Examen final
INF8225 IA : techniques probabilistes et d'apprentissage

Toute documentation est autorisée.
Veuillez travailler individuellement. Le professeur Pal répondra aux questions dans le chat du Zoom.

Question	Points
Q1	2
Q2	16
Q3	8
Total	26

Hiver 2021, le 1er mai 9h30 au 12h30.

Correction – Question 1 (H21 Final Exam)

a) Qu’est-ce qu’un «adversarial example» ?
→ Un exemple 'adversarial' est une entrée modifiée de manière subtile, souvent imperceptible à l'œil nu,
mais qui induit une mauvaise prédiction du modèle. Ces perturbations sont calculées pour tromper le modèle.

b) Qu’est-ce que «experience replay» ?
→ En apprentissage par renforcement, le 'experience replay' consiste à stocker les expériences (état, action, récompe
dans une mémoire, pour ensuite les réutiliser de manière aléatoire lors de l'entraînement. Cela brise la corrélation ent
et améliore la stabilité et la performance de l’apprentissage.

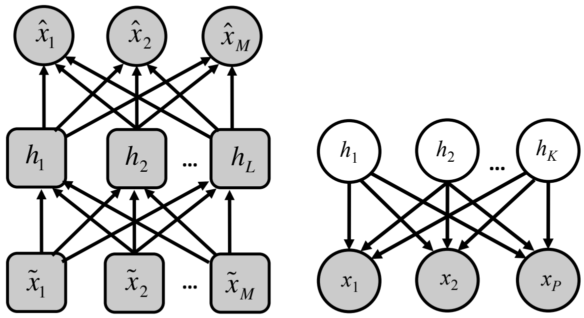

Figure 1 – **FR** (À gauche) Un exemple du type d'auto-encodeur discuté à la question 2a) ci-dessous. (À droite) Un réseau bayésien avec des variables aléatoires continues pour visualiser le modèle de «PPCA» - utilisé à la question 2b) ci-dessous. **EN** (Left) An example of the type of autoencoder discussed in question 2a) below. (Right) A Bayesian Network illustrating the model of PPCA with continuous random variables – used in question 2b) below.

Question 1 (Réponse courte - une ou deux phrases seulement. Short answer – one or two sentences only (2 points))

a) **FR** Qu'est-ce qu'un «adversarial example» ? **EN** What is an adversarial example ?

b) **FR** Qu'est-ce que «experience replay» ? **EN** What is "experience replay" ?

Question 2 (Version française - «AEs», «PCA», «VAEs», et «GANs» ? (16 points))

a) Donnez une équation pour la fonction de perte d'un auto-encodeur n'ayant qu'une couche d'entrée, une couche de code cachée et une couche de sortie. Vous avez N exemples de vecteurs continus de dimensionnalité M. Assurez-vous que cette équation montre comment la fonction de perte complète est composée à l'aide de matrices de poids, de biais et de fonctions d'activation.

b) Considérons le modèle de la figure 1 (à droite) ci-dessus, que nous avons utilisée pour décrire l'analyse probabiliste en composantes principales (PPCA). Qu'est-ce que la distribution utilisée pour $p(h_1)$ dans PPCA ? Donnez le modèle mathématique complet pour $P(\mathbf{x}, \mathbf{h})$. Quelle est la fonction de perte que nous aimerions optimiser étant donné $i = 1, \ldots, N$ exemples $\tilde{\mathbf{x}}_i$?

c) Qu'est-ce qu'un auto-encodeur variationnel et comment est-il lié : aux auto-encodeurs réguliers (a) et au PCA (b) ?

d) Qu'est-ce qu'un réseau contradictoire génératif (GAN) et comment peut-on apprendre un GAN ? Utilisez des équations et expliquez ce que chaque terme de votre équation signifie en mots.

Question 2 (English Version - AEs, PCA, VAEs & GANs. (16 points))

a) Give an equation for the loss function of an autoencoder having only an input layer, a hidden code layer and an output layer. You have N examples of M dimensional continuous vectors. Make sure this equation shows how the complete loss function is composed using weight matrices, biases and activation functions.

b) Consider the model in figure 1(right) above, which we have used to describe probabilistic principal component analysis (PPCA). What distribution is used for $p(h_1)$ in PPCA? Give the full mathematical model for $P(\mathbf{x}, \mathbf{h})$. What is the loss function that we would like to optimize given $i = 1, \ldots, N$ examples $\tilde{\mathbf{x}}_i$?

c) What is a variational autoencoder and how is it related to: regular autoencoders (a) and PCA (b)?
d) What is a generative adversarial network (GAN) and how does one train a GAN? Use equations and explain what each term of your equation means in words.

Correction – Question 2 (H21 Final Exam)

a) Perte d'un autoencodeur (AE) simple:
→ Données : $x_i \in \blacksquare^M$, $i = 1,...,N$
→ Encodeur : $h_i = \sigma(W\blacksquare\, x_i + b\blacksquare)$
→ Décodeur : $x\blacksquare_i = \sigma(W\blacksquare\, h_i + b\blacksquare)$
→ Fonction de perte : $L = (1/N) \sum ||x_i - x\blacksquare_i||^2$ (reconstruction MSE)

b) PPCA (Probabilistic PCA):
→ $p(h) = N(0, I)$,
→ $p(x|h) = N(Wh + \mu, \sigma^2 I)$,
→ Modèle complet : $p(x,h) = p(x|h)\, p(h)$
→ Fonction objectif : log-vraisemblance marginalisée sur h : $\log \sum p(x_i|h)p(h)$

c) VAE vs AE vs PCA :
→ VAE : encodeur donne une distribution $q(h|x)$, et la sortie est échantillonnée.
→ Perte VAE : reconstruction ($\log p(x|z)$) + régularisation ($KL(q(z|x) || p(z))$)
→ VAE ≈ AE + probabilistic latent space. PCA peut être vu comme une version linéaire et gaussienne du VAE.

d) GAN :
→ Deux réseaux : Générateur $G(z)$, Discriminateur $D(x)$
→ Minimax loss : $\min_G \max_D [E[\log D(x)] + E[\log(1 - D(G(z)))]]$
→ D apprend à distinguer vrai/faux, G apprend à tromper D.

Question 3 (Version française - processus de décision markovien (8 points))

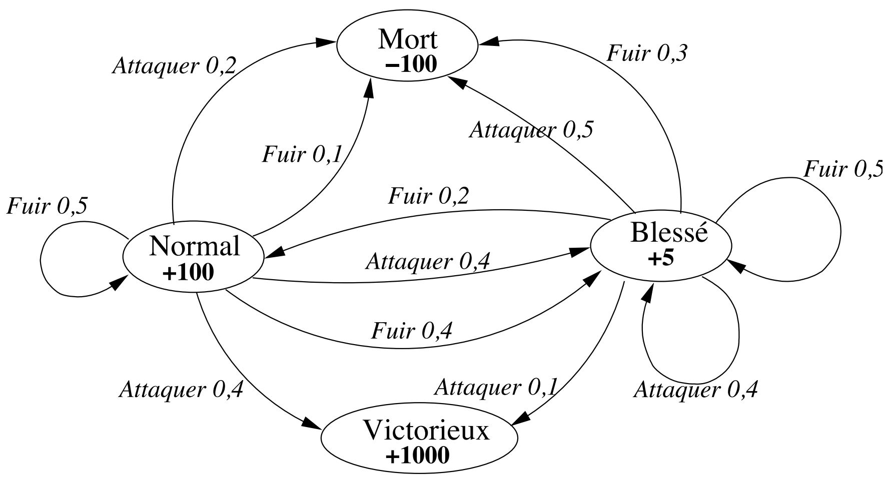

FIGURE 2 – Espace de transition markovien

Imaginez un héros de jeu vidéo qui doit affronter un adversaire. La figure 2 illustre cette situation comme un problème de décision markovien comprenant quatre états et deux actions possibles. Chaque noeud représente un état, et la récompense pour chaque état est indiquée dans le noeud (par exemple, la récompense est de 100 pour l'état *Normal*). Pour chaque action, on indique sa probabilité. Lorsqu'on se retrouve dans l'état *Mort* ou *Victorieux* l'agent passe de manière déterministe à l'état Normal, peu importe l'action prise.

a) (2 points) Donnez les deux matrices pour l'état suivant s' étant donné l'état courant s et l'action a, $P(s'|s, a)$. Donnez un tableau ou une matrice selon la structure des tableaux suivants.

a = Attaquer	s'			
s	M	B	N	V
Mort (M)				
Blessé (B)				
Normal (N)				
Victorieux (V)				

TABLE 1 – $P(s'|s, a = Attaquer)$

a = Fuir	s'			
s	M	B	N	V
Mort (M)				
Blessé (B)				
Normal (N)				
Victorieux (V)				

TABLE 2 – $P(s'|s, a = Fuir)$

b) (4 points) Lorsqu'on applique l'algorithme d'itération de la valeur, avec un coefficient $\gamma = 1$, et en utilisant un vecteur pour les états dans l'ordre $V_i(s) = \begin{bmatrix} V(M) & V(B) & V(N) & V(V) \end{bmatrix}^T$ on obtient les valeurs suivantes pour les deux premières itérations . C'est-à-dire, pour chaque action a_j où a_0 = Attaquer, et a_1 = Fuir, dans les deux premières itérations nous avons :

$$V_1^{a_0} = V_1^{a_1} = V_1 = \begin{bmatrix} -100 \\ 5 \\ 100 \\ 1000 \end{bmatrix},$$

Correction – Question 3 (H21 Final Exam)

Value Iteration (Itération 3) avec $\gamma = 1$
Récompenses : M=-100, B=+5, N=+100, V=+1000
V2 = [M=0, B=57, N=482, V=1100]

Action: Attaquer (a0)
V3(M) = 0.0*R + 0.1*62 + 0.4*582 + 0.5*1200 = 232.8
V3(B) = 0.0*R + 0.5*62 + 0.2*582 + 0.3*1200 = 253.8
V3(N) = 0.4*R + 0.4*62 + 0.0*582 + 0.2*1200 = -27.6
V3(V) = terminal → 1.0*R = 2195

Action: Fuir (a1)
V3(M) = ... = 58.2
V3(B) = ... = 195.6
V3(N) = ... = 195.2
V3(V) = terminal → 1.0*R = 2195

V3 = max(Va0, Va1) = [232.8, 253.8, 195.2, 2195.0]
Politique optimale = action qui maximise V par état.

$$V_2^{a_0} = \begin{bmatrix} 0 \\ 57 \\ 482 \\ 1100 \end{bmatrix}, V_2^{a_1} = \begin{bmatrix} 0 \\ -2.5 \\ 142 \\ 1100 \end{bmatrix} \rightarrow V_2 = \begin{bmatrix} \max(0,0) \\ \max(57,-2.5) \\ \max(482,142) \\ \max(1100,1100) \end{bmatrix} = \begin{bmatrix} 0 \\ 57 \\ 482 \\ 1100 \end{bmatrix}$$

Donnez les valeurs de $V_3^{a_0}$, $V_3^{a_1}$ et V_3 à la troisième itération de l'algorithme. Indiquez les étapes de votre travail.

c) (2 points) Comment obtiendriez-vous la politique de ce MDP ?

Question 3 (English Version - Markov Decision Processes (8 points))

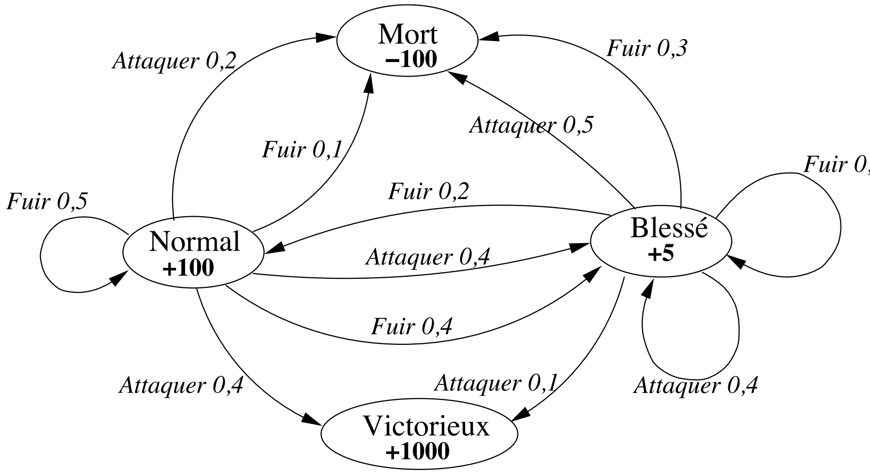

Figure 3 – Transition probabilities for a Markov Decision Process (MDP)

Imagine a video game hero who must face an opponent. Figure 3 illustrates this problem as a Markov decision process (MDP) including four states and two possible actions. Each node represents a state, and the reward for each state is indicated in the node (e.g., a reward of 100 is received for the state *Normal*). For each action the state transition diagram indicates the probability of transitioning to different states. When the agent is in state *Dead* (*Mort*) or *Victorious* (*Victorieux*) the agent deterministically transitions into the Normal state no matter which action is taken. The other state is *Injured* (*Blessé*) and the actions are either *Attack* (*Attaquer*) or *Flee/Escape* (*Fuir*).

a) (2 points) Give the two matrices for the next state s' given the current state s and the action a, $P(s'|s,a)$. Give a table or a matrix for your response using the following structures.

a = Attaquer	s'			
s	M	B	N	V
Mort (M)				
Blessé (B)				
Normal (N)				
Victorieux (V)				

Table 3 – $P(s'|s, a = Attaquer)$

a = Fuir	s'			
s	M	B	N	V
Mort (M)				
Blessé (B)				
Normal (N)				
Victorieux (V)				

Table 4 – $P(s'|s, a = Fuir)$

b) (4 points) When one applies the value iteration algorithm with a coefficient of $\gamma = 1$ and using a vector for the states in the order $V_i(s) = \begin{bmatrix} V(M) & V(B) & V(N) & V(V) \end{bmatrix}^T$, one will obtain the following values for the first two iterations. That is to say that for each action in the first two iterations we have:

$$V_1^{a_0} = V_1^{a_1} = V_1 = \begin{bmatrix} -100 \\ 5 \\ 100 \\ 1000 \end{bmatrix},$$

$$V_2^{a_0} = \begin{bmatrix} 0 \\ 57 \\ 482 \\ 1100 \end{bmatrix}, V_2^{a_1} = \begin{bmatrix} 0 \\ -2.5 \\ 142 \\ 1100 \end{bmatrix} \rightarrow V_2 = \begin{bmatrix} \max(0,0) \\ \max(57,-2.5) \\ \max(482,142) \\ \max(1100,1100) \end{bmatrix} = \begin{bmatrix} 0 \\ 57 \\ 482 \\ 1100 \end{bmatrix}$$

Give the values for $V_3^{a_0}$, $V_3^{a_1}$ et V_3 in the third iteration of the algorithm. Indicate the intermediate steps in your work.

c) (2 points) How would you obtain the policy for this MDP ?