Université de Montréal

Apprentissage de stratégies de calcul adaptatives pour les réseaux neuronaux profonds

par

Aton Kamanda

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en informatique

Juillet 17, 2023

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé

Apprentissage de stratégies de calcul adaptatives pour les réseaux neuronaux profonds

présenté par

Aton Kamanda

a été évalué par un jury composé des personnes suivantes :

Eugene Syriani
(président-rapporteur)

Houari Sahraoui
(directeur de recherche)

Ioannis Mitliagkas
(membre du jury)

Résumé

La théorie du processus dual stipule que la cognition humaine fonctionne selon deux modes distincts : l'un pour le traitement rapide, habituel et associatif, appelé communément "système 1" et le second, ayant un traitement plus lent, délibéré et contrôlé, que l'on nomme "système 2". Cette distinction indique une caractéristique sous-jacente importante de la cognition humaine : la possibilité de passer de manière adaptative à différentes stratégies de calcul selon la situation. Cette capacité est étudiée depuis longtemps dans différents domaines et de nombreux bénéfices hypothétiques semblent y être liés [**56**]. Cependant, les réseaux neuronaux profonds sont souvent construits sans cette capacité à gérer leurs ressources calculatoires de manière optimale. Cette limitation des modèles actuels est d'autant plus préoccupante que de plus en plus de travaux récents semblent montrer une relation linéaire entre la capacité de calcul utilisé et les performances du modèle lors de la phase d'évaluation [**39**].

Pour résoudre ce problème, ce mémoire propose différentes approches et étudie leurs impacts sur les modèles, tout d'abord, nous étudions un agent d'apprentissage par renforcement profond qui est capable d'allouer plus de calcul aux situations plus difficiles. Notre approche permet à l'agent d'adapter ses ressources computationnelles en fonction des exigences de la situation dans laquelle il se trouve, ce qui permet en plus d'améliorer le temps de calcul, améliore le transfert entre des tâches connexes et la capacité de généralisation. L'idée centrale commune à toutes nos approches est basée sur les théories du coût de l'effort venant de la littérature sur le contrôle cognitif qui stipule qu'en rendant l'utilisation de ressource cognitive couteuse pour l'agent et en lui laissant la possibilité de les allouer lors de ses décisions il va lui-même apprendre à déployer sa capacité de calcul de façon optimale. Ensuite, nous étudions des variations de la méthode sur une tâche référence d'apprentissage profond afin d'analyser précisément le comportement du modèle et quels sont précisément les bénéfices d'adopter une telle approche. Nous créons aussi notre propre tâche "Stroop MNIST" inspiré par le test de Stroop [**74**] utilisé en psychologie afin de valider certaines hypothèses sur le comportement des réseaux neuronaux employant notre méthode. Nous finissons par mettre en lumière les liens forts qui existent entre apprentissage dual et les méthodes de

distillation des connaissances. Notre approche a la particularité d'économiser des ressources computationnelles lors de la phase d'inférence.

Enfin, dans la partie finale, nous concluons en mettant en lumière les contributions du mémoire, nous détaillons aussi des travaux futurs, nous approchons le problème avec les modèles basés sur l'énergie, en apprenant un paysage d'énergie lors de l'entrainement, le modèle peut ensuite lors de l'inférence employer une capacité de calcul dépendant de la difficulté de l'exemple auquel il fait face plutôt qu'une simple propagation avant fixe ayant systématiquement le même coût calculatoire. Bien qu'ayant eu des résultats expérimentaux infructueux, nous analysons les promesses que peuvent tenir une telle approche et nous émettons des hypothèses sur les améliorations potentielles à effectuer.

Nous espérons, avec nos contributions, ouvrir la voie vers des algorithmes faisant un meilleur usage de leurs ressources computationnelles et devenant par conséquent plus efficace en termes de coût et de performance, ainsi que permettre une compréhension plus intime des liens qui existent entre certaines méthodes en apprentissage machine et la théorie du processus dual.

Mots clés: Apprentissage profond, Apprentissage par renforcement profond, Théorie du processus dual, Efficacité computationnelle, Distillation des connaissances, Modèles basés sur l'énergie, Contrôle cognitif.

Abstract

The dual-process theory states that human cognition operates in two distinct modes: one for rapid, habitual and associative processing, commonly referred to as "system 1", and the second, with slower, deliberate and controlled processing, which we call "system 2". This distinction points to an important underlying feature of human cognition: the ability to switch adaptively to different computational strategies depending on the situation. This ability has long been studied in various fields, and many hypothetical benefits seem to be linked to it [**56**]. However, deep neural networks are often built without this ability to optimally manage their computational resources. This limitation of current models is all the more worrying as more and more recent work seems to show a linear relationship between the computational capacity used and model performance during the evaluation phase [**39**].

To solve this problem, this thesis proposes different approaches and studies their impact on models. First, we study a deep reinforcement learning agent that is able to allocate more computation to more difficult situations. Our approach allows the agent to adapt its computational resources according to the demands of the situation in which it finds itself, which in addition to improving computation time, enhances transfer between related tasks and generalization capacity. The central idea common to all our approaches is based on cost-of-effort theories from the cognitive control literature, which stipulate that by making the use of cognitive resources costly for the agent, and allowing it to allocate them when making decisions, it will itself learn to deploy its computational capacity optimally.

We then study variations of the method on a reference deep learning task, to analyze precisely how the model behaves and what the benefits of adopting such an approach are. We also create our own task "Stroop MNIST" inspired by the Stroop test [**74**] used in psychology to validate certain hypotheses about the behavior of neural networks employing our method. We end by highlighting the strong links between dual learning and knowledge distillation methods.

Finally, we approach the problem with energy-based models, by learning an energy landscape during training, the model can then during inference employ a computational capacity

dependent on the difficulty of the example it is dealing with rather than a simple fixed forward propagation having systematically the same computational cost. Despite unsuccessful experimental results, we analyze the promise of such an approach and speculate on potential improvements.

With our contributions, we hope to pave the way for algorithms that make better use of their computational resources, and thus become more efficient in terms of cost and performance, as well as providing a more intimate understanding of the links that exist between certain machine learning methods and dual process theory.

Keywords: Deep learning, Deep reinforcement learning, Dual process theory, Computational efficiency, Knowledge distillation, Energy-based models, Cognitive control.

Table des matières

Liste des tableaux

Liste des figures

Liste des sigles et des abréviations

MNIST	Base de données modifiée du National Institute of Standards and Technology, de l'anglais Modified National Institute of Standards and Technology database *Cross-entropy*
MDL/LDM	Longueur de description minimale, en anglais *Minimum description lenght (MDL)*
EBM/MBE	Modéle basé sur l'energie, de l'anglais *Energy based model (EBM)*
ACC/CCA	Contrôle computationnel adaptatif, de l'anglais *Adaptive computationnal control (ACC)*
MRR	Rang réciproque moyen, de l'anglais *Mean Reciprocal Rank*
MAP	Moyenne de moyenne de précision, de l'anglais *Mean Average Precision*

Remerciements

Je tiens tout d'abord à exprimer ma profonde gratitude envers toutes les personnes qui ont joué un rôle essentiel dans la possibilité d'accomplir cette maîtrise. Votre soutien, vos enseignements et vos encouragements ont été d'une valeur inestimable tout au long de mon parcours.

Je voudrais plus particulièrement exprimer mes remerciements les plus sincères à mes collaborateurs Lucas Maes et Martin Weyssow et mon directeur de recherche Prof. Houari Sahraoui, qui m'ont guidé avec patience et expertise tout au long de ce projet. Vos conseils avisés et vos discussions stimulantes ont été des moments précieux qui ont enrichi ma compréhension des sujets que j'ai étudiés et du monde de la recherche scientifique en général.

Enfin, je souhaite adresser mes remerciements spéciaux à toutes les personnes qui, directement ou indirectement, ont contribué à ma formation académique et ont permis mon parcours possible. Votre influence, vos conseils et vos connaissances partagées ont façonné ma pensée critique et ont élargi mes horizons.

Je suis profondément reconnaissant envers chacun(e) d'entre vous pour avoir fait partie de mon voyage académique. Votre impact sur mon parcours intellectuel et personnel ne peut être surestimé.

Chapitre 1

Introduction

1.1. Contexte

La distinction entre deux modes de pensée, l'un automatique, rapide et habituel, et l'autre contrôlé, lent et délibéré, a été remarquée depuis longtemps en psychologie [**35**], plus récemment, cette distinction a été caractérisée dans le contexte de la prise de décision comme le "système 1" pour la stratégie de traitement rapide, automatique et habituel et le "système 2" pour la stratégie lente, délibérée et contrôlée [**36**]. Cette distinction apparaît également dans l'apprentissage par renforcement, elle a permis de délimiter deux types d'apprentissage, le comportement habituel dit ("sans modèle") et le comportement plus orienté vers un objectif précis et la planification dit ("basé sur un modèle").

L'une des façons d'envisager le compromis entre ces deux stratégies de calculs est de considérer la complexité de l'échantillon et la complexité temporelle des algorithmes. La complexité de l'échantillon fait référence au nombre d'exemples d'apprentissage dont un algorithme d'apprentissage a besoin pour atteindre un certain niveau de précision. La complexité temporelle fait référence à la durée d'exécution d'un algorithme. Si l'on prend l'exemple de l'apprentissage pas renforcement, les algorithmes sans modèle ont une complexité d'échantillonnage élevée, mais une faible complexité temporelle - en d'autres termes, l'apprentissage est lent, mais l'inférence est rapide. Les algorithmes basés sur un modèle ont la propriété inverse : une complexité d'échantillonnage relativement faible, en supposant que le modèle puisse être appris efficacement, mais une complexité temporelle élevée. Étant donné que la quantité de données et de calculs à laquelle un modèle a accès est systématiquement limitée, il semble que ce compromis soit inévitable, qu'il s'agisse d'un agent artificiel ou naturel, d'autant que ces deux types de traitement de l'information sont mutuellement exclusifs. En plus des avantages purement en termes d'économie de ressources de calcul qui semble nécessiter un système gérant ce compromis, des travaux théoriques récents semblent indiquer

qu'une telle distinction entre des stratégies de calculs différentes permet au modèle de gagner des bénéfices computationnels très avantageux pour les réseaux de neurones artificiels, notamment en terme de généralisation et de robustesse au surapprentissage [**56**].

1.2. Problématique

Il semble donc avantageux d'un point de vue théorique d'avoir des modes de calculs distinct en fonction de la situation. Cependant, les réseaux neuronaux actuels contrastent avec cette observation, ils présentent un budget de calcul fixe contrôlé par l'architecture du réseau, empêchant toute distinction dans la stratégie de calcul adoptée qui soit influencée par les états qu'ils traversent. Ce paradigme pose un problème de plus en plus préoccupant, d'autant que l'importance de la puissance de calcul pour accroître les performances des modèles sur les tâches continue d'être démontré dans de nombreux contextes tels que la planification [**72**], les grands modèles de langage [**39**] et, plus récemment, l'apprentissage par renforcement basé sur un modèle [**27**]. Cette situation est problématique pour une raison simple : étant donné que la puissance de calcul influe sur les performances du modèle sur la tâche, à budget de calcul fixe, il semble sous-optimal que ces modèles déploient toute leur puissance de calcul sur chaque exemple, quelle que soit la situation. Cette intuition a été confirmée empiriquement, [**2**] crée une métrique rigoureuse de la difficulté d'un échantillon de donnée et montre que dans un grand nombre de tâches, pour une partie importante des échantillons, les prédictions faites par le réseau neuronal sont déjà décidées avant les couches finales, ce qui rend une grande partie des calculs inutiles. En outre, leur article montre qu'il existe une relation linéaire entre la difficulté d'un échantillon et la quantité de calcul nécessaire pour le prédire. Cela signifie qu'avec un budget de calcul donné et fixe, de meilleures performances pourraient être obtenues avec une meilleure stratégie d'allocation.

En plus de cela, un modèle peut être confronté à des situations qui nécessitent une réaction rapide ou une planification très précise et réfléchie, qui sont des stratégies de calcul opposées en raison de la nature même de leurs exigences matérielles. Ces phénomènes ont été largement étudiés dans la littérature sur le contrôle cognitif et sont connus sous le nom de "Speed accuracy tradeoff" (compromis entre vitesse et précision) [**30**].

Malheureusement, le coût de calcul est souvent ignoré dans la plupart des tests de référence, dans les processus de décision markoviens classiques, formalisme principale de l'apprentissage par renforcement, le temps que prend l'agent pour passer d'un état à l'autre n'est pas pris en compte dans la récompense finale et par conséquent pas dans la tâche en elle-même. Il a été remarqué que ce décalage entre les hypothèses sous-jacentes aux tâches de référence classique et la réalité du calcul en temps réel peut conduire à des résultats indésirables [**63**].

Dans ce mémoire, nous proposons plusieurs méthodes pour remédier à ces limitations, ces méthodes permettent au modèle de déterminer de manière adaptative la difficulté d'un échantillon et d'allouer des ressources de calcul en fonction de l'évaluation de la difficulté de la situation.

1.3. Contributions

Dans ce mémoire, nous proposons différentes approches pour implémenter la théorie du processus dual [**56**] dans les réseaux de neurones artificiels et leur permettre de changer de stratégie de calcul en fonction de la situation. Nous analysons aussi les hypothétiques avantages d'une telle approche en nous basant sur des résultats théoriques et empiriques de différents domaines.

Tout d'abord, nous proposons deux nouvelles méthodes distinctes pour implémenter la théorie du processus dual, la première : "Contrôle computationnel adaptatif" (CCA ou ACC en anglais) est une méthode basée sur la distillation de la connaissance, ancrant ses fondements théoriques dans la théorie du principe de longueur de description minimale [**20**], un concept à la frontière de l'informatique théorique et des statistiques, nous montrons qu'un réseau de neurone artificiel utilisant cette méthode valide plusieurs avantages computationnels hypothétiques avancés par [**56**] comme un gain en termes de temps de calcul ainsi qu'une meilleure capacité de généralisation. L'ajout majeur entre notre contribution et les travaux précédents est l'idée d'ajouter un coût cognitif [**42, 43**] à l'utilisation des ressources afin que le modèle apprenne une utilisation optimale de celle-ci lors de son entrainement. Nous montrons aussi que le comportement du réseau de neurone face aux données est cohérent avec le comportement attendu d'un point de vue théorique par un réseau qui appliquerait les principes de la théorie du processus dual. Nous analysons les forces et faiblesses de variations de notre méthode sur les mêmes tâches. Nous évaluons cette méthode via un agent d'apprentissage par renforcement profond ainsi que sur MNIST [**13**]. Nous proposons également un nouveau benchmark inspiré du test de Stroop en psychologie [**74**], "Stroop MNIST": qui permet d'évaluer le comportement des réseaux de neurones ayant un système de calcul adaptatif.

La seconde méthode que nous avons élaborée propose d'exploiter les avantages des modèles basés sur l'énergie [**46**], en entraînant une fonction d'énergie paramétrisée pour prédire le coût pendant l'entrainement lorsqu'il est connu, le modèle peut ensuite utiliser cette fonction d'énergie paramétrisée pour prédire le coût de sa solution pendant l'inférence et descendre le gradient de l'énergie par rapport à des variables d'intérêts comme les neurones ou les variables latentes pour proposer des solutions alternatives variant en termes de complexité de calcul, la forme locale du paysage de la fonction d'énergie correspondra ainsi naturellement à la difficulté de l'échantillon d'entrée, puisque les solutions faciles convergeront plus

rapidement vers le minima et nécessiteront donc moins de temps de calcul. Bien qu'ayant été infructueuse lors de l'implémentation, nous présentons quand même cette méthode, car nous pensons qu'il s'agit d'une méthode prometteuse pour les méthodes de calcul adaptative.

Nous espérons que les contributions discutées dans ce mémoire constituent un pas vers l'implémentation et la compréhension des stratégies de calculs adaptatives dans les réseaux de neurones artificiels modernes.

1.4. Structure

Ce mémoire est décomposé de la manière suivante : le premier chapitre introduit clairement quelles sont les motivations du mémoire et dans quel contexte les problématiques défiés se situent, le deuxième chapitre est une vue d'ensemble des concepts majeurs nécessaires à la bonne compréhension des méthodes discutées.

Le troisième chapitre regroupe les travaux connexes essentiels qui détaillent l'historique interdisciplinaire de la théorie de l'apprentissage dual, ses liens avec l'apprentissage machine ainsi que l'historique des concepts majeurs employés dans nos méthodes.

Le quatrième chapitre décrit en détail la première approche basée sur la distillation de la connaissance, son fonctionnement, ses liens théoriques avec la théorie de l'apprentissage dual et les résultats expérimentaux, ainsi qu'une étude approfondie des variations possibles.

Enfin, le cinquième chapitre conclut les contributions apportées lors de ce mémoire et discute des travaux futurs. En particulier, nous développons une approche potentielle utilisant les modèles basés sur l'énergie.

Chapitre 2

Généralités et définitions

2.1. Apprentissage machine

L'apprentissage machine fournit des méthodes automatisées qui peuvent détecter des caractéristiques dans les données brutes et les utiliser pour accomplir certaines tâches d'intérêt. Si l'on formalise le problème, la plupart des tâches d'apprentissage machine consiste, étant donné un jeu de donnée $\mathcal{D}$, un ensemble de données d'entrée $x_i \in \mathbb{R}^d$ tiré de cet ensemble de donnée où d est la dimension de la donnée d'entrée x , et des sorties possibles, $y_i \in \mathbb{R}^o$ où o est la dimension de la sortie y, l'objectif est de trouver une fonction $f : \mathcal{X} \to \mathcal{Y}$ qui minimise un certain critère d'erreur $\mathcal{L}$. L'apprentissage machine se distingue en de nombreuses branches qui se sont beaucoup développés au cours des dernières années. Par exemple, l'apprentissage supervisé, une des premières forme d'apprentissage machine à avoir connu du succès, consiste à déduire une classification ou une régression à partir de données d'apprentissage étiquetées, c'est-à-dire que le modèle apprend avec une paire d'exemple (x,y) ou la solution y lui est donnée, récemment l'apprentissage non supervisé ou auto-supervisé, qui consiste à extraire des caractéristiques à partir d'ensembles de données constitués de données d'entrée sans réponse étiquetées, a gagné en efficacité et a même rattrapé les méthodes supervisées sur des tâches de références [**10, 3**]. Ces méthodes ont suscité un intérêt tout particulier, car elles ne nécessitent pas l'étiquetage de données d'entrainement pour le modèle. Cela signifie, la phase d'apprentissage ne nécessite que fournir les données d'entrée x pour que le modèle puisse extraire les caractéristiques et résoudre une tâche d'intérêt. Enfin, un autre type d'apprentissage machine, l'apprentissage par renforcement, consiste à construire des agents effectuant des actions dans un environnement afin de maximiser les récompenses cumulées [**77**].

Une des plus grosses révolutions des dernières années en apprentissage machine a été réalisé principalement grâce aux développements récents de l'apprentissage profond, tous les types d'apprentissage évoqués précédemment ont grandement bénéficié des avancées en apprentissage profond, en particulier, elles ont connu des améliorations spectaculaires lorsqu'il

s'agit d'apprendre de données à haute dimension qui est la propriété de beaucoup de données d'intérêt tel que les séries temporelles, les images et les vidéos.

2.2. Apprentissage profond

Pour résoudre le problème de trouver une fonction $f : \mathcal{X} \to \mathcal{Y}$ qui minimise un certain critère d'erreur $\mathcal{L}(x,y)$, l'idée clé de l'apprentissage profond est d'utiliser un réseau de neurones artificiel, un réseau de neurones artificiel se caractérise par une succession de multiples couches effectuant un calcul menant à un différent niveau en termes de représentation. Chaque couche alterne entre une transformation linéaire et une transformation non-linéaire, tel que $h = A\left(W \cdot x + b\right)$, où A est une fonction d'activation non-linéaire, h est l'input qui remplacera le x à la prochaine couche du réseau, W et b sont des paramètres entrainables du réseau de neurones appelé respectivement poids synaptiques et biais. Lorsque le réseau termine et qu'il n'y a plus de transformation restante, on considère cela comme la sortie du réseau de neurones y.

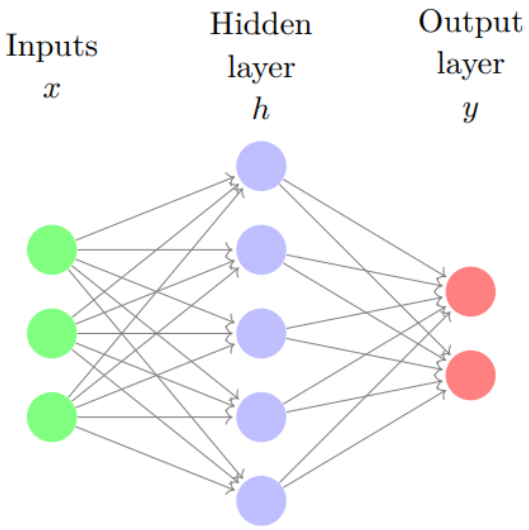


Fig. 2.1. Un réseau de neurones simple avec une seule couche cachée, figure de [**17**].

De façon formelle, un réseau de neurones artificiels est donc une fonction différentiable $f : \mathcal{X} \to \mathcal{Y}$ paramétrisée par des paramètres θ tels que $y = f(x;\theta)$. Lors de la phase d'apprentissage, le modèle modifie ses paramètres θ appelés poids synaptiques afin de minimiser l'erreur de la fonction de coût $\mathcal{L}(x,y)$. La méthode la plus courante pour optimiser les paramètres d'un réseau de neurone est basée sur la descente de gradient via l'algorithme de rétropropagation du gradient [**64**]. L'utilisation de cette technique nécessite que toutes les opérations de la fonction entre l'entrée et la sortie soient différentiables. Grâce à l'algorithme de rétropropagation du gradient, le gradient de la fonction de coût par rapport aux poids synaptiques du modèle $\nabla_\theta \mathcal{L}(x,y)$ est obtenu. Ensuite, dans le cas le plus simple,

à chaque itération, l'algorithme modifie ses paramètres internes θ afin d'ajuster la fonction de coût dans une étape appelée pas de gradient : $\theta_{t+1} = \theta_t - \alpha \nabla_\theta \mathcal{L}$ où α appelé le taux d'apprentissage est un scalaire déterminant à quel point la magnitude de la mise à jour des poids est élevé. L'enchaînement de toutes ces opérations permet aux modèles de passer de paramètres initialisés aléatoirement ayant des performances faibles à des poids synaptiques ayant un faible taux d'erreur et capable de résoudre la tâche donnée.

2.3. Apprentissage par renforcement profond

L'apprentissage par renforcement est une partie de l'apprentissage machine qui se concentre plus sur le contrôle optimal et la prise de décision, par ce fait et par son histoire, il possède un formalisme qui lui est propre. Le formalisme de l'apprentissage par renforcement repose sur les processus de décision de Markov [**4**]. En apprentissage par renforcement, l'agent interagit avec son environnement dans le cadre d'un processus de contrôle stochastique à temps discret où un agent interagit avec son environnement de la manière suivante : l'environnement génère des états $s_0 \in S$, l'agent recueille une observation initiale $o_0 \in O$. À chaque pas de temps t, l'agent doit prendre une action, $a_t \in A$ à la suite de cette action l'agent obtient une récompense $r_t \in R$, l'état transite à $s_{t+1} \in S$, et l'agent obtient une nouvelle observation $o_{t+1} \in O$.

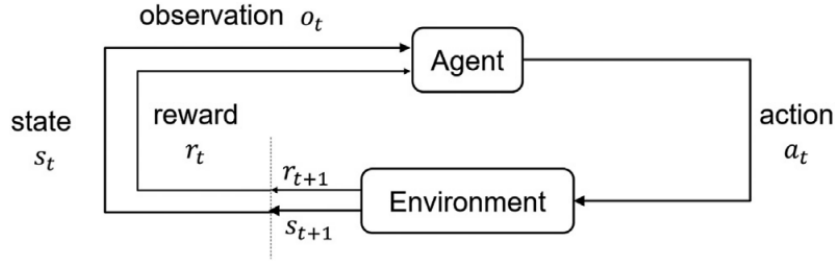


Fig. 2.2. La boucle de l'apprentissage par renforcement, figure adapté de [**76**].

Un processus de décision est dit markovien s'il possède les propriétés suivantes $\mathbb{P}(o_{t+1} \mid o_t, a_t) = \mathbb{P}(o_{t+1} \mid o_t, a_t, \ldots,, o_0, a_0)$ et $\mathbb{P}(r_t \mid o_t, a_t) = \mathbb{P}(r_t \mid o_t, a_t, \ldots, o_0, a_0)$, cette propriété appelée propriété de Markov signifie que l'avenir du processus de décision markovien ne dépend que de l'observation actuelle, et l'agent n'a donc aucun intérêt à regarder l'historique complet. Un processus de décision markovien dans le contexte de l'apprentissage par renforcement est donc un 5-tuple où $\mathcal{S}$ est l'espace d'état, $\mathcal{A}$ est l'espace d'action, $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ est la fonction de transition qui détermine les probabilités conditionnelles de transition entre les états, $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathcal{R}$ est la fonction de récompense, $\mathcal{R}$ est un ensemble de récompense continue possible $R_{\max} \in \mathbb{R}^+$ $\gamma \in [0,1)$ est le facteur de décompte. Dans l'apprentissage par renforcement, deux concepts définissent l'agent et son comportement, la politique noté π qui définit la qui manière dont

un agent sélectionne les actions, elle peut être soit déterministe tel que $\pi(s) : \mathcal{S} \rightarrow \mathcal{A}$ soit stochastique tel que $\pi(s,a) : \mathcal{S} \times \mathcal{A} \rightarrow [0,1]$ ou $\pi(s,a)$ dénote la probabilité que a soit choisit dans l'état $s.$, le but de la politique est d'optimiser la fonction de valeur $V^{\pi}(s) : \mathcal{S} \rightarrow \mathbb{R}$ telle que $V^{\pi}(s) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid s_t = s, \pi\right]$ ou $r_t = \underset{a \sim \pi(s_t,\cdot)}{\mathbb{E}} R\left(s_t, a, s_{t+1}\right)$ et $\mathbb{P}\left(s_{t+1} \mid s_t, a_t\right) = T\left(s_t, a_t, s_{t+1}\right)$ avec $a_t \sim \pi\left(s_t, \cdot\right)$ ou $\sim$ signifie que l'action à l'instant t est échantillonnée de la politique.

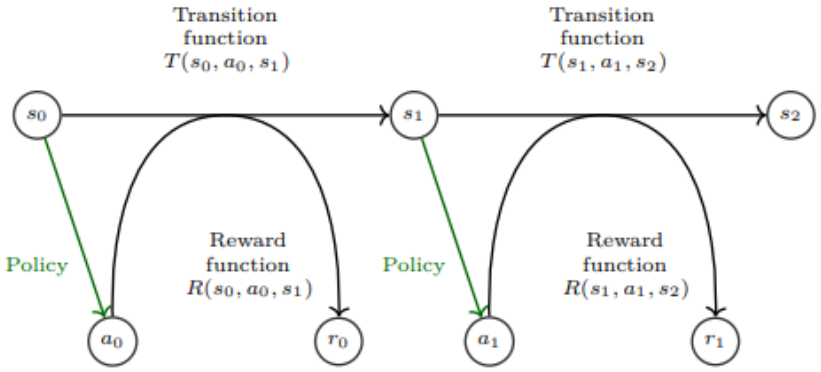


Fig. 2.3. Un processus de décision markovien, à chaque étape l'agent prend une action et reçoit une récompense, la fonction de transition détermine les probabilités conditionnelles de transition entre les états, figure de [**17**].

L'utilisation de l'apprentissage profond dans le cadre de l'apprentissage par renforcement est rendu évident par une raison simple, dans de nombreux scénarios de contrôles d'intérêt, l'environnement produit des observations à haute dimension, la politique et la fonction de valeur étant des fonctions prenant en entrée ces observations à haute dimension, l'utilisation de l'apprentissage profond pour approximer ces fonctions s'est révélé être d'une efficacité remarquable [**54**], l'apprentissage par renforcement utilise donc l'apprentissage profond à trois niveaux : pour apprendre une représentation de la fonction de valeur et calculer la valeur attendue pour un état donné, c'est ce que l'on nomme un "critique", pour apprendre une représentation de la fonction de la politique et une relation complexe entre les états et les actions, c'est ce que l'on nomme un "acteur", et enfin pour apprendre un modèle de l'environnement qui va estimer les probabilités de la fonction de transition entre les états ainsi que les récompenses associées en fonction des actions ce qui va permettre la planification, c'est ce qui est généralement appelé un "modèle du monde". L'absence ou la présence d'un modèle du monde est ce qui différencie l'apprentissage sans modèle et l'apprentissage basé sur le modèle, cette distinction a son importance dans la théorie du processus dual central à cette thèse de maîtrise.

2.4. Modèles basés sur l'énergie

L'objectif principal de l'apprentissage automatique est d'encoder les dépendances entre des variables d'intérêt, appelons-les x et y. Les modèles basés sur l'énergie (EBM pour "Energy based models") [**47**] capturent les dépendances en associant un scalaire appelé énergie (une mesure de compatibilité) à chaque configuration des variables. Dans ce formalisme, l'inférence consiste à fixer la valeur des variables observées et de trouver les valeurs des variables restantes qui minimisent l'énergie, ce qui permet par la suite de faire une prédiction ou prendre une décision. Formellement, le but d'un EBM est de construire une fonction $E(x,y)$ qui fait correspondre chaque point x d'un espace d'entrée à un scalaire unique, appelé énergie en fonction d'une certaine valeur possible y. Par exemple, dans un contexte où nous voudrions faire de la prédiction ou de la classification, ou de la prise de décision, calculer l'énergie $E(x,y)$ sera similaire à se poser la question: "Quelle valeur de y est la plus compatible avec ce x ? Par exemple, dans un contexte dans lequel nous souhaitons que le modèle conduise un véhicule autonome équipé d'une caméra, le x sera le flux vidéo et les senseurs et le y sera l'ensemble des décisions possibles du modèle tel que "tourner à gauche", "tourner à droite" ou "aller tout droit".

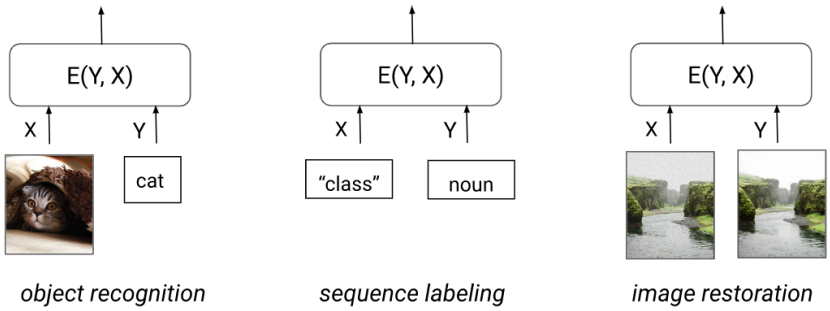


Fig. 2.4. Les modèles basés sur l'énergie déployés dans 3 contextes différents, la tâche que le modèle va résoudre va dépendre de la nature du x et du y fournit en entrée, figure de Stefano Ermon, Yang Song.

Une collection de valeurs d'énergie peut être transformée en une densité de probabilité $p(x)$ par le biais de la distribution de Gibbs.

$$p(y \mid \mathbf{x}) = \frac{e^{-E(\mathbf{x},y)/T}}{\int_{y'} e^{-E(\mathbf{x},y')/T}} = \frac{e^{-E(\mathbf{x},y)/T}}{e^{-E(\mathbf{x})/T}}$$

où T est la température, et le dénominateur est appelé fonction de partition. Il existe de nombreuses situations pratiques où le calcul de la fonction de partition est intractable, souvent parce qu'il y a une cardinalité élevée, intuitivement calculer cette fonction de partition dans le contexte de la classification d'image reviendrait à calculer la probabilité d'une image par rapport à toutes les images possibles. Comme il n'est pas nécessaire de procéder à une normalisation appropriée, les approches basées sur l'énergie évitent les problèmes liés

à l'estimation de la constante de normalisation dans les modèles probabilistes. En outre, un avantage souvent reconnu des modèles basés sur l'énergie est que l'absence de condition de normalisation permet une plus grande flexibilité dans la conception que les modèles probabilistes. La plupart des modèles probabilistes peuvent être considérés comme des types particuliers de modèles basés sur l'énergie dans lesquels la fonction d'énergie satisfait à certaines conditions de normalisation et où la fonction de perte, optimisée par l'apprentissage, à une forme particulière [**47**].

Dans les modèles basés sur l'énergie, l'apprentissage consiste à trouver une fonction d'énergie qui associe les énergies faibles aux valeurs correctes des variables restantes et les énergies élevées aux valeurs incorrectes. Une fonction de coût, minimisée pendant l'apprentissage, est utilisée pour mesurer la qualité des fonctions d'énergie possible. L'apprentissage basé sur l'énergie fournit un cadre unifié pour de nombreuses approches probabilistes et non probabilistes de l'apprentissage, en particulier pour l'apprentissage non probabiliste.

2.5. Processus dual

La théorie du processus dual propose que les processus cognitifs humains sont régis par deux systèmes distincts : le système 1, intuitif et automatique, et le système 2, analytique et contrôlé. Ce cadre a été largement adopté pour expliquer divers aspects de la prise de décision, du raisonnement et du comportement humains.

L'un des travaux fondateurs de la théorie du double processus est celui de Kahneman et Frederick [**37**], qui met en évidence la différenciation entre les deux systèmes et leurs rôles dans la prise de décision. Le système 1 fonctionne rapidement et sans effort, en s'appuyant sur des heuristiques et des jugements intuitifs, tandis que le système 2 implique une réflexion délibérée et laborieuse, caractérisée par une analyse rationnelle.

Une autre ligne de recherche qui contribue à la théorie du double processus est l'étude des biais cognitifs. Tversky et Kahneman [**81**] ont exploré divers biais découlant de la tendance du système 1 à s'appuyer sur des raccourcis, ce qui entraîne des écarts systématiques par rapport à la prise de décision rationnelle. Ces biais, tels que l'heuristique de disponibilité et l'effet d'ancrage, démontrent l'influence des processus automatiques sur les jugements humains.

Des études plus récentes ont étendu la théorie du double processus à différents domaines. Par exemple, Evans [**16**] examine comment la théorie s'applique aux processus de raisonnement. Il suggère que le système 1 guide souvent les réponses initiales, mais que le système 2 peut annuler ces réponses par une analyse minutieuse. En outre, Stanovich et West [**73**] soulignent les différences individuelles dans l'engagement des deux systèmes, mettant en évidence le rôle de la capacité cognitive dans la médiation de leurs interactions.

La théorie du double processus a donc largement contribué à notre compréhension de la cognition et du comportement humains. De la prise de décision au raisonnement et à la persuasion, l'interaction entre les processus du système 1 et du système 2 offre un cadre complet pour expliquer les complexités de l'esprit humain. Récemment, des bénéfices hypothétiques possibles de ce type de traitement de l'information pour les modèles d'apprentissage machine ont été formalisés par [**56, 55**].

Chapitre 3

Travaux Connexe

3.1. Contrôle cognitif

Nos travaux sont très inspirés par la littérature sur le contrôle cognitif, Stroop MNIST la tâche que nous avons créée pour tester notre méthode s'inspire de la tâche de Stroop, une expérience psychologique classique d'étude du traitement cognitif contrôlé [**74**].

La théorie du processus dual dont les origines peuvent être retracées à [**35**] a été popularisé par [**36**] en tant que système 1 pour le traitement automatique habituel et système 2 pour le traitement délibéré contrôlé, [**80**] utilise une politique distillée qui capture le comportement commun entre les tâches et régularise les politiques pour se rapprocher d'une politique partagée. Récemment, il a été remarqué que de nombreuses facultés faisant défaut au réseau de neurones artificiels actuel semblaient être liées au contrôle cognitif [**34**].

De manière cruciale, [**56**] explicite ce lien et montre que la différenciation en un processus dual peut être comprise comme émergeant de l'optimisation et de l'apprentissage dès lors qu'une politique dite "par défaut" compressé essaie de capturer le comportement d'une autre ayant plus d'expressivité [**56**] arrive à rendre compte d'une quantité significative de résultats expérimentaux sur le contrôle cognitif. En plus de s'inspirer de ces travaux récents, une de nos méthodes pour appliquer la théorie du processus dual s'inspire des modèles computationnels du cortex cingulaire antérieur (CCA) [**69, 70, 48, 32**] qui semble être la zone cérébrale impliquée dans la décision de passer d'un traitement habituel ou contrôlé à un moment donné. Enfin, une de nos contributions principale consiste à rendre l'utilisation de ressources cognitive coûteuses pour l'agent afin qu'il apprenne à les utiliser optimalement, cette idée est fortement inspiré de la littérature sur le coût de l'effort cognitif et les hypothétiques avantages qu'il contient [**33, 43, 7**].

Récemment, plusieurs publications ont étudié des architectures prenant inspiration dans la littérature sur le contrôle cognitif, par exemple, [**8**] utilise un contrôleur neuronal pour la génération contrôlable de langage naturel en modulant les activations pendant la passe avant, [**62**] utilise un système de modules neuronaux interagissant de manière éparse qui apprennent

à interagir les uns avec les autres à l'aide de graphes de connectivité appris. L'idée d'utiliser la neuromodulation en fonction des caractéristiques pour induire un comportement a également été introduite par FiLM [**59**].

3.2. Stratégie de calcul adaptative et compression de modèle

Des travaux récents ont commencé à aborder les avantages pour les modèles d'adopter des stratégies de calcul adaptatives, [**14**] utilise les modèles basés sur l'énergie et apprennent un paysage énergétique, ce qui ajuste le budget de calcul sous-jacent en exécutant une procédure d'optimisation plus complexe par rapport à la difficulté du paysage, [**45**] propose un mode 2 qui exécute un modèle du monde pour effectuer la planification et un mode 1 qui tente de faire une version compressée de son action via la minimisation de la divergence.

Notre approche peut aussi être liée à celle du calcul conditionnel [**5**], qui consiste à activer le réseau de neurones de manière sélective en n'activant que certaines parties du réseau à la fois en fonction de la situation pour optimiser l'utilisation des ressources matérielles.

Enfin, notre approche est liée à la compression de modèle [**1**], en particulier l'utilisation du décrochage variationnelle [**41**] dans une de nos méthodes est directement lié aux méthodes de compression bayésiennes [**50**]. Notre méthode s'inscrit aussi dans le courant utilisation la distillation des connaissances pour utiliser des modèles plus compressé [**60, 18**].

3.3. Distillation des connaissances

La distillation des connaissances [**31**] consiste à transférer les connaissances d'un grand réseau de neurones appelé "professeur" dans un réseau de neurones plus petit appelé "étudiant". Cette méthode s'est avérée particulièrement efficace lorsque la quantité de données d'apprentissage est limitée ou que la taille du modèle de l'élève est très petite. Pour cette raison, cette méthode a été utilisée pour compresser les réseaux de neurones artificiels dans de nombreux travaux [**60, 18**].

La distillation de la connaissance a aussi été utilisée pour les mêmes raisons dans l'apprentissage par renforcement avec des performances de compression impressionnantes [**65**].

De façon cruciale, pour une de nos méthodes, [**80**] utilise une politique "distillée" qui tient compte des comportements communs entre les différentes tâches auxquels l'agent est exposé et le pénalise s'il s'écarte de ce comportement. [**43**] remarque un parallèle entre le comportement d'un agent ayant une telle architecture et le comportement tel que compris à travers le prisme du contrôle cognitif.

Enfin [**56**] propose une théorie basé sur le principe de description minimum [**20**] employant la distillation comme implémentation.

3.4. Apprentissage par renforcement sans modèle et basé sur un modèle

Une des distinctions les plus importantes en apprentissage par renforcement est la distinction entre apprentissage par renforcement sans modèle et basé sur un modèle. Les algorithmes populaires hors politique sans modèle DDPG [**49**] et SAC [**22**] représentent des avancées dans l'apprentissage par renforcement profond, basées sur un large corpus de littérature [**75, 53, 29, 38**]. DDPG et SAC apprennent tous deux une politique π_θ et une fonction de valeur Q_θ, mais n'apprennent pas de modèle. L'apprentissage base sur un modèle consiste à apprendre un modèle de l'environnement qui peut être utilisé pour la planification [**15, 24, 51, 66, 58**] ou pour l'entraînement d'un algorithme sans modèle avec des données générées [**61, 21, 26, 67**]. Par exemple, [**84, 21, 24, 26**] apprend un modèle dynamique en utilisant une perte de prédiction vidéo, [**83, 40**] considèrent la RL basée sur un modèle dans le cadre hors ligne, et MuZero/EfficientZero [**66, 82**] apprennent un modèle dynamique latent en utilisant la prédiction de la récompense. Plusieurs travaux antérieurs visent à développer des algorithmes qui combinent des éléments de l'apprentissage par renforcement sans modèle et basés sur un modèle [**57, 9, 61, 23, 6, 52, 12**] et [**9, 51**] utilise un modèle appris pour améliorer la politique et l'apprentissage de la valeur à travers des trajectoires générés. [**71**] étend SAC avec un modèle de prédiction d'état appris et contraint les trajectoires planifiées à être proches de celles de SAC, tandis que nous remplaçons la politique paramétrée par une planification avec TD-MPC et apprenons un modèle de dynamique latente orienté vers les tâches. De façon générale, la plupart de ces techniques ne s'inspirent pas de la théorie du processus dual dans leurs conceptions. Dans ce mémoire, nous avons choisi d'expérimenter en nous basant sur [**25, 24**] car ces travaux sont les plus représentatifs de l'état de l'art de l'apprentissage par renforcement profond [**27**].

Chapitre 4

Implémentation du processus dual via la distillation des connaissances

Vue d'ensemble

Cette section décrit la première méthode employée pour implémenter la théorie du processus dual dans les réseaux de neurones artificiel, cette méthode se fonde sur des travaux récents ayant des fondements théoriques solides expliquant les bénéfices de la théorie du processus dual d'un point de vue statistique et de la théorie de l'information [**56, 55**], cette méthode utilise la distillation de la connaissance pour faire en sorte qu'une politique dites "par défaut", inspiré du système 1, habituel rapide et automatique compresse le comportement d'une politique dites "contrôle" plus lente, plus coûteuse, mais plus expressive représentant le système 2. En procédant de la sorte, nous pouvons tirer avantage des deux mondes, lorsqu'un exemple est difficile, soit intrinsèquement, soit parce qu'il est nouveau pour l'agent [**19**], le modèle peut employer le réseau de neurone contrôle pour tirer avantage de plus de capacité de calcul afin de faire moins d'erreur et apprendre plus vite [**39**], et au fur et à mesure de l'entrainement quand certains exemples deviennent faciles pour le modèle, il peut utiliser le réseau de neurone par défaut qui a l'avantage d'être moins sujet au surapprentissage, possède une meilleure capacité de généralisation à des nouveaux environnements [**80**] et est plus rapide lors de l'inférence [**55**]. En rendant l'utilisation du réseau de neurone contrôle coûteuse pour l'agent via un terme de régularisation appris, l'agent apprend naturellement lors de l'entrainement à utiliser la politique contrôle uniquement lorsque nécessaire. Nous introduisons également la tâche que nous avons créée pour évaluer les modèles utilisant le contrôle cognitif et contrôler pour la robustesse au surapprentissage Stroop MNIST. Enfin, nous décrivons des méthodes alternatives que nous avons expérimentées base sur les mêmes principes, une utilisant l'entropie du softmax du réseau par défaut pour déterminer l'activation du réseau contrôle et l'autre modifiant la longueur de l'horizon lors de la planification,

nous finissons cette section en mettant en lumière les différentes hypothèses, limitations et travaux ultérieurs que nos travaux permettent de considérer.

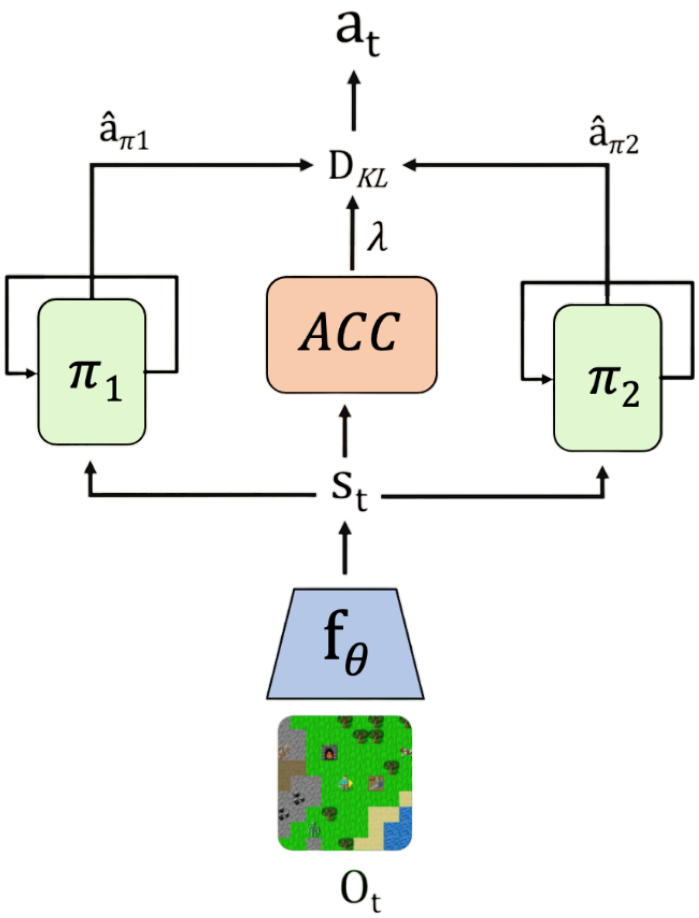


Fig. 4.1. Illustration de notre architecture principale CCA (ACC en anglais) lorsque l'agent est exposé à un exemple, un réseau de neurone spécifique représenté ici en orange va être chargé de calculer un terme de régularisation $\boldsymbol{\lambda}$, ce terme est un coefficient modulant le coût de la divergence entre les deux modèles, permettant ainsi d'apprendre à assigner au système 1 la politique par défaut $\boldsymbol{\pi_1}$ ou au système 2 $\boldsymbol{\pi_2}$la politique contrôle en modulant le terme de régularisation $\boldsymbol{\lambda}$. Si la divergence D_{KL} est nulle ou quasiment nulle la politique contrôle est désactivée pour les prochains exemples jusqu'à ce qu'elle redépasse un certain seuil établi par un hyperparamètre .

4.1. Approche

Notre approche s'appuie sur le formalisme "MDL-C" [**55**], un formalisme récent pour l'apprentissage par renforcement basé sur le principe de longueur de description minimale (MDL) [**20**], le principe de longueur de description minimale stipule que la meilleure description des données est celle du modèle qui compresse le mieux les données. En d'autres

termes, l'apprentissage d'un modèle des données, par exemple pour la prédiction, c'est capturer les régularités dans les données et toute régularité dans ces données peuvent être utilisées pour mieux les compresser. Ainsi, plus nous pouvons compresser les données, plus nous en apprenons sur elles et mieux, nous pouvons les prédire.

L'idée clé dans MDL-C consiste à désigner, comme "données" à compresser, le comportement du réseau de neurone, le papier original se concentre sur l'apprentissage par renforcement profond, dans ce contexte particulier le comportement du réseau de neurone est donc la politique comportementale d'un agent d'apprentissage par renforcement, nous décidons de nous inspirer de leurs notations en désignant la politique "contrôle" par $\boldsymbol{\pi_2}$, et en suivant la logique du MDL-C, nous supposons également un "modèle" de cette politique, le réseau de neurone "par défaut" $\boldsymbol{\pi_1}$. Lorsque les deux réseaux sont actifs, le softmax des logits de leurs activations respectives sont comparées via la divergence de Kullback-Leibler et ajouté au coût comme un terme de régularisation, procédant ainsi à une distillation de la connaissance [**31**]. Cela nous donne le terme régularisation suivant lors de l'entrainement :

$$\mathcal{L}_{KL} = D_{KL}\left(\pi_2\left(a_{\pi_2} \mid x_t; \theta\right) \| \pi_1\left(a_{\pi_1} \mid x_t; w\right)\right)$$

Ce terme a pour effet que la politique par défaut $\boldsymbol{\pi_1}$ est entrainée à correspondre à la politique contrôle, en ajoutant à cela le fait que $\boldsymbol{\pi_1}$ doit être plus compacte que $\boldsymbol{\pi_2}$, ici via l'utilisation du décrochage variationnel, encourageant le modèle à être éparse [**41, 50**] c'est-à-dire à utiliser très peu de paramètres synaptiques non nuls. Cela a pour effet que $\boldsymbol{\pi_1}$ ne peut et doit capturer les régularités dans le comportement de $\boldsymbol{\pi_2}$, ce qui correspond intuitivement au concept d'habitude. Et $\boldsymbol{\pi_2}$ est pénalisé s'il s'éloigne trop de $\boldsymbol{\pi_1}$ ce qui a été associé au coût cognitif de dévier de ses habitudes [**43**].

Nos contributions majeures aux approches précédentes sont les suivantes, dans le papier originel de MDL-C et les travaux connexes associés, les deux politiques ne peuvent jamais interagir indépendamment, cela signifie que même lorsque la divergence entre les deux politiques $\mathcal{L}_{KL} = D_{KL}$ est nulle $\boldsymbol{\pi_2}$ est quand même appelé inutilement. En outre, le coût de la divergence entre les softmax des deux politiques est toujours fixe. Cela signifie que lors que le modèle n'a jamais d'incitant à apprendre quand utiliser la politique contrôle. Pour surmonter ces limitations, inspiré par la littérature sur le coût du contrôle cognitif [**43**], nous ajoutons au modèle un mécanisme permettant de sélectionner l'activation d'un réseau de neurone plutôt que l'autre en fonction de la situation, en lui permettant d'apprendre un coefficient $\boldsymbol{\lambda}$ modulant le coût de la divergence $\mathcal{L}_{KL}$. La subtilité de notre approche est que transformer ce coefficient $\boldsymbol{\lambda}$ en terme appris, permet de reproduire un des rôles computationnel hypothétique d'une partie du contrôle cognitif [**69**] le réseau de neurone calculant ce coefficient $\boldsymbol{\lambda}$ que l'on va nommer $\boldsymbol{ACC}$ va apprendre à quel point une situation vaut la peine de dévier de la politique par défaut et donc de payer un coût cognitif supplémentaire. En plus de cela, nous permettons à $\boldsymbol{\pi_2}$ et $\boldsymbol{\pi_1}$ d'être exécuté indépendamment, nous supposons

que $\mathcal{L}_{KL}$ est une bonne mesure de la difficulté d'un exemple, intuitivement si la divergence pour un exemple est grande, alors les réseaux ne sont pas d'accord sur la réponse à apporter à cet exemple, il s'agit donc d'un exemple qui est plus difficile pour l'agent, les méthodes, dites ensemblistes tirent avantages de ce postulat pour créer une mesure de l'ambiguïté d'un exemple [**19, 68**], si l'exemple n'est pas intrinsèquement difficile alors une grande valeur pour $\mathcal{L}_{KL}$ signifie que le réseau de neurone par défaut n'a pas encore suffisamment compressé le comportement de l'agent pour cette tâche et donc que l'utilisation de $\boldsymbol{\pi_2}$ pour trouver des solutions plus rapidement par son expressivité est avantageuse [**39**]. Ainsi, dans notre méthode lorsque, $\mathcal{L}_{KL}$ est nulle ou en dessous d'un certain seuil établi par un hyperparamètre, $\boldsymbol{\pi_2}$ est désactivé pour les prochains exemples. En plus de permettre un gain en terme d'utilisation de ressources, la possibilité pour les deux politiques d'interagir indépendamment fait qu'elles développeront des caractéristiques différentes et complémentaires lors de l'entrainement de façon similaire a ce qui a déjà été observé dans [**44**], intuitivement, $\boldsymbol{\pi_2}$ pourrait se concentrer sur les caractéristiques des exemples plus difficiles et $\boldsymbol{\pi_1}$ des exemples plus simples, ce qui serait plus cohérent avec la littérature sur le contrôle cognitif, comme il a été remarqué par [**56**].

Ainsi, nous obtenons un objectif d'optimisation pour les deux politiques qui pondère le terme RL standard, favorisant une récompense attendue élevée (R), par rapport aux deux termes de l'objectif LDM dans le contexte de l'apprentissage par renforcement :

$$\mathbb{E}_{\pi}[R] - [L(\pi_1) + \lambda L(\pi_2 \mid \pi_1)] \tag{4.1.1}$$

Où L($\boldsymbol{\pi}$) est la longueur de description du réseau de neurone, c'est-à-dire le nombre de bits de la théorie de l'information nécessaires pour coder ce modèle, une mesure de la complexité.

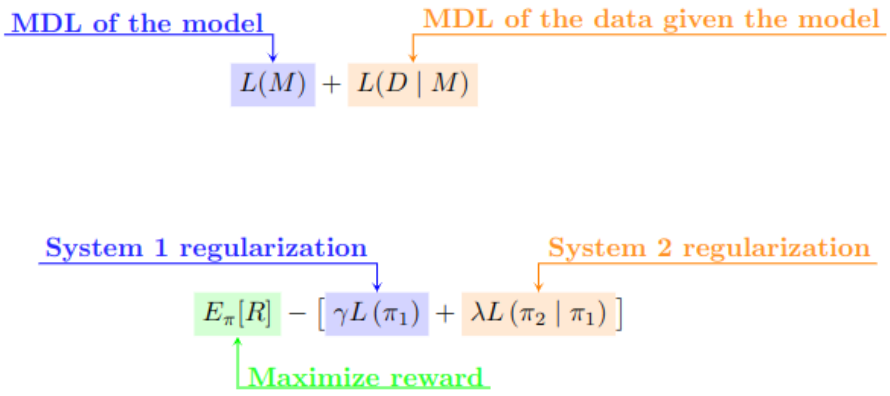


Fig. 4.2. Implémentation du principe de longueur de description minimal dans notre approche, la pénalisation de la complexité de $\boldsymbol{\pi_1}$ équivaut à réduire la longueur de description du modèle alors que la régularisation forçant $\boldsymbol{\pi_1}$ à être proche de $\boldsymbol{\pi_2}$ est l'équivalent de réduire la longueur de description des données sachant le modèle.

Lorsque tous ces éléments sont mis ensemble, cela nous donne la fonction de coût finale de MDL-C adapté à notre approche, c'est-à-dire :

$$\mathcal{L}_{\text{MDL-C}} = E_\pi[R] - \Big[\beta \bar{D}_{KL}(q(w;\phi)\|p(w)) + [\lambda D_{KL}\left(\pi_2\left(a_{\pi_2} \mid x_t;\theta\right)\|\pi_1\left(a_{\pi_1} \mid x_t;w\right)\right)]$$

Voilà qui conclut le formalisme de l'approche et le formalisme de MDL-C pour plus de détails voir [**56**] [**55**] [**20**].

$$\mathcal{L}_{\text{MDL-C}} = E_\pi[R] - \big[\, \beta \bar{D}_{KL}(q(w;\phi)\|p(w)) + [\lambda D_{KL}\left(\pi_2\left(a_{\pi_2} \mid x_t;\theta\right)\|\pi_1\left(a_{\pi_1} \mid x_t;w\right)\right) \,\big]$$

Fig. 4.3. La fonction de coût de MDL-C (en jaune) adapté à notre approche, il s'agit de l'équation dans 4.2 développé à notre cas d'utilisation précis, : le modèle essaie de maximiser sa récompense attendue comme tout agent d'apprentissage par renforcement (en vert) la pénalisation de la complexité de $\boldsymbol{\pi_1}$ (en bleu) équivaut à la divergence Kullback-Leibler entre avec les poids synaptique du modèle et des poids ayant un certain à priori dans leur distribution facilitant la simplicité du modèle voir [**56**], enfin la divergence entre les deux modèles est pénalisé (en orange), et modulé par le coefficient $\boldsymbol{\lambda}$.

4.1.1. Implémentation

Dans notre implémentation, $\boldsymbol{\pi_1}$ est un réseau neuronal compressé régularisé par le biais du décrochage variationnel [**41**]. Le décrochage variationnel suppose que les poids synaptiques d'un réseau neuronal sont soumis à un bruit gaussien multiplicatif, étant donné qu'à mesure que le bruit augmente, l'information véhiculée par les poids du réseau diminue, la régularisation peut être comprise comme une régularisation des poids en faveur de la compacité ou de la simplicité de façon similaire à [**56**]. Par souci de compacité ou de simplicité, $\boldsymbol{\pi_2}$ est un réseau neuronal plus coûteux en termes de calcul. Comme nous utilisons le décrochage variationnel $\boldsymbol{\pi_1}$ et $\boldsymbol{\pi_2}$ peuvent avoir la même architecture tout en imposant à $\boldsymbol{\pi_1}$ de rester moins complexe par le fait qu'il soit régularisé pour éliminer ses poids synaptiques, $\boldsymbol{ACC}$ est chargé de calculer $\boldsymbol{\lambda}$ le coefficient du terme de régularisation D_{KL} qui influencera si le réseau contrôle $\boldsymbol{\pi_2}$ sera actif ou non au prochain exemple.

Dans l'intégralité de nos implémentations, les réseaux neuronaux sont des réseaux de neurones récurrents, plus précisément des GRU (Gated recurrent unit) [**11**] qui reçoivent le même état caché par leurs entrées perceptuelles.

À chaque pas de temps t, après être passé dans un encodeur, $\boldsymbol{f_\theta}$ l'état est distribué aux trois modules, le réseau par défaut $\boldsymbol{\pi_1}$ et le réseau calculant le coût cognitif $\boldsymbol{ACC}$ sont toujours actifs, $\boldsymbol{\pi_2}$ n'est actif que lorsque la divergence $\mathcal{L}_{KL} = D_{KL}\left(\pi_2\left(a \mid x_t;\theta\right)\|\pi_1\left(a \mid x_t;w\right)\right)$ a été au-dessus d'un certain seuil. Cela mène à une politique contrôlée $\boldsymbol{\pi_2}$, qui exploitera son

expressivité lorsque que le modèle estime que le coût attendu de la déployer en vaut la peine. Cela est appris directement par réseau de neurone **ACC** qui, par le fait qu'il ne reçoit l'état en entre et ne peut influencer le résultat des actions et donc les récompenses obtenues qu'en générant le terme de régularisation, $\boldsymbol{\lambda}$ devrait apprendre quand il est nécessaire d'activer la politique par défaut ou la politique contrôle en fonction des situations.

Pour tester notre méthode, nous employons l'agent d'apprentissage par renforcement profond "Dreamer", Dreamer est un choix d'architecture intéressant sur lequel construire notre méthode pour plusieurs raisons, tout d'abord, il est très représentatif des méthodes à l'état de l'art en apprentissage par renforcement profond basé sur le modèle [**27**], c'est une architecture souvent utilisée comme base de référence [**28, 68**], c'est une architecture coûteuse computationnellement, et contrairement à certaines méthodes de planification Dreamer utilise un softmax comme sortie lors de la planification dans son modèle du monde ce qui le rend compatible avec notre méthode pour plus détails sur l'implémentation de Dreamer voir [**25**].

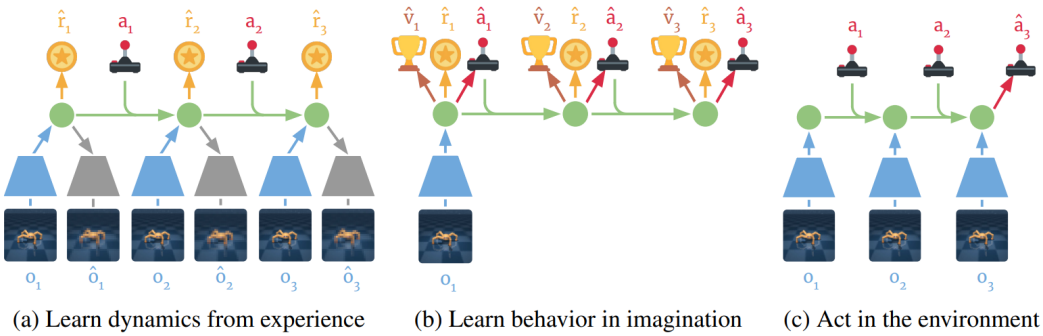


Fig. 4.4. L'agent d'apprentissage par renforcement profond "Dreamer", après avoir récolté des données dans l'environnement, il apprend un modèle du monde qui lui permet de simuler des observations et apprendre une politique sans voir des nouvelles données, la nouvelle politique est ensuite réutilisée pour récolter des donnes plus intéressantes dans le vrai environnement. Figure de [**25**].

4.1.2. Stroop MNIST : Une tâche pour tester la robustesse au surapprentissage et les modèles utilisant le contrôle cognitif

Le test de Stroop, développé par le psychologue John Ridley Stroop en 1935 [**74**], est l'un des tests psychologiques les plus utilisés dans la recherche sur le contrôle cognitif. Il est conçu pour mesurer l'interférence cognitive et évaluer les capacités des individus à gérer des informations contradictoires ou incongruentes. Ce test repose sur l'idée que la lecture d'un mot peut interférer avec la capacité de nommer la couleur dans laquelle ce mot est écrit, générant ainsi un conflit cognitif. L'interférence cognitive se produit lorsque la tâche de nommer la couleur est compromise par la tendance automatique à lire le mot. Dans la condition incongruente, l'interférence cognitive est maximale, car il y a un conflit entre la couleur du mot et le mot lui-même. Les participants mettent généralement plus de temps à nommer la couleur dans cette condition par rapport aux conditions congruente et neutre. Ainsi, intuitivement, les cas congruents ou le mot est écrit dans la même couleur sont des échantillons faciles, et les cas incongruents ou les deux sont différents sont échantillons plus difficiles.

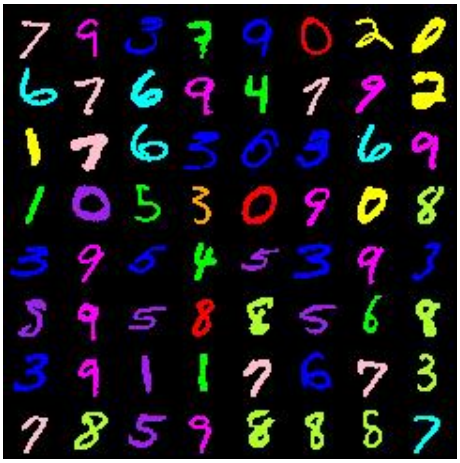
Fig. 4.5. Le test de Stroop, lors des cas dits congruents, les participants doivent nommer la couleur d'une série de mots écrits dans la même couleur (par exemple, le mot "rouge" écrit en rouge, lors des cas dit incongruents, les participants doivent nommer la couleur d'une série de mots écrits dans une couleur différente (par exemple, le mot "rouge" écrit en vert). Dans la condition neutre, les participants doivent lire le mot sans se soucier de sa couleur.

Cette tâche nous donne donc un point de référence tout trouver pour évaluer le comportement de notre architecture, mais aussi évaluer la robustesse de nos modèles aux surapprentissage, en effet le test de Stroop peut être lié au concept de surapprentissage en apprentissage machine, ce qui cause la confusion lors des exemples incongruent est l'association forte entre la couleur et le mot. Il serait donc intéressant de voir à quel point les modèles arrivent à performer en faisant abstraction de ces associations. Nous nous inspirons de cette tâche pour

proposer une tâche similaire aux modèles d'apprentissage profond, nous utilisons les données de la tâche de référence en apprentissage profond MNIST [**13**].

Nous créons ainsi une tâche qui associe à chaque nouvelle exécution une couleur avec un chiffre de MNIST avec une certaine probabilité. Pour rester fidèle au test de Stroop, nous générons une majorité d'exemples congruents ou un chiffre va être associé à une couleur avec haute probabilité, dans nos expérimentations, nous choisissons 80%, le reste sont des exemples incongruents prenant une autre couleur que celle à laquelle ils ont été assignés à l'exécution du programme avec une probabilité égale pour toutes les autres couleurs (comme il s'agit de MNIST, il y a 10 chiffres donc une couleur en particulier à 11% de chance d'apparition pour une couleur donné d'apparaitre sur exemple incongruent.

Le modèle est ensuite exposé à la même tâche que pour MNIST prédire le bon chiffre pour un exemple donné [**13**] comme il est exposé à une association excessive entre une couleur est un chiffre Stroop MNIST va permettre de mettre en lumière quel sont les modèles qui résistent à ces associations et lesquels ne le font pas.



((a)) Un lot d'exemples congruents ou la majorité des chiffres sont associé à la même couleur.

((b)) Un lot d'exemples incongruents ou la majorité des chiffres sont associé une couleur differente.

Fig. 4.6. Deux lots de données générés avec Stroop MNIST en apprenant avec une majorité d'exemples congruents puis en exposant le modèle à des exemples incongruents, nous pouvons reproduire une situation similaire au test Stroop pour les réseaux de neurones artificiels et étudier comment ils se comportent.

L'intérêt principal de Stroop MNIST est donc premièrement d'observer à quel point inclure des associations congruentes et incongruentes dans les exemples modifie la précision des modèles en les "piégeant" à surapprendre ces associations. Mais Stroop MNIST est aussi intéressant pour sa similarité avec le test de Stroop originel, il permet de tester le comportement des modèles et observer s'il correspond avec certaines observations de la littérature sur le contrôle cognitif.

Une fois une tâche de Stroop MNIST généré, nous pouvons entrainer un modèle, observer son comportement et tirer des conclusions non seulement par rapport au contrôle cognitif, mais aussi par rapport à la théorie soutenant les choix architecturaux de ce mémoire MDL-C [**55**].

4.2. Expériences & Résultats

Afin d'évaluer notre méthode, nous avons utilisé principalement six taches différentes, les tâches de références de l'apprentissage par renforcement, DeepMind Control Suite (DMC)[**79**], l'architecture que nous utilisons Dreamer étant particulièrement lourde computationellement, pour faciliter les expérimentations, nous avons testé la majorité de nos hypothèses sur le comportement du modèle sur Stroop MNIST. Ces tâches sont très complémentaires, car les tâches de référence de l'apprentissage par renforcement permettent d'évaluer le gain en termes de performance sur un agent de référence, et Stroop MNIST permet une étude qualitative rigoureuse inspirée par une tâche référence du contrôle cognitif.

4.2.1. Contrôle optimal sur DMC

Nous évaluons notre méthode sur 6 tâches du DeepMind Control suite [**78**] nous nous inspirons de [**25**], les observations données à l'agent sont des images RGB de taille 64x64x3 Les actions vont de 1 à 12 dimensions, les récompenses vont de 0 à 1, les épisodes durent 1000 pas et ont des états initiaux aléatoires. Nous utilisons une répétition d'actions fixe de R = 2 pour toutes les tâches. Nous évaluons notre agent sur 1 million de pas d'environnement. Tout d'abord, pour nous assurer qu'une comparaison équitable, nous avons comparé notre implémentation de Dreamer en Pytorch à l'implémentation originelle en Tensorflow qui se faisait vieillissante et posait tout un tas de problème à mettre à l'échelle pour tester notre méthode. Nous avons évalué notre implémentation par rapport aux résultats officiels de Dreamer pour nous assurer de la validité de nos résultats et avons obtenu les résultats suivants :

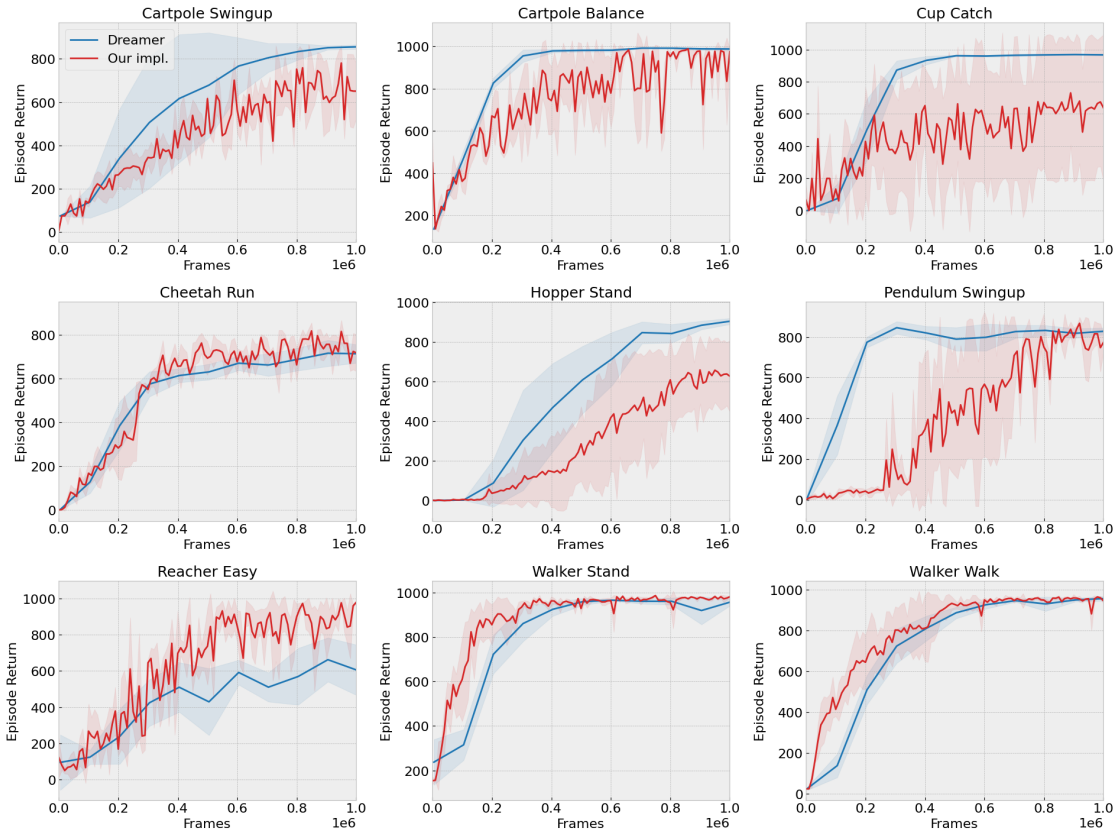


Fig. 4.7. Notre implémentation en Pytorch a des résultats compétitif avec ceux du papier original [**25**] ce qui confirme la validité de notre cadre experimental.

Notre implémentation avait du mal à atteindre les mêmes performances que Dreamer sur certaines tâches, nous avons finis par émettre l'hypothèse que ces différences étaient dues aux différences de bibliothèques d'implémentation et à la difficulté inhérente à reproduire des expériences d'apprentissage par renforcement profond. Nous avons exécuté plusieurs implémentations différentes pour confirmer cette hypothèse et il s'est avéré que la bibliothèque impliquée influence beaucoup la variance des résultats.

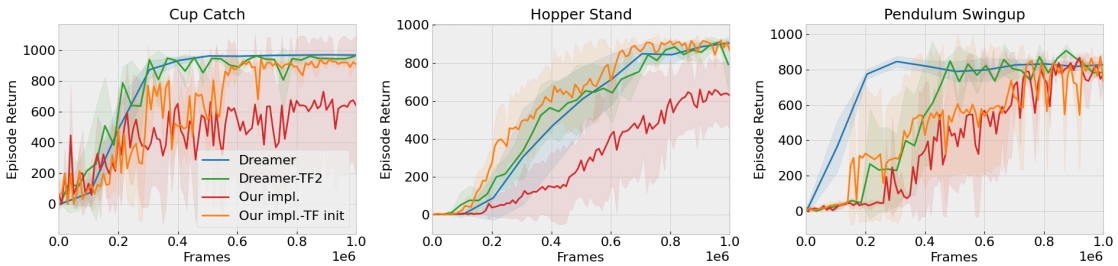


Fig. 4.8. En procédant à des différences mineures en termes d'implémentation, les résultats peuvent être assez differents.

L'implémentation de notre méthode principale d'expérimentation est en PyTorch toutes les expériences ont été réalisées sur un seul GPU **"NVIDIA GeForce RTX 3080"** équipé d'un CPU **"AMD Ryzen Threadripper PRO 3945WX 12-Cores"**. L'implémentation de base Dreamer prend en moyenne notre implémentation 20 h 26 et ACC 18h49 qui est un gain de temps non négligeables pour des architectures aussi lourdes. Nous parvenons à avoir ce gain en termes de capacité de calcul sans avoir de baisse de performance notable.

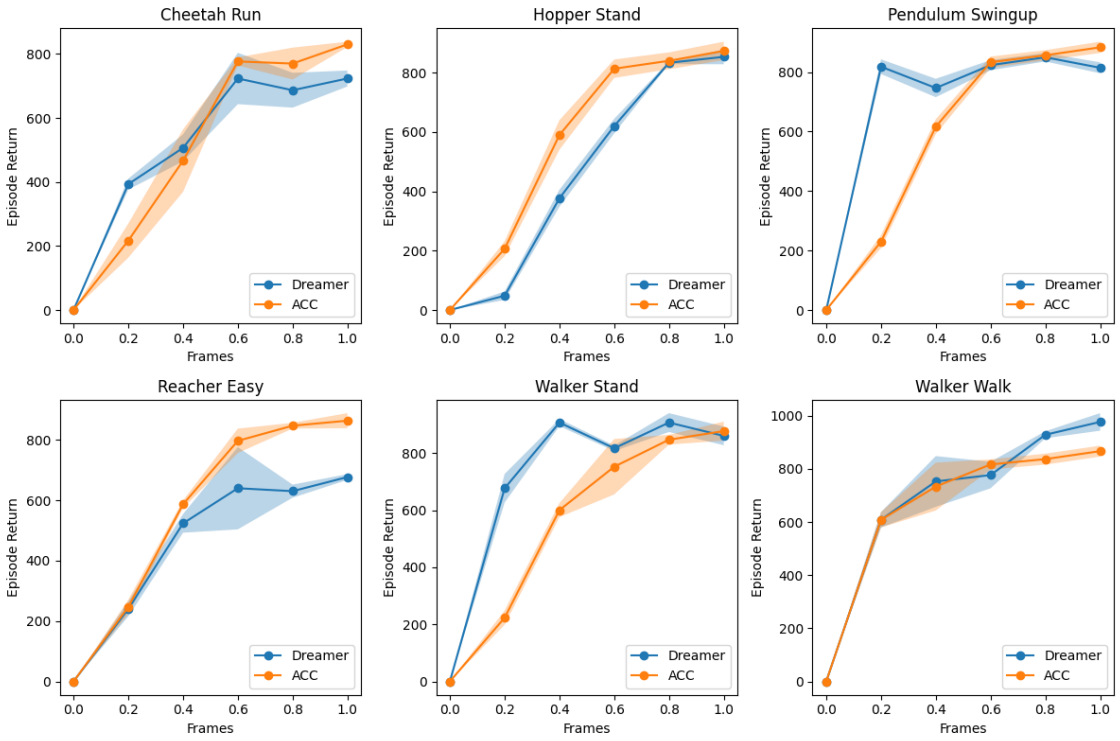


Fig. 4.9. Résultats similaires à Dreamer Figure de [**25**].

Tâche	**Dreamer**	**ACC**
Cheetah Run	20h37	18h37
Hopper Stand	19h17	17h40
Pendulum Swingup	21h17	18h37
Reacher Easy	20h25	18h17
Walker Stand	21h05	19h56
Walker Walk	19h57	19h24
`Temps d'entrainement moyen`	**20h26**	**18h49**

Tableau 4.1. Temps d'entrainement sur les tâches de contrôle classique, ACC est systématiquement plus rapide que Dreamer pour des performances comparables.

En plus des gains en termes de temps de calcul, nous nous inspirons de la méthodologie de [**55**] et faisant passer aux modèles une série de tâches séquentielles pour analyser sa

capacité à généraliser à des nouveaux environnements, le formalisme est le suivant , les tâches sont échantillonnées une à une, uniformément sans remplacement, la politique par défaut $\boldsymbol{\pi_1}$ est conservée pour toutes les tâches, mais $\boldsymbol{\pi_2}$ est réinitialisé. L'objectif de l'agent est d'accélérer l'apprentissage sur chaque tâche successive, mesuré par le regret cumulé. Pour tester, nous utilisons les 3 taches "Walker" du DMC, "Walker run", "Walker stand" et "Walker walk". Pour des raisons de ressources computationelles, nous n'avons pas pu lancer toutes les expériences mais nous observons de façon similaire à [**55**] que l'approche permet une meilleure généralisation à des nouveaux environnements sur les premières frames observées pendant l'entrainement.

4.2.2. Analyse quantitative et qualitative sur Stroop MNIST

Nous commençons par une analyse quantitative du modèle sur Stroop MNIST en termes de performance.

Afin qu'il soit sensible aux couleurs RGB nous évaluons sur Stroop MNIST avec un réseau convolutif ayant l'architecture suivante :

```
self.conv1 = nn.Conv2d(3, 32, kernel_size=3, stride=1)
self.conv2 = nn.Conv2d(32, 64, kernel_size=3, stride=1)
self.fc1 = nn.Linear(64 * 5 * 5, 128)
self.fc2 = nn.Linear(128, 10)
```

Chaque convolution est suivie de la fonction d'activation ReLu et d'une opération de max pooling [**44**]. C'est notre modèle de référence, notre méthode correspond à cette architecture sauf qu'il y a deux réseaux convolutifs et l'un d'eux est soumis au décrochage variationnel et la réponse finals du modèle est une combinaison linéaire des softmax pour coller à [**56**].

Pour évaluer les deux modèles, nous utilisons la même méthodologie que pour MNIST [**13**] nous calculons leur précision sur les données d'évaluation avec 3 seeds et calculons leurs moyennes. Comme attendu, notre méthode étant plus robuste au surapprentissage obtient de bien meilleures performances.

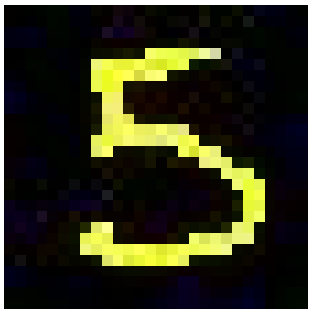
Run	**Convnet**	**ACC**
1	65%	82%
2	57%	88%
3	62%	84%
`Précision moyenne`	**61.3%**	**84.6%**

Tableau 4.2. Précision sur 3 runs différentes sur l'ensemble d'évaluation pour Stroop MNIST ACC est systématiquement plus performant d'une grande marge, nous faisons l'hypothèse que c'est une conséquence de la robustesse de notre méthode au surapprentissage.

Nous procédons ensuite à une évaluation qualitative du modèle, si nos hypothèses sont correctes alors plusieurs comportements devraient être observés sur le modèle, premièrement

la politique par défaut $\boldsymbol{\pi_1}$ devrait être biaisé par les associations de couleur comme dans [**56**]. Ici, c'est ce que nous observons que comme attendu $\boldsymbol{\pi_1}$ est piégé par les exemples congruents alors que $\boldsymbol{\pi_2}$ pas.

((a)) Exemple congurent de ce jet de donnée, 2 a été associé au jaune à la creation 80% des 2 dans l'ensemble d'entrainement sont donc jaune.



((b)) Exemple incongruent le 5 est jaune contrairement ce qui va mener les modèles ayant surappris à le prendre pour un 2.

Fig. 4.10. Nous analysons le comportement du modèle face à ce type d'exemples, ici 2 a été associé à la couleur jaune à la création de la tache et nous allons observer si comme attendu le modèle classe le chiffre 5 comme étant un 2.

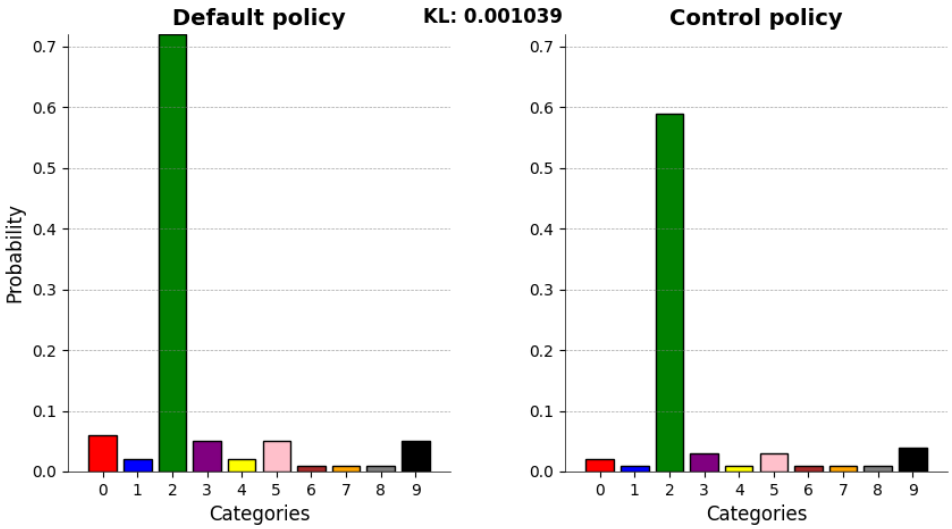


Fig. 4.11. Les distributions des logits des deux politiques pour l'exemple du 2 jaune 4.10 ici les deux politiques classent bien le chiffre avec la politique par défaut qui a plus d'assurance dans son choix

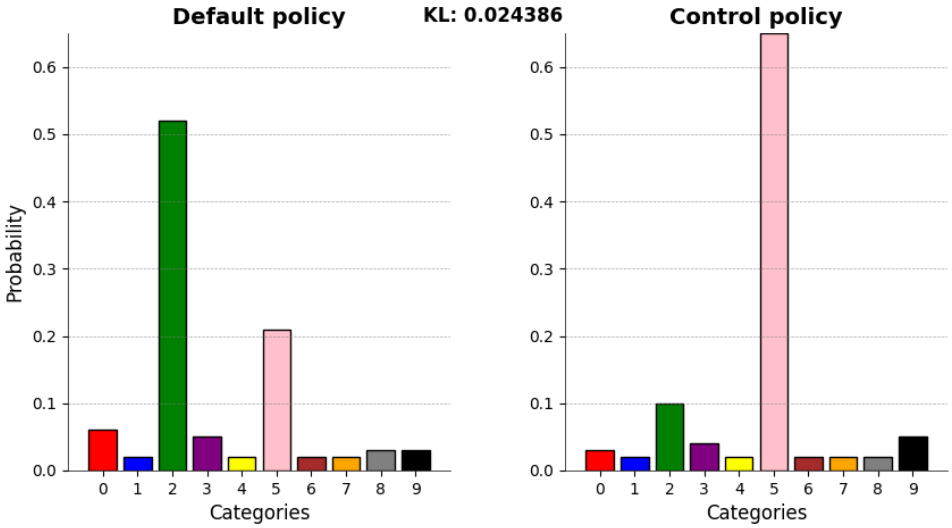


Fig. 4.12. Mais lorsqu'ils font fassent à un exemple incongruent ici le 5 jaune 4.10 la politique par défaut se trompe en pensant que c'est toujours un 2 à cause de la couleur jaune alors que la politique contrôle donne la toujours bonne réponse.

4.2.3. Méthode basée sur l'entropie

Une des alternatives intéressante a notre méthode ACC que nous avons essayé est de plutôt que d'utiliser un réseau de neurone pour déterminer l'activation de $\boldsymbol{\pi_2}$ nous pouvons effectuer une propagation avec $\boldsymbol{\pi_1}$ qui est plus rapide lors d'inférence et utiliser l'entropie du

softmax de $\boldsymbol{\pi_1}$ comme une mesure de l'incertitude et la difficulté de l'exemple, si l'entropie dépasse un certain seuil alors $\boldsymbol{\pi_2}$ est appelé et l'action finale a est une combinaison linéaire des deux softmax. L'avantage évident de cette méthode est sa simplicité par rapport à ACC, elle dispense de l'utilisation d'un réseau de neurone supplémentaire et nécessite moins de changement en terme d'implémentation néanmoins, ce que cette méthode nécessite pour être efficace, bien sélectionné les paramètres, en effet peut être vu comme un hyperparamètre définissant à quel point nous voulons utiliser de la capacité de calcul avant même l'entrainement du modèle, mais cela est trop différent de ce que nous cherchions. Une hypothèse qui rend cette option intéressante est le fait qu'implicitement dans l'entrainement π_1 va être incité à représenter l'entropie de ses décisions avec sagesse, car toute décision mal calibrée serait directement punie par la régularisation, l'étude d'une telle méthode pour améliorer la calibration des modèles pourrait être intéressante.

Un problème qui nous est vite apparu pour la méthode basée sur l'entropie est que l'entropie est mathématiquement définie sur un domaine allant de 0 à $-log$ en base 2 du nombre de classes, dans le contexte de la classification pour MNIST cela équivaut à -log2(10) donc 3.32 mais dans un autre contexte avec un nombre de classes différent comme imagenet et ses 1000 classes, on aurait -log2(1000) et donc 9.965 cela signifie que l'hyperparamètre du seuil changera dès que le nombre de classes change ce qui est une contrainte tout de même assez problématique

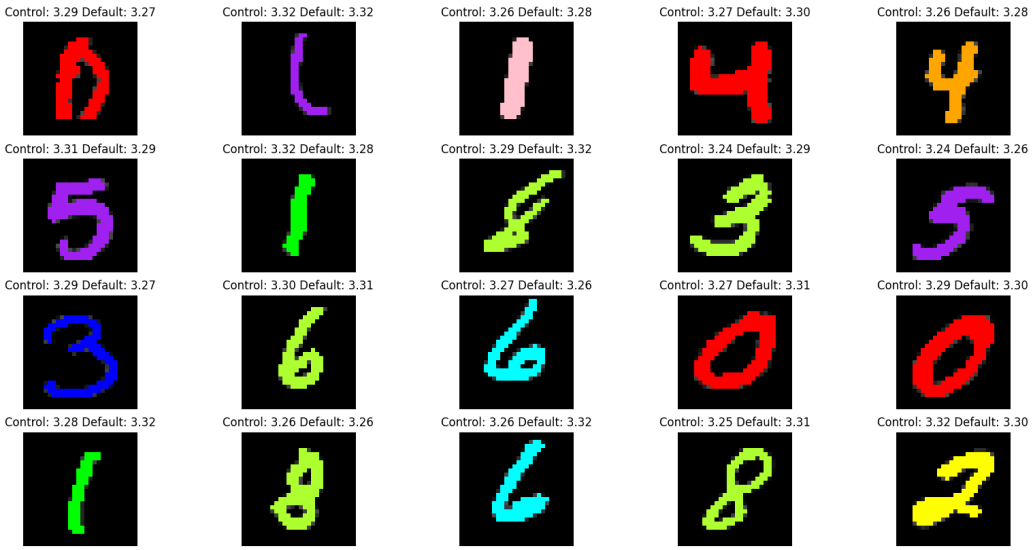


Fig. 4.13. Les entropies des distributions des deux politiques pour un lot de Stroop MNIST. La variance de l'entropie étant très faible dans ce contexte, elle donne en réalité très peu d'information.

4.2.4. Méthode base sur l'horizon de la planification

Une potentielle implémentation de la théorie du processus dual que nous avons voulu tester lors de ce mémoire est basé sur la longueur de l'horizon, l'intuition est la suivante : dans le contexte de la planification, les algorithmes d'apprentissage par renforcement, acteur critique, Π est typiquement une politique paramétrée par un réseau neuronal qui apprend à approximer $\Pi_\theta(\cdot|\mathbf{s}) \approx \arg\max_{\mathbf{a}} \mathbb{E}[Q_\theta(\mathbf{s}, \mathbf{a})]$, c'est-à-dire la politique globalement optimale. En contrôle, Π est traditionnellement mis en œuvre comme une procédure d'optimisation de trajectoire. Pour rendre le problème traitable, on obtient généralement une solution *locale* au problème d'optimisation de la trajectoire à chaque étape t en estimant les actions optimales $\mathbf{a}_{t:t+H}$ sur un horizon fini H et en exécutant la première action $\mathbf{a}_t$, connue sous le nom de *contrôle prédictif de modèle* (MPC) :

$$\Pi_\theta^{\text{MPC}}(\mathbf{s}_t) = \arg\max_{\mathbf{a}_{t:t+H}} \mathbb{E}\left[\sum_{i=t}^{H} \gamma^i \mathcal{R}(\mathbf{s}_i, \mathbf{a}_i)\right]$$

où γ, contrairement à l'itération Q ajustée, est généralement fixé à 1, c'est-à-dire qu'il n'y a pas d'actualisation. Intuitivement, l'équation **??** peut être considérée comme un cas particulier de l'objectif standard de contrôle optimal à coût additif, ce qui est intéressant avec le contrôle prédictif de modèle, c'est que la distinction entre apprentissage par renforcement basé sur le modèle et apprentissage sans modèle peut être vu en terme d'horizon de planification, cela a été remarqué par [**45**], en effet dans cette vision l'apprentissage basé sur le modèle planifie sur un horizon déterminé H, et l'apprentissage sans modèle pourrait être vu comme le même type de comportement, mais avec un H réduit ou nul si cet horizon devenait une variable adaptative comme le $\boldsymbol{\lambda}$ dans 4.1.2 alors, on pourrait avoir un agent qui décide également en fonction d'un état s à quel point il décide d'allouer une capacité de calcul élevé à la situation, il déciderait d'utiliser la planification pour des situations difficiles et ambigües et un H faible pour les situations simples. Le changement de H au cours du temps serait similaire à des passages du mode 2 au mode 1.

Pour tester notre méthode, nous avons d'abord voulu vérifier qu'un nombre plus élevé pour H était en effet associé avec une récompense plus élevée, notre intuition a été confirmée empiriquement

Nous avons ensuite voulu rendre H apprenable pendant l'entrainement, nous avons opté pour une approche similaire à ***ACC*** avec un réseau de neurone supplémentaire qui génère un H à chaque timestep cependant nous avons testé une implémentation de cette idée qui n'a pas donné de résultats concluants, le problème principal que nous avions est que le réseau de neurone restait bloque sur un H donne, nous avons donc stoppé les expérimentations même si nous pensons que des versions plus sophistiqué de cette approche peuvent avoir du potentiel.

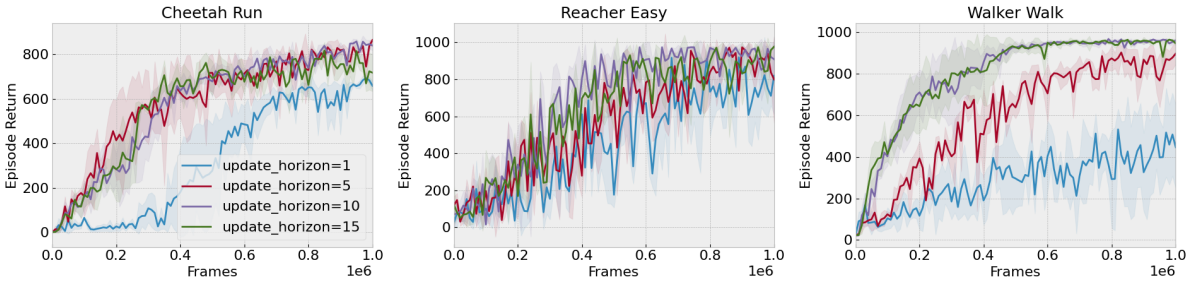


Fig. 4.14. Les performances augmentent avec la taille de l'horizon sur toutes les taches testées.

4.3. Discussions

L'idée originale de la distillation des connaissances vient de l'intuition que l'entrainement et l'inférence sont deux tâches distinctes et par conséquent ont des besoins architecturaux différents qui rendent l'utilisation d'un même modèle pour les deux sous optimal. Cette idée peut être directement lié à la théorie du processus dual et a MDL-C tel que développé dans ce mémoire et [**55**]. Lorsque l'on est confronté à une situation inédite, l'effort cognitif n'a pas besoin d'être déployé à chaque fois lorsque nous avons trouvé une solution. La plupart des algorithmes de recherche ne nécessite pas d'utiliser les mêmes ressources calculatoires pour utiliser les solutions qu'ils ont trouvées, c'est particulièrement vrai dans le contexte de l'apprentissage profond ou l'entrainement peut être vu comme la recherche des poids synaptique ayant un faible taux d'erreur sur les donnes d'entrainement, opération très couteuses, mais une fois fait beaucoup moins onéreuse à déployer, c'est aussi la même logique pour la planification dans l'apprentissage basé sur le modèle.

Notre travail met en œuvre cette idée en laissant à l'agent le soin de décider quand il est suffisamment à l'aise avec la tâche pour y faire face sans avoir besoin de recourir à plus de ressource que nécessaire.

Nous avons montré qu'il est possible d'y parvenir avec un simple module qui agit comme un arbitre du passage entre les deux modes. Nous pensons que les travaux présentés dans ce mémoire et travaux connexes vont s'avérer particulièrement importants pour des domaines comme la compression de modèle [**50**] et la généralisation hors de la distribution [**56**].

4.3.1. Limites

Une limitation importante de notre approche est liée à une limitation de la compression de modèle en général, et effet une avenue prometteuse pour la compression de modèles sont les méthodes qui rendent le modèle et ses paramètres éparses [**50**], dans notre implémentation plutôt que d'utiliser un réseau plus petit en termes de paramètre dès le début ou d'autres méthodes de compression, nous avons décidé d'utiliser le décrochage variationnel

[**41**] comme [**56**]. Le décrochage variationnel est particulièrement intéressant, il permet de se débarrasser de jusqu'a 90% des paramètres d'un modèle tout en conservant les performances [**50**]. Néanmoins, malgré ce ratio impressionnant, les méthodes éparses ne permettent que des gains "négligeables" en termes de gain de temps, d'inférence donner valeur.

La communauté de l'apprentissage profond est très consciente de ce problème qui est lié au hardware des GPU actuel à mesure que des hardware dédiés aux méthodes éparse verront le jour les méthodes comme propose ici gagneront grandement en efficacité.

4.3.2. Hypothèses et travaux ultérieurs

Enfin, nous finissons cette section en émettant quelques hypothèses et en dressant la piste pour des travaux ultérieurs lies aux méthodes discute.

Premièrement, notre méthode est très liée à la théorie de "la valeur attendue du contrôle" [**69**] qui stipule qu'un rôle computationnel probable d'une partie du cerveau implique dans le contrôle cognitif le cortex cingulaire antérieur, est de calculer " La valeur attendue du contrôle" c'est-à-dire à quel point cela vaut le coût d'exécuter un effort cognitif, des nombreux tests psychologiques et données neuroscientifiques ont été bâtis pour tester cette idée, il serait intéressant de la même manière que pour Stroop MNIST d'étudier si notre architecture a un comportement similaire à celui observé dans la littérature.

Une piste intéressante à explorer serait celle évoquée par [**45**], plutôt que d'avoir deux politiques assez similaire, on pourrait faire que $\boldsymbol{\pi_2}$ est une politique de planification très coûteuse et itérative et le rôle de $\boldsymbol{\pi_1}$ serait de compresser cette solution, même si à priori cela ne change rien aux fondamentaux des méthodes employées dans ce papier, c'est une application extrêmement logique de MDL-C.

Enfin, une particularité de notre approche est que d'ordinaire la distillation de la connaissance se produit après entrainement du modèle, ici, elle se fait constamment durant l'entrainement et les modèles continuent même à s'influencer lors de l'inférence, il serait intéressant d'étudier comment ces différentes variations de la distillation de la connaissance impact les modèles.

Chapitre 5

Conclusion

5.1. Discussion

De plus en plus de travaux ont commencé à se rendre compte de la nécessité pour les modèles actuels de développer des stratégies de calculs s'adaptant en fonction du contexte [**45**], et des travaux théoriques commencent à faire la lumière sur les bénéfices computationnels de ce genre d'architecture [**55, 56**], les travaux discutés dans ce mémoire s'inscrivent dans ce courant, en essayant plusieurs approches différentes, en analysant l'impact de leurs implémentations sur les modèles et en développant une nouvelle tâche, nous faisons un pas de plus vers l'intégration de ces méthodes dans les modèles d'apprentissage machine actuel et leur compréhension.

Malgré les progrès récents, il reste encore beaucoup de facettes à évaluer, à quel point les bénéfices, se mettent-ils, à l'échelle avec l'augmentation de la taille du modèle ? Une autre problématique que nous n'avons pas eu le temps d'évaluer dans ce mémoire est la question de la nature discrète ou continue du compromis entre système 1/système 2, basé sur le modèle et sans modèle. C'est un débat compliqué qui ne fait pas encore consensus dans la communauté et demandera encore des efforts pour être abouti. Nous espérons aussi dans le futur l'apparition de benchmark complémentaire à Stroop MNIST qui permettrait l'évaluation des modèles à stratégie de calcul adaptatif plus élaboré, typiquement comme le test de Stroop beaucoup de nouveaux benchmarks pourrait s'inspirer de la littérature sur le contrôle cognitif.

Nous espérons, avec nos contributions, ouvrir la voie vers des algorithmes faisant un meilleur usage de leurs ressources computationnelles et devenant par conséquent plus efficace en termes de coût et de performance, ainsi que permettre une compréhension plus intime des liens qui existent entre certaines méthodes en apprentissage machine et la théorie du processus dual.

5.2. Travaux futurs

Les réseaux neuronaux ont connu un succès remarquable dans un large éventail d'applications, de la classification d'images [**44**] au traitement du langage naturel [**39**]. Cependant, l'une des principales limitations des réseaux de neurones actuels est leur incapacité à gérer les prédictions multimodales où un seul échantillon d'entrée peut correspondre à plusieurs réponses différentes et aussi également plausibles. Dans un contexte de prédiction vidéo, il existe un nombre infini de clips vidéo qui sont des continuations plausibles d'une image, de clips vidéo qui sont des suites plausibles d'un clip donné. En outre, ils ne sont pas en mesure d'allouer de manière adaptative plusieurs étapes de calcul pour les échantillons les plus difficiles, ce qui peut nuire à leurs performances globales et conduire à des erreurs de calcul, ce qui peut entraver leurs performances globales et conduire à une utilisation sous-optimale des ressources informatiques comme soulevé dans 4.2 de calcul. Pour relever ces défis, nous faisons l'hypothèse similaire à [**14**] que l'optimisation du temps d'inférence sur un paysage énergétique appris, le réseau neuronal est capable d'apprendre des réponses réponses multimodales pour des entrées ambiguës, et ajuste naturellement son budget de de calcul en exécutant plus d'étapes d'optimisation pour les problèmes plus difficiles qui conduisent à un paysage énergétique plus complexe. d'énergie plus complexe. Même si l'exploration de ce type de méthode s'est avéré infructueuse dans le cadre de ce mémoire, nous pensons qu'elles représentent une avenue prometteuse pour employer les méthodes de calculs adaptatives.

Les modèles basés sur l'énergie (EBM) [**47**] fournissent un moyen naturel de surmonter les limitations des réseaux neuronaux classiques, un EBM divise le problème de la capture des dépendances entre les variables en deux processus distincts : **inférence** qui consiste à fixer la valeur des variables observées et à trouver les valeurs des variables restantes qui minimisent l'énergie et **apprentissage** qui consiste à trouver une fonction d'énergie qui associe des énergies faibles aux valeurs correctes des variables restantes et des énergies plus élevées aux valeurs incorrectes en utilisant une perte.

Nous proposons une nouvelle méthode pour exploiter les avantages des modèles basés sur l'énergie, en formant une fonction d'énergie paramétrée pour prédire la perte pendant la formation, le réseau neuronal est alors capable d'utiliser l'optimisation du temps d'inférence pour descendre le gradient de l'énergie par rapport à la variable latente et proposer des explications complexes aux dépendances entre les données. Cela permettrait l'apparition de deux propriétés cruciales, la capacité de représenter des solutions multimodales et d'adapter les ressources de calcul à la difficulté de l'échantillon. En entraînant un réseau neuronal à paramétrer un paysage énergétique sur toutes les sorties possibles, le réseau neuronal représente une collection possiblement infinie de prédictions plausibles, la forme locale de ces

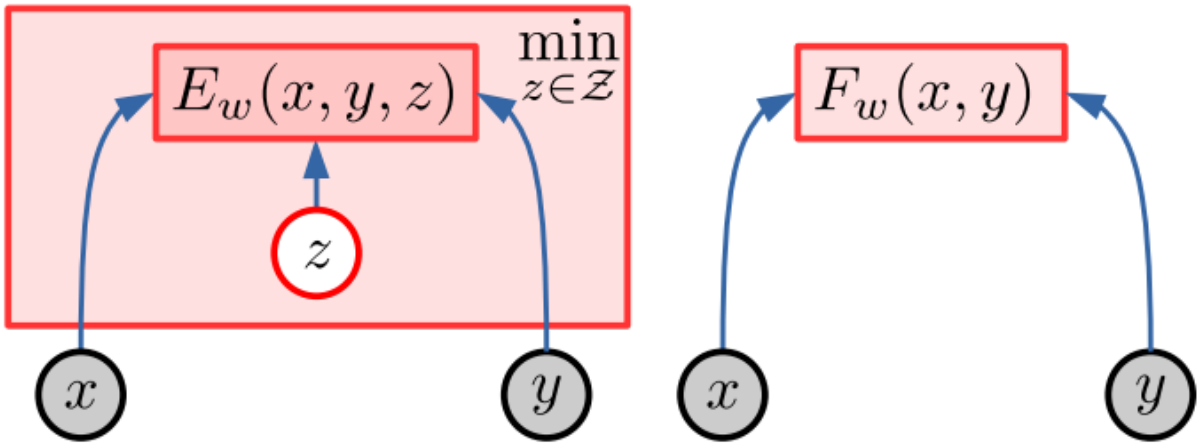


Fig. 5.1. Pour évaluer le degré de compatibilité entre x et y, l'EBM peut avoir besoin d'une variable latente. La variable latente peut être considérée comme paramétrant l'ensemble des relations possibles entre un x et un ensemble de y compatibles. Les variables latentes représentent des informations sur y qui ne peuvent être extraites de x. Par exemple, si x est une vue d'un objet et y une autre vue du même objet, z peut paramétrer le déplacement de la caméra entre les deux vues. L'inférence consiste à trouver la latente qui minimise l'énergie $\check{z} = \text{argmin}_{z \in \mathcal{Z}} E_w(x, y, z)$ Dans l'exemple à double vue, l'inférence trouve le mouvement de caméra qui explique le mieux comment x pourrait être transformé en y. L'énergie résultante $F_w(x, y) = E_w(x, y, \check{z})$ ne dépend que de x et y. Dans l'exemple de la double vue, l'inférence trouve le mouvement de caméra qui explique le mieux comment x pourrait être transformé en y. explique comment x a pu être transformé en y. Figure de [**45**].

prédictions sur le collecteur correspond naturellement à la difficulté de l'échantillon d'entrée puisque les solutions faciles convergeront plus rapidement vers les minima et nécessiteront donc une étape de calcul moins importante.

Un EBM à variables latentes (LVEBM) est une fonction d'énergie paramétrée qui dépend de x, y et z : $E_w(x, y, z)$. Lorsqu'on lui présente une paire (x, y), la procédure d'inférence de l'EBM trouve une valeur de la fonction d'énergie latente. EBM trouve une valeur de la variable latente z qui minimise l'énergie.

$$\check{\boldsymbol{z}} = \underset{z \in \mathcal{Z}}{\text{argmin}} E_w(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$$

Notre méthode consiste à entraîner un réseau neuronal profond pour calculer la fonction d'énergie $F_w(x, y)$ paramétrée avec un vecteur de paramètres w. $E_w(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) : \mathbb{R}^I \times \mathbb{R}^L \times \mathbb{R}^O \rightarrow \mathbb{R}^+$

L'idée principale de notre approche est d'utiliser la fonction d'énergie apprise pour obtenir le gradient par rapport à la variable latente du réseau neuronal qui tente de résoudre la tâche par optimisation du temps d'inférence. Intuitivement, pour un seul échantillon, le réseau neuronal tente de trouver les explications latentes qui expliquent le mieux les dépendances entre X et Y.

Une étape d'explication individuelle est alors représentée comme suit

$$\boldsymbol{z}^t = \boldsymbol{z}^{t-1} - \lambda \nabla_{\boldsymbol{z}} E_w \left(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}^{t-1} \right)$$

Où λ est la taille de chaque étape de descente de gradient. Comme pour [**14**], nous nous arrêtons lorsque $\boldsymbol{z}^t$ est un minima locaux du paysage énergétique $E_w\left(\boldsymbol{z}^t\right) = E_w\left(\boldsymbol{z}^{t-1}\right)$ où des étapes d'optimisation supplémentaires ne sont pas nécessaires.

Nous n'avons pas été capable de produire une implémentation satisfaisante de cette méthode, le problème principale auquel nous faisions fasse est que le modèle était particulièrement lent lors de l'entrainement, en effet notre méthode nécessite qu'à chaque exemple en plus du pas de gradient par rapport au cout $\nabla_\theta \mathcal{L}(x,y)$ le modèle calcule le gradient de la latente $\nabla_{\boldsymbol{z}} E_w\left(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}^{t-1}\right)$ et exécute une propagation avant. Toute implémentation de cette architecture ne finissait pas le cycle d'entrainement sur les données pas même sur MNIST, quelques échantillons prenaient déjà plusieurs dizaines de minutes. Étant donné le temps restreint dans le contexte d'une thèse de mémoire, les contraintes matérielles et la difficulté des méthodes employés, nous n'avons pas eut le temps de trouver des solutions à cette difficulté technique, mais nous sommes confiant qu'à mesure que les méthodes de calculs s'améliorent, des méthodes de ce type vont se retrouver plus communes et plus efficaces.

Références bibliographiques

[1] Jose M Alvarez et Mathieu Salzmann : Compression-aware training of deep networks. *In* I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan et R. Garnett, éditeurs : *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[2] Robert J. N. Baldock, Hartmut Maennel et Behnam Neyshabur : Deep learning through the lens of example difficulty. *CoRR*, abs/2106.09647, 2021.

[3] Adrien Bardes, Jean Ponce et Yann LeCun : Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *CoRR*, abs/2105.04906, 2021.

[4] Richard Bellman : A Markovian Decision Process. *Journal of Mathematics and Mechanics*, 6(5):679–684, 1957.

[5] Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau et Doina Precup : Conditional computation in neural networks for faster models. *CoRR*, abs/1511.06297, 2015.

[6] Mohak Bhardwaj, Sanjiban Choudhury et Byron Boots : Blending mpc & value function approximation for efficient reinforcement learning. *ArXiv*, abs/2012.05909, 2021.

[7] Matthew Botvinick et Todd Braver : Motivation and cognitive control: from behavior to neural mechanism. *Annual Review of Psychology*, 66:83–113, janvier 2015.

[8] Zachary C. Brown, Nathaniel Robinson, David Wingate et Nancy Fulda : Towards Neural Programming Interfaces, février 2021. arXiv:2012.05983 [cs].

[9] Jacob Buckman, Danijar Hafner, G. Tucker, Eugene Brevdo et Honglak Lee : Sample-efficient reinforcement learning with stochastic ensemble value expansion. *In NeurIPS*, 2018.

[10] Ting Chen, Simon Kornblith, Mohammad Norouzi et Geoffrey E. Hinton : A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709, 2020.

[11] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho et Yoshua Bengio : Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[12] Ignasi Clavera, Yao Fu et P. Abbeel : Model-augmented actor-critic: Backpropagating through paths. *ArXiv*, abs/2005.08068, 2020.

[13] Marie-Catherine de Marneffe et Christopher D. Manning : The Stanford typed dependencies representation. *In Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation - CrossParser '08*, pages 1–8, Manchester, United Kingdom, 2008. Association for Computational Linguistics.

[14] Yilun Du, Shuang Li, Joshua B. Tenenbaum et Igor Mordatch : Learning Iterative Reasoning through Energy Minimization, juin 2022. arXiv:2206.15448 [cs].

[15] Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex X. Lee et Sergey Levine : Visual foresight: Model-based deep reinforcement learning for vision-based robotic control. *ArXiv*, abs/1812.00568, 2018.

[16] Jonathan St B T Evans : *Dual-process theories of the social mind*. Oxford University Press, 2008.

[17] Vincent François-Lavet, Peter Henderson, Riashat Islam, Marc G. Bellemare et Joelle Pineau : An introduction to deep reinforcement learning. *CoRR*, abs/1811.12560, 2018.

[18] Jianping Gou, Baosheng Yu, Stephen John Maybank et Dacheng Tao : Knowledge distillation: A survey. *CoRR*, abs/2006.05525, 2020.

[19] Jordi Grau-Moya, Grégoire Delétang, Markus Kunesch, Tim Genewein, Elliot Catt, Kevin Li, Anian Ruoss, Chris Cundy, Joel Veness, Jane Wang, Marcus Hutter, Christopher Summerfield, Shane Legg et Pedro Ortega : Beyond bayes-optimality: meta-learning what you know you don't know, 2022.

[20] Peter Grünwald : *The Minimum Description Length Principle*. 01 2007.

[21] David Ha et Jürgen Schmidhuber : Recurrent world models facilitate policy evolution. *In Advances in Neural Information Processing Systems 31*, pages 2451–2463. Curran Associates, Inc., 2018.

[22] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, G. Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, P. Abbeel et Sergey Levine : Soft actor-critic algorithms and applications. *ArXiv*, abs/1812.05905, 2018.

[23] Muhammad Burhan Hafez, Cornelius Weber, Matthias Kerzel et Stefan Wermter : Curious meta-controller: Adaptive alternation between model-based and model-free control in deep reinforcement learning. *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2019.

[24] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee et James Davidson : Learning latent dynamics for planning from pixels. *In International Conference on Machine Learning*, pages 2555–2565, 2019.

[25] Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba et Mohammad Norouzi : Dream to control: Learning behaviors by latent imagination. *CoRR*, abs/1912.01603, 2019.

[26] Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba et Mohammad Norouzi : Dream to control: Learning behaviors by latent imagination. *ArXiv*, abs/1912.01603, 2020.

[27] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba et Timothy Lillicrap : Mastering diverse domains through world models, 2023.

[28] Nicklas Hansen, Xiaolong Wang et Hao Su : Temporal difference learning for model predictive control, 2022.

[29] H. V. Hasselt, A. Guez et D. Silver : Deep reinforcement learning with double q-learning. *In Aaai*, 2016.

[30] Richard P. Heitz : The speed-accuracy tradeoff: history, physiology, methodology, and behavior. *Frontiers in Neuroscience*, 8, 2014.

[31] Geoffrey Hinton, Oriol Vinyals et Jeff Dean : Distilling the Knowledge in a Neural Network, mars 2015. arXiv:1503.02531 [cs, stat].

[32] Clay B. Holroyd et Tom Verguts : The Best Laid Plans: Computational Principles of Anterior Cingulate Cortex. *Trends in Cognitive Sciences*, 25(4):316–329, avril 2021.

[33] Clay B. Holroyd et Tom Verguts : The Best Laid Plans: Computational Principles of Anterior Cingulate Cortex. *Trends in Cognitive Sciences*, 25(4):316–329, avril 2021.

[34] Russin Jacob, C. O'Reilly Randall et Bengio Yoshua : Deep learning needs a prefrontal cortex, juin 2020.

[35] William James : *The principles of psychology, Vol I.* Henry Holt and Co, New York, 1890.

[36] Daniel Kahneman : *Thinking, fast and slow*. Allen Lane, London, 2011.

[37] Daniel Kahneman et Shane Frederick : Maps of bounded rationality: Psychology for behavioral economics. *American Economic Review*, 93(5):1449–1475, 2002.

[38] Dmitry Kalashnikov, Jacob Varley, Yevgen Chebotar, Benjamin Swanson, Rico Jonschkowski, Chelsea Finn, Sergey Levine et Karol Hausman : Mt-opt: Continuous multi-task robotic reinforcement learning at scale. *ArXiv*, abs/2104.08212, 2021.

[39] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu et Dario Amodei : Scaling laws for neural language models. *CoRR*, abs/2001.08361, 2020.

[40] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli et Thorsten Joachims : Morel : Model-based offline reinforcement learning. *ArXiv*, abs/2005.05951, 2020.

[41] Diederik P. Kingma, Tim Salimans et Max Welling : Variational dropout and the local reparameterization trick, 2015.

[42] Wouter Kool et Matthew Botvinick : A labor/leisure tradeoff in cognitive control. *Journal of Experimental Psychology: General*, 143(1):131–141, 2014.

[43] Wouter Kool et Matthew Botvinick : Mental labour. *Nature Human Behaviour*, 2(12):899–908, décembre 2018.

[44] Alex Krizhevsky, Ilya Sutskever et Geoffrey E Hinton : Imagenet classification with deep convolutional neural networks. *In* F. Pereira, C.J. Burges, L. Bottou et K.Q. Weinberger, éditeurs : *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[45] Yann Lecun : A path towards autonomous machine intelligence, juin 2022.

[46] Yann Lecun, Sumit Chopra et Raia Hadsell : *A tutorial on energy-based learning*. 01 2006.

[47] Yann Lecun, Sumit Chopra et Raia Hadsell : *A tutorial on energy-based learning*. 01 2006.

[48] Sang Wan Lee, Shinsuke Shimojo et John P. O'Doherty : Neural Computations Underlying Arbitration between Model-Based and Model-free Learning. *Neuron*, 81(3):687–699, février 2014.

[49] T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver et Daan Wierstra : Continuous control with deep reinforcement learning. *CoRR*, abs/1509.02971, 2016.

[50] Christos Louizos, Karen Ullrich et Max Welling : Bayesian compression for deep learning, 2017.

[51] Kendall Lowrey, Aravind Rajeswaran, Sham M. Kakade, Emanuel Todorov et Igor Mordatch : Plan online, learn offline: Efficient learning and exploration via model-based control. *ArXiv*, abs/1811.01848, 2019.

[52] G. Margolis, Tao Chen, Kartik Paigwar, Xiang Fu, Donghyun Kim, Sangbae Kim et Pulkit Agrawal : Learning to jump from pixels. *In CoRL*, 2021.

[53] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra et Martin Riedmiller : Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[54] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra et Martin A. Riedmiller : Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[55] Ted Moskovitz, Ta-Chu Kao, Maneesh Sahani et Matthew M. Botvinick : Minimum Description Length Control, juillet 2022. arXiv:2207.08258 [cs].

[56] Ted Moskovitz, Kevin Miller, Maneesh Sahani et Matthew M. Botvinick : A unified theory of dual-process control, 2022.

[57] Anusha Nagabandi, Gregory Kahn, Ronald S. Fearing et Sergey Levine : Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566, 2018.

[58] Tung D. Nguyen, Rui Shu, Tu Pham, Hung Hai Bui et Stefano Ermon : Temporal predictive coding for model-based planning in latent space. *In ICML*, 2021.

[59] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin et Aaron Courville : FiLM: Visual Reasoning with a General Conditioning Layer, décembre 2017. arXiv:1709.07871 [cs, stat].

[60] Antonio Polino, Razvan Pascanu et Dan Alistarh : Model compression via distillation and quantization. *CoRR*, abs/1802.05668, 2018.

[61] Vitchyr H. Pong, Shixiang Shane Gu, Murtaza Dalal et Sergey Levine : Temporal difference models: Model-free deep rl for model-based control. *ArXiv*, abs/1802.09081, 2018.

[62] Nasim Rahaman, Martin Weiss, Francesco Locatello, Chris Pal, Yoshua Bengio, Bernhard Schölkopf, Li Erran Li et Nicolas Ballas : Neural attentive circuits, 2022.

[63] Simon Ramstedt et Christopher J. Pal : Real-time reinforcement learning. *CoRR*, abs/1911.04448, 2019.

[64] David E. Rumelhart, Geoffrey E. Hinton et Ronald J. Williams : Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, octobre 1986.

[65] Andrei A. Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu et Raia Hadsell : Policy Distillation, janvier 2016. arXiv:1511.06295 [cs].

[66] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, L. Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy P. Lillicrap et David Silver : Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588 7839:604–609, 2020.

[67] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, P. Abbeel, Danijar Hafner et Deepak Pathak : Planning to explore via self-supervised world models. *ArXiv*, abs/2005.05960, 2020.

[68] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner et Deepak Pathak : Planning to explore via self-supervised world models. *CoRR*, abs/2005.05960, 2020.

[69] Amitai Shenhav, Matthew M. Botvinick et Jonathan D. Cohen : The Expected Value of Control: An Integrative Theory of Anterior Cingulate Cortex Function. *Neuron*, 79(2):217–240, juillet 2013.

[70] Amitai Shenhav, Jonathan D. Cohen et Matthew M. Botvinick : Dorsal anterior cingulate cortex and the value of control. *Nature Neuroscience*, 19(10):1286–1291, octobre 2016.

[71] Harshit Sikchi, Wenxuan Zhou et David Held : Learning off-policy with online planning. *In Conference on Robot Learning*, pages 1622–1633. PMLR, 2022.

[72] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel et Demis Hassabis : Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, octobre 2017.

[73] Keith E Stanovich et Richard F West : Individual differences in reasoning: Implications for the rationality debate? *Behavioral and brain sciences*, 23(5):645–665, 2000.

[74] J. R. Stroop : Studies of interference in serial verbal reactions. *Journal of Experimental Psychology*, 18(6):643–662, décembre 1935.

[75] Richard Sutton : Learning to predict by the method of temporal differences. *Machine Learning*, 3:9–44, 08 1988.

[76] Richard S. Sutton et Andrew G. Barto : *Reinforcement Learning: An Introduction*. The MIT Press, second édition, 2018.

[77] Richard S. Sutton et Andrew G. Barto : *Reinforcement Learning: An Introduction*. The MIT Press, second édition, 2018.

[78] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki *et al.* : Deepmind control suite. Rapport technique, DeepMind, 2018.

[79] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy P. Lillicrap et Martin A. Riedmiller : Deepmind control suite. *CoRR*, abs/1801.00690, 2018.

[80] Yee Whye Teh, Victor Bapst, Wojciech Marian Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess et Razvan Pascanu : Distral: Robust multitask reinforcement learning. *CoRR*, abs/1707.04175, 2017.

[81] Amos Tversky et Daniel Kahneman : Judgment under uncertainty: Heuristics and biases. *Science*, 185(4157):1124–1131, 1974.

[82] Weirui Ye, Shaohuai Liu, Thanard Kurutach, P. Abbeel et Yang Gao : Mastering atari games with limited data. *ArXiv*, abs/2111.00210, 2021.

[83] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y. Zou, Sergey Levine, Chelsea Finn et Tengyu Ma : Mopo: Model-based offline policy optimization. *ArXiv*, abs/2005.13239, 2020.

[84] Marvin Zhang, Sharad Vikram, Laura Smith, P. Abbeel, Matthew J. Johnson et Sergey Levine : Solar: Deep structured latent representations for model-based reinforcement learning. *ArXiv*, abs/1808.09105, 2018.