Apprentissage par renforcement profond pour la classification précoce de séquences temporelles

Coralie MARTINEZ[1], Emmanuel RAMASSO[2], Guillaume PERRIN[1], Michèle ROMBAUT[3]

[1]bioMérieux, Marcy l'Etoile, France

[2]FEMTO-ST Institute, Univ. Bourgogne Franche-Comté, Besançon, France

[3]Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France

martinezcoralie.mc@gmail.com, emmanuel.ramasso@univ-fcomte.fr
guillaume.perrin@biomerieux.com, Michele.Rombaut@gipsa-lab.grenoble-inp.fr

Résumé – Dans cet article, nous traitons du problème de classification précoce de séquences temporelles. Il s'agit d'un problème de prise de décision séquentielle où l'objectif est de minimiser le temps de prédiction. Nous modélisons le problème par un Processus de Décision Markovien Partiellement Observable (POMDP). Nous proposons d'utiliser une méthode d'apprentissage par renforcement afin d'entraîner un agent à décider entre les actions de classer la séquence incomplète ou d'attendre une observation supplémentaire. Nous utilisons un algorithme existant qui approxime la politique de l'agent par un réseau de neurone profond. Nous avons adapté l'algorithme afin de pouvoir faire un apprentissage soit sur une base d'entraînement fixe, soit en ligne avec l'acquisition dynamique de nouvelles données d'entraînement. Nous proposons aussi un échantillonnage et un stockage hiérarichisé par action et par classe ainsi qu'une initialisation spécifique des épisodes d'entraînement afin de pallier le déséquilibre de la mémoire de l'agent.

Abstract – In this article, we address the problem of early classification on temporal sequences. We frame early prediction as a sequential decision making problem and we define a Partially Observable Markov Decision Process (POMDP). We solve the POMDP by training an agent for early prediction with reinforcement learning. The agent learns to make adaptive decisions between classifying incomplete sequences now or delaying its prediction to gather more data points. We adapt an existing algorithm for batch and online learning. We propose prioritized sampling, prioritized storing and a specific episode initialization to address the fact that the agent's memory is unbalanced.

1 Introduction

Il existe de nombreuses applications où des séquences temporelles sont acquises de manière dynamique, c'est-à-dire qu'elles peuvent être complétées avec de nouvelles mesures au cours du temps. Lorsque les mesures sont coûteuses ou lorsqu'il est essentiel d'agir le plus tôt possible, la classification précoce des séquences temporelles est d'une importance primordiale. Il s'agit donc de trouver un compromis entre la qualité de la classification et le temps de prédiction le plus court.

Dans la littérature, la classification sur d'autres types de données dynamiques a été proposée par plusieurs auteurs. Dans [1], un Processus de Décision de Markov (MDP) est utilisé pour le problème de la classification de texte où il n'est pas toujours nécessaire de lire un document entier pour en classer le contenu. Le problème de l'acquisition d'information dans le domaine médical a été abordé par [2, 3] afin d'optimiser le compromis entre la précision et coût d'accès aux informations pour des problèmes de classification.

Dans [4], nous avons proposé une approche DRL (Deep Reinforcement Learning) utilisant l'algorithme Deep-Q-Network (DQN) [5] pour résoudre le problème de classification précoce à plusieurs classes. L'optimisation simultanée à la fois de la qualité et de la précocité de la classification repose sur un compromis spécifié par l'utilisateur à travers les récompenses de l'agent. La méthode [4] proposée rencontre cependant un certain nombre de problèmes. La mémoire de l'agent (base d'apprentissage) est déséquilibrée. En effet, après chaque acquisition d'une nouvelle mesure, l'agent choisit entre attendre des données supplémentaires ou classer la séquence incomplète. Les actions de classer entraînent la fin du processus d'acquisition et l'action d'attente est surreprésentée. De plus, la fin de la séquence est rarement atteinte et les débuts de séquences sont également surreprésentés.

Dans cet article, nous apportons les contributions suivantes :

(1) Nous définissons le processus de prédiction précoce comme un POMDP répondant aux deux objectifs compétitifs de précocité et de qualité de classification.

(2) Le POMDP est résolu par un apprentissage de la politique de l'agent par renforcement, en ligne ou sur une base d'entraînement fixe. La politique est apprise à l'aide de l'algorithme Double Deep-Q-Network (DDQN) de [5] que nous avons adapté pour résoudre le problème de mémoire non équilibrée : nous utilisons un échantillonnage et un stockage par action et par classe lors de l'entraînement et nous redéfinissons l'initialisation des épisodes d'entraînement.

(3) Nous montrons expérimentalement que les réseaux de neurones profonds "naïfs" statiques formés pour la classification à chaque instant sont moins efficaces en termes de précision / vitesse par rapport à la solution proposée.

2 Apprentissage par renforcement profond

L'apprentissage par renforcement est basé sur l'interaction d'un agent dans un environnement inconnu. La décision se fait par un processus d'essais et d'erreurs. Dans chaque état s de l'espace d'état $\mathcal{S}$, l'agent sélectionne une action a dans l'ensemble des actions possibles $\mathcal{A}$. Le choix de l'action a est dicté par sa politique π telle que $a = \pi(s)$. En réponse, l'agent reçoit une récompense $r = R(s, a)$ et passe à l'état suivant $s' = T(s, a)$. Les interactions $< s, a, r, s' >$ entre l'agent et l'environnement se poursuivent jusqu'à ce que l'agent atteigne un état terminal menant à la fin de l'épisode.

A tout instant $t \in \mathbb{N}^+$, l'agent cherche à choisir les actions générant le rendement maximal défini comme la somme des futures récompenses $\sum_{k=0}^{\infty} \gamma^k r_{t+k}$. Le coefficient $\gamma \in [0, 1]$ est configuré de façon à privilégier les récompenses immédiates plutôt que les récompenses futures. La politique optimale π^* conduit au rendement maximal.

La valeur d'action Q d'un état $s \in \mathcal{S}$ subordonnée à une action a est définie comme le retour que l'agent peut espérer obtenir en sélectionnant l'action a dans l'état s en suivant sa politique π.

$$Q_\pi(s, a) = \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t = s, a_t = a]$$

La fonction de valeur d'action optimale définie par $Q_*(s, a) = \max_\pi Q_\pi(s, a)$ permet de définir la politique optimale $\pi_*(s) = \arg\max_a Q_*(s, a)$.

Dans [5], les auteurs approximent la fonction de valeur d'action optimale Q_* par un réseau neuronal profond $Q(s, a, \Theta)$ avec les paramètres Θ en minimisant la fonction de perte :

$$L(\Theta) = (r + \gamma Q(s', \arg\max_a Q(s', a, \Theta), \Theta^-) - Q(s, a, \Theta))^2$$

L'optimisation du réseau neuronal profond est réalisée par une descente de gradient sur un ensemble d'interactions $< s, a, r, s' >$ préalablement placées dans la mémoire de l'agent au cours des épisodes d'entraînement.

3 Classification de séquences temporelles

Soit $X = (x_1, ..., x_T) \in \mathbb{R}^{p \times T}$ une séquence temporelle de longueur maximale $T \in \mathbb{N}^+$ où x_i est un vecteur de $p \in \mathbb{N}^+$ caractéristiques au pas de temps $i \in [1, T]$. Une séquence temporelle partielle est définie par $X_{:t} = (x_1, ..., x_t) \in \mathbb{R}^{p \times t}$ avec $t \leq T$. On dispose d'un ensemble de données d'apprentissage $\mathcal{D} = \{(X^j, l^j)\}_{j=1..n}$ avec n des paires de séquences temporelles complètes X et leur étiquette de classe $l \in \mathcal{L}$, avec $\mathcal{L}$ l'ensemble des étiquettes.

Classifieur précoce Dans les applications réelles, les observations $x_i \in \mathbb{R}^p$ à chaque pas de temps $i \in [1, T]$ sont obtenues séquentiellement. L'objectif n'est pas de prédire l'instant optimal $t^* \in [1, T]$ pour la classification, mais de choisir en ligne, à chaque pas de temps t, d'effectuer la classification sur la séquence partielle $X_{:t}$ ou d'attendre afin d'obtenir des observations supplémentaires.

Nous représentons le problème de prise de décision séquentielle par un POMDP défini par le tuple $\{\mathcal{S}, \mathcal{A}, T, R, \mathcal{O}, \gamma\}$ où $\mathcal{S}$ est l'espace d'état, $\mathcal{A}$ est l'espace d'action, T est le modèle de transition, R est la fonction de récompense, $\mathcal{O}$ est l'espace d'observation et γ est le facteur d'atténuation.

L'état $s \in \mathcal{S}$ est caractérisé par le tuple $s = (X, l, t)$ avec $(X, l) \in \mathcal{D}$ la séquence temporelle X et son étiquette associée l, et avec $t \in [1, T]$ le nombre de pas de temps observés dans la séquence. On définit l'observation $o = X_{:t}$ de l'état $s = (X, l, t)$ la séquence partielle de X où les x_i sont collectés jusqu'au temps t. Le processus est dit partiellement observable car l'agent n'accède pas à l'information complète sur l'état s mais plutôt à une observation partielle o de cet état.

$\mathcal{A}$ est l'espace des actions : $\mathcal{A} = \mathcal{A}_c \cup a_d$, avec a_d l'action d'attente d'une nouvelle observation et $\mathcal{A}_c$ l'ensemble des actions de classification : $\mathcal{A}_c = \mathcal{L}$.

Le modèle de transition T entre les états est défini par :

$$T((X, l, t), a) = \begin{cases} \emptyset & \text{si } a \in \mathcal{A}_c \\ (X, l, t+1) & \text{si } a = a_d \end{cases}$$

Les actions de classification terminent le processus.

Récompense Les récompenses permettent de fixer les objectifs de précocité et de précision. On définit $R(s, a)$ comme étant la récompense pour choisir l'action $a \in \mathcal{A}$ à l'état s.

(1) Actions de classification : Lorsque la classe prédite correspond à la classe réelle, $R((X, l, t), a = l) = +1$. Sinon $R((X, l, t), a \neq l) = -1$. D'autres fonctions de récompense peuvent être choisies comme, par exemple dans [3] où une classification correcte correspond à $R((X, l, t), a = l) = 0$.

(2) Action d'attente : Pour éviter des récompenses parcimonieuses, l'action d'attente est pénalisée à chaque pas de temps telle que la pénalité augmente avec la quantité d'information reçue par l'agent. Nous définissons la pénalité d'attente à l'instant t par $R((X, l, t), a_d) = -\lambda \times \kappa^t / (\kappa^T - 1)$, où $\kappa > 1$. Le paramètre $\lambda \in \mathbb{R}^+$ permet de définir le compromis entre les deux objectifs : plus λ est grand, plus la précocité est privilégiée par rapport à la qualité de la classification.

Particularités du POMDP $\mathcal{A} = \mathcal{A}_c \cup a_d$ est l'ensemble des actions avec a_d pour l'action d'attente et $\mathcal{A}_c$ les actions de prédiction des classes $l \in \mathcal{L}$ qui terminent le processus d'acquisition. La probabilité d'atteindre le temps t dans un épisode tend à être nulle lorsque t augmente. Aussi, lorsque l'agent classe

à l'instant t, l'épisode est composé de $t-1$ actions d'attente pour une action de classification. Les actions de classification sont donc les plus rares, contrairement à l'action d'attente.

4 Apprentissage de la fonction *Q (s, a)*

L'espace d'action $\mathcal{A}$ étant discret et de faible dimension, nous modélisons la politique π de l'agent par la fonction de valeur d'action $Q(o,a)$. L'espace d'observations étant continu, nous utilisons l'algorithme DDQN de [5] pour approximer la fonction $Q(o,a)$ à l'aide d'un réseau de neurones profond $Q(o,a,\Theta)$. L'avantage de la méthode est l'apprentissage simultané de (1) la classification optimale des séquences et (2) la décision de classer ou d'attendre, cette dernière contribuant à identifier un temps optimal de prédiction.

Apprentissage sur une base d'entraînement fixe Dans le cadre d'applications réelles, la taille de l'ensemble d'apprentissage peut être fixe et ne pas évoluer au cours du temps. Dans ce cas, on construit donc la mémoire exhaustive $\mathcal{M}$ de toutes les interactions $< s,a,r,s' >$ possibles. L'utilisation de l'algorithme propose un tirage uniforme sur les interactions stockées en mémoire. Mais la représentation des différentes actions est très déséquilibrée, avec peu d'actions de classification par rapport à l'action d'attente. La mémoire peut également être déséquilibrée dans la représentation des classes suivant la composition du jeu de données d'entraînement. Nous proposons donc d'imposer une stratégie d'*échantillonnage par action* et par classe qui sera utilisée dans l'algorithme d'apprentissage.

A partir de l'ensemble des interactions dans la mémoire et pour chaque libellé de classe $l \in \mathcal{L}$, nous échantillonnons un mini-lot aléatoire d'interactions $\{< o,a,r,o' >\} \sim \mathcal{M}$ tel que l'observation o soit associée à une séquence temporelle X du libellé de classe l. Nous imposons qu'une fraction $\mu \in [0,1]$ du mini-lot soit associée à des actions de classification $a \in \mathcal{A}_c$, avec μ le paramètre d'échantillonnage par action.

Apprentissage en ligne La résolution du problème de classification précoce avec l'apprentissage par renforcement peut également être effectuée en ligne avec simultanément la collecte de nouvelles données d'entraînement et l'optimisation de la politique π. Pour s'adapter aux spécificités du POMDP de prédiction précoce, nous proposons une adaptation de l'algorithme DDQN avec une stratégie simple d'*initialisation des épisodes*, d'*échantillonnage par action* et de *stockage par action*.

L'action d'attente a_d étant sur-représentée, nous proposons de réserver une fraction de la mémoire aux actions de classification $\mathcal{A}_c$. Avec cette stratégie, les actions de classification seront *stockées prioritairement* dans la mémoire, par rapport aux action d'attente.

L'initialisation statique de l'épisode d'entraînement au temps $t=1$ entraîne une surreprésentation des premiers temps de séquences dans la mémoire. Nous adaptons donc l'algorithme DDQN par une stratégie d'*initialisation des épisodes* à des instants aléatoires dans la séquence temporelle pour obliger l'agent à explorer et à s'entraîner sur tous les instants d'acquisition de la séquence. Nous utilisons également la même stratégie d'*échantillonnage par action* et par classe que lors de l'apprentissage sur une base d'entraînement fixe.

5 Évaluation expérimentale

Dans [4], nous comparons la méthode d'apprentissage par renforcement à d'autres algorithmes de classification précoce sur des jeux de données publics issus de l'archive UCR de [6]. Nous montrons que l'agent entraîné pour la classification précoce apprend simultanément des règles de classification et de prise de décision sur le temps de prédiction tout en améliorant sa politique au cours des épisodes d'entraînement jusqu'à atteindre un meilleur compromis entre la qualité de la classification et sa précocité. Nous atteignons des performances de classification similaires aux méthodes état de l'art avec des temps de prédiction plus rapide.

Dans cette étude expérimentale, nous montrons l'amélioration apportée par le classifieur précoce, prenant des décisions de classification ou d'attente, par rapport à un réseau de neurones profond "naïf" statique équivalent entraîné à la classification à chaque instant. Les évaluations expérimentales sont faites sur un ensemble de données provenant d'un projet mené par la société bioMérieux. Les données sont des séries temporelles à plusieurs variables provenant d'organismes vivants. Les 3155 séquences temporelles $X = (x_1, ..., x_T)$ ont une longueur $T = 77$ et chaque point de données $x_{i\in[1,T]}$ est un vecteur de dimension 5. Les séquences sont associées à quatre classes d'organismes vivants.

Nous sélectionnons de manière aléatoire un ensemble d'hyper-paramètres dans un espace combinatoire restreint proche des paramètres optimaux présentés dans [7]. Nous dédions un agent par paramétrage d'hyper-paramètres. Nous réalisons l'apprentissage pour un nombre fixe d'épisodes jusqu'à ce que le réseau neuronal atteigne 100 000 mises à jour.

L'instant de prédiction $t = t_{j,pred}$ sur une séquence $(X^j, l^j) \in \mathcal{D}$ est défini comme l'instant pour lequel la valeur d'action $Q(o = X^j_{:t}, a \in \mathcal{A}_c)$ d'une action de classification dépasse la valeur d'action d'attente $Q(o = X^j_{:t}, a_d)$. Le temps de prédiction moyen t_{pred} de l'agent est la moyenne des temps de prédiction sur $\mathcal{D}$, tel que :

$$t_{pred} = \sum_{j=1}^{n} t_{j,pred}/n$$

Nous notons $\widehat{X^j}$ la classe prédite par l'agent sur la séquence X^j. La précision Acc du classifieur sur l'ensemble des n séquences $\mathcal{D} = \{(X^j, l^j)\}_{j=1..n}$ est :

$$Acc = \sum_{j=1}^{n} \mathbb{1}(\widehat{X^j} = l^j)/n$$

Dans [7], les auteurs sélectionnent la politique optimale comme celle qui obtient le score de récompense le plus élevé.

Dans le cas particulier de la prédiction précoce, la sélection de politique optimale π^* dépend de l'application et du compromis souhaité entre précocité et qualité de la classification. L'utilisateur peut choisir une plage de temps de prédiction moyen et sélectionner la politique qui obtient la meilleure qualité de classification Acc sur cette plage de temps.

Dans la figure 1, nous reportons les performances moyennes des cinq politiques les plus performantes du classifieur précoce pour différentes plages de temps de prédiction moyen t_{pred}.

Pour comparer expérimentalement le classifieur précoce avec l'approche "naïve" statique, nous désactivons la capacité de prise de décision du classifieur précoce, c'est-à-dire la partie apprentissage par renforcement. Nous entraînons un ensemble de réseaux de neurones statiques prenant chacun en entrée les données partielles $X_{:t}$, t parcourant $[1, T]$. Les réseaux des classifieurs statiques et du classifieur précoce sont similaires, à l'exception de la couche de sortie. Celle du classifieur précoce a une fonction d'activation linéaire et possède un neurone supplémentaire pour l'action d'attente par rapport à celle du classifieur statique qui a autant de neurones que de classes et finit par une fonction d'activation softmax. Nous reportons sur la figure 1 les performances moyennes des cinq réseaux de neurones statiques les plus performants sur les même plages de temps de prédiction t_{pred} que pour le classifieur précoce.

Le réseau de neurone profond "naïf" statique et le classifieur précoce ont tous les deux une faible Acc pour $t_{pred} < 20$ en raison du manque d'informations dans les séquences temporelles partielles. Puis, le classifieur précoce fournit 5 politiques avec une Acc plus élevée que le classifieur statique. L'amélioration en Acc pour un t_{pred} équivalent est due à la capacité de l'agent d'adapter sa classification individuellement à chaque séquence temporelle. L'agent peut choisir de classer rapidement les séquences qui peuvent facilement être classées ou d'exiger davantage d'observations sur les séquences dépourvues de motifs discriminants. Ceci rend la classification plus efficace que les réseaux statiques utilisant le même nombre d'observations dans toutes les séquences, quelle que soit leur complexité.

Il est intéressant de noter que nous ne pouvons pas évaluer le classifieur précoce dans les temps de prédiction tardifs ($t_{pred} > 55$). Pour atteindre son objectif de prise de décision rapide, l'agent a en effet préféré ne pas attendre la fin des séquences et a toujours fourni des décisions plus rapides.

6 Conclusion

La problématique de classification précoce a été modélisée par un Processus de Décision Markovien Partiellement Observable (POMDP). L'approche choisie d'Apprentissage par Renforcement (RL) de la politique a permis de traiter simultanément les objectifs compétitifs de bonne classification et de décision rapide. Nous avons traité la problématique du déséquilibre des classes, des actions et des temps dans la mémoire de l'agent pour l'apprentissage. Enfin, nous avons comparé expérimentalement notre approche avec un réseau de neurones profond "naïf" statique équivalent entraîné à la classification à

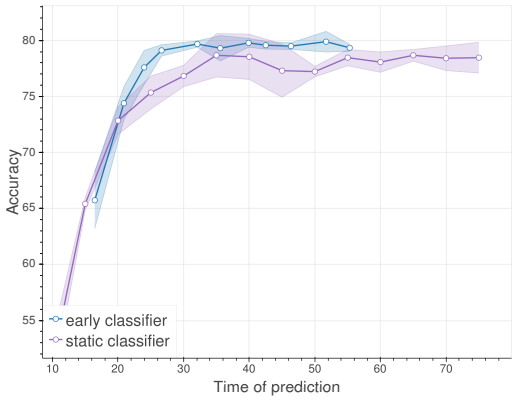

FIGURE 1 – Les performances en Acc à différents pas de temps t_{pred} des 5 meilleures politiques du POMDP et des 5 meilleurs classifieurs "naïfs" statiques. Les lignes pleines représentent la moyenne et les bandes représentent l'écart-type de Acc.

chaque instant, montrant l'apport de l'apprentissage par renforcement pour prendre des décisions spécifiques à la complexité de chaque série temporelle.

Références

[1] G. Dulac-Arnold, L. Denoyer, P. Gallinari, Text classification : A sequential reading approach, in : European Conference on Information Retrieval, Springer, 2011, pp. 411–423.

[2] Y.-S. Peng, K.-F. Tang, H.-T. Lin, E. Chang, Refuel : Exploring sparse features in deep reinforcement learning for fast disease diagnosis, in : Advances in Neural Information Processing Systems, 2018, pp. 7333–7342.

[3] J. Janisch, T. Pevnỳ, V. Lisỳ, Classification with costly features using deep reinforcement learning, in : AAAI Conference on Artificial Intelligence, 2019.

[4] C. Martinez, G. Perrin, E. Ramasso, M. Rombaut, A deep reinforcement learning approach for early classification of time series, in : 2018 26th European Signal Processing Conference (EUSIPCO), IEEE, 2018, pp. 2030–2034.

[5] H. Van Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double q-learning, in : Thirtieth AAAI Conference on Artificial Intelligence, 2016.

[6] H. A. Dau, A. Bagnall, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, E. Keogh, The ucr time series archive, arXiv preprint arXiv :1810.07758.

[7] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529.