Faculté des Arts et Sciences - Département d'Informatique et de Recherche opérationnelle

TITRE DU COURS:	**Intelligence artificielle : Introduction**	
SIGLE:	IFT3335	PROFESSEUR: Jian-Yun Nie
EXAMEN:	Final H2020	DATE: 29 avril 2020
LIEU:	en ligne	HEURES: 12:30-15:30

Directives pédagogiques: Toute documentation est autorisée. Remettre les réponses sur Studium avant la fermeture de la remise à **15:45**. Vos réponses peuvent être dans des fichiers en format de votre choix : texte, Word, PDF, ou photo (vous pouvez répondre sur des feuilles et prendre les photos).

Question 1. (20%) Généralité

1. Le test de Turing a été imaginé par Alain Turing pour tester si un système d'IA est intelligent. Par rapport aux 4 types d'approches à l'IA illustrées en bas, quel est le type d'approche que le test de Turing vise à tester? Donnez une courte explication pour votre réponse. Agir comme humain

Penser comme humain	Penser rationnellement
Agir comme humain	Agir rationnellement

2. Si on veut tester un autre type d'IA, par exemple, "penser comme humain", est-ce qu'on peut continuer à utiliser le test de Turing? Répondez à cette question pour tous les autres types d'approches. Expliquez pourquoi oui ou pourquoi non. non

3. Dans l'histoire de développement d'IA, surtout en traitement de langue naturelle, ELIZA a été un grand succès. Rappelons que ce système vise à simuler le comportement (dialogue) d'un psychanalyste avec un patient (utilisateur). Voici un segment de dialogue entre ELIZA et un utilisateur, qui ressemble à une conversation avec un vrai psychanalyste :

   young woman: Men are all alike.
   eliza: In what way?
   young woman: They're always bugging us about something specific or other.
   eliza: Can you think of a specific example?
   young woman: Well, my boyfriend made me come here.
   eliza: Your boyfriend made you come here?
   young woman: He says I'm depressed much of the time.
   eliza: I am sorry to hear you are depressed.
   young woman: It's true. I'm unhappy.
   eliza: Do you think coming here will help you not to be unhappy?

   Si l'utilisateur n'arrive pas à savoir que c'est un système qui a générer des réponses, est-ce que, selon vous, c'est une preuve que le système ELIZA a réussi à passer le test de Turing, et qu'on doit considérer que ELIZA est intelligent? Donnez une explication à votre réponse. Oui dans ce contexte et selon le test de Turing.

4. Le système ELIZA fonctionne d'une manière simpliste : Il détecte certains mots déclencheurs dans la phrase de l'utilisateur, et active un patron correspondant pour générer une réponse. Dans la plupart des cas, la réponse réutilise certains segments de la phrase de l'utilisateur, ou elle est générique, En sachant cela, est-ce que votre réponse est la même? Est-ce que votre réponse est en contradiction avec le test de Turing? Expliquez votre réponse. Ceci montre la limite du test de Turing – il ne peut pas tester la façon d'implanter l'intelligence.

5. Dans le temps moderne (aujourd'hui), on tente de créer un agent conversationnel en utilisant des réseaux de neurones profonds. Une approche typique – encoder-decoder – consiste à créer une représentation interne (un vecteur) pour la phrase d'utilisateur (la phase d'encodage), et génère une nouvelle phrase à partir de cette représentation (la phase de décodage). Les processus d'encodage et de décodage sont entraînés en utilisant beaucoup de pairs de vraies conversations. Sans aller en détails techniques, est-ce que selon vous, ce système basé sur des réseaux de neurones profonds utilise un principe radicalement différent de ELIZA? Si ce système est capable de bien simuler la conversation humaine, est-ce que votre réponse quant à l'intelligence du système sera différente que pour ELIZA? Expliquez votre réponse.

Un principe différent – apprendre au lieu d'être programmé.

Question 2. (10%) Recherche

Un algorithme de recherche général est Meilleur-d'abord (Best-first). Cet algorithme utilise la fonction f(n) = g(n) + h(n) pour évaluer chaque nœud n, et choisit le nœud qui a la meilleure valeur de f(n).

1. Considérez les cas particuliers suivants, et donnez le nom de l'algorithme correspondant :
   a. On définit h(n)=0; coût uniforme
   b. On définit g(n)=0; greedy
   c. On impose h(n) ≤ h*(n). A*
2. Comme h(n)=0 est un cas spécial de h(n) ≤ h*(n), pour garantir cette dernière propriété, on peut toujours définir h(n)=0. Est-ce que c'est une bonne stratégie? Expliquez pourquoi. Non. La complexité sera très grande.

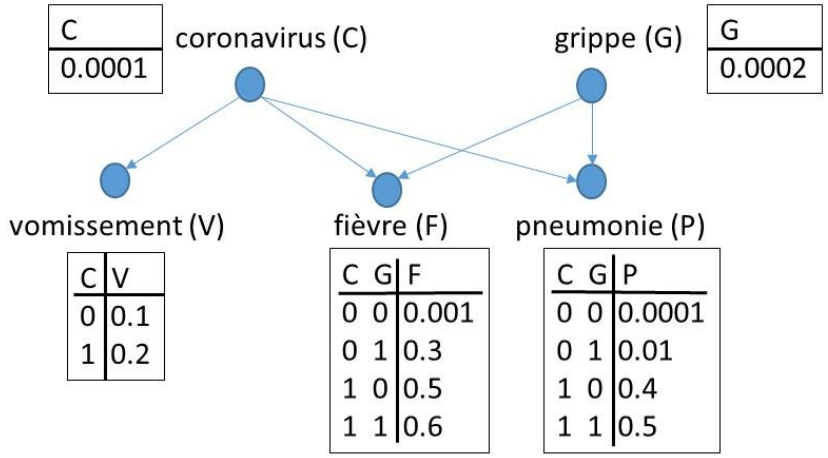
Question 3. (10%) Logique et inférence

1. Pourquoi s'intéresse-t-on à la logique en IA? Pourquoi veut-on faire du raisonnement en appliquant les règles d'inférence plutôt qu'utiliser une approche basée sur les modèles (model checking)?

   Problème de complexité pour model checking. On veut aussi développer un système qu'on peut généraliser dans tous les domaines.
2. La logique classique n'offre pas beaucoup de moyen pour traiter l'incertitude. Notamment, il est difficile d'associer une valeur d'incertitude (par exemple, une probabilité) à un raisonnement logique. Dans le système expert MYCIN, on propose une façon d'associer une valeur d'incertitude (appelée facteur de certitude) à un fait et à une règle. Cette approche a été abandonnée plus tard. Pourquoi ce n'est pas une bonne idée de faire comme dans MYCIN? Quelle solution pouvez-vous proposer à la place? Les déficiences théoriques dans MYCIN. La façon n'est pas rigoureuse. À la place : probabilité.

Question 4. (20%) Réseau bayésien

Considérez le réseau bayésien suivant où les variables C, G, V, F et P sont binaires. On a aussi les tables de probabilités conditionnelles associées à chaque nœud.



C	V
0	0.1
1	0.2

C	G	F
0	0	0.001
0	1	0.3
1	0	0.5
1	1	0.6

C	G	P
0	0	0.0001
0	1	0.01
1	0	0.4
1	1	0.5

On vous demande de calculer les probabilités suivantes (où chaque variable en minuscule signifie que la variable est vraie) :

a. P(c|v) (la probabilité d'avoir le coronavirus en observant le vomissement)
   =P(c,v)/P(v)=P(c) P(v|c) / (P(c) P(v|c) + P(-c) P(v|-c))
   = 0.0001*0.2 / (0.0001*0.2 + 0.9999*0.1)
b. P(f) (la probabilité qu'il y a fièvre)
   =somme_C,G P(f,C,G) = somme_C,G P(C) P(G) P(f|C,G)
   = 0.0001*0.0002*0.6 + 0.0001*0.9998*0.5 + 0.99998*0.0002*0.3 + 0.9999*0.9998*0.001
c. P(f|c) =P(f,c|c)=somme_G P(f,c,G)/P(c) = somme_G P(c)P(G)P(f|c,G)/P(c)
   =(0.0001*0.0002*0.6+0.0001*0.9998*0.5)/0.0001
d. P(f|c,g) =0.6

e. P(c|f,p) =P(c,f,p)/P(f,p)=somme_G (P(c,f,p,G) / somme_C,G P(f,p,C,G)
= somme_G (P(c)P(G)*P(f|c,G)* P(p|c,G)) / somme_C,G (P(C)P(G)*P(f|C,G)* P(p|C,G))

Pour ces questions, si vous n'avez pas le temps de faire le calcul final exactement, il vous suffit de dériver les formules jusqu'à mettre les chiffres dans les formules.

Question 5. (25%) Apprentissage

Dans une application pour aider à la décision, nous avons les cas d'exemples collectés comme dans le tableau gauche en bas. Il y a 16 cas. Chaque cas est décrit par 4 attributs A, B, C et D, qui ont des valeurs numériques. La dernière colonne contient la décision qu'on doit prendre (positive ou négative).

	A	B	C	D	Décision
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	1.2	positive
3	5	3.4	1.6	0.2	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	négative
10	6.4	3.2	4.7	1.5	négative
11	6.9	3.1	4.9	1.5	négative
12	5.5	2.3	4	1.3	négative
13	6.5	2.8	4.6	1.5	négative
14	5.7	2.8	4.5	1.3	négative
15	6.3	3.3	4.7	1.6	négative
16	4.9	2.4	3.3	1	négative

	A	B	C	D	Décision
1	0	1	0	0	positive
2	1	1	0	0	positive
3	1	1	0	0	positive
4	1	1	0	0	positive
5	1	1	0	0	positive
6	0	1	0	0	positive
7	0	1	0	0	positive
8	1	1	0	0	positive
9	1	1	1	1	négative
10	1	1	1	1	négative
11	1	1	1	1	négative
12	1	0	0	0	négative
13	1	0	1	1	négative
14	1	0	1	0	négative
15	1	1	1	1	négative
16	0	0	0	0	négative

1. Dans cette question, on vous demande d'abord à utiliser l'arbre de décision pour prendre la décision. Pour cela, on discrétise d'abord les valeurs des attributs selon le schéma suivant :

A	B	C	D	Valeur discrète
>= 5	>= 3.0	>= 4.2	>= 1.4	1
< 5	< 3.0	< 4.2	< 1.4	0

Cette discrétisation produit le tableau à droite. On utilise ces valeurs discrétisées pour construire l'arbre de décision. On vous demande de choisir l'attribut à utiliser au premier niveau selon le gain d'information. Déterminez cet attribut, en montrant vos calculs.

Distributtion des cas : 8 / 8. Entropie = -0.5 log 0.5 – 0.5 log o.5 = 1.

Distributions selon les attributs : nombre : #1 : (#pos, #neg) / #0 : (#pos, #neg)

A : 12: (5,7) /4: (3,1) B: 12: (8,4) / 4: (0,4)

C: 6: (0,6) / 10: (8,2) D: 5: (0,5) / 11: (8,3)

Remainder(A) = (12/16) (-5/12 log 5/12 – 7/12 log 7/12) + (4/16) (-3/4 log 3/4 – 1/4 log 1/4)

Remainder(B) = (12/16) (-8/12 log 8/12 – 4/12 log 4/12) + (4/16) (-0 log 0 – 4/4 log 4/4)

Remainder(C) = (6/16) (-0 log 0 – 6/6 log 6/6) + (10/16) (-8/10 log 8/10 – 2/10 log 2/10)

Remainder(D) = (5/16) (-0 log 0 – 5/5 log 5/5) + (11/16) (-8/11 log 8/11 – 3/11 log 3/11)

Choisir C car IG(C) = 1-remainder(C) est le plus grand.

2. Expliquez intuitivement la raison de choisir cet attribut plutôt qu'un autre attribut. Pourquoi en choisissant cet attribut, l'arbre construit nous aide dans la classification? L'attribut C permet de mieux séparer pos/neg.

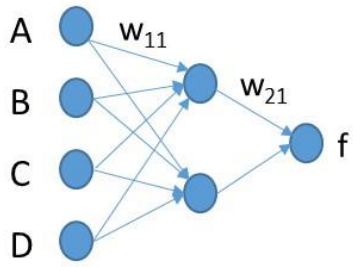
3. Si on utilise la classification bayésienne naïve (Naïve Bayes), on doit déterminer une séries de probabilités de base afin de pouvoir calculer P(Décision | A, B, C, D). Expliquez les probabilités de base qu'on doit avoir, et déterminez les valeurs de d'au moins 3 probabilités en utilisant le tableau des cas à droite.

   P(Pos) = 0.5, P(Neg) = 0.5

   P(A=1|Pos) =5/8 , P(A=0|Pos)=3/8, P(A=1|Neg)7/8, P(A=0|Neg)=1/8,

   …

4. On revient à notre tableau initial avec les valeurs en réelle (tableau gauche), et on veut utiliser un réseau de neurones pour faire la classification. L'architecture de ce réseau est comme suit :



   Les valeurs d'entrées sont celles des attributs A, B, C et D. Il y a 2 neurones cachés, et un neurone de sortie qui produit une valeur f. On utilise sigmoïde comme fonction d'activation pour les neurones cachés et le neurone de sortie. On désire produire f=1 pour les décisions positives et f=-1 pour les décisions négatives.

   Considérons l'entraînement en utilisant le premier cas (#1) dans le tableau à gauche. Supposons que f produite pour ce cas est f=0.2, mais la valeur désirée est 1. Expliquez comment on doit réviser les poids w_{21} et w_{11}. Vous devez montrer les formules pour leur calcul, sans toutefois calculer la valeur exacte.

   Cas considéré :

4.8	3.4	1.9	0.2

   Pour W21 (vers un neurone de sortie): W21 = W21 + alpha*Err*$g'(in)*x_j$,

   Où $g'(in)$ = g(in)(1- g(in)) pour sigmoïde. x_j est la valeur transmise par le lien W21, alpha = est le learning rate, Err=(1-0.2).

   Si on utilise la règle de Delta: Delta_i = Err * g'(in_i), W21 = W21 + alpha*Delta_i*x_j

   Pour W11 (vers un neurone cache):

   Delta_j = g'(in_j) * sum_i W_ji Delta_i

   Où W_ji est le poids W11 et celui vers un autre neurone caché

   Delta_i est le Delta d'un neurone cache (comme calculé dans la précédente étape). Il faut aussi calculer pour l'autre neurone caché.

   W11 = W11 + alpha *x_j * Delta_j, x_j est la valeur transmise à ce neurone d'entrée A (i.e. 4.8).

5. Considérons maintenant le cas d'apprentissage non supervisé. On suppose que nous ne savons les décisions dans la colonne de décision dans le tableau à gauche. On veut créer 2 clusters automatiquement selon les données. Comment peut-on faire? Est-ce qu'on peut s'attendre que les clusters créés correspondent aux classes positive et négative comme dans la colonne de décision? Sinon, à quoi peut servir de tels clusters?

   Par un algorithme de clustering. Par exemple k-means.

   On commence par deux seeds aléatoire (comme centroide initiaux).

   - Associer chaque cas au centroide le plus proche (distance euclidienne, par exemple)
   - Déterminer le centroide de chaque cluster,

- Recommencer les deux étapes.

On ne peut pas garantir que les clusters correspondent bien aux classes. Un algorithme de clustering permet de regrouper les données qui se ressemblent. Ça permet d'organiser les données automatiquement (peut être avant une étape de classification).

Question 6. (15%) Langue naturelle

1. Les traitements d'une langue naturelle comportent plusieurs niveaux, dont morphologique, syntaxique, et sémantique. Considérez la phrase suivante :

   Jean mange une pomme.

   On vous demande de donner le résultat de chaque niveau d'analyse pour cette phrase.
   Vous pouvez supposer que vous possédez un dictionnaire contenant toutes les informations nécessaires pour chaque mot, une grammaire et une représentation sémantique de base (que vous pouvez choisir selon votre besoin).
   Morphologique : Jean – Pronom mas., mange – verbe manger, sing., une – article fém, sing., pomme – nom, fém, sing.
   Syntacique : arbre syntaxique
   Sémantique : manger(Jean, pomme)
2. On peut utiliser un modèle statistique de langue pour déterminer si une phrase est probable dans cette langue. Supposons que les probabilités d'uni-gramme sont données comme suit :

Jean	0.0001
mange	0.0003
une	0.001
pomme	0.0002

   Déterminer la probabilité de la phrase « Jean mange une pomme » selon le modèle uni-gramme.
   = 0.0001 * 0.0003 * 0.001 * 0.0002
3. Quelle est la probabilité de « mange une Jean pomme » qui n'est pas correcte? Quelles limitations pouvez-vous observer sur ces exemples pour le modèle de langue uni-gramme? Comment peut-on développer un meilleur modèle statistique de langue? Expliquez votre solution.
   La probabilité est la même. Unigramme ne tient pas compte de l'ordre des mots. Un meilleur modèle est bigramme (n-gramme avec n >1).