

MORPION Q-LEARNING

Apprentissage par
renforcement

2026



SOMMAIRE

- 1. Contexte**
- 2. Q-Learning - Principe**
- 3. Architecture du projet**
- 4. Interface graphique**
- 5. Implémentations réalisées**
- 6. Difficultés rencontrées**
- 7. Résultats théoriques**
- 8. Résultats - Évaluation**
- 9. Axes d'amélioration**
- 10. Conclusion**



Contexte

- Jeu: Morpion (Tic-Tac-Toe) sur grille 3×3
- Objectif: Créer un agent intelligent via Q-Learning
- Cible: >85% de victoires contre adversaire aléatoire
- Méthode: Apprentissage par renforcement sans programmation explicite

1
2
3
4
5
6
7
8
9
10

tour !

2. Q-Learning - Principe

- **Équation:**

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max Q(s',a') - Q(s,a)]$$

Hyperparamètres

- $\alpha = 0.2$ (taux d'apprentissage)
- $\gamma = 0.95$ (discount factor)
- $\epsilon = 1.0 \rightarrow 0.05$ (exploration/exploitation)

- **Stratégie ϵ -greedy:**

- Avec probabilité ϵ : action aléatoire (exploration)
- Avec probabilité $1-\epsilon$: meilleure action (exploitation)



3. Architecture du projet

```
TP_Morpion_TP1/
├── images/      # Captures d'écran de l'app
│   ├── bg_wds.jpg # Image de fond
│   └── image1-6.png # Screenshots pour le rapport
├── main.py      # Point d'entrée
├── gui.py       # Interface Pygame
├── environment.py # Environnement de jeu
├── agent.py      # Agent Q-Learning
├── trainer.py    # Module d'entraînement
├── agent_trained.pkl # Agent sauvegardé
├── requirements.txt # Dépendances
└── README.md     # Ce fichier (rapport)
```

4. Interface graphique

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9

Choix des couleurs

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum eleifend tellus et iaculis iaculis. Cras ac nisl id est scelerisque pretium vel et dui. Sed ac arcu molestie, sollicitudin diam in, placerat quam. Vestibulum sodales risus eros, non tincidunt metus vulputate sit amet. Suspendisse non blandit tellus.

Choix de la police

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum eleifend tellus et iaculis iaculis. Cras ac nisl id est scelerisque pretium vel et dui. Sed ac arcu molestie, sollicitudin diam in, placerat quam. Vestibulum sodales risus eros, non tincidunt metus vulputate sit amet. Suspendisse non blandit tellus.



5. Implémentations réalisées



01

Environnement de jeu complet

02

Agent Q-Learning avec stratégie ϵ -greedy

03

Entraînement symétrique (50% X, 50% O)

04

•Interface graphique moderne (Pygame)

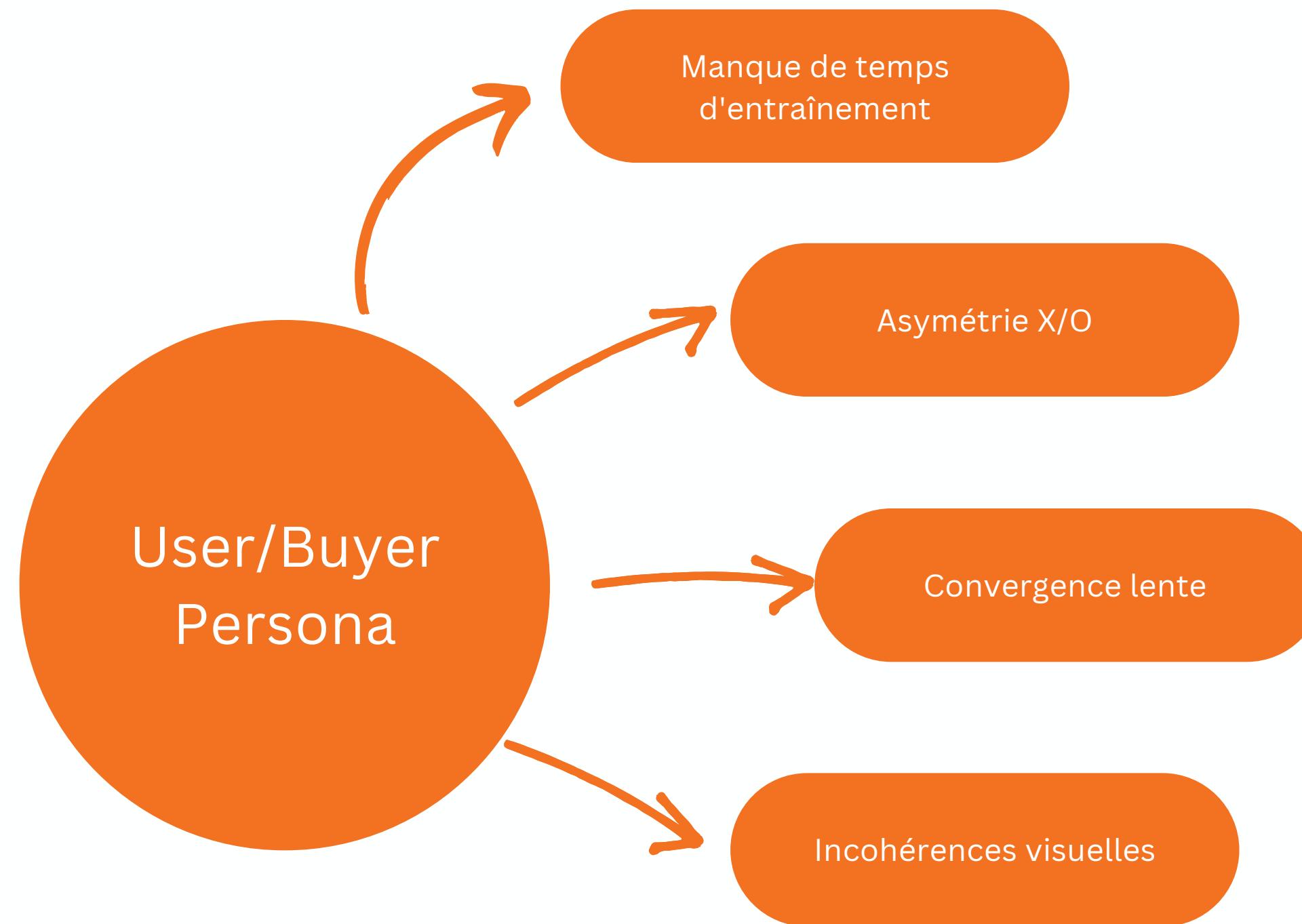
05

3 modes: Humain vs Humain, Humain vs IA, IA vs IA

06

Sauvegarde/chargement de l'agent
ET
Module d'évaluation quantitative

6. Difficultés rencontrées



7. Résultats théoriques

Initialisation...
Episodes: 100000 (X et O)
Paramètres: alpha=0.2, gamma=0.95

Episode 10000 | Win: 48.0% | eps: 0.6065
Episode 20000 | Win: 53.9% | eps: 0.3679
Episode 30000 | Win: 58.6% | eps: 0.2231
Episode 40000 | Win: 61.7% | eps: 0.1353

- ..Évolution avec entraînement complet:
 - .. 0 épisodes → ~50% (aléatoire)
 - .. 10 000 épisodes → ~75% (stratégies basiques)
 - .. 30 000 épisodes → 85-90% (stratégie mature)
 - .. 100 000 épisodes → 89-92% (quasi-optimal)
-
- Limitation actuelle:
- Agent non optimal par manque de temps d'entraînement



8. Axes d'amélioration



Q-Learning • Machine Learning

Humain vs Humain

Humain vs IA

IA vs IA

Entraîner l'agent

Évaluer l'agent

- 1. Plus d'entraînement
- 2. Deep Q-Learning
- 3. Adversaire intelligent
- 4. Grilles plus grandes (4×4 , 5×5)
- 5. Interface enrichie (animations, sons)

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9



- ✓ Q-Learning fonctionne pour le morpion
- ✓ Entraînement symétrique crucial (évite biais X/O)
- ✓ Interface complète et fonctionnelle



**THANK
YOU!**