Questions préparatoires
IA : techniques probabilistes et d'apprentissage

Question	Points
Q1	10
Q2	20
Q3	30
Total	60

Hiver 2024

Question 1 (Les énoncés vrai ou faux et des questions écrites (10 points))

Répondez True ou False pour les énoncés ci-dessous.

1) Si l'on adapte un modèle polynomial simple aux données pour faire des prévisions futures, la meilleure stratégie est toujours d'utiliser le polynôme le plus élevé possible.

2) Un perceptron à une seule couche est un modèle discriminatif pas génératif.

3) Le problème d'optimisation associé à l'apprentissage dans un réseau neuronal à une seule couche utilisant une fonction d'activation logistique est-il un problème convexe ?

4) Le problème d'optimisation associé à un perceptron multicouches avec une couche cachée est-il un problème convexe ?

5) Pourquoi de nombreux chercheurs et ingénieurs en apprentissage de machines ont-ils des problèmes convexes ?

Répondez avec quelques lignes de texte.

6) Expliquez pourquoi les chercheurs en apprentissage profond ont la tendance à utiliser la descente du gradient stochastique.

7) Quelles sont les motivations et les justifications théoriques pour l'utilisation du « dropout » ?

Question 1 (English version - True or False and Written Questions (10 points))

Answer either True or False for the statements below, explain your answer for each question, especially if the answer could changed depending on your perspective.

1) If one is fitting a simple polynomial model to data to make future predictions, the best strategy is always to use the highest order polynomial possible.

False. High order polynomials can overfit the training data. One could select the model complexity using a validation set.

2) A single layer perceptron is a discriminative not a generative model.

True. However one could use the empirical distribution of the input and then generate a label from the distribution given by the perceptron and create a strange kind of generative model.

3) Is the optimization problem associated with learning in a single layer neural network using a logistic activation function a convex problem ?

Yes. We have made note of this fact many times in class.

4) Is the optimization problem associated with a multilayer perceptron with one hidden layer a convex problem ?

No. This is partially what held back neural network research for many years because this was perceived as being a big problem.

5) Why do many machine learning researchers and engineers like convex problems ?

There is a single global minimum. Everyone using the model an a good optimizer will obtain the same solution.

Answer with a few lines of text.

6) Explain why deep learning researchers tend to use stochastic gradient descent.

The datasets are a so large that it is much better to make progress on a small sample of the data than it is to wait until the full gradient has been computed before making a gradient descent step.

7) What are the motivations and theoretical justifications for the use of dropout. It reduces neurons from becoming co-dependant and it can be viewed as a method for creating a pseudo-ensemble of models. It makes models more robust to situations where certain features are misssing. One could mask pixels on training images, but dropout masks the features derived from the pixels.

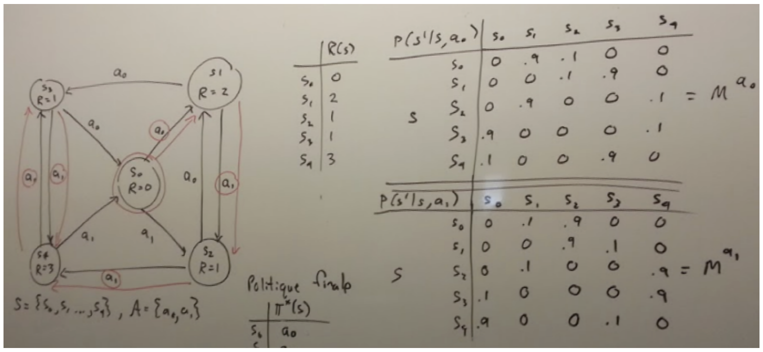


FIGURE 1 – Un problème de processus de décision de Markov présenté en classe. A Markov Decision Process problem presented in class.

Question 2 (version française) (20 points)

a) Étant donné le problème du processus de décision de Markov (MDP) présenté en classe et illustré dans la figure ci-dessus, montrez comment calculer la première itération de l'itération de la politique.

b) Expliquer la différence entre les méthodes basées sur un modèle et celles sans modèle dans l'apprentissage par renforcement.

c) Expliquer la formulation générale d'un processus de décision de Markov « MDP » et résumer les approches de : l'itération de la valeur, l'itération de la politique, et l'apprentissage Q.

d) Expliquez comment vous pourriez concevoir un système pour servir comme un agent d'intelligence artificielle dans un jeu vidéo du type « monde ouvert » à l'aide d'un processus de décision de Markov pour contrôler le personnage.

Question 2 (English version) (20 points)

a) Given the Markov Decision Process (MDP) problem presented in class and illustrated in the figure above, show how to compute the first iteration of policy iteration.

See the example presented on the board in class.

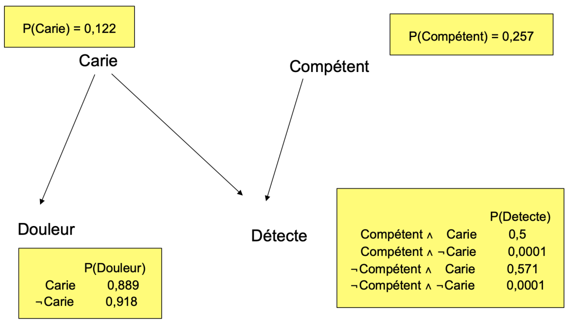
b) Explain the difference between model based and model free methods in reinforcement learning.

Model based RL uses an explicit model $P(s'|s,a)$ to implement key computations and perform learning, whereas model free methods don't.

c) Explain the general formulation of a Markov Decision Process and summarize the approaches of : value iteration, policy iteration and Q-learning.

MDPs : Defined on Slide 16, lesson 8.
Value Iteration : See slide 23, lesson 8.
Policy Iteration : See slide 38, lesson 8.
Q-learning : See slide 58, lesson 8.

d) Explain how you could design a system that is to behave as an artificially intelligent agent in an open world video game using a Markov Decision Process to control the character. One could use the same strategy as the Atari Game playing DeepMind work, where a Q function is parameterized by a convolutional neural network, and the action space is discrete. Q learning relies on an implicit MDP assumption. One could also formulate a tabular MDP with states and actions defined in a very simple discretized state space, e.g. for agent location, different types of observations, and actions.



Carie	Compétent	Douleur	Détecte	#
0	0	0	0	3
0	0	1	0	45
0	1	*	0	**17**
1	0	1	0	3
1	1	0	1	1
1	1	1	0	1
*	*	1	1	**30**
			Total	100

FIGURE 2 – (à gauche) Un réseau Bayésien avec les paramètres initiaux indiqués sur la figure. (à droite) Données à avec des valeurs manquantes indiquées par un *. (left) A Bayesian Network with initial parameters shown in the figure. (right) Data to with missing values indicated by a *.

Question 3 (version française) (30 points)

a) Expliquez à l'aide de mathématiques (et non de valeurs spécifiques) comment utiliser l'algorithme "Expectation Maximization (EM)" pour apprendre les paramètres du réseau bayésien à l'aide de données de la forme de la figure ci-dessus.

b) Comparez et contrastez PCA avec les auto-encodeurs, incluez une discussion sur les auto-encodeurs de débruitage.

c) Comment sont des matrices de Toeplitz liées aux réseaux convolutifs ? (Astuce : utiliser l'Internet, ceci est une question préparatoire donc l'utilisation de l'Internet est permis et encouragé.)

d) Comparer les RNNs simples avec les LSTMs et les GRUs (expliquer cela mathématiquement).

e) (12 points) Choisissez trois articles de recherche qui ont été présentés en classe et résumez-les dans vos propres mots en vous concentrant en particulier sur : a) la contribution technique apportée par l'article, et b) un résumé de la méthode, en utilisant les mathématiques pour apporter votre explication plus claire autant que possible.

Question 3 (English version) (30 points)

a) Explain using mathematics (not specific values), how use the "Expectation Maximization (EM)" Algorithm to learn the parameters of the Bayesian Network using data of the form in the figure above.

Start with a valid but random assignment of all conditional and unconditional probabilities. Perform an E-Step : Compute P(Pain | Cavity, Competent, Detection), Compute P(Cavity, Competent | Pain, Detection). Use the probabilities to probabilistically reweight the counts (see the example in the class notes). Perform M-Step : Update the probability tables using the standard maximum likelihood estimation equations and these weighted counts for the different configurations. Repeat the E and M steps until convergence.

b) Compare and contrast PCA with Autoencoders, include a discussion of denoising autoencoders.

PCA can be viewed as maximum likelihood estimation of a linear Gausssian continous hidden variable probablity model where we optimize the marginal probability of the observed data, i.e. $P(\mathbf{x})$, obtained by marginalizing over a hidden latent variable $\mathbf{z}$, from the joint distribution $P(\mathbf{x}|\mathbf{z})P(\mathbf{z})$. In contrast, autoencoders optimize $P(\tilde{\mathbf{x}}|\mathbf{x})$ using a neural network, typically with a lower dimensional bottleneck layer. Denoising autoencoders add noise to $\mathbf{x}$ (e.g. zero-ing out values in random dimensions of x, or adding Gaussian noise) and train the model to remove the noise.

c) How are Toeplitz matrices related to convolutional networks ? (Hint : Use the internet, this is a practice question where that is allowed and encouraged.)

Convolutions can be written as band diagonal matrices having the Toeplitz structure.

d) Compare vanilla RNNs with LSTMs and GRUs (using math).

RNNs update the hidden state of a neural network using a simple linear structure where :
$\mathbf{h}_t = \sigma_h\left(\mathbf{W}_h \mathbf{x}_t + \mathbf{U}_h \mathbf{h}_{t-1} + \mathbf{b}_h\right)$.
LSTMs add input, forget and output gates to manage controlled access of information input and extracted from a memory cell. This structure tends to allow gradient information to flow much more efficiently backwards in time when optimizing an RNN via gradient descent. A GRU is like a long short-term memory (LSTM) with a gating mechanism to input or forget certain features, but it lacks a context vector or output gate, resulting in fewer parameters than the LSTM.

e) (12 points) Pick three research articles that were presented in class and summarize them in your own words focus in particular on : a) the technical contribution that the article provided, and b) a summary of the method, using mathematics to make your explanation clearer whenever possible.

Example : The StyleGAN paper developed a way to inject randomness throughout the generative model, not just at the start. There is an injection of an affine mapping of a transformed version of the usual latent GAN $\mathbf{z}$. It is transformed through multiple fully connectd layers. It also used adaptive instance normalization at different layers in the generator and this was found to lead to controllable elements of style for the generated content. Adaptive Instance Norm is computed using : $\operatorname{AdaIN}\left(\mathbf{x}_i, \mathbf{y}\right) = \mathbf{y}_{s,i} \frac{\mathbf{x}_i - \mu(\mathbf{x}_i)}{\sigma(\mathbf{x}_i)} + \mathbf{y}_{b,i}$.