation de l'état
Propagation de l'état
Propagation de l'état
AVEC ÉTAT


1000 Connexions:
10 lots de 100 connexions
Lot 1
Rétropropagation du gradient
Lot 2
Rétropropagation du gradient
Lot 9
Rétropropagation du gradient
Lot 10
Connexion 001 : $x_1 \dots x_{41}$
...
Connexion 100 : $x_1 \dots x_{41}$
Connexion 101 : $x_1 \dots x_{41}$
...
Connexion 200 : $x_1 \dots x_{41}$
...
...
Connexion 801 : $x_1 \dots x_{41}$
...
Connexion 900 : $x_1 \dots x_{41}$
Connexion 901 : $x_1 \dots x_{41}$
...
Connexion 1000 : $x_1 \dots x_{41}$
Réinitialisation de l'état
Réinitialisation de l'état
Réinitialisation de l'état
SANS MÉLANGE

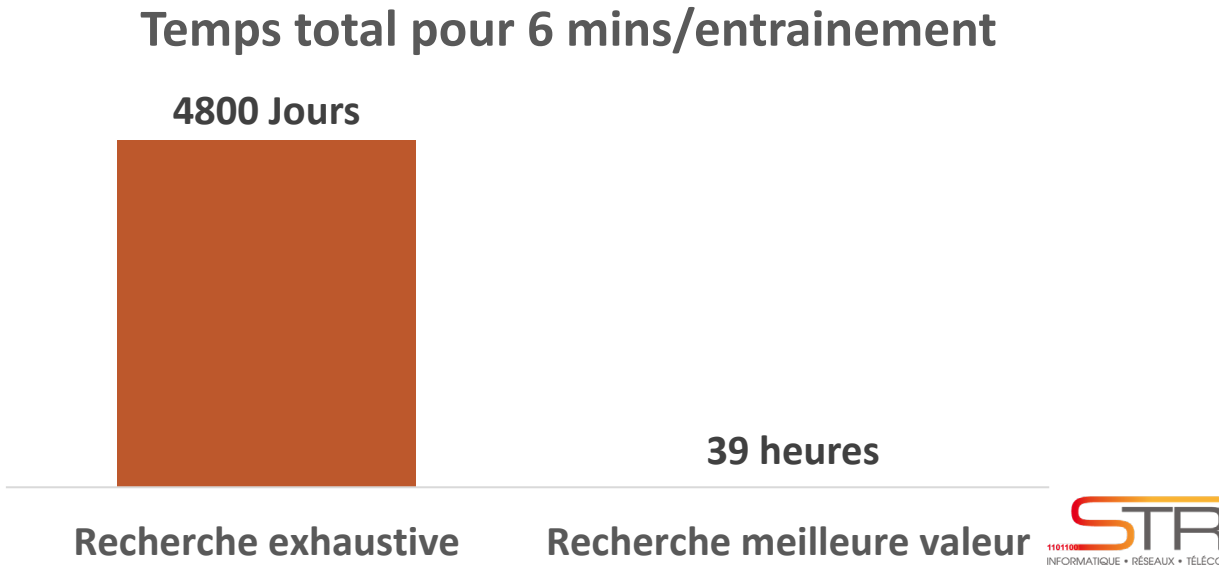

1000 Connexions:
10 lots de 100 connexions
Lot 3
Rétropropagation du gradient
Lot 6
Rétropropagation du gradient
Lot 4
Rétropropagation du gradient
Lot 8
Connexion 301 : $x_1 \dots x_{41}$
...
Connexion 400 : $x_1 \dots x_{41}$
Connexion 601 : $x_1 \dots x_{41}$
...
Connexion 700 : $x_1 \dots x_{41}$
...
...
Connexion 401 : $x_1 \dots x_{41}$
...
Connexion 500 : $x_1 \dots x_{41}$
Connexion 801 : $x_1 \dots x_{41}$
...
Connexion 900 : $x_1 \dots x_{41}$
Réinitialisation de l'état
Réinitialisation de l'état
Réinitialisation de l'état
AVEC MÉLANGE

Équilibre des paramètres

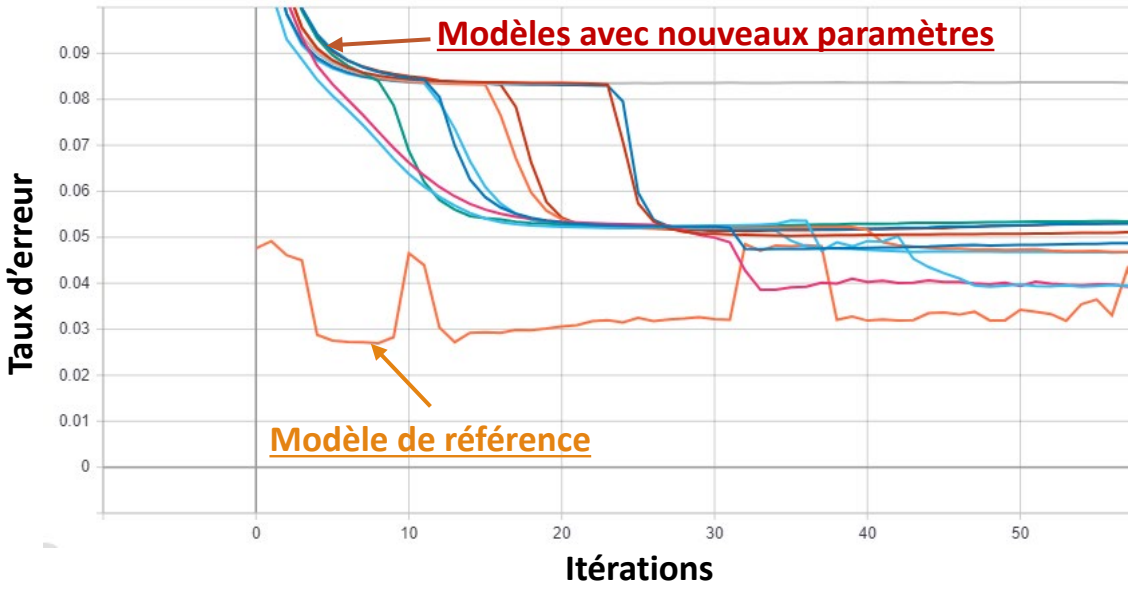


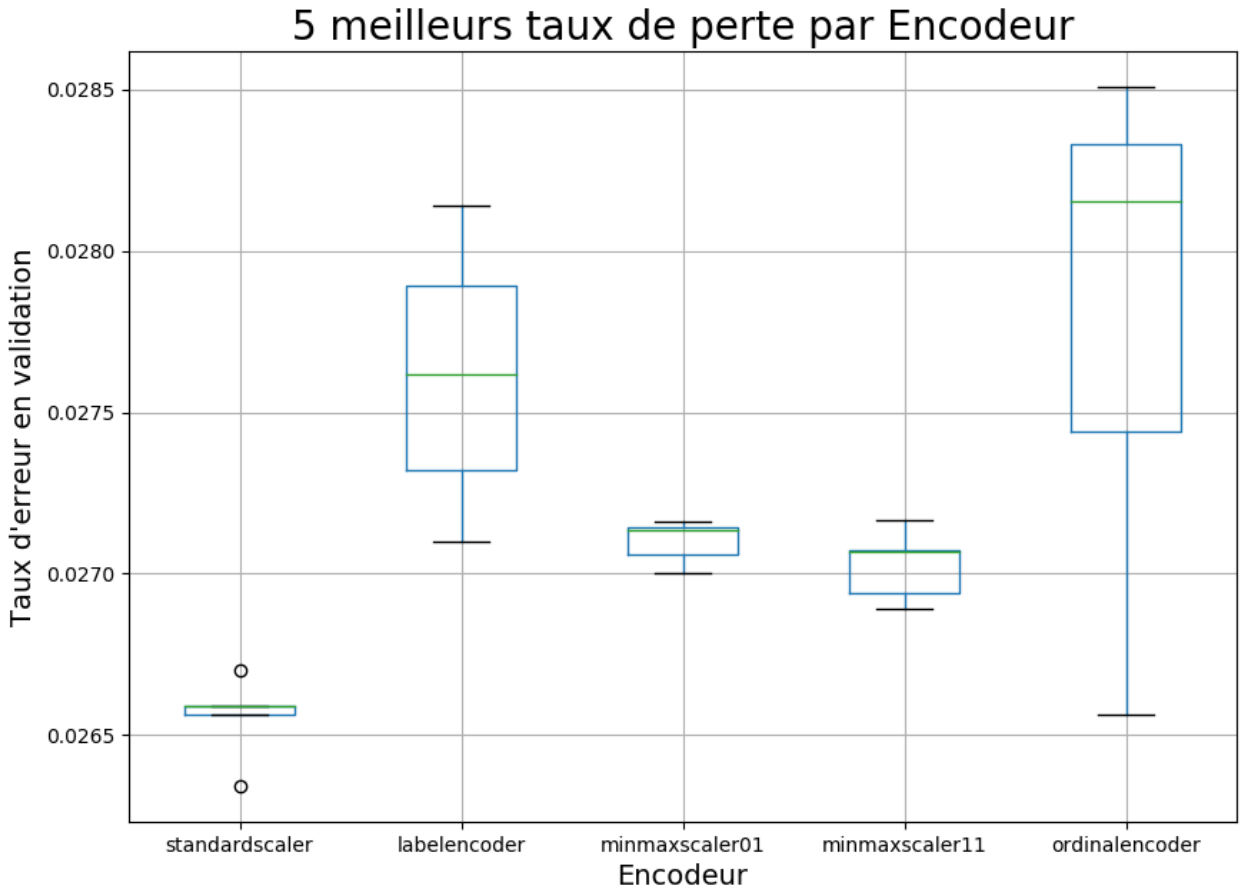


Recherche Exhaustive	Recherche de meilleure valeur
5 encodeurs	(5 encodeurs
∗ 6 optimiseurs	+ 6 optimiseurs
∗ 4 fonctions d'activation	+ 4 fonctions d'activation
∗ 6 valeurs de perte	+ 6 valeurs de perte
∗ 4 nombres de couches	+ 4 nombres de couches
∗ 10 nombres d'unités	+ 10 nombres d'unités
∗ 4 tailles de lot	+ 4 tailles de lot)
∗ 10 entrainement	∗ 10 entrainements
= **1 152 000** entrainements	= **390** entrainements



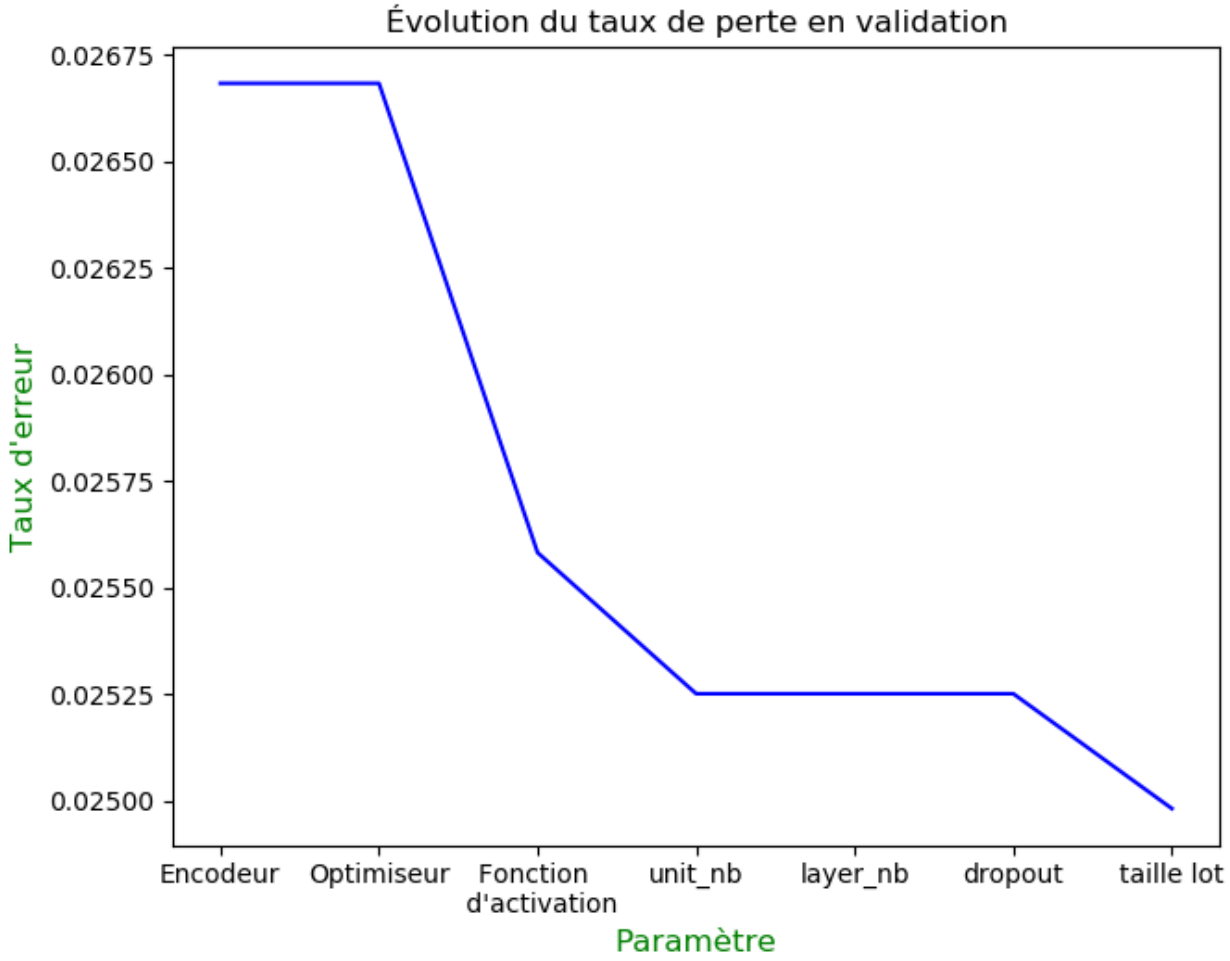
Fonction d'erreur:
Erreur Quadratique Moyenne





COMPARAISON DES MODÈLES

RECHERCHE DU MEILLEUR ENCODAGE



APPLICATION DE L'ALGORITHME

Prédiction

Réel	Normal	Probe	DoS	U2R	R2L	% Correct
Normal	**59541**	0	1035	0	0	98.29
Probe	3555	**0**	611	0	0	00.00
DoS	16674	0	**213179**	0	0	92.74
U2R	69	0	1	**0**	0	00.00
R2L	16017	0	318	0	**0**	00.00
% Correct	62.11	00.00	99.08	00.00	00.00	PRÉCISION : **92.25 %**

Jeu d'entrainement KDD'99 10% avec état

Prédiction

Réel	Normal	Probe	DoS	U2R	R2L	% Correct
Normal	**59896**	242	436	2	17	98.84
Probe	91	**3359**	716	0	0	80.62
DoS	5293	937	**223623**	0	0	97.28
U2R	58	0	7	**2**	3	02.85
R2L	12386	78	109	0	**3774**	23.08
% Correct	77.06	72.76	99.43	50,00	99.47	PRÉCISION : **93.45 %**

Jeu d'entrainement KDD'99 10% sans état

Type	Connexion	Part
Normal	492 708	**19.86%**
Probe	41 102	**00.84%**
DoS	3 883 370	**79.30%**
U2R	52	**00.001%**
R2L	1 126	**00.02%**
Total	4 898 431	100 %

Répartition des connexions

	Connexion	TYPE
1	normal	NORMAL
...	normal	NORMAL
41114	normal	NORMAL
41115	perl	U2R
41116	normal	NORMAL
...	normal	NORMAL
77908	normal	NORMAL
77909	smurf	DOS
...	smurf	DOS
114852	smurf	DOS
114853	spy	R2L
114854	spy	R2L
114855	neptune	DOS
...	neptune	DOS
194320	neptune	DOS

Exemple d'une séquence de connexions

COMPOSITION DU JEU

Prédiction

Réel	Normal	Probe	DoS	U2R	R2L	% Correct
Normal	**59928**	229	436	0	0	98.90
Probe	306	**3239**	621	0	0	77.74
DoS	5343	740	**223770**	0	0	97.35
U2R	69	0	1	**0**	0	00.00
R2L	12497	55	3795	0	**0**	00.00
% Correct	76.69	75.97	97.87	00.00	00.00	PRÉCISION : **92.25 %**

Jeu d'entrainement KDD'99 entier sans mélange

Prédiction

Réel	Normal	Probe	DoS	U2R	R2L	% Correct
Normal	**60112**	230	251	0	0	99.20
Probe	254	**3594**	318	0	0	86.26
DoS	5877	1111	**222865**	0	0	96.95
U2R	63	0	1	**0**	6	00.00
R2L	13435	121	6	0	**2785**	17.03
% Correct	75.38	71.08	99.74	00.00	99.78	PRÉCISION : **93.03 %**

Jeu d'entrainement KDD'99 entier avec mélange

Type	Connexion	Part
Normal	97 278	**19.69%**
Probe	4 107	**00.83%**
DoS	391 458	**79.24%**
U2R	52	**00.01%**
R2L	1126	**00.23%**
Total	494 021	100 %

Répartition des connexions

	Connexion	TYPE
1	ftp_write	R2L
2	ftp_write	R2L
3	normal	NORMAL
...	normal	NORMAL
160	normal	NORMAL
161	back	DOS
...	back	DOS
1161	back	DOS
1162	imap	R2L
1163	normal	NORMAL
...	normal	NORMAL
1563	normal	NORMAL
1564	loadmodule	U2R

Exemple d'une séquence de connexions

COMPOSITION DU JEU

Prédiction

Réel	Normal	Probe	DoS	U2R	R2L	% Correct
Normal	**59959**	250	333	0	51	98.95
Probe	85	**3315**	722	0	44	79.57
DoS	5335	984	**223534**	0	0	97.25
U2R	67	0	1	**0**	2	00.00
R2L	13007	63	64	0	**3213**	19.65
% Correct	76.42	71.87	99.50	00.00	97.06	PRÉCISION : **93.25 %**

Jeu d'entrainement KDD'99 10% sans mélange

Prédiction

Réel	Normal	Probe	DoS	U2R	R2L	% Correct
Normal	**59896**	242	436	2	17	98.84
Probe	91	**3359**	716	0	0	80.62
DoS	5293	937	**223623**	0	0	97.28
U2R	58	0	7	**2**	3	02.85
R2L	12386	78	109	0	**3774**	23.08
% Correct	77.06	72.76	99.43	50,00	99.47	PRÉCISION : **93.45 %**

Jeu d'entrainement KDD'99 10% avec mélange

Type	Connexion	Part
Normal	67 343	**53.46%**
Probe	11 656	**09.25%**
DoS	45 927	**36.45%**
U2R	52	**00.04%**
R2L	995	**00.79%**
Total	125 972	100 %

Répartition des connexions

	Connexion	TYPE
1	normal	NORMAL
2	normal	NORMAL
3	nmap	PROBE
4	normal	NORMAL
5	neptune	DOS
6	neptune	DOS
7	normal	NORMAL
8	neptune	DOS
9	normal	NORMAL
10	ipsweep	PROBE
11	guess_passwd	R2L
12	neptune	DOS
13	normal	NORMAL
14	neptune	DOS
15	neptune	DOS

Exemple d'une séquence de connexions

COMPOSITION DU JEU

Prédiction

Réel	Normal	Probe	DoS	U2R	R2L	% Correct
Normal	**9370**	200	140	0	1	96.48
Probe	248	**1722**	451	0	0	71.12
DoS	551	360	**6547**	0	0	87.78
U2R	62	0	5	**0**	0	00.00
R2L	2284	75	453	0	**75**	02.59
% Correct	88.96	95.30	90.33	00.00	50.18	PRÉCISION : **78.58 %**

Jeu d'entrainement NSL KDD'99 sans mélange

Prédiction

Réel	Normal	Probe	DoS	U2R	R2L	% Correct
Normal	**9260**	204	126	6	115	95.35
Probe	275	**1690**	452	0	4	69.80
DoS	800	374	**6280**	0	4	84.20
U2R	38	0	1	**15**	13	22.38
R2L	1327	45	664	2	**849**	29.40
% Correct	79.14	73.06	83.47	65.21	86.19	PRÉCISION : **80.26 %**

Jeu d'entrainement NSL KDD'99 avec mélange

Prédiction

Réel	Normal	Probe	DoS	U2R	R2L	% Correct
Normal	**60262**	243	78	4	6	99.5
Probe	511	**3471**	184	0	0	83.3
DoS	5299	1328	**223226**	0	0	97.1
U2R	168	20	0	**30**	10	13.2
R2L	14527	294	0	8	**1360**	08.4
% Correct	74,6	64.8	99.9	71.4	98.8	PRÉCISION : **92.71 %**

Résultats Gagnant KDD Cup '99

Prédiction

Réel	Normal	Probe	DoS	U2R	R2L	% Correct
Normal	**60182**	154	221	0	36	99.3
Probe	889	**2348**	928	0	1	56.4
DoS	723	195	**228935**	0	0	99.6
U2R	68	0	2	**0**	0	00.0
R2L	16229	9	81	0	**28**	00.2
% Correct	77,1	86.8	99.5	00.0	43.1	PRÉCISION : **93.72 %**

Résultats article Staudemeyer (4 params)

Prédiction

Réel	Normal	Probe	DoS	U2R	R2L	% Correct
Normal	**59868**	275	440	0	10	98.80
Probe	438	**3063**	665	0	0	73.52
DoS	5863	310	**223680**	0	0	97.31
U2R	69	0	1	**0**	0	00.0
R2L	15307	454	586	0	**0**	00.0
% Correct	73.41	74.67	99.24	00.0	00.0	PRÉCISION : **92.25 %**

Résultats modèle entrainée GRU (4 params)

Prédiction

Réel	Normal	Probe	DoS	U2R	R2L	% Correct
Normal	**59896**	242	436	2	17	98.84
Probe	91	**3359**	716	0	0	80.62
DoS	5293	937	**223623**	0	0	97.28
U2R	58	0	7	**2**	3	02.85
R2L	12386	78	109	0	**3774**	23.08
% Correct	77.06	72.76	99.43	50,00	99.47	PRÉCISION : **93.45 %**

Résultats modèle entrainée LSTM (4 params)

- **Les Réseaux de Neurones Récurrent sont-ils vraiment efficaces ?**

→ Seulement si le jeu comporte des séquences « mémorisables »

- **Que faire si ce n'est pas le cas ?**

→
- Méthode d'apprentissage spécifique au *jeu de données*
- \+
- Méthode d'apprentissage spécifique au *type de données*

} Laborieux

- **Faut-il harmoniser les jeux de données ?**

→ Beaucoup (trop?) de réseaux et de type d'attaques différents

---

La solution la plus pertinente ne serait-elle pas l'apprentissage d'un jeu de données *correspondant* au réseau cible de l'IDS ?

Machine Learning

- Principes de l'apprentissage
- Type de réseaux de neurones
- Création d'un modèle d'apprentissage

Initiation à la recherche

- Problématiques et fonctionnement du chercheur
- Recherches sur l'état de l'art
- L'esprit critique du chercheur

Détection d'intrusion

- Découverte des méthodes de détection d'intrusion
- Application de l'auto-apprentissage à la détection
- Enjeux de la détection d'intrusion sur un réseau

MERCI DE VOTRE ATTENTION !

Liste des références:

- Ralf. C Staudemeyer : *Applying long short-term memory recurrent neural networks to intrusion detection* – 2015
- Antonio Gulli, Sujit Pal : *Deep Learning with Keras* – 2017
- Rodolfo Bonnin : *Machine Learning for developers* – 2017
- François Chollet : *Deep Learning with Python* – 2017
- Arnaud Oglaza : *État de l'art sur l'apprentissage automatique*
- S. Hochreiter et J. Schmidhuber : Long Short-Term Memory – 1997
- R. Caruana, S. Lawrence, et C. Lee Giles : *Overfitting in Neural Nets: Backpropagation, Conjugate Gradient, and Early Stopping* – 2000
- K. Cho *et al :* Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation – 2014

- www.github.com/sylvainlapeyrade/RNN_Intrusion-Detection_Keras