

# Environnement logiciel pour l'apprentissage de l'exploration visuelle d'une image

Louis Cuni    Pablo Donato

3I013 – Introduction à la recherche

Université Pierre et Marie Curie

2 mai 2017

# Contexte

Objectif 1

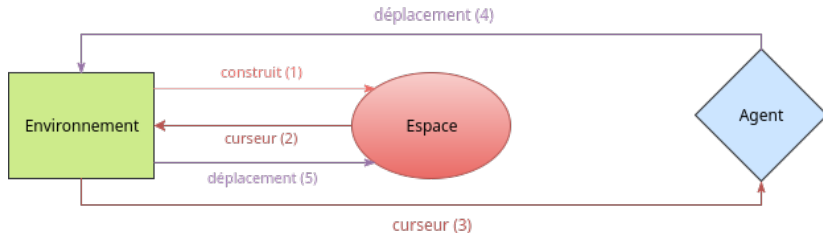
Environnement

Objectif 2

Agent



# Interaction agent-environnement



- 1 Construction de l'**espace**
- 2 Boucle d'interaction

# Sommaire

1 Environnement

2 Agent

3 Étude expérimentale

# Chargement de la grille

## MNIST

- images + labels
- training set + testing set

# Chargement de la grille

## MNIST

- images + labels
- training set + testing set

## Sous-ensemble de chiffres

- 1 Chargement des labels
- 2 Sélection des indices
- 3 Chargement des images
- 4 Sélection des labels

```
1  # (1)
2  self.labels = mnist_loader.load_idx_data(mnist_labels_path)
3
4  # (2)
5  num_examples = np.prod(size)
6  i = 0
7  labels_i = []
8  while len(labels_i) < num_examples:
9      if self.labels[i] in digits:
10         labels_i.append(i)
11         i += 1
12
13  # (3)
14  self.images = mnist_loader.load_idx_data(mnist_images_path, pos=labels_i)
15
16  # (4)
17  self.labels = self.labels[labels_i].reshape(size[:-1] + self.labels.shape[1:])
18  self.images = self.images.reshape(size[:-1] + self.images.shape[1:])
```

# Apprentissage par renforcement

## IA

- Classe d'algorithmes d'apprentissage
- Inspiré de la psychologie animale

# Apprentissage par renforcement

## IA

- Classe d'algorithmes d'apprentissage
- Inspiré de la psychologie animale

## Cadre formel : MDP

- 1 Ensemble d'états  $S$
- 2 Ensemble d'actions  $A$
- 3 Ensemble de récompenses  $R$
- 4 Fonction de probabilités  $p$

Espace	Attribut	Classe
$D = \{0, 1, \dots, 10\}$	digit_space	Discrete
$P = P_x \times P_y$	position_space	MultiDiscrete
$A = P \times D$	action_space	Tuple
$S = [0; 255]^{h \times w}$	observation_space	Box



# Espaces et wrappers

## Wrapper

- $\Leftrightarrow$  Design pattern *décorateur* pour `gym.Env`
- Conversion espaces d'actions/états (`ActionWrapper`/`ObservationWrapper`)

# Espaces et wrappers

## Wrapper

- $\Leftrightarrow$  Design pattern *décorateur* pour `gym.Env`
- Conversion espaces d'actions/états (`ActionWrapper`/`ObservationWrapper`)

## Pour `NumGrid`

- `DirectionWrapper`
- `DiscreteDirectionWrapper`
- `DiscretePositionWrapper`
- `DiscreteActionWrapper`

---

```
numgrid = NumGrid()  
numgrid = DiscreteDirectionWrapper(numgrid)  
numgrid = DiscretePositionWrapper(numgrid)  
numgrid = DiscreteActionWrapper(numgrid)
```

---

# Méthode step

## Fonction

- 1 pas de temps boucle d'interaction
- Fournit les informations à l'agent

# Méthode step

## Fonction

- 1 pas de temps boucle d'interaction
- Fournit les informations à l'agent

## Opérations

- 1 Vérification position curseur
- 2 Déplacement curseur
- 3 Choix récompense
- 4 Changement de zone
- 5 Vérification fin épisode
- 6 Retour des informations

---

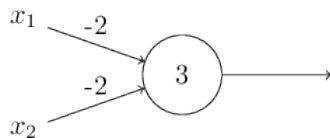
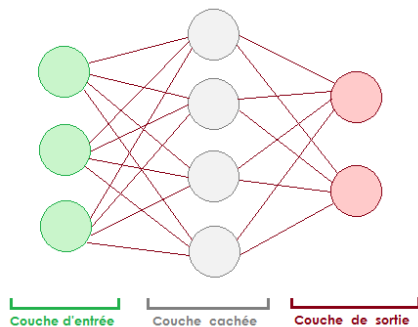
```

1  reward = 0
2  done = False
3  info = {'out_of_bounds': False, 'digit': self.current_digit}
4  # (1)
5  if not self.position_space.contains(pos):
6      info['out_of_bounds'] = True
7  # (2)
8  else:
9      self.cursor_pos = np.array(pos)
10 # (3)
11 if digit < 10:
12     if digit != info['digit']:
13         reward -= 3
14     else:
15         reward += 3
16     # (4)
17     self.cursor_pos = np.array(self.position_space.sample())
18 # (5)
19 self.steps += 1
20 if self.steps >= self.num_steps:
21     done = True
22 # (6)
23 return self.cursor, reward, done, info

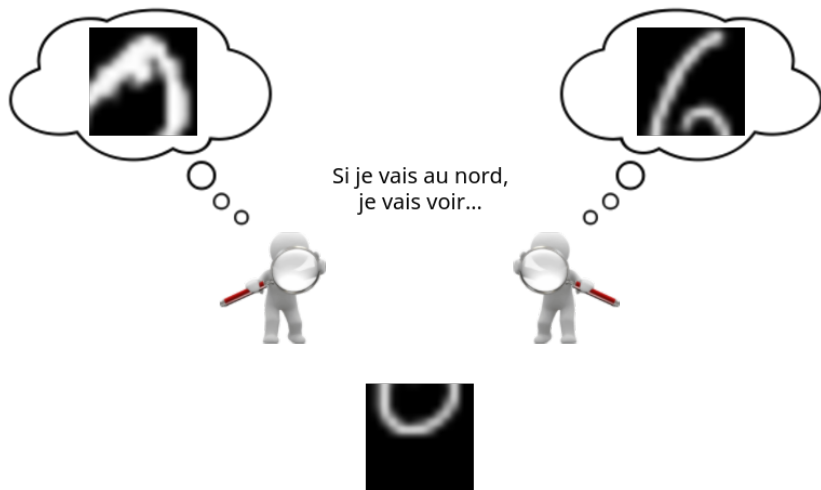
```

---

# Réseaux de neurones






# Prédicteurs



# Prise de décisions






# Prise de décisions




			$t$
0	0	0	0
0	0	0	






# Prise de décisions

			$t$
0	0	0	0
0	0	0	
0.81	0.86	0.83	1
0	0	0	




# Prise de décisions

			$t$
0	0	0	0
0	0	0	
0.81	0.86	0.83	1
0	1	0	




# Prise de décisions

			$t$
0	0	0	0
0	0	0	
0.81	0.86	0.83	1
0	1	0	
0.84	0.83	0.80	2
0	1	0	




# Prise de décisions

			$t$
0	0	0	0
0	0	0	
0.81	0.86	0.83	1
0	1	0	
0.84	0.83	0.80	2
1	1	0	




# Prise de décisions

			$t$
0	0	0	0
0	0	0	
0.81	0.86	0.83	1
0	1	0	
0.84	0.83	0.80	2
1	1	0	
0.82	0.89	0.85	3
1	1	0	




# Prise de décisions

			$t$
0	0	0	0
0	0	0	
0.81	0.86	0.83	1
0	1	0	
0.84	0.83	0.80	2
1	1	0	
0.82	0.89	0.85	3
1	2	0	

# Prise de décisions

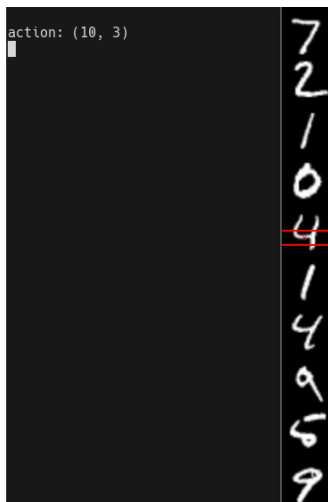
			$t$
0	0	0	0
0	0	0	
0.81	0.86	0.83	1
0	1	0	
0.84	0.83	0.80	2
1	1	0	
0.82	0.89	0.85	3
1	2	0	
0.83	0.91	0.86	4
1	2	0	

# Prise de décisions

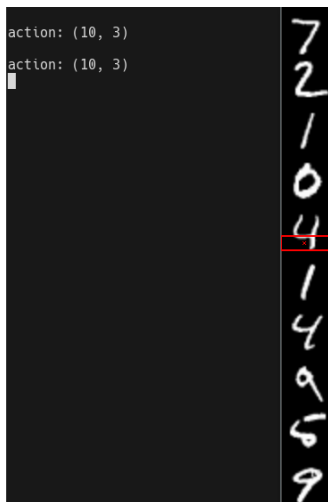
			$t$
0	0	0	0
0	0	0	
0.81	0.86	0.83	1
0	1	0	
0.84	0.83	0.80	2
1	1	0	
0.82	0.89	0.85	3
1	2	0	
0.83	0.91	0.86	4
1	3	0	



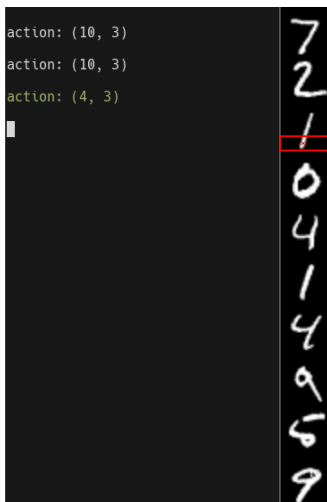
# Exemple d'une séquence d'identification



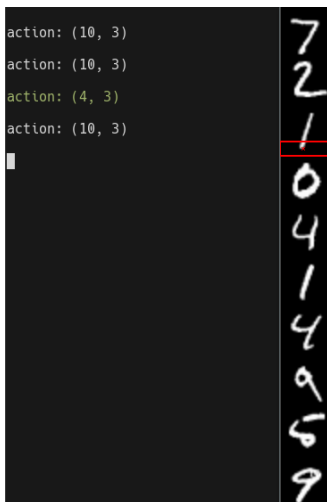
# Exemple d'une séquence d'identification



# Exemple d'une séquence d'identification

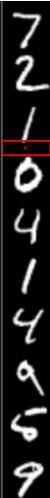


# Exemple d'une séquence d'identification



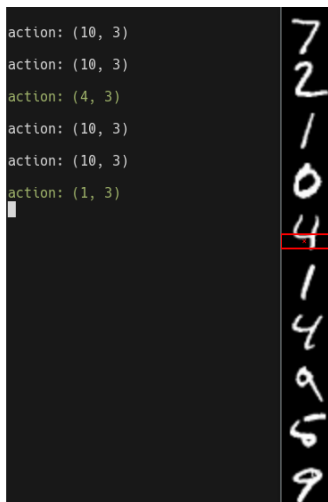
# Exemple d'une séquence d'identification

```
action: (10, 3)
action: (10, 3)
action: (4, 3)
action: (10, 3)
action: (10, 3)
-
```



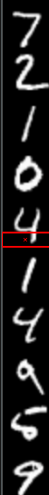
A vertical sequence of handwritten digits: 7, 2, 1, 0, 4, 1, 4, 9, 5, 9. The digit '0' is highlighted with a red box.

# Exemple d'une séquence d'identification



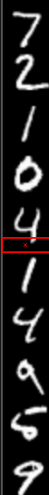
# Exemple d'une séquence d'identification

```
action: (10, 3)
action: (10, 3)
action: (4, 3)
action: (10, 3)
action: (10, 3)
action: (1, 3)
action: (10, 3)
```



# Exemple d'une séquence d'identification

```
action: (10, 3)
action: (10, 3)
action: (4, 3)
action: (10, 3)
action: (10, 3)
action: (1, 3)
action: (10, 3)
action: (10, 3)
```




A vertical strip of handwritten digits: 7, 2, 1, 0, 4, 1, 4, 9, 5, 9. The digit '1' at the 6th position is highlighted with a red box.



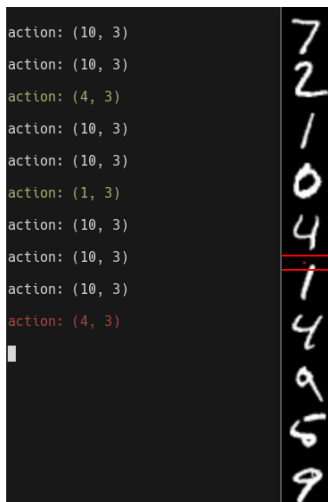
# Exemple d'une séquence d'identification

```
action: (10, 3)
action: (10, 3)
action: (4, 3)
action: (10, 3)
action: (10, 3)
action: (1, 3)
action: (10, 3)
action: (10, 3)
action: (10, 3)

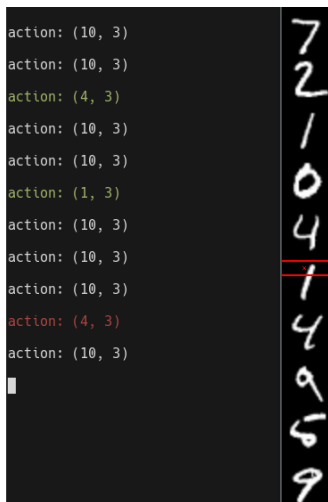
```



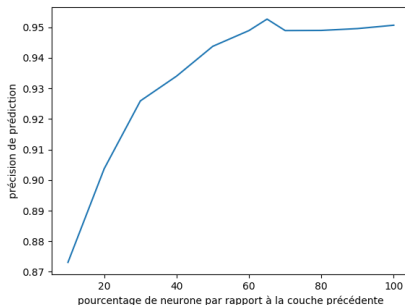
# Exemple d'une séquence d'identification



# Exemple d'une séquence d'identification

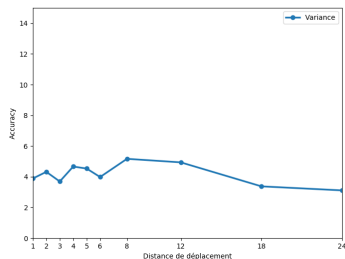
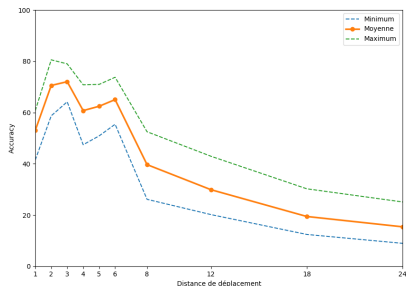


# Architecture des prédicteurs



Taille du curseur	Chiffre	Directions	Distance de déplacement
$8 \times 8$	0	{↓}	1 pixel

# Distance de déplacement



Taille de la grille	Chiffres	Taille du curseur	Directions	Seuil de décision
1 × 50	Tous	27 × 8	{↓}	5

# Conclusion

## Ce qui a été accompli

- Environnement opérationnel, performant, générique
- Agent honorable sur 1 direction (accuracy max. = **80.5%**)

## Ce qu'il reste à faire...

- Tests + poussés de l'environnement avec des algos d'AR
- Décision → accuracy satisfaisante sur 4 directions
- Algorithme d'AR pour optimiser les déplacements de l'agent

Merci pour votre attention !