ARF Examen

Exercice 1 (4 points) – Questions indépendantes

Q 1.1 Résumer en quelques lignes les principes clés de l'algorithme SVM, son lien avec le perceptron et les avantages et les inconvénients.

Q 1.2 Quels sont les similarités et les différences entre le bagging et le boosting ?

Q 1.3 Pour estimer la probabilité d'un événement, est-il préférable d'utiliser un modèle appris par régression aux moindres carrés ou par régression logistique ?

Q 1.4 Soit un problème de décision, dans lequel on dispose de K machines à sous et le but est de choisir sur T pas de temps les machines à jouer de manière à maximiser les gains collectés. On suppose que chaque machine $i \in \{1, .., K\}$ possède une distribution de gains stationnaire de moyenne μ_i et que les tirages successifs sont indépendants.

Q 1.4.1 Représenter le MDP correspondant (schéma états + transitions du problème)

Q 1.4.2 Soit une politique qui tire sur les K premiers pas de temps une fois chaque machine (au temps $t = K$, toutes les machines ont été tirées une fois), puis qui par la suite se concentre sur la machine qui a permis le meilleur gain sur ce tirage d'initialisation pour les $T - K$ pas de temlps restants. Que pensez-vous de cette politique ? Peut-on proposer une meilleure politique ?

Exercice 2 (4 points) – t-SNE

L'algorithme t-SNE est une algorithme pour la visualisation de données en très grande dimension : à partir d'un ensemble de données $\{\mathbf{x}^1, \ldots, \mathbf{x}^n\} \in X$ décrites dans un espace $\mathcal{X} \subset \mathbb{R}^d$, il s'agit de trouver une projection 2D ou 3D des points dans un espace $\mathcal{Y} \subset \mathbb{R}^{d'}$ en préservant au mieux les structures spatiales locales de l'espace d'origine. Pour modéliser ces structures et pouvoir comparer les deux espaces, t-SNE utilise une modélisation probabiliste des voisinages des points en faisant comme hypothèse que cette distribution doit être très semblable dans $\mathcal{X}$ et dans le nouvel espace $\mathcal{Y}$. L'algorithme tente ainsi d'attribuer à chaque point $\mathbf{x}^i$ de X un point $\mathbf{y}^i$ dans $\mathcal{Y}$ qui permet de minimiser la distance entre les deux distributions.

Q 2.1 Considérons un point $\mathbf{x}^i$ des données et une similarité gaussienne $s(\mathbf{x}^i, \mathbf{x}) = Ke^{-\|\mathbf{x}^i - \mathbf{x}\|/\sigma_i^2}$. Soit $p_{j|i}$ la distribution de probabilité définie sur les points $\{\mathbf{x}^j,\ j \neq i\}$ telle que la probabilité soit proportionnelle à $s(\mathbf{x}^i, \mathbf{x}^j)$: un point distant de $\mathbf{x}^i$ aura une faible probabilité d'être choisie. Donner l'expression de $p_{j|i}$ (on considérera $p_{i|i} = 0$). Nous noterons par la suite $P_i = (p_{1|i}, \ldots, p_{n|i})$ la distribution du point de vue du point $\mathbf{x}^i$.

Q 2.2 De manière analogue, on définit les probabilités $q_{i|j}$ dans l'espace de projection $\mathcal{Y}$ à partir des coordonnées projetées $\{y^1, \ldots, y^n\}$ et $Q_i = (q_{1|i}, \ldots, q_{n|i})$. Afin de comparer les distributions dans les deux espaces, on utilise la divergence de Kullback-Leiber (KL) : soit P et Q deux distributions de probabilité avec les mêmes modalités $(p_1, \ldots, p_n)$ et $(q_1, \ldots, q_n)$, la KL divergence est définie par $KL(P||Q) = \sum_{i=1}^{n} p_i log\frac{p_i}{q_i}$.

Q 2.2.1 Montrer que cette distance est bien positive. Quand est ce que $KL(P||Q) = 0$? Est-ce une vraie distance ?

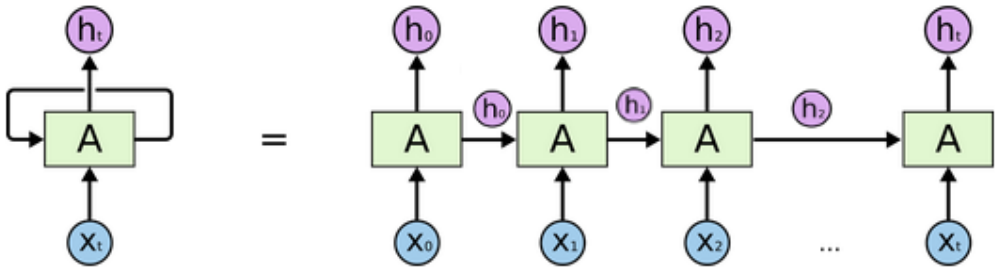
Q 2.2.2 Donner l'expression de $KL(P_i||Q_i)$.

Q 2.3 L'objectif de t-SNE est de trouver les coordonnées Y en minimisant la distance entre les

distributions. Proposer un coût à optimiser et un algorithme de résolution en détaillant les calculs. Donner le code python associé.

Exercice 3 (6 points) – **Réseaux de neurones récurrents**

Les réseaux de neurones récurrents (RNN) sont des réseaux de neurones adaptés au traitement de séquences, dans lesquels un même module neuronal est répété en boucle tout au long de la chaîne de traitement. La figure suivante présente à gauche une représentation compacte d'un RNN avec un module A, puis sur la droite sa version "dépliée" sur une chaîne de $t+1$ entrées.



Soit une séquence de T observations $\mathbf{x} \in \mathbb{R}^{d \times T}$, dont l'observation $\mathbf{x}_t$ à chaque pas de temps est un vecteur colonne de taille d. Au début du processus d'inférence, l'entrée $\mathbf{x}_0$ est présentée au module et une sortie $\mathbf{h}_0 \in \mathbb{R}^l$ est calculée. Cette sortie de dimension l représente une *mémoire* du module en charge d'accumuler les informations utiles à chaque pas. Elle est calculée en effectuant une transformation linéaire de l'entrée avec une matrice de poids W suivie d'une fonction d'activation *tanh* appliquée à l'ensemble des sorties de la transformation linéaire. Pour la suite du processus d'inférence, à chaque pas de temps t de la séquence, la sortie $\mathbf{h}_{t-1}$ de l'étape précédente $t-1$ est concaténée à la nouvelle observation $\mathbf{x}_t$ pour former l'entrée du module : nous noterons ce vecteur concaténé $\mathbf{z}_t = \begin{pmatrix} \mathbf{h}_{t-1} \\ \mathbf{x}_t \end{pmatrix} \in \mathbb{R}^{d+l}$ (nous noterons également $\mathbf{z}_0$ le vecteur d'entrée au début du processus résultant de la concaténation d'un vecteur colonne de l zéros et $\mathbf{x}_0$).
Notons que, dans le modèle considéré, la sortie $\mathbf{h}_t$ au pas de temps t sert donc non seulement à former l'entrée du module au pas de temps t+1, mais également à prédire un évènement dont on connaît en apprentissage la sortie attendue y_t et pour laquelle on peut alors calculer un coût $L_t = \Delta(\phi(\mathbf{h}_t), y_t)$, avec $\phi(\mathbf{h}_t)$ une fonction de décodage de $\mathbf{h}_t$. Nous ne considérerons pas dans cet exercice le processus de décodage de la mémoire (ou de l'espace latent) qui permet à partir de $\mathbf{h}_t$ de prédire la valeur de sortie attendue de la séquence, nous supposerons uniquement que la dérivée de l'erreur au temps t est connue (cf question 3.6).

Q 3.1 Quel est le nombre de neurones dans la couche cachée du réseau ? Combien a-t-on de paramètres W à apprendre pour le traitement de séquences de taille T ?

Q 3.2 Donner la fonction $f : \mathbb{R}^{l+d} \to \mathbb{R}^l$ permettant de passer de l'entrée $\mathbf{z}_t$ à la sortie $\mathbf{h}_t$.

Q 3.3 Sachant que $\dfrac{\partial \tanh x}{\partial x} = 1 - \tanh(x)^2$, donner le gradient de $f(z)$ par rapport à W, avec $\mathbf{z}$ un vecteur colonne de taille $l + d$.

Q 3.4 Donner maintenant le gradient de la sortie d'un module $\mathbf{h}_t$ par rapport à son entrée $\mathbf{z}_t$.

Q 3.5 En déduire la dérivée de $\mathbf{h}_t$ par rapport à $\mathbf{h}_{t-1}$.

Q 3.6 Connaissant $\delta(\mathbf{h}_t)$ la dérivée de l'erreur L_t au temps t par rapport à sa sortie correspondante $\mathbf{h}_t$: $\delta(\mathbf{h}_t) = \dfrac{\partial L_t}{\partial \mathbf{h}_t}$, donner la dérivée du coût L_t par rapport à $\mathbf{h}_{t-1}$ pour tout $t > 0$.

Q 3.7 En déduire la dérivée de l'erreur au temps t en fonction de la sortie au temps $s \leq t$.

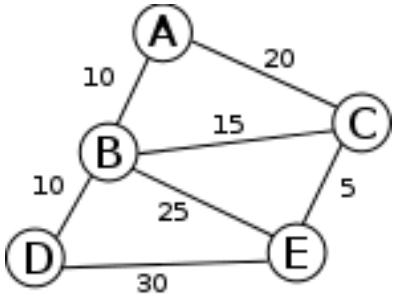
Q 3.8 Donner enfin la dérivée de l'erreur L_t en fonction de W

Q 3.9 En déduire la dérivée de l'erreur $L = \sum_{t \in \{0,..,T-1\}} L_t$ en fonction de W

Q 3.10 En déduire un algorithme de descente de gradient sur un ensemble de séquences X.

Exercice 4 (6 points) – Apprentissage par renforcement

On considère le réseau routier suivant, où les cerles représentent des villes et les arcs des routes associées à des temps de trajet moyens (en minutes) :



On considère que l'on part de la ville A et que l'on souhaite rejoindre la ville E au plus vite.

Q 4.1 L'ensemble des états S du problème correspondent aux villes numérotées de A à E de notre réseau routier. Quelles sont les différentes actions $a \in A$ possibles ? Si l'on considère des transitions déterministes, donner les valeurs prises par la fonction de transition $t : S \times A \times S \to [0, 1]$ et par la fonction de récompense $r : S \times A \to \mathbb{R}$ pour modéliser notre problème de transport routier sous la forme d'un MDP.

Q 4.2 Rappeler à quoi correspond V la fonction valeur des états selon une politique π (détailler les différentes composantes) : $V(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} t(s, \pi(s), s')V(s')$

Q 4.3 Quelle sont les différences entre les algorithmes Policy Iteration et Value Iteration ?

Q 4.4 Faire tourner manuellement l'algorithme Value Iteration avec $\gamma = 1$ jusqu'à convergence (en partant de valeurs initiales $V = (0, 0, 0, 0, 0)$). L'algorithme converge-t-il vers une solution satisfaisante ?

Q 4.5 À quoi sert γ ? Quel est le risque avec $\gamma = 1$? Et qu'obtient-on avec $\gamma = 0$?

Q 4.6 Considérons maintenant le cas suivant : Selon des problèmes de trafic qui surviennent fréquemment sur la route séparant la ville A de la ville C, on sait qu'une fois qu'on l'emprunte sur 3, on est obligé de rebrousser chemin jusqu'à la ville de départ (A ou C), ce qui nous fait perdre 20 minutes pour rien. Dire ce que cela implique pour la fonction de récompense de notre MDP.

Q 4.7 Donner une nouvelle formulation de V incluant alors cette nouvelle définition de la fonction de récompense, et montrer que l'on peut se ramener facilement à la formulation classique donnée ci-dessus. En déduire les nouvelles valeurs prises par la fonction de récompense $r(s, a)$. Donner également les nouvelles valeurs pour la fonction de transition $t(s, a, s')$.

Q 4.8 Refaire tourner votre algorithme pour observer les différences de comportement avec l'instance initiale du problème (en partant des même valeurs d'initialisation).

Q 4.9 On considère maintenant que lorsque l'on passe par B, on risque d'avoir une avarie (proba 1/3), qui fait que l'on roule ensuite au ralenti et double tous les temps de parcours ultérieurs. Donner la nouvelle représentation du MDP correspondant (uniquement le schéma reliant les différents états possibles avec explication de l'idée générale, pas indispensable d'y inscrire les probabilités de transition ni les nouvelles récompenses).