Titre: Title:	L'intelligence artificielle au service de l'optimisation de l'énergie électrique dans un réseau intelligent
Auteur: Author:	Amine Bellahsen
Date:	2020
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Bellahsen, A. (2020). L'intelligence artificielle au service de l'optimisation de l'énergie électrique dans un réseau intelligent [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/5373/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/5373/
Directeurs de recherche: Advisors:	Hanane Dagdougui
Programme: Program:	Maîtrise recherche en mathématiques appliquées

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

L'intelligence artificielle au service de l'optimisation de l'énergie électrique dans un réseau intelligent

AMINE BELLAHSEN
Département de mathématiques et de génie industriel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Mathématiques appliquées

Août 2020

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

L'intelligence artificielle au service de l'optimisation de l'énergie électrique dans un réseau intelligent

présenté par **Amine BELLAHSEN**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Charles AUDET, président
Hanane DAGDOUGUI, membre et directrice de recherche
Luc-Désiré ADJENGUE, membre

DÉDICACE

À mes parents

REMERCIEMENTS

En tout premier lieu, je tiens à remercier Hanane Dagdougui, ma directrice de recherche au sein du département de mathématiques et de génie industriel, pour son encadrement, le soutien qu'elle m'a apporté et pour m'avoir permis de réaliser cette maîtrise à Polytechnique Montréal. Sa connaissance du sujet et ses nombreux conseils m'ont aidé à découvrir et m'épanouir dans un domaine nouveau qu'est la recherche.

Je remercie également Polytechnique Montréal et Télécom SudParis, écoles dans lesquelles j'ai pu compléter ma formation par le biais de ce double diplôme. Les accords entre ces deux institutions m'ont permis d'enrichir davantage mes connaissances scientifiques et culturelles au travers de la découverte d'un nouveau pays.

Ensuite, je souhaite remercier les partenaires de ce projet à savoir le Conseil de Recherches en Sciences Naturelles et en Génie (CRSNG) du Canada, Hydro-Québec ainsi que Schneider Electric pour leur apport financier et technique au sein de cette maîtrise et finalement le campus de l'École de Technologie Supérieure (ÉTS) pour nous avoir procuré les données nécessaires pour cette maîtrise. J'espère que ces travaux pourront apporter de nouvelles connaissances aidant à leur amélioration.

Cette maîtrise recherche m'a de plus permis de rencontrer plusieurs acteurs tout au long de cette aventure et je remercie chacun d'entre eux. Ils m'ont apporté des conseils, des connaissances, des idées et du soutien, qui m'ont aidé à construire et finaliser mon projet. Parmi ces personnes, j'accorde une pensée particulière à Dalal Asber, chercheuse à Hydro-Québec, et à Jonathan Jalbert, professeur adjoint au département de mathématiques appliquées et de génie industriel à Polytechnique Montréal, pour leur aide et leur implication durant ma recherche ainsi qu'à mes collègues de bureau Vincent Taboga, Ehsan Rezaei et Ismail Zejli pour leurs idées et leur patience.

Finalement, je remercie mes parents, ma famille et mes amis pour leurs encouragements et leur soutien moral tout au long de ce projet, sans lesquels rien n'aurait été possible.

RÉSUMÉ

Afin de maintenir l'équilibre entre l'offre et la demande d'électricité, l'utilisation d'une planification opérationnelle fiable est un aspect important de la gestion du système électrique. Pour de nombreux services, les consommateurs doivent payer plus cher l'électricité pendant les périodes de pointes. Par conséquent, si les consommateurs possèdent une connaissance à l'avance des informations sur les pointes de la demande électrique, ces coûts supplémentaires peuvent être évités car ils pourront mieux gérer leur utilisation de l'énergie électrique. Pour cela, des prévisions précises de la demande d'électricité et donc des informations crédibles sur les pointes de puissance vont assurer un approvisionnement en électricité fiable, car d'un point de vue du fournisseur elles donnent le temps d'anticiper son offre, et d'un point de vue du consommateur elles peuvent être d'une grande utilité pour réduire le coût de l'électricité. Maintenant, lorsqu'il s'agit de prédire la consommation de puissance future d'un groupe de consommateurs avec des profils hétérogènes comme c'est le cas dans les bâtiments résidentiels, commerciaux ou institutionnels, il est nécessaire de savoir que les consommateurs peuvent avoir selon plusieurs critères leur propre comportement, une autre difficulté s'ajoute donc au problème. En effet, il devient primordial de tenir en compte l'aspect comportemental de chacun et donc de comprendre comment les consommateurs agissent à chaque moment. Sans bien connaître cet aspect, le modèle de prédiction ne sera pas robuste aux changements brusques dans les habitudes quotidiennes de chacun. Il est également crucial de pouvoir porter de l'assistance chaque consommateur dans la gestion de sa consommation d'énergie, tout en essayant de maximiser son confort car cela peut permettre un gaspillage minimal d'énergie avec des changements de comportement mineurs de la part des consommateurs puisque la flexibilité permise par la demande d'électricité combinée aux incitatifs de la part du système électrique peuvent créer davantage de possibilités pour la gestion de la demande. Globalement, appliquer ce principe sur une plus grande échelle contribuera pleinement à la réduction des émissions de gaz à effet de serre. Ce mémoire est donc construit à travers deux grands objectifs : le premier étant de créer des modèles précis de prédiction de la charge électrique, le second concerne la gestion de la demande électrique et la maximisation du confort du consommateur, ce dernier sera traduit par le contrôle de la température de chaque pièce à une consigne du consommateur.

Le premier objectif s'oriente autour d'une étude comparative et approfondie de trois principaux algorithmes d'apprentissage profond largement utilisés pour la prédiction de séries chronologiques à court terme. Cette étude fera rentrer en jeu un ensemble de données se composant de l'historique de puissance consommée sur cinq bâtiments hétérogènes d'un

campus universitaire à Montréal. L'historique de puissance est récolté à l'aide de compteurs intelligents chaque 15 minutes pendant les années 2015 et 2016. D'autres données externes liées à la météo et le calendrier seront ajoutées par la suite afin d'extraire les effets saisonniers et prendre en compte le comportement des consommateurs. Le but de cette étude sera de trouver le meilleur algorithme et la meilleure stratégie afin de construire un modèle capable de prédire avec une grande précision la consommation de puissance future à court terme au niveau du campus et sur plusieurs horizons. Aussi, ce modèle nous permettra d'avoir des informations fiables concernant les pointes d'électricité.

La deuxième partie de cette recherche, faite en collaboration avec un étudiant doctorant, vise à fournir une étude approfondie sur la gestion de l'énergie et du confort en utilisant l'apprentissage par renforcement, qui est un processus essai-erreur où un agent autonome apprend à remplir une tâche en interagissant avec son environnement et en prenant ses propres décisions de façon à optimiser une récompense au cours du temps. Par conséquent, l'agent devra trouver la bonne politique qui lui permettra d'obtenir une récompense maximale sur toute la période de contrôle. Des agents capables de bien agir sur des charges contrôlées par thermostat ont été construits en utilisant des données synthétiques, puis testés sur diverses combinaisons de pièces, chaque pièce contenant une charge contrôlée par thermostat. Par la suite cette étude a pu prouver que donner les résultats de prédiction de la charge électrique comme information à l'agent était très efficace car comparée au schéma où ces résultats ne sont pas fournis, elle a permis de réduire jusqu'à 73 % la consommation d'énergie pendant les périodes de pic. Enfin, cette étude a été également l'objet d'une évolution vers la mise en oeuvre d'un agent généralisé fiable, appelé GA2TC, capable de bien gérer l'énergie électrique et de maximiser le confort dans des bâtiments, quelle que soit leur architecture ou leur dimension. Cet agent a été développé dans le but d'éviter de construire une solution spécialement pour chaque bâtiment et, à la place, d'utiliser un seul agent adaptable à tous types de pièces et donc de bâtiments. Ce travail permettra une fluidité et une avancée rapide dans le domaine électrique ainsi que dans des domaines similaires, où la difficulté réside principalement dans la modélisation de solutions spécifiques à certains types de problèmes.

En conclusion, ce mémoire contribue à la découverte de nouvelles approches basées sur de l'apprentissage machine ainsi qu'à la mise en oeuvre de leurs applications dans le réseau intélligent (*Smart Grid*). Il fournit également une avancée et une nouvelle approche dans la manière de gérer les systèmes électriques dans les bâtiments, en prenant en compte dans certains cadres les prévisions de consommations futures. Toutefois, des limites techniques liées au manque de ressources de calcul pour perfectionner nos modèles de prévision et de gestion, et à la modélisation simplifiée du monde réel quant aux environnements des bâtiments, offrent de nouveaux défis à relever et des perspectives d'études élargies.

ABSTRACT

In order to maintain the balance between electricity demand and supply, the use of reliable operational planning is an important aspect of power system management. For many services, consumers must pay more for electricity during periods of high demand like during cold weather, big events or football matches. Therefore, if consumers have prior knowledge of electricity peak information, these additional costs can be avoided as they will help them better manage their use of electrical energy. For this, accurate forecasts of energy demand and thus reliable information on the expected peak of power will ensure a reliable supply of electricity, because from a supplier's point of view, it gives time to anticipate its supply, and from a consumer's point of view, it can be of great use in reducing the cost of electricity. Now, when it comes to predicting the future power consumption of a group of consumers with heterogeneous profiles, as it is the case in residential, commercial or institutional buildings, it is necessary to know that consumers have their own behaviour according to several criteria, so another difficulty is added to the problem. Indeed, it becomes essential to take into account the behavioural aspect of each person, and therefore to understand how consumers act at each moment. Without a good knowledge of this aspect, the prediction model will not be robust enough to abrupt changes in the daily habits of each individual. It is also crucial to be able to assist each consumer in managing their energy consumption, while trying to maximize their comfort, as this can allow minimal energy waste with minor changes in consumer behaviour since the flexibility allowed by the electricity demand combined with incentives from the power system can create more opportunities for energy saving. On a broader level, applying this principle on a larger scale will make a significant contribution to reducing global greenhouse gas emissions. This paper is therefore built around two main objectives: the first is to create accurate models for predicting the electrical demand, and the second is to minimize and manage the consumed electrical power and maximize consumer comfort, which will be translated into the control of the temperature set point each room at a consumer instruction.

The first objective is based on a comparative and in-depth study of three main deep learning algorithms widely used for short-term time series prediction. This study will bring into a real data set consisting of the power consumption history of five heterogeneous buildings on a university campus in Montreal. The power history is collected using smart meters every 15 minutes during the years 2015 and 2016. Other external data related to weather and calendar will be added later on to extract seasonal effects and take into account consumer behaviour. The goal of this study will be to find the best algorithm and strategy to build a model capable

of predicting with high accuracy the future power consumption at the campus level, and thus to have reliable information about electricity peaks.

The second part of this research, done in collaboration with a PhD student, aims to provide an in-depth study of energy and comfort management using reinforcement learning, which is a trial-and-error process where an autonomous agent learns to perform a task by interacting with its environment and making its own decisions in order to optimize a reward over time. Therefore, the agent will need to find the right policy to reach maximum reward over the entire control period. Agents capable of acting well on thermostatically controllable loads have been constructed using synthetic data and then tested on various combinations of rooms, each room containing a thermostatically controllable load. Subsequently this study proved that giving the prediction results of the electrical load as information to the agent was very effective as it allowed us to reduce energy consumption by up to 73% compared with having no prediction information. Finally, this study was also the subject of an evolution towards the implementation of a reliable generalized agent, called GA2TC, capable of managing electrical energy and maximizing comfort in buildings, regardless of their architecture or size. This agent has been developed with the aim of avoiding constructing a solution specifically for each building, and instead using a single agent adaptable to all types of rooms and therefore buildings. This work will allow fluidity and rapid progress in the electrical field as well as in similar fields, where the difficulty lies mainly in modeling specific solutions to specific types of problems.

In conclusion, this research contributes to the discovery of new approaches based on machine learning as well as to the implementation of their applications in the smart grid field. It also provides an advanced and innovative approach in the way of managing electrical systems in buildings, taking into account in some frameworks the forecasts of future consumption. However, technical limitations related to the lack of computational resources to refine our prediction and control models, and simplified real-world modeling of building environments, offer new challenges and expanded study opportunities.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES ET ABRÉVIATIONS

DR	Réponse à la demande
CRSNG	Conseil de Recherches en Sciences Naturelles et en Génie
SVR	Machine à vecteurs de support de régression
LSTM	Réseau de neurones récurrent à mémoire court et long terme
SVM	Machine à Vecteurs de Support
GRU	*Gated recurrent networks*
HVAC	Chauffage, ventilation et climatisation
TCL	Charge contrôlée par thermostat
MPC	Commande prédictive
RL	Apprentissage par renforcement
DRL	Apprentissage par renforcement profond
MLP	Perceptron multicouche
ANN	Réseau de neurones artificiels
RNN	Réseau de neurones récurrents
Seq2Seq	Séquence à Séquence
MIMO	*Multi Input Multi Output*
MAPE	Erreur moyenne absolue en pourcentage
MDP	Processus de décision de Markov
TD	*Temporal Difference*
DQN	*Deep Q-networks*
PG	*Policy Gradient*
DDPG	*Deep Deterministic Policy Gradient*
A2C	*Advantage Actor Critic*
TRPO	*Trust Region Policy Optimization*
KL	Kullback-Leibler
FIM	Information de Fisher
ACKTR	*Actor Critic using Kronecker-Factored Trust Region*
K-FAC	*Kronecker-factored Approximate Curvature*
ICT	*Information and communication technologies*
SRNN	Réseau de neurones récurrents simple
HE	*Human expertise*
ACF	*Autrocorrelation function*
GA2TC	*Generalized Agent Training for Temperature Control*

CHAPITRE 1 INTRODUCTION

1.1 Mise en contexte

Les effets du dioxyde de carbone sur le réchauffement de la planète sont connus depuis cent cinquante ans. Cependant, les travaux de Jonh Tyndall et de Svante Arrhenius [2, 3] qui les ont découvert pour la première fois sont restés largement inaperçus pendant des décennies. En conséquence, le monde s'est réchauffé de plus d'un degré Celsius, le niveau de la mer a augmenté de 22 centimètres depuis 1880, ce qui menace la population côtière et les animaux marins, la biomasse des mammifères a diminué de 82 % depuis la préhistoire et d'importantes mesures doivent être prises par conséquent [4–6].

Dans le paradigme du réseau intelligent, il est devenu essentiel d'aider les consommateurs à mieux gérer leur consommation d'énergie à l'aide de différents programmes. L'aspect important de la gestion de la consommation d'électricité consiste à réduire la consommation d'électricité pendant les périodes où les coûts du marché sont élevés ou lorsque la fiabilité du réseau est menacée. De telles actions créent des avantages non seulement pour le fournisseur d'électricité mais aussi pour les consommateurs. Pour répondre de manière appropriée à ces problèmes de réponse à la demande (DR), les sociétés d'agrégation de la demande électrique commencent à offrir aux consommateurs la possibilité de participer à des activités de DR en utilisant leurs charges contrôlables [7], soit en réduisant les charges lors d'événements spécifiques, soit en gérant les charges contrôlables sur le marché en temps réel, soit en proposant des récompenses financières basées sur la quantité de charge flexible fournie.

Afin de rendre les programmes de DR opérationnels, des outils de prévision des charges et des systèmes de gestion de l'énergie fiables seront indispensables et doivent être mis en œuvre au niveau des agrégateurs. De plus, pour assurer une distribution d'électricité efficace et fiable pour chaque utilisateur, la prévision à court terme est le choix le plus approprié, notamment en raison de son importance pour permettre une planification optimale de la distribution des ressources électriques, en minimisant tout risque financier, et en améliorant la fiabilité de la maintenance à court terme [8]. La DR d'un nombre significatif de consommateurs, lorsqu'elle est recueillie et contrôlée avec précision par les agrégateurs, peut jouer un rôle croissant sur le marché de gros de l'électricité existant. Dans ce cas, elle peut aider le gestionnaire de réseau de distribution à gérer la DR à un haut niveau tout en exploitant le potentiel de flexibilité et aider les consommateurs à bénéficier de récompenses ou de factures d'énergie moins élevées. Afin d'assister ce processus, à ces outils de prévision doivent s'ajouter des outils de contrôle des charges où une optimisation de l'énergie consommée est effectuée au plus bas niveau

possible afin de réduire au maximum le gaspillage d'énergie sans provoquer des changements majeurs sur le comportement des consommateurs pour ainsi garder leur confort.

1.2 Problématique et objectifs

Ce projet de maîtrise donne suite à un partenariat entre Polytechnique Montréal et le Conseil de Recherches en Sciences Naturelles et en Génie (CRSNG) du Canada, Hydro-Québec ainsi que Schneider Electric dans le cadre du programme de recherche de la chaire de recherche industrielle en optimisation des réseaux électriques intelligents. L'objectif principal est de créer un cadre de prédiction et de contrôle de la charge électrique en maximisant le confort des consommateurs.

Pour cela, cette recherche se base sur deux objectifs spécifiques :

- Développer une étude comparative de méthodes d'apprentissage profond pour la prévision à court terme de la charge agrégée pour des bâtiments hétérogènes en apportant une estimation aux pointes de la demande électrique. De plus, l'étude proposera diverses stratégies pour capturer l'aspect comportemental des consommateurs dans les bâtiments.
- Proposer des modèles basés sur l'apprentissage par renforcement afin de pourvoir contrôler la charge électrique en maintenant le confort en termes de chauffage et climatisation de multiple maisons résidentiels. Ensuite, un nouvel algorithme général sera proposé pour établir ce contrôle dans des maisons en dépit de leurs architectures.

1.3 Structure du mémoire

Le présent mémoire est constitué de sept chapitres. Tout d'abord, cette introduction définit le contexte de l'étude et le domaine d'application, puis la problématique et les objectifs qui s'en suivent. Ensuite, le chapitre 2 propose une revue de littérature sur les thèmes liés au domaine d'étude, et une mise en avant des travaux effectués au sujet de la prévision et la gestion de la charge dans les bâtiments. Par la suite, le chapitre 3 décrit les algorithmes de prévision utilisés d'une part puis les algorithmes d'apprentissage par renforcement utilisés d'une autre part. Le chapitre 4 présente le cas d'étude concernant les prédictions en tant que papier de recherche et le chapitre 5 présente deux cas d'études dans le cadre de la gestion de la charge. Le chapitre 6 dresse le portrait des résultats rapportés au sein de ce mémoire sous forme d'une discussion générale. Finalement, le chapitre 7 conclut ce mémoire en exposant les contributions de cette recherche ainsi que ses limitations. Plusieurs perspectives d'application et d'amélioration y sont proposées.

CHAPITRE 2 REVUE DE LITTÉRATURE

En général, un réseau électrique intègre l'électricité des systèmes de production, de transmission, de distribution et de contrôle afin de fournir de l'électricité aux consommateurs. Ensuite, l'intelligence y est mise en œuvre pour contribuer à rendre le réseau intelligent. Le réseau intelligent (*Smart Grid*) tente d'améliorer les opérations et les opérations de maintenance en automatisant les opérations afin que les composants communiquent entre eux [9]. Un réseau intelligent est un réseau bidirectionnel qui apporte une grande amélioration au réseau électrique traditionnel, il assure un fonctionnement plus économique et réactive. Le réseau intelligent incorpore les technologies modernes de l'information et de communication avec des dispositifs électroniques avancées de manière intégrée tout au long de la production, transmission puis la consommation.

Avec l'augmentation de la demande électrique dans le réseau intelligent, ce dernier doit faire face à de nouveaux défis tels que l'augmentation de la capacité de production et de transmission et la gestion du pic de puissance. Cela d'un côté augmente les coûts d'électricité surtout pendant les périodes de pointe, puis oblige le réseau à trouver des solutions optimales de la gestion de son énergie tout en minimisant ces demandes de pointe [10]. Par exemple au Québec, la consommation d'électricité peut augmenter considérablement en raison de la forte pénétration du chauffage électrique en hiver. Le secteur résidentiel est responsable, en semaine, de deux pics : un pic le matin entre 6h00 et 9h00, et un pic le soir de 16h00 à 20h00 [11]. Pour pallier ce problème, il est important de trouver des solutions efficaces et pratiques pour les deux plus grands acteurs du réseau intelligent : les consommateurs qui veulent minimiser les coûts, et les fournisseurs qui cherchent à faire plus de profits et à réduire la demande de pointe.

2.1 Prédiction de la demande à court terme

2.1.1 Contexte

Afin de lutter contre la surcharge électrique et ne pas se soumettre à son coût élevé, certaines compagnies d'électricité achètent le double du volume d'énergie prévu. D'autres adoptent une méthode de réserve d'énergie, dont l'entretien et le fonctionnement sont coûteux. Par conséquent, pour prévenir les pannes de réseau et les surcharges, il est essentiel de disposer de bons modèles de prévision de la charge, ce qui est aussi primordial pour mieux gérer la charge sur les marchés à l'avance et en cours de journée [12]. Récemment, avec l'expansion rapide des

technologies intelligentes de l'information et de la communication [13], la façon conventionnelle de voir les systèmes d'alimentation électrique s'est déplacée vers les réseaux intelligents car ces infrastructures intelligentes peuvent générer des volumes très importants et des débits élevés dépassant les capacités des systèmes d'alimentation électrique traditionnels. Par exemple à ce jour, Hydro-Québec, comme beaucoup d'autres fournisseurs d'énergie à travers le monde, a remplacé plus de 98% de ses compteurs par des compteurs intelligents [14]. Afin de tirer profit de cette quantité massive de données générées, l'utilisation de l'apprentissage profond [15], qui est l'une des approches les plus actives dans de nombreux domaines de recherche, semble être un choix essentiel pour augmenter la performance des prévisions et mieux saisir la volatilité existante dans chaque profil de consommation. En outre, comme des compteurs intelligents sont parfois placés à l'extrémité du réseau au niveau des consommateurs, on peut imaginer que les prévisions peuvent même être faites à leur niveau.

Pour assurer une distribution d'électricité efficace et fiable, la prévision à court terme est le choix le plus approprié, notamment en raison de son importance pour permettre une planification optimale de la distribution des ressources électriques, en minimisant les risques financiers éventuels, et en améliorant la fiabilité de la maintenance à court terme [8].

L'objectif de cette recherche étant de prévoir la consommation à court terme dans un point de vue agrégateur, en prenant en compte la demande de chacun des différents utilisateurs du réseau, puis de gérer ce dernier, nous sommes contraints à plusieurs défis. L'exploitation et la planification du système deviennent plus difficiles lorsqu'il s'agit de bâtiments de profils différents. En effet, les bâtiments résidentiels ne consomment pas de la même manière que les bâtiments commerciaux ou institutionnels, et ne partagent donc pas les mêmes profils. Ces différences créent des défis dans le processus de construction de modèles précis de prévision de la charge pour un district contenant de nombreux bâtiments. En outre, par rapport à la prévision de la charge au niveau des utilisateurs individuels, les recherches au niveau de l'agrégateur restent insuffisantes [16,17] malgré les progrès dus à l'évolution des infrastructures de réseau intelligent [13].

2.1.2 Approches pour la prédiction de la demande

De nombreuses études ont été rapportées dans la littérature pour résoudre le problème de la prévision de la charge à court terme, en particulier avec les progrès de l'apprentissage profond. De plus, plusieurs applications et approches ont été couvertes dans beaucoup de recherches. En effet, Fan et al. [18] ont proposé une étude comparative entre des algorithmes d'apprentissage profonds, tous des variantes des réseaux de neurones récurrents, en utilisant différentes stratégies connues pour la prédiction plusieurs étapes en avance. Leurs algorithmes ont été

évalués sous différents angles basés sur les données opérationnelles de divers bâtiments. Tan et al. [19] ont abordé les problèmes liés au contrôle de la demande d'énergie à très court terme en utilisant un nouveau modèle de prédiction basé sur l'apprentissage profond d'ensemble. Dans leur approche, ils sont parvenus à obtenir une bonne précision et une bonne robustesse de leur modèle tout en utilisant une fonction de perte qui ajoute l'erreur de prévision de la demande de pointe et en mesurant l'impact de leur modèle à l'aide d'une nouvelle métrique appelée l'erreur absolue de pointe en pourcentage. Une étude comparative a été réalisée pour montrer que leur modèle est plus performant que les méthodes statistiques classiques ou même que les réseaux de neurones. Une combinaison de deux modèles de prévision de pointe pour les séries chronologiques a été réalisée par Tang et al [20] afin de créer un réseau de neurones récurrent bidirectionnel à multiples couches visant à créer des résultats précis pour la prévision de la charge à court terme. Pour leurs expériences, ils ont testé la méthode proposée sur deux jeux de données, surpassant les algorithmes de machine à vecteurs de support de régression (SVR) et de réseaux de neurones classiques. Kong et al. [21] ont proposé une approche également basée sur l'utilisation des réseaux de neurones pour la prévision de la charge résidentielle à court terme. Un cadre de prévision à apprentissage profond basé sur les réseau de neurones récurrent à mémoire court et long terme (LSTM) a été mis en œuvre pour résoudre les problèmes de volatilité concernant l'historique de consommation d'un ménage individuel. Encore une fois, les performances de cette approche ont été comparées à celles de plusieurs stratégies telles que les réseaux de neurones simples de type *feed forward* ou les K-plus proches voisins, montrant que leur modèle est le mieux adapté. Les auteurs de Rahman et al. [22] ont utilisé une construction de données spécifique afin d'éliminer la saisonnalité et la tendance de la charge avant d'alimenter un réseau LSTM en données. Somu et al. [23] ont développé un modèle de prévision de la consommation d'énergie qui utilise les réseaux LSTM et un algorithme amélioré d'optimisation du sinus cosinus pour une prévision précise et robuste de la consommation d'énergie des bâtiments. Les auteurs ont mis en œuvre une approche pour trouver les valeurs optimales des hyperparamètres de LSTM. Wen et al. [24] ont proposé un modèle d'apprentissage profond pour prévoir la demande de charge des bâtiments résidentiels. En conséquence, le modèle proposé est également utilisé comme méthode efficace pour combler les données manquantes grâce à l'apprentissage à partir de données historiques présentes.

Avant de modéliser tout problème de séries chronologiques, il est crucial de sélectionner les variables qui sont les plus importantes et les plus corrélées avec l'historique de la charge afin de faire des prévisions fiables, des variables liées à des instants précis dans l'historique de puissance aux variables externes telles que celles liées à la météo. Par exemple, Xie et al. [25] ont réalisé des recherches expérimentales sur l'effet de l'ajout de la température et

de l'humidité relative sur la demande d'électricité en Californie du Nord, et ont démontré une diminution des erreurs de prévision. Quant à Wu et al. [26], ils prennent des facteurs de charge hétérogènes de sources multiples tels que la charge historique, la température de l'air, les précipitations, la direction du vent et le prix de l'électricité, etc. pendant la prévision de la charge et développent une méthode à noyaux multiples. Toutefois, la corrélation entre les facteurs et la variation de la charge électrique n'est pas analysée en détail. Zeng et Jin [27] présentent une méthode de prévision de la charge basée sur des données multi-sources et un réseau topologique au jour le jour. L'une des conclusions auxquelles ils sont parvenus est que les éléments météorologiques, en particulier la température, la pression atmosphérique et la pression de vapeur saturante de l'eau, ont une influence dominante sur la variation de la charge, outre les variables temporelles de la charge. Enfin, Wang et al. [28] ont développé une étude en trois grandes étapes afin de construire un modèle de prédiction probabiliste en utilisant des forêts aléatoires et l'algorithme *Gradient Boosting* comme méthodes de prévision. Ils ont pu démontrer à partir d'études empiriques que leurs prévisions dépendaient beaucoup du temps, des conditions météorologiques et de la période de temps concernée. Cependant, leur approche ne prêtait pas une grande attention à la sélection de leurs variables temporelles et leur modélisation de température.

Comme mentionné précédemment, avec les progrès des infrastructures de compteurs intelligents, il est devenu intéressant d'étudier la prévision de la charge à un niveau d'agrégation en utilisant les données collectées par ces compteurs intelligents comme cela a été fait dans plusieurs recherches. Dans leur recherche [29], Kong et al. ont proposé un cadre basé sur le LSTM à partir des données d'un véritable compteur intelligent résidentiel, afin de prévoir la charge au niveau de l'agrégateur. Pour cela, ils ont utilisé deux approches : la première est en utilisant d'abord les prévisions au niveau des individus puis en les agrégeant, la deuxième est en utilisant les données agrégées pour faire des prévisions directement. Ils ont conclu que la première approche est meilleure. Chen et al. [16] ont proposé dans leur papier une approche pour la prédiction agrégée un jour à l'avance basée sur la propagation d'affinité qui est un algorithme de partitionnement de données (*clustering*), en ciblant 1000 foyers dans la région de Londres. La première étape étant de grouper les foyers selon une mesure d'affinité, la deuxième était de prendre chaque groupe et effectuer une réduction de dimension via un auto-encodeur pour capturer les caractéristiques historiques essentielles pour chaque groupement. La dernière étape consiste à combiner la sortie de l'auto-encodeur en entrée avec la météo et les caractéristiques sociales puis les fournir à un réseau de neurones de fusion pour la prédiction de chaque groupe. Ces prédictions sont ensuite agrégées au niveau de l'aggrégateur. Ils finissent par conclure que leur méthode dépasse les performances de la plupart des approches directes utilisant des algorithmes d'apprentissage profond. Dans

la même optique, Wang et al. [30] ont appliqué la méthode de regroupement hiérarchique pour faire du partitionnement de données sur deux jeux de données contenant plus de 5000 consommateurs résidentiels en cherchant le nombre optimal de *clusters*. Ensuite, ils utilisent un réseau de neurones simple pour réaliser leurs prédictions par groupe. Ils ont par la suite comparé cette approche avec celle qui consiste à prédire la consommation individuellement puis faire la somme pour agréger. Les résultats montrent la deuxième idée est bien pire que la méthode des regroupements en raison de la grande variation des profils de charge individuels. Néanmoins, à la limite de notre connaissance, même si les recherches dans cette optique de prédictions agrégées connaissent plus de succès ces dernières années, la plupart voire la totalité des papiers de recherches se focalisent sur les bâtiments résidentiels ou les ménages, et ne considèrent pas de combinaison de bâtiments hétérogènes tels que les bâtiments commerciaux ou institutionnels pour ce type de problème.

L'objectif final de ce chapitre étant d'utiliser les prévisions agrégées pour extraire de l'information concernant le pic de puissance, plusieurs études ont été menées dans ce domaine. Dans [27], Zeng et Jin ont proposé une nouvelle méthode dynamique afin de prévoir la valeur du pic de puissance quotidien après s'être principalement concentrés sur l'impact de différentes données multi-sources, comme les facteurs naturels et sociaux qui conduisent à des variations de charge comme la température et les vacances. Les auteurs ont intégré un cadre comme suit : pour chaque jour de prévision, ils ont utilisé une marche aléatoire avec algorithme de redémarrage pour sélectionner des jours similaires afin de construire un ensemble de formation adéquat pour finalement appliquer la méthode SVR sur celui-ci afin de prévoir la pointe quotidienne. Ils ont fini par surpasser les autres méthodes qui se concentraient sur d'autres approches de sélection de caractéristiques/variables. Nagi et al. [31] introduisent une méthode en combinant la machine à vecteurs de support (SVM) pour les prédictions et une carte auto adaptative pour le partitionnement de données, sur trois différents jeux de données. Leur approche concerne les prévisions à moyen terme de pic journalier de la charge, et démontre qu'elle dépasse les performances des autres papiers appliqués sur les mêmes jeux de données. Yu et al. [32] ont réussi à développer une nouvelle approche pour la prédiction du pic journalier. Cette approche se base sur la combinaison d'un type particulier de réseaux de neurones récurrents : les *Gated Reccurent Networks* (GRU) pour les prédictions avec la méthode de la déformation temporelle dynamique appliquées aux séries chronologiques pour capturer les similarités entre les différentes courbes de charge quotidienne. La méthode de la déformation temporelle permet de décrire le degré de similitude pour deux séquences dans leur ensemble, et une grande partie des informations locales également, ce qui n'est pas le cas des mesures de similarités classiques telles que la distance euclidienne. Ils démontrent empiriquement que leur méthode surpasse les méthodes statistiques plus classiques.

2.2 Gestion de la demande

2.2.1 Contexte

Le contrôle dans un réseau intelligent peut être vu sous différents angles, et peut être appliqué à divers acteurs de ce réseau, allant des centrales électriques [33] aux véhicules électriques [34] en passant par les énergies renouvelables [35]. Ce contrôle vise à maintenir un équilibre optimal entre disponibilité, coût, fiabilité et efficacité. Et vu la quantité de charges qui s'ajoutent en continu au réseau, il est indispensable d'automatiser beaucoup de processus de contrôle notamment en incluant des techniques d'optimisation.

De nombreuse recherches se sont focalisées sur les techniques d'optimisation et leurs applications sur le système électrique au cours des dernières décennies. Cette optimisation du réseau peut être réalisée en surveillant le comportement de la charge [36], en déplaçant la demande d'électricité des heures de pointe vers les heures creuses et en remédiant rapidement à toute panne de courant [37]. Elle permet également de construire des réservoirs d'énergie pour répondre à la demande supplémentaire [38], d'ajouter des véhicules électriques rechargeables au réseau [34], de recueillir des données en temps réel sur les utilisateurs et de les partager avec les sociétés de services publics [39]. Plus important encore, les techniques d'optimisation peuvent déterminer la production d'électricité optimale parmi les différentes sources en fonction de la demande, de la planification, de la répartition, etc. afin de minimiser la demande de pointe et les coûts opérationnels tout en prenant en compte la plupart des fois la maintenance des préférences ainsi que les conditions thermales des utilisateurs comme principaux objectifs de l'optimisation [40]. Ces techniques, de leur côté, peuvent beaucoup varier allant des algorithmes génétiques [41] à l'optimisation par essaims particulaires (*Particule Swarm Optimization*) [42] en passant par de modèles plus complexes qui décrivent l'état de l'art dans ce domaine tels que l'apprentissage par renforcement [43].

Par ailleurs, malgré la variété de toutes ces applications dans le réseau intelligent, la plus importante à traiter est la gestion de l'énergie dans les bâtiments [44]. En effet, comme 30% des émissions mondiales de gaz à effet de serre proviennent du secteur du bâtiment, afin de réduire l'empreinte de la consommation d'énergie sur la planète, nous devons trouver des moyens de consommer le plus efficacement possible [45]. Au Canada, environ 40% de la consommation mondiale d'énergie est liée aux bâtiments et la moitié est utilisée par les systèmes de chauffage, ventilation et climatisation (HVAC). De plus, en raison de l'inertie thermique de l'air et des murs, une partie des opérations du système de chauffage et de refroidissement peut être décalée de la période d'utilisation réelle. En effet, l'énergie thermique est stockée dans la masse du bâtiment, ce qui permet le préchauffage ou le pré-refroidissement.

La flexibilité offerte par ce type de charge, également appelée charge contrôlée par thermostat (TCL), est essentielle pour équilibrer la production et la consommation d'énergie en lissant la courbe de consommation du bâtiment. De plus, les fournisseurs d'énergie peuvent proposer des prix d'électricité plus avantageux en dehors des périodes de pic de demande en entraînant des avantages financiers substantiels pour les consommateurs [45].

En outre, avec l'intégration de la réponse à la demande (DR), les bâtiments peuvent contribuer à l'efficacité énergétique grâce à la flexibilité des TCLs, puis générer des profits et réaliser des économies sur leurs factures d'électricité. Grâce à des programmes de DR basés sur des incitations, les consommateurs peuvent participer à un programme dans lequel un agrégateur ou un réseau électrique peut directement couper, déplacer ou réduire leur consommation d'énergie pour réduire la consommation pendant les périodes de demande de pointe. Par exemple, le contrôle des points de consigne de température, le préchauffage ou le pré-refroidissement des bâtiments peuvent être considérés comme un ensemble d'actions visant à réduire et à mieux gérer la demande de pointe [46].

2.2.2 Approches pour la gestion de la demande

Beaucoup de recherches ont été faites dans l'optique de vouloir contrôler la charge électrique dans des bâtiments, en essayant parfois de maintenir le confort des utilisateurs ou la maximisation des économies concernant le prix de l'électricité. Parmi ces approches beaucoup se sont focalisées sur la gestion de l'énergie dans des bâtiments résidentiels, en utilisant diverses techniques d'optimisation. Plusieurs approches pour le contrôle de la charge ont été implémentées et testées sur divers cas d'études. On peut diviser ces méthodes de contrôle en quatre principales catégories [47] :

- Régulateurs classiques : ils font partie des méthodes les plus utilisées, telles que la méthode marche/arrêt qui utilise un seuil supérieur et inférieur pour réguler le processus du contrôle, puis la régulation P/PI/PID (Proportionnel, Intégral, Dérivé) qui est un système de régulation en boucle fermée qui se base sur la dynamique des erreurs pour moduler la variable contrôlée pour un processus plus précis. La méthode marche/arrêt est la plus directe à implémenter, cependant avec l'inertie thermique que possède les TCLs il est difficile à ce modèle de suivre les déplacements en retard des processus. La régulation PID, quant à elle, prend beaucoup de temps pour trouver ses paramètres optimaux, qui sont propres à chaque problème. Ces modèles ont déjà été appliqués pour divers objectifs tels que le contrôle de la température dans des pièces ou des chauffages [48–50] avec une grande importance à la recherche des bons paramètres lorsqu'il s'agit des régulateurs PID.

- Régulateurs strictes (*Hard Control*) : ils se basent sur une théorie pour les systèmes de

contrôle composée du contrôle non linéaire, du contrôle robuste, du contrôle optimal et de la commande prédictive (MPC) qui utilise un modèle précis de l'environnement basé sur les lois physiques et l'ajustement des paramètres en fonction des données pour calculer une stratégie de contrôle. La MPC est l'une des plus prometteuses en raison de sa capacité à intégrer dans la formulation du contrôleur le rejet des perturbations, la gestion des contraintes et les stratégies de contrôle et d'économie d'énergie. Cependant, pour la MPC l'analyse et la modélisation des données prennent beaucoup de temps, puis cette méthode nécessite des investissements sensiblement plus élevés comparés aux méthodes classiques, qui peuvent ne pas être compensés par des économies supplémentaires dans un court laps de temps [51]. Yan et al. [52] présentent une application des algorithmes génétiques pour la gestion du chauffage et de la climatisation des bâtiments en réduisant la consommation d'énergie du système de source de refroidissement de 7%. Une multitude de recherches montrent les applications des régulateurs strictes pour le contrôle et la conservation de l'énergie consommée dans les bâtiments [53, 54].

- Régulateurs souples (*Soft Control*) : ce sont des techniques relativement nouvelles rendues possibles par l'avènement des contrôleurs numériques. Ils utilisent des principes tels que les réseaux de neurones. Les réseaux de neurones viennent dans ce schéma remplacer les régulateurs conventionnels, néanmoins ils ont besoin d'un grand nombre de données pour qu'ils soient précis et robustes, ce qui n'est pas toujours le cas pour plusieurs systèmes. Le problème avec ces modèles réside dans la difficulté de leur interpretabilité malgré leurs bons résultats, car ils agissent comme des boîtes noires. En plus d'être de bons outils pour la prédiction de la charge (chauffage, refroidissement ou puissance), ils peuvent être combinés avec des approches d'optimisation comme c'est le cas dans [55], où un réseau de neurones a été utilisé pour prévoir des données météorologiques et de charge, puis l'optimisation par essaims particulaires a été appliquée pour minimiser la consommation d'énergie tout en maintenant le confort des utilisateurs. Ils finissent par obtenir de très bons résultats même en ajoutant des contraintes liées à l'air. Garnier et al. [56] ont construit une stratégie de contrôle prédictif basée sur un réseau de neurones multi-zone pour satisfaire la contrainte de confort thermique d'un bâtiment non résidentiel. Les résultats montrent que le régulateur prédictif prenant en compte le transfert de chaleur entre les pièces adjacentes offre une amélioration à la fois de l'efficacité énergétique et du confort thermique. D'autres applications de ces méthodes ont été utilisées pour la prédiction, le contrôle et l'optimisation [57–59].

- Régulateurs hybrides : ils se basent sur les combinaisons de régulateurs strictes et souples tels que la régulation floue - PID [60]. Ce type de méthodes bénéficie des qualités des techniques de contrôle stricte et souple, mais doit faire face aux problèmes de chacune, tels que le manque de données pour la partie souple, où la difficulté à trouver les bonnes conditions pour la partie

stricte. Pal et al. [61] présentent un contrôle hybride comprenant une combinaison entre PI et la logique floue pour le contrôle de la pression de l'air d'alimentation dans des HVACs.

Toutefois, la performance et la fiabilité de ces approches dépendent fortement de la précision du modèle décrivant la dynamique thermique du bâtiment, et beaucoup d'approches utilisent généralement un modèle thermique simplifié pendant l'exécution du contrôle pour prédire l'évolution de la température des bâtiments [62]. De plus, la température du bâtiment est influencée par de nombreux facteurs, notamment la structure et les matériaux du bâtiment, l'environnement (par exemple, la température ambiante, l'humidité et l'intensité du rayonnement solaire), et les gains de chaleur internes des occupants, les systèmes d'éclairage et autres équipements. Par conséquent, la température du bâtiment présente souvent des comportements aléatoires dans le cadre d'une modélisation incomplète.

Par conséquent, certains travaux récents ont commencé à développer des approches orientées vers l'usage et l'exploitation des données en temps réel pour le contrôle de la climatisation basé sur l'apprentissage par renforcement (RL), qui est un processus d'essai-erreur dans lequel un agent apprend à accomplir une tâche en interagissant avec son environnement. En effet, le RL a été appliqué aux réseaux intelligents pour contrôler divers systèmes énergétiques tels que les réseaux de chauffage urbain [63], le stockage d'énergie avec des batteries [64], les appareils intelligents [65] ou les systèmes de chauffage, ventilation et climatisation (HVAC) [66–68], pour prendre des actions discrètes comme continues. Les travaux de [69] développent une approche RL assistée, où des arbres aléatoires sont utilisés pour approximer la valeur de l'action et décider des stratégies de contrôle simple marche/arrêt. Néanmoins, ils présentent un coût de calcul élevé tout au long du processus d'apprentissage, il faut donc trouver un compromis entre temps et précision. De plus, avec le développement des réseaux de neurones, de nouvelles méthodes d'approximation ont émergés notamment l'apprentissage par renforcement profond (DRL), ce qui permet d'avoir plus de flexibilité et d'efficacité dans pour une multitude de problèmes complexes. Le DRL a été largement utilisé dans les jeux Atari et a pu surpasser les performances de l'homme dans une plusieurs jeux tels que le jeu de Go [70, 71] et ont été bien impliquées dans l'avancement en terme de contrôle des HVAC ces dernières années. En 2017 Wei et al. [62] ont été les premiers à appliquer le DRL pour le contrôle des HVAC, obtenant des résultats satisfaisants. Leurs expériences démontrent que le DRL est plus efficace que l'approche traditionnelle basée sur des règles, pour réduire les coûts énergétiques tout en maintenant la température ambiante dans la plage souhaitée, et donc le confort du consommateur. Dans leur publication [68], Zhang et al. développent un cadre pratique basé sur le DRL pour le contrôle de la climatisation de bâtiments administratifs, puis l'implémentent dans un cas pratique d'un système de chauffage radiant existant. Leurs résultats montrent que cette méthode peut économiser 16.7% de la demande de chauffage

avec une probabilité de plus de 95%.

2.2.3 Généralisation de la gestion de la demande

La généralisation est souvent considérée comme une caractéristique essentielle des systèmes intelligents avancés et comme une question centrale dans la recherche sur l'intelligence artificielle [72, 73]. Elle comprend à la fois l'interpolation vers des environnements similaires à ceux observés pendant l'entraînement et l'extrapolation en dehors de la distribution de l'entraînement. Cette dernière est particulièrement difficile mais est cruciale pour le déploiement des systèmes dans le monde réel car ils doivent être capables de gérer des situations imprévues. De la même manière, on pourrait espérer pouvoir réaliser du contrôle adaptatif généralisé pour les HVAC.

Ces dernières années, la généralisation dans la DRL a été reconnue comme un problème important et fait l'objet de multiples projets de recherche [74, 75]. Duan et al. [76] visent à apprendre une politique, ce qui définit la manière de décision des actions de l'agent, qui s'adapte automatiquement à la dynamique de l'environnement et à évaluer sa méthode sur des environnements de navigation en labyrinthe où la position de l'objectif change en obtenant plus de 96% de réussite. Dans la même optique, Harries et al. [77] ont créé un environnement Gym [78] appelé MazeExplorer conçu pour tester la capacité des agents de RL à généraliser leurs connaissances dans des environnements plus complexes. La tâche de l'agent est de résoudre un jeu de labyrinthe en 3D, comme dans l'environnement 2D CoinRun [79], en considérant que la carte des labyrinthes change pendant le processus de formation. L'agent voit donc un environnement légèrement différent à chaque étape de l'entraînement. Néanmoins, en testant leur agent avec différentes méthodes de DRL sur des environnements jamais vus pendant l'entraînement, ils ne réussissent pas à généraliser autant que pour le problème en 2D. Cependant, ce problème reste prometteur en vu de la progression et l'émergence de multiples nouvelles approches pour plusieurs nouveaux cas d'application. Cependant, à notre connaissance, aucun travail n'a encore été réalisé pour appliquer les techniques de généralisation à l'aide de l'apprentissage par renforcement pour le contrôle de l'énergie dans les bâtiments.

CHAPITRE 3 MÉTHODOLOGIE DE PRÉVISION ET GESTION DE LA CHARGE

3.1 Prédiction

Cette section apporte une idée générale sur les modèles utilisés pour la prédiction de séries chronologiques multivariées. Ils sont aussi expliqués de manière complémentaire ou similaire dans le Chapitre 4 qui est l'article qui porte sur les prédictions à court terme de la charge agrégée dans des bâtiments hétérogènes.

3.1.1 Le perceptron multicouche

Un perceptron multicouche (MLP) [15] est un type de réseau de neurones artificiels (ANN), et sont utilisés dans le cadre de l'approximation universelle [80]. Les ANNs sont une famille de modèles probabilistes qui peuvent être considérés comme une abstraction mathématique du système nerveux biologique.

Un MLP est premièrement composé d'unités élémentaires appelées «neurones». Un neurone, représenté en Figure 3.1, est une fonction à n variables (i.e. entrées) $\boldsymbol{x_1, x_2, \ldots, x_n}$ liées à une sortie $\boldsymbol{y}$ avec l'équation suivante :

$$\boldsymbol{y = h\left(\Theta_0 + \sum_{i=1}^{n} \Theta_i . x_i\right)} \tag{3.1}$$

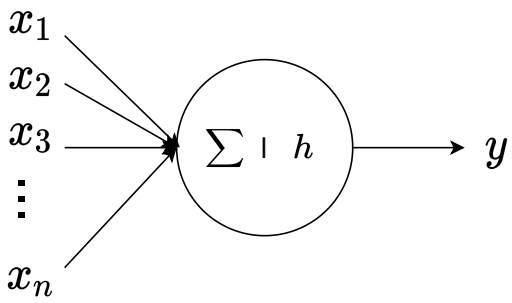

Figure 3.1 Représentation d'un neurone

avec $\boldsymbol{h}$ une fonction d'activation qui est utile pour caractériser le neurone. Il existe une variété de fonctions d'activation, toutes adaptés à des problèmes spécifiques. S'il était lieu d'avoir une sortie du neurone entre $\mathbf{0}$ et $\mathbf{1}$ (e.g. une probabilité), la fonction **sigmoïde** (i.e. $\boldsymbol{h : \mathbb{R} \rightarrow [0, 1]}$) serait un choix adéquat.

L'architecture du MLP est organisée comme suit :

- Les neurones sont organisés en couches successives entièrement connectées. La sortie d'un neurone est connectée à tous les neurones de la couche suivante et il n'existe pas de connexion entre les neurones d'une même couche.
- Les connexions entre les neurones sont pondérées par $\boldsymbol{\Theta_i} \in \mathbb{R}$ dans l'équation (3.1). Ainsi, l'entrée d'un neurone donné est une combinaison linéaire pondérée des sorties de tous les neurones de la couche précédente.
- La première couche d'entrée représente les variables des données utilisées.
- La dernière couche de sortie représente les valeurs de la charge prédite dans le futur.
- Les couches intermédiaires cachées représentent la "Boîte noire" du système, afin de créer plus de connexions entre les neurones. Plus les couches cachées sont multiples, plus le modèle est complexe et peut mieux approximer la sortie.

De plus, le théorème d'approximation universelle [81] assure qu'un ANN à une seule couche cachée avec une fonction d'activation linéaire peut approximer toute fonction continue arbitrairement bien lorsqu'il possède suffisamment de neurones. Cela marche également pour les autres fonctions d'activations (e.g. sigmoid, tanh) et plusieurs couches cachées. Cependant, en pratique avoir un grand nombre de neurones peut mener à des problèmes de stabilité et de prédiction. C'est pour cette raison que les MLPs supposent qu'il est plus efficace de multiplier le nombre de couches au lieu du nombre de neurones.

La Figure 3.2 représente une architecture d'un modèle MLP.

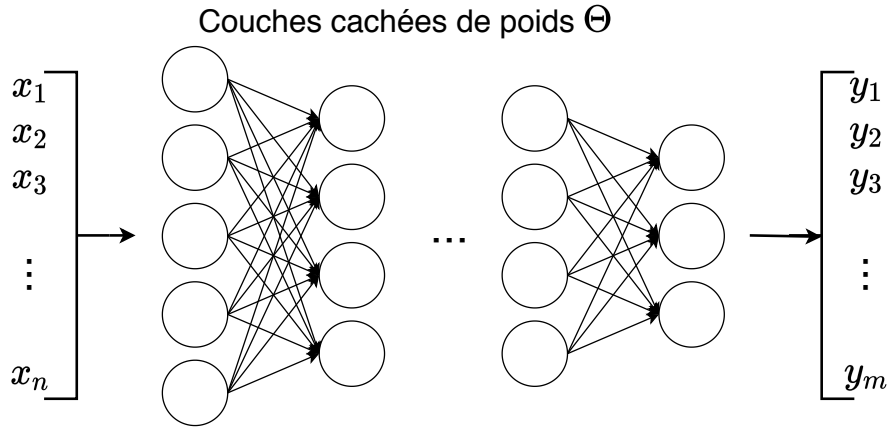

Figure 3.2 Réprésentation d'une MLP

Où $\boldsymbol{\Theta}$ regroupe toutes les pondérations des connexions entre les différents neurones de chaque couche $\boldsymbol{j}$ et les neurones $\boldsymbol{k} \in \boldsymbol{\{1, \ldots, r_{j-1}\}}$ de sa couche précédente. On considère $\boldsymbol{r_j}$ est le

nombre de neurones à la couche j, on a $\Theta_{jk} = (\Theta^1_{jk}, \Theta^2_{jk}, \ldots, \Theta^{r_j}_{jk}) \in \mathbb{R}^{r_j \times r_{j-1}}$. Pendant l'entraînement, le modèle devra idéalement trouver la bonne combinaison de Θ qui va réduire l'écart entre la prédiction et la vraie valeur. Un MLP est donc un modèle qui peut être écrit comme suit :

$$(y_1, y_2, \ldots, y_m) = f(x_1, x_2, \ldots, x_n, \Theta) \tag{3.2}$$

3.1.2 Réseaux de neurones récurrents

Malgré les bonnes performances qu'un MLP peut générer dans de multiples problèmes, il possède une manière directe de lier chaque vecteur d'entrée à la sortie. L'aspect séquentiel des données n'est donc pas forcément considéré. C'est pour cette raison qu'un nouveau type de réseaux de neurones artificiels ont pu émerger cette décennie afin de pallier ce problème.

Les réseaux de neurones récurrents (RNNs) [15], qui est un autre type de ANN, sont bien connus pour leur bon fonctionnement dans les tâches d'apprentissage où les données d'entrée sont séquentielles. Partageant les poids entre les couches cachées à chaque pas de temps, l'architecture RNN est un moyen naturel de modéliser les données de séries chronologiques, où chaque pas de temps de l'entrée dépend de celles du passé. De plus, grâce à la propriété de partage de poids, les RNNs ont la capacité à gérer des entrées de longueur variable contrairement au modèles MLP.

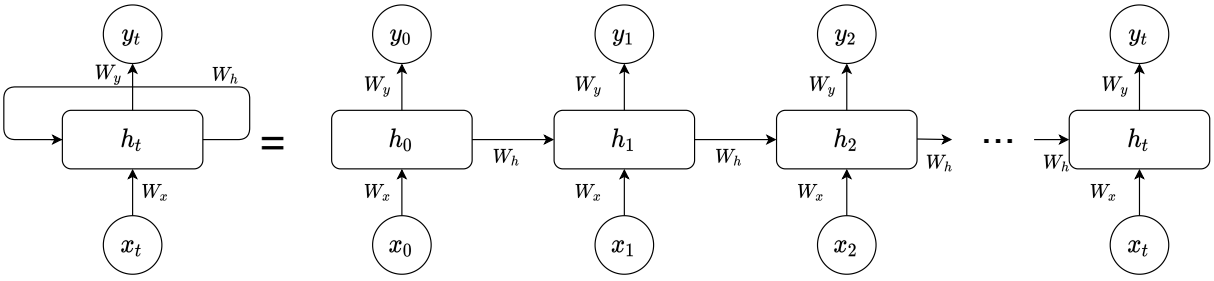

Figure 3.3 Exemple global d'un RNN

Les RNN traditionnels se composent d'une couche d'entrée, d'une couche de sortie tout comme le MLP, ajouté à cela une couche cachée récurrente, comme illustré en Figure 3.3. Ils sont constitués d'une série de matrices de poids et de fonctions d'activation. Explicitement, l'ensemble d'équations qui relient un ensemble d'entrées aux prédictions est comme suit :

$$h_{t+1} = \sigma_h(W_x x_{t+1} + W_h h_t + b_h) \tag{3.3}$$

$$y_{t+1} = \sigma_y(W_y h_{t+1} + b_y) \tag{3.4}$$

où $\boldsymbol{\sigma_h}$ et $\boldsymbol{\sigma_y}$ sont deux fonctions d'activation. Dans notre cas, $\boldsymbol{\sigma_h} = \mathbf{tanh}$ et $\boldsymbol{\sigma_y}$ est l'identité. $\boldsymbol{b_h} \in \mathbb{R}^{n_h}, \boldsymbol{b_y} \in \mathbb{R}^{n_y}$ constituent simultanément des vecteurs de biais et $\boldsymbol{n_h}, \boldsymbol{n_y}$ sont le nombre de neurones dans la couche cachée $\boldsymbol{h}$ et la couche de sortie $\boldsymbol{y}$, et $\boldsymbol{W_x} \in \mathbb{R}^{n_h \times n_x}, \boldsymbol{W_h} \in \mathbb{R}^{n_h \times n_h}, \boldsymbol{W_y} \in \mathbb{R}^{n_y \times n_h}$ sont des matrices de poids partagées sur des pas de temps avec $\boldsymbol{n_x}$ est le nombre de neurones à la couche d'entrée $\boldsymbol{x}$. Au cours de la formation, le modèle apprend une bonne combinaison des matrices de poids et biais afin de minimiser une fonction de perte globale.

3.1.3 Réseaux récurrents à mémoire court et long terme

Bien que les RNN partagent des poids à chaque pas de temps et n'aient pas un grand nombre de paramètres, il souffre de la disparition ou l'explosion du gradient [82] qui lui causent un mauvais apprentissage surtout pour les prédictions sur le long terme.

Pour résoudre ce problème, les réseaux récurrents à mémoire court et long terme (LSTM) ont été proposés [83]. Ce sont des versions modifiées du RNN, qui facilitent la mémorisation des données transmises en mémoire. Cela résout le problème de la disparition de gradient du RNN. La Figure 3.4 représente une différence de la couche cachée récurrente dans un RNN d'un LSTM.

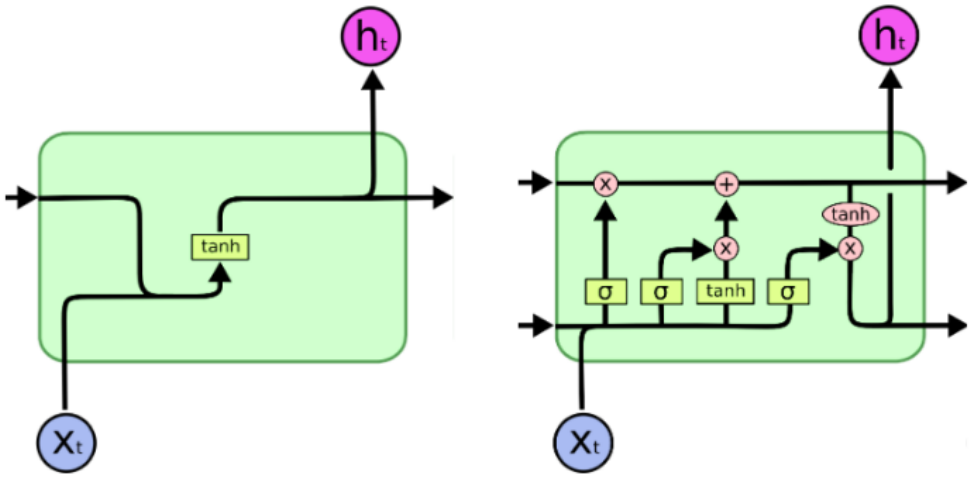

Figure 3.4 Unité RNN simple (gauche) vs Unité LSTM (droite) [1]

Les unités d'un LSTM peuvent se présenter sous plusieurs formes, mais partagent toutes le même principe qui se base sur les **Gates** afin de garder ce qui importe dans la mémoire du modèle et enlever ce qui est inutile. L'une des architectures classiques d'unité d'un LSTM est

explicitement décrite dans l'équation suivante :

$$\boldsymbol{f_t} = \boldsymbol{\sigma}(\boldsymbol{x_t}.\boldsymbol{U^f} + \boldsymbol{h_{t-1}}.\boldsymbol{W^f}) \tag{3.5}$$

$$\boldsymbol{i_t} = \boldsymbol{\sigma}(\boldsymbol{x_t}.\boldsymbol{U^i} + \boldsymbol{h_{t-1}}.\boldsymbol{W^i}) \tag{3.6}$$

$$\boldsymbol{o_t} = \boldsymbol{\sigma}(\boldsymbol{x_t}.\boldsymbol{U^o} + \boldsymbol{h_{t-1}}.\boldsymbol{W^o}) \tag{3.7}$$

$$\tilde{\boldsymbol{C}}_t = \boldsymbol{tanh}(\boldsymbol{x_t}.\boldsymbol{U^g} + \boldsymbol{h_{t-1}}.\boldsymbol{W^g}) \tag{3.8}$$

$$\boldsymbol{C_t} = \boldsymbol{f_t} \odot \boldsymbol{C_{t-1}} + \boldsymbol{i_t} \odot \tilde{\boldsymbol{C}}_t \tag{3.9}$$

$$\boldsymbol{h_t} = \boldsymbol{tanh}(\boldsymbol{C_t}) \odot \boldsymbol{o_t} \tag{3.10}$$

Avec $\boldsymbol{W^{i,f,o,g}} \in \boldsymbol{R^{p_H \times p_H}}$ et $\boldsymbol{U^{i,f,o,g}} \in \boldsymbol{R^{p_H \times p}}$ sont les poids que le réseau doit apprendre, $\boldsymbol{\sigma}$ la fonction **sigmoïde** et $\odot$ est le produit d'Hadamard.

Le principe des *gates* $\boldsymbol{i_t}, \boldsymbol{f_t}, \boldsymbol{o_t}$ est qu'elles servent à filtrer l'information venant du pas précèdent, l'information venant du pas actuel puis l'information finale. Le but étant de garder en mémoire que les informations importantes selon l'explication suivante :

- $\boldsymbol{f_t}$ (*Forget gate*) : prend une valeur entre 0 et 1. Quand cette valeur se rapproche de 0, l'ancien état ne compte pas beaucoup d'après l'équation 3.9.
- $\boldsymbol{i_t}$ (*Input gate*) : prend une valeur entre 0 et 1. Quand cette valeur se rapproche de 0, l'état actuel ne compte pas beaucoup d'après l'équation 3.9.
- $\boldsymbol{o_t}$ (*Output gate*) : sert à filtrer la sortie du réseau d'après l'équation 3.10.

3.1.4 Modèle séquence à séquence

Les modèles séquence à séquence (Seq2Seq) [84] sont une variante de réseaux de neurones récurrents. Ils sont beaucoup utiles pour la prédiction *Multi Input Multi Output (MIMO)* où l'on procure plusieurs variables dans le temps afin d'avoir des prédictions sur plusieurs pas en avance. Ces modèles sont composés d'une partie encodeur et d'une partie décodeur comme illustré dans la Figure 3.5. La partie encodeur est, comme pour les RNNs, composée d'une entrée, de couches cachées récurrentes et d'une sortie dirigée vers le décodeur. Le décodeur, quant à lui, est composée de plusieurs couches cachées récurrentes, avec une sortie pour chacune. Chaque sortie décrit la prédiction à un pas de temps à la fois, et cela de manière séquentielle. Le décodeur est la partie qui différencie le Seq2Seq des simples RNNs, car le RNN simple ne considère l'aspect temporel que pour l'entrée et non pas pour la sortie. Les couches cachées récurrentes peuvent être basées par l'implémentation des simples RNN ou bien d'autres variantes tout comme le LSTM.

On remarque que $\boldsymbol{y_{t+5}}$ prend les informations de la couche cachée au temps $\boldsymbol{t+4}$ qui prend

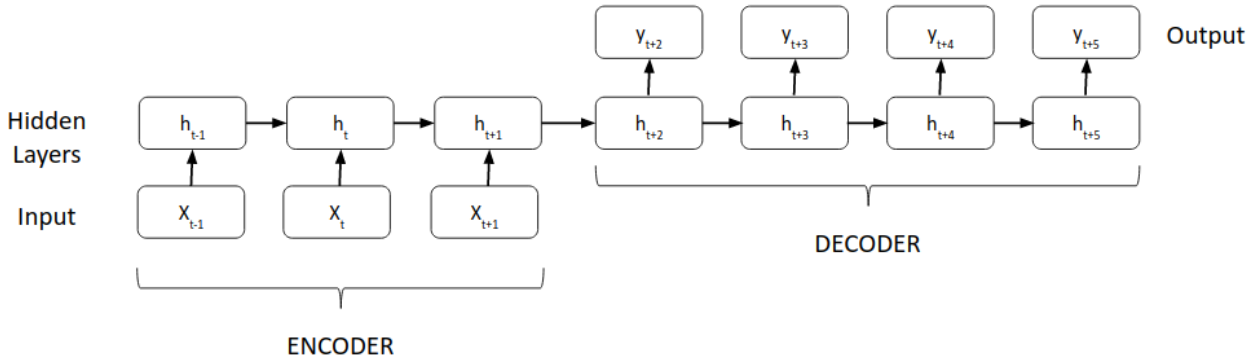

Figure 3.5 Schéma basique d'une architecture Seq2Seq

les informations de toutes les couches cachées qui précèdent de manière séquentielle, ce qui permet de capturer beaucoup plus de temporalité dans le problème surtout lorsque nous faisons de la prédiction sur le long terme.

3.1.5 Entraînement et test des modèles

Le fait d'entraîner un modèle revient à trouver la bonne combinaison de poids $\boldsymbol{\Theta}$ qui minimise l'écart entre les vraies valeurs et les prédictions. L'erreur quadratique est une fonction d'erreur populaire utilisée largement dans les problèmes de régression, également en séries chronologiques :

$$\boldsymbol{\mathcal{L}}\left(\boldsymbol{\Theta}\right) = \frac{\mathbf{1}}{\boldsymbol{m}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{m}} ||\boldsymbol{y}^{(i)} - \boldsymbol{f}(\boldsymbol{x}^{(i)}, \boldsymbol{\Theta})||^{\mathbf{2}} \tag{3.11}$$

$$\hat{\boldsymbol{\Theta}} \in \underset{\boldsymbol{\Theta}}{\mathbf{argmin}}\, \boldsymbol{\mathcal{L}}(\boldsymbol{\Theta}) \tag{3.12}$$

où $\boldsymbol{y}^{(i)}$ représente la vraie valeur de l'élément $\boldsymbol{i}$ qu'on souhaite prédire, $\boldsymbol{f}(\boldsymbol{x}^{(i)}, \boldsymbol{\Theta})$ représente la prédiction pour le même élément, et $\boldsymbol{m}$ le nombre d'échantillon dont nous disposons. Le but est de trouver le $\hat{\boldsymbol{\Theta}}$ qui minimise l'erreur $\boldsymbol{\mathcal{L}}$. La méthode de la descente de gradient stochastique est la méthode numérique la plus directe pour résoudre ce genre de problèmes, mais des nouvelles variantes de cette méthode semblent mieux performer théoriquement et pratiquement tels que la méthode ADAM [85].

Un ANN dépend fortement de beaucoup de hyperparamètres qui jouent un rôle très important dans la qualité du modèle de prédiction. Ces hyperparamètres, tels que le nombre de couches, le nombre de neurones, les fonctions d'activation et le traitement des données, doivent être trouvées de manière à ce qu'ils puissent générer les prédictions les plus précises possibles.

Nous divisons nos données en un jeu d'entraînement, de validation et un de test. L'ensemble de validation est utilisé pour évaluer le modèle formé sur les données d'entraînement et nous aider à trouver les hyperparamètres qui devraient a priori donner de bonnes performances sur l'ensemble de test. Une fois les bons hyperparamètres trouvés grâce à l'ensemble de validation, le modèle correspondant sera testé sur l'ensemble test pour donner une performance non biaisée. Il faut aussi éviter que le modèle ne sur-apprenne pas les données d'entraînement, pour qu'il puisse bien généraliser dans les autres ensembles tests. Pour cela, nous allons utiliser l'arrêt précoce (*Early Stopping*) [86] qui est une méthode de régularisation qui consiste à arrêter l'entraînement du modèle après quelques pas de temps quand son erreur sur l'ensemble de validation commence à augmenter.

Afin de tester notre modèle, nous utiliserons l'erreur moyenne absolue en pourcentage (MAPE) exprimée selon l'équation suivante :

$$\boldsymbol{MAPE} = \frac{\mathbf{1}}{\boldsymbol{m}} \sum_{\boldsymbol{t=1}}^{\boldsymbol{m}} \left| \frac{\boldsymbol{G_t - P_t}}{\boldsymbol{G_t}} \right| \tag{3.13}$$

Où $\boldsymbol{G_t}$ correspond à la vraie valeur de la charge et $\boldsymbol{P_t}$ correspond à la valeur de la charge prédite, à l'instant $\boldsymbol{t}$.

Le nombre de neurones dans la couche de sortie des modèles dépendra du nombre de pas futurs dont on souhaite avoir la valeur de la charge électrique. Ceci est également le cas pour les autres modèles de prédiction présentés ci-dessous.

3.1.6 Cadre d'étude

Le but de la prédiction est de trouver le meilleur algorithme de prédiction, la meilleure manière de sélectionner les variables nécessaires ainsi que la bonne stratégie qui permet d'obtenir les prédictions les plus précises au niveau d'un campus contenant plusieurs bâtiments de profils différents. Le chapitre 4 présente une étude comparative approfondie et complète de ce processus. Le cadre d'étude a été entièrement implémenté par Python et la librairie d'apprentissage profond Keras [87].

3.2 Gestion et optimisation de la charge

La revue de littérature nous a montré que l'une des techniques les plus en tendance ces dernières années pour la gestion de la charge dans les réseaux intelligents est l'apprentissage par renforcement (RL). le RL fait référence à un type de méthodes d'optimisation basées sur un processus d'essai et d'erreur, dans lequel un agent apprend à accomplir une tâche en

interagissant avec son environnement [43], i.e. tout ce qui est extérieur à l'agent concerné. À chaque itération du processus, l'agent reçoit des observations sur l'environnement. Sur la base de cette observation et de ses connaissances préalables, l'agent RL choisit une action afin de changer l'état de l'environnement et achever sa tâche. L'agent recevra par la suite le nouvel état de l'environnement ainsi qu'un signal de récompense pour prouver la qualité des actions prises. Pour trouver une stratégie de contrôle optimale, l'agent essaye de maximiser la somme actualisée des futures récompenses afin d'apprendre une fonction, appelée politique, qui lie les observations avec les meilleures actions possibles. La Figure 3.6 schématise le cycle par lequel un agent passe de manière générale.

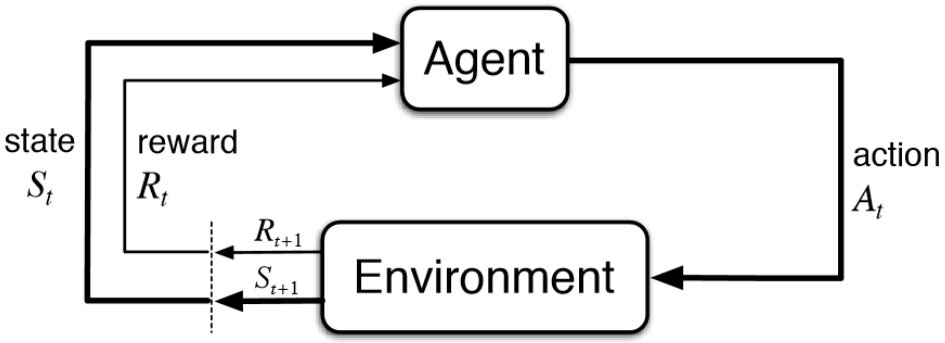

Figure 3.6 Principe de l'apprentissage par renforcement

Nous modélisons le problème comme un processus de décision de Markov (MDP) qui comprend principalement les parties suivantes :

- Un espace d'états $\boldsymbol{S}$, un espace d'actions $\boldsymbol{A}$.
- Une distribution dynamique de transition entre états $\boldsymbol{p(s_{t+1}|s_t, a_t)}$ qui suit la loi de Markov $\boldsymbol{p(s_{t+1}|s_1, a_1, \ldots, s_t, a_t) = p(s_{t+1}|s_t, a_t)}$ pour n'importe quelle trajectoire $\boldsymbol{s_1, a_1, \ldots, s_t, a_t}$ dans l'espace des (états, actions).
- Une fonction de récompense $\boldsymbol{r : S \times A \to \mathbb{R}}$.

Une politique est utilisée pour sélectionner les actions dans le cadre du MDP. Elle peut être déterministe $\boldsymbol{\pi : S \to A}$, mais en général, la politique est stochastique et désignée par $\boldsymbol{\pi : S \to P(A)}$, où $\boldsymbol{P(A)}$ est l'ensemble des mesures de probabilité sur $\boldsymbol{A}$ et $\boldsymbol{\pi(a_t|s_t)}$ est la densité de probabilité conditionnelle de $\boldsymbol{a_t}$ associée à la politique. L'agent utilise sa politique pour interagir avec le MDP afin de donner une trajectoire d'états, d'actions et de récompenses, $\boldsymbol{h_{1:T} = (s_1, a_1, r_2 ..., s_T, a_T, r_{T+1})}$ sur $\boldsymbol{S \times A \times \mathbb{R}}$.

Le rendement $\boldsymbol{G_t}$ est la récompense totale actualisée à partir du pas de temps $\boldsymbol{t}$, $\boldsymbol{G_t = \sum_{k=0}^{\infty} \gamma^k . r_{t+k+1}}$ où $\boldsymbol{0 < \gamma < 1}$ afin d'accorder une importance aux récompenses futures

mais avec un impact moins conséquent que pour les récompenses immédiates. Les fonctions de valeurs d'états et d'actions (*State-Value-Function*, *Action-Value-Function*) sont définies comme étant l'espérance de la récompense totale, $\boldsymbol{V_\pi(s) = \mathbb{E}[G_t|S_t = s; \pi]}$ qui représente l'importance de l'état en moyennant les actions prises à ce même état $\boldsymbol{s}$ et $\boldsymbol{Q_\pi(s, a) = \mathbb{E}[G_t|S_t = s, A_t = a; \pi]}$ qui représente la qualité de l'action $\boldsymbol{a}$ appliquée à l'état $\boldsymbol{s}$. L'objectif de l'agent est d'obtenir une politique qui maximise la récompense cumulative actualisée à partir de l'état de départ.

Afin de catégoriser les différents algorithmes RL et montrer les contrastes existants, nous utiliserons quelques algorithmes représentés dans la Figure 3.7.

(2).png (2).png (2).png (2).png

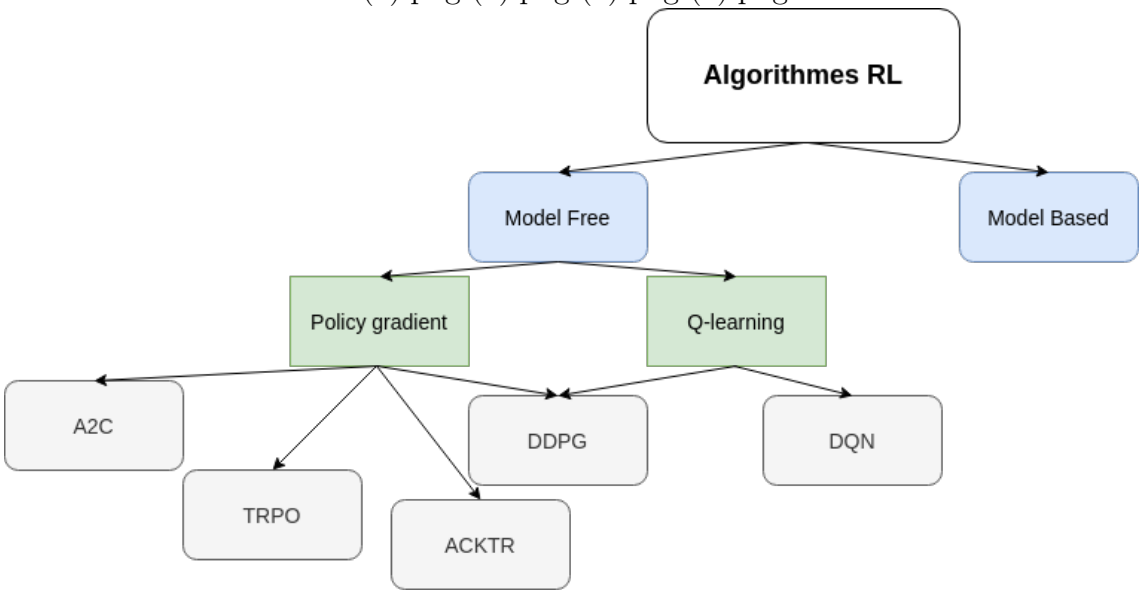

Figure 3.7 Liste non exhaustive des algorithmes modernes de RL

Afin de rester le plus fidèle aux modèles et à leur description, nous allons parfois noter des termes de quelques approches et raisonnements avec leur terminologie originale en anglais.

En apprentissage par renforcement, on retrouve deux types d'algorithmes :

- *Model-Based* : se basent sur une connaissance préalable de la fonction qui prédit les transitions d'états $\boldsymbol{p(s_{t+1}|s_t, a_t)}$ et les récompenses. Ils sont fortement influencés par la théorie du contrôle. Ils ont plus d'hypothèses et d'approximations, ce qui signifie qu'elles sont limitées à des tâches spécifiques.
- *Model-Free* : n'ont pas vraiment accès au modèle de transition ni aux récompenses et donc n'en tirent tout simplement aucune information au préalable. Ils sont plus populaires et ont été plus largement développées et testées que les méthodes *Model-*

based. De plus quand les actions à prendre sont continues (e.g. si l'action était l'angle de direction des roues dans un jeu de voiture), l'usage d'un modèle de transition serait extrêmement coûteux en temps et en mémoire.

Dans ce mémoire, nous allons nous intéresser aux méthodes de types *Model-Free* à savoir les approches *Q-learning* et *Policy Gradient*, car cela correspondra mieux à la modélisation de notre problème, qui consistera à prendre des actions continues et parce que nous n'avons pas de réelles études nous permettant de donner à l'agent l'accès aux modèles de transitions et de récompenses au préalable.

Avant de rentrer dans les détails du fonctionnement de ces algorithmes, nous définissons quelques concepts dont nous nous servirons par la suite :

- Épisode : signifie l'entière séquence d'interaction de l'agent avec son environnement jusqu'à ce qu'elle arrive à à l'état terminal (e.g. l'arrivée au héros du jeu vidéo à son but).
- Approche $\boldsymbol{\epsilon}$-greedy : lors de l'entraînement d'un agent RL, on souhaite qu'il apprenne la bonne politique en explorant tous les cas de figure possibles d'états et d'actions. Cependant, cela est compliqué à satisfaire vu les grands espaces d'états et d'actions sur lesquelles il travaille. Cet algorithme prend la meilleure action la plupart du temps (i.e. exploitation), mais fait de l'exploration aléatoire de temps en temps avec une probabilité $\boldsymbol{\epsilon}$.

Nous allons alors présenter les concepts généraux sur lesquelles reposent les méthodes Q-learning, Policy Gradients puis Actor-Critic. Par la suite, et pour des raisons de simplicité, nous allons aborder les algorithmes utilisant ces principes de manière moins abstraite, le principe étant de comprendre leur fonctionnement et leurs différences.

3.2.1 Q-learning

Le principe de cette approche est d'apprendre à associer à chaque couple (État, Action) une valeur $\boldsymbol{Q(s, a)} \in \mathbb{R}$ pour des états et actions de valeurs discrètes, avec $\boldsymbol{s} \in \mathbb{R}^{\boldsymbol{n}}$ où $\boldsymbol{n}$ désigne la longueur du vecteur d'état $\boldsymbol{s}$ et $\hat{\boldsymbol{a}} \in \mathbb{R}^{\boldsymbol{p}}$ où $\boldsymbol{p}$ désigne le nombre d'action à prendre en même temps étant à l'état $\boldsymbol{s}$. Une fois entraîné de manière épisodique, et en étant à l'état $\boldsymbol{s}$, l'agent choisira alors l'action $\hat{\boldsymbol{a}}$ telle que :

$$\hat{\boldsymbol{a}} = \boldsymbol{\pi}(\boldsymbol{s}) \in \underset{\boldsymbol{a}}{\mathbf{argmax}}\, \boldsymbol{Q}(\boldsymbol{s}, \boldsymbol{a}) \tag{3.14}$$

L'entraînement d'un agent dans ce contexte se base sur l'équation de Bellman [43] pour la fonction de valeurs d'états $\boldsymbol{V}$ en utilisant la formule suivante :

$$\boldsymbol{G_t} = \sum_{\boldsymbol{k=0}}^{\infty} \boldsymbol{\gamma^k . r_{t+k+1}} = \boldsymbol{r_{t+1} + \gamma . G_{t+1}} \tag{3.15}$$

Cela nous donne :

$$\begin{aligned} \boldsymbol{V(s)} = \mathbb{E}[\boldsymbol{G_t}|\boldsymbol{S_t = s}] &= \mathbb{E}[\sum_{\boldsymbol{k=0}}^{\infty} \boldsymbol{\gamma^k . r_{t+k+1}}|\boldsymbol{S_t = s}] && (3.16)\\ &= \mathbb{E}[\boldsymbol{r_{t+1} + \gamma . G_{t+1}}|\boldsymbol{S_t = s}] && (3.17)\\ &= \mathbb{E}[\boldsymbol{r_{t+1} + \gamma . V(s_{t+1})}|\boldsymbol{S_t = s}] && (3.18) \end{aligned}$$

L'équation de Bellman peut également être exprimée pour la fonction de valeurs d'actions $\boldsymbol{Q}$ sous la forme suivante :

$$\begin{aligned} \boldsymbol{Q(s,a)} &= \mathbb{E}[\boldsymbol{r_{t+1} + \gamma . V(s_{t+1})}|\boldsymbol{S_t = s, A_t = a}] && (3.19)\\ &= \mathbb{E}[\boldsymbol{r_{t+1} + \gamma . E_a(Q(s_{t+1}, a))}|\boldsymbol{S_t = s, A_t = a}] && (3.20) \end{aligned}$$

Le Q-learning [88] s'inspire d'un principe appelé le *Temporal Difference (TD) learning* [43] dont l'idée clé est de mettre à jour la fonction de valeur $\boldsymbol{V(s_t)}$ vers un retour estimé $\boldsymbol{r_{t+1} + V(s_{t+1})}$ (connu sous le nom de *TD target*). La mesure dans laquelle nous voulons mettre à jour la fonction de valeur est contrôlée par l'hyperparamètre du taux d'apprentissage $\boldsymbol{\alpha}$:

$$\begin{aligned} \boldsymbol{V(s_t)} &\leftarrow \boldsymbol{(1-\alpha).V(s_t) + \alpha . G_t} && (3.21)\\ \boldsymbol{V(s_t)} &\leftarrow \boldsymbol{(1-\alpha).V(s_t) + \alpha.(r_{t+1} + \gamma . V(s_{t+1})} && (3.22)\\ \boldsymbol{V(s_t)} &\leftarrow \boldsymbol{V(s_t) + \alpha.(r_{t+1} + \gamma . V(s_{t+1}) - V(s_t))} && (3.23) \end{aligned}$$

On obtient de même pour la fonction de valeur d'action $\boldsymbol{Q}$:

$$\boldsymbol{Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha.(r_{t+1} + \gamma . Q(s_{t+1}, a_{t+1}) - Q(s_t, , a_t))} \tag{3.24}$$

En Q-learning, le choix de l'action $\boldsymbol{a_{t+1}}$ dans l'équation 3.24 est fait suivant l'équation :

$$\boldsymbol{Q(s_{t+1}, a_{t+1}) = \max_a Q(s_{t+1}, a)} \tag{3.25}$$

Enfin pour résumer, l'algorithme Q-learning est décrit d'une manière générale en dessous : Théoriquement, on peut mémoriser $\boldsymbol{Q(.,.)}$ pour tous les couples état-action dans le Q-learning,

Algorithm 1 Q-learning

Initialisation des valeurs $\boldsymbol{Q(s,a)}\ \forall \boldsymbol{s,a}\quad \boldsymbol{Q(s,a)=0}$
Initialisation n = nombre d'épisodes à parcourir
for Épisode dans [1,. . . ,n] **do**
    Initialisation pas de temps $\boldsymbol{t=0}$
    while Épisode non fini **do**
        Sélection d'un état aléatoire $\boldsymbol{s_t}$
        Choix d'action $\boldsymbol{a_t}$ selon $\boldsymbol{a_t = \mathbf{argmax}_a\, Q(s_t,a)}$ ou $\boldsymbol{\epsilon}$-greedy (i.e. exploration)
        Application de $\boldsymbol{a_t}$ à $\boldsymbol{s_t}$
        Observation de la récompense $\boldsymbol{r_{t+1}}$ et prochain état $\boldsymbol{s_{t+1}}$
        Mise à jour $\boldsymbol{Q(s_t,a_t) = Q(s_t,a_t) + \alpha.(r_{t+1} + \gamma.\mathbf{max}_a\, Q(s_{t+1},a) - Q(s_t,,a_t))}$
        $\boldsymbol{t = t+1}$
    end while
end for

comme dans un tableau gigantesque. Cependant, cela devient rapidement impossible à calculer lorsque l'espace d'état et d'action est grand. Ainsi pour pallier ce problème, l'utilisation des modèles d'apprentissage machine pour approcher les valeurs de $\boldsymbol{Q}$ semble être une bonne solution. Par exemple, si le paramètre $\boldsymbol{\theta}$ représente la matrice de poids du modèle d'approximation, approximer $\boldsymbol{Q(s,a)}$ revient à calculer $\boldsymbol{Q(s,a;\theta)}$.

L'utilisation des réseaux de neurones combinés au Q-learning donne naissance au *Deep Q-Networks* (DQN) [70], qui consiste à avoir un réseau de neurones appelé *Policy Network* ou Q-Network, qui prend en entrée un vecteur définissant un état $\boldsymbol{s}$ dans le but d'estimer en sortie la valeur de $\boldsymbol{Q(s,a)}$ pour toute action $\boldsymbol{a}$. On peut donc utiliser l'algorithme de descente du gradient pour résoudre l'équation de Bellman en considérant que $\boldsymbol{r_{t+1} + \mathbf{max}_a\, Q(s_{t+1},a;\theta)}$ soit la cible à atteindre pour $\boldsymbol{Q(s_t,a_t)}$ en utilisant l'erreur des moindres carrés $\boldsymbol{J}$, où $\boldsymbol{s'}$ représente l'état où l'agent se retrouve en appliquant $\boldsymbol{a}$ à l'état $\boldsymbol{s}$:

$$\boldsymbol{J(\theta_i) = (r + \gamma.\max_{a'} Q(s',a';\theta_{i-1}) - Q(s,a,\theta_i))^2} \tag{3.26}$$

$$\boldsymbol{\nabla_{\theta_i} J(\theta_i) = -2.\nabla_{\theta i} Q(s,a,\theta_i).(r + \gamma.\max_{a'} Q(s',a';\theta_{i-1}) - Q(s,a,\theta_i))} \tag{3.27}$$

où $\boldsymbol{\theta_{i-1}, \theta_i}$ représentent simultanément les poids du *Policy Network* à l'itération $\boldsymbol{i\text{-}1}$ et $\boldsymbol{i}$.

Néanmoins en pratique, cette approche diverge à cause des corrélations entre les différentes itérations dans la séquence. Pour résoudre ce problème, les DQN visent à améliorer et à stabiliser considérablement la procédure de formation du Q-learning par deux mécanismes intéressants :

- Experience replay : Toutes les séquences $\boldsymbol{e_t = (s_t, a_t, r_{t+1}, s_{t+1})}$ de l'épisode sont

stockées dans une mémoire de relecture $\boldsymbol{D_T} = \boldsymbol{e_1}, ..., \boldsymbol{e_T}$. Lors des mises à jour de Q-learning, un échantillon est tiré au hasard de $\boldsymbol{D_T}$ et peut donc être utilisé plusieurs fois. L'expérience replay améliore l'efficacité des données, supprime les corrélations dans les séquences d'observation et lisse les changements dans la distribution des données.

- Target Network : Afin de rendre l'entraînement plus stable et surmonter les oscillations à court terme, il est suggéré de ne pas utiliser le même réseau de neurones pour calculer $\boldsymbol{Q(s_t, a_t; \theta)}$ et $\boldsymbol{\max_a Q(s_{t+1}, a; \theta)}$. On rajoute un deuxième réseau de neurones appelé *Target Network*, similaire au premier, avec les poids $\boldsymbol{\theta^-}$. Ses poids seront mis à jour après un nombre de pas $\boldsymbol{k}$ dans le temps et il servira à calculer $\boldsymbol{\max_a Q(s_{t+1}, a; \theta^-)}$.

Finalement, l'entraînement se fera à l'aide de la descente de gradient appliqué à l'erreur des moindres carrés $\boldsymbol{(r + \gamma . \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta))^2}$. La Figure 3.8 représente le fonctionnement général d'un DQN.

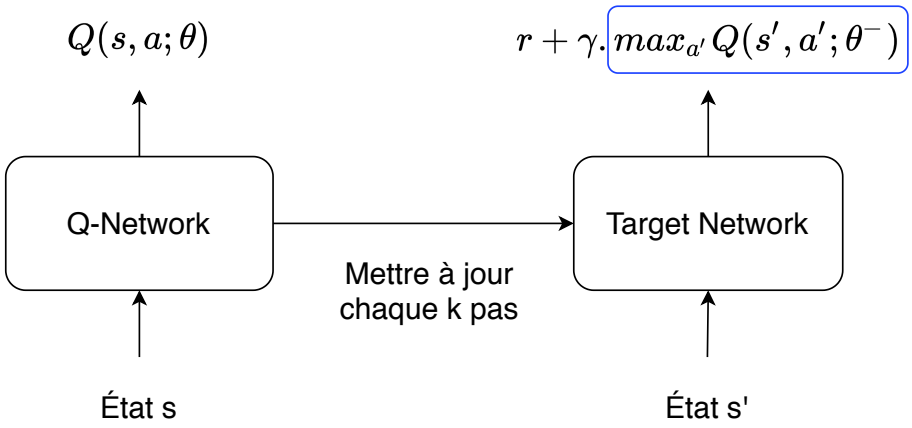

Figure 3.8 Représentation du fonctionnement général d'un DQN

Il faut noter que le DQN permet désormais de travailler dans des environnements où les états peuvent être de nature continue. Cependant, sa structure oblige à travailler avec des actions de nature discrète. Afin de pouvoir utiliser des actions continues, d'autres approches ont été développées telles que les méthodes Policy Gradients.

3.2.2 Policy Gradient

Tandis que les méthodes de Q-learning visent à apprendre la fonction de valeur d'état/action et à sélectionner les actions en conséquence, les méthodes de Policy Gradient (PG) [43] apprennent plutôt la politique directement avec une fonction paramétrée par rapport à $\boldsymbol{\theta}$, $\boldsymbol{\pi(a|s; \theta)}$ qui désigne la distribution des actions $\boldsymbol{a}$ conditionnellement à $\boldsymbol{s}$. Généralement, un

réseau de neurones de paramètres $\boldsymbol{\theta}$ est utilisé afin de faire cette estimation de la politique.

On définit la fonction de récompense $\boldsymbol{J}$ (contraire d'une fonction de perte) dans le cas d'espaces continus dans l'équation 3.28. L'objectif est d'entraîner l'algorithme afin qu'il maximise cette fonction par rapport à $\boldsymbol{\theta}$ pour ainsi espérer maximiser le rendement $\boldsymbol{G_t}$.

$$\boldsymbol{J(\theta) = \sum_{s \in S} d_\pi(s).V_\pi(s) = \sum_{s \in S} d_\pi(s). \sum_{a \in A} \pi(a|s;\theta).Q_\pi(s,a)} \tag{3.28}$$

où $\boldsymbol{d_\pi(s)}$ représente la fréquence de visites de l'état s durant l'épisode en suivant la politique $\boldsymbol{\pi}$, $\boldsymbol{V_\pi(s)}$ représente la fonction de valeur d'état, $\boldsymbol{\pi(a|s;\theta)}$ l'estimation de la probabilité de prendre l'action $\boldsymbol{a}$ en étant dans l'état $\boldsymbol{s}$, $\boldsymbol{Q_\pi(s,a)}$ représente la fonction de valeur d'action.

Partant du principe que nous souhaitons utiliser l'algorithme du gradient pour trouver le $\boldsymbol{\theta}$ qui maximise $\boldsymbol{J}$, nous devons alors être capable de calculer $\boldsymbol{\nabla_\theta J(\theta)}$. Le *Policy Gradient Theorem* [43] est très important, et pose les bases théoriques de diverses méthodes de type PG. Ce théorème montre que ce gradient peut s'exprimer comme suit :

$$\boldsymbol{\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[Q_\pi(s,a).\nabla_\theta ln(\pi(a|s;\theta)]} \tag{3.29}$$

Ce théorème est généralisé [43] pour inclure une comparaison de la valeur de l'action à une référence arbitraire b(s) :

$$\boldsymbol{\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[(Q_\pi(s,a) - b(s)).\nabla_\theta ln(\pi(a|s;\theta)]} \tag{3.30}$$

Où $\boldsymbol{b(s)}$ peut être n'importe quelle fonction, même une variable aléatoire, tant qu'elle ne varie pas avec a. L'équation reste valable car la quantité soustraite est égale à zéro [43]. Le but principal de cette généralisation est de réduire la variance du gradient et donc la vitesse d'apprentissage, et cela sans changer son biais. L'un des choix les plus communs pour $\boldsymbol{b(s)}$ est $\boldsymbol{V_\pi(s)}$; cela nous donne $\boldsymbol{A(s,a) = Q_\pi(s,a) - V_\pi(s)}$ appelé la fonction *Avantage*, et qui représente l'importante de choisir l'action $\boldsymbol{a}$ plutôt que l'espérance suivant la totalité des actions en étant à l'état $\boldsymbol{s}$.

Il est également important de noter que ces algorithmes marchent aussi bien pour les espaces d'actions discrètes que pour les espaces d'actions continues, et dans les deux cas le réseau de neurones produit une distribution des actions $\boldsymbol{\pi(a|s;\theta)}$ pour chaque état $\boldsymbol{s}$. Pour les actions discrètes, le nombre de neurones à la couche de sortie sera égal au nombre d'actions possibles. Pour les actions continues, il est sujet de prédire les paramètres de la distribution $\boldsymbol{\pi(a|s;\theta)}$ (e.g. prédire la moyenne $\boldsymbol{\mu_\theta}$ et la variance $\boldsymbol{\sigma_\theta}$ de la distribution $\boldsymbol{\pi(a|s;\theta)}$ si elle est supposée normale).

3.2.3 Actor-Critic

La méthode Actor Critic tend à combiner DQN et PG pour l'entraînement de l'agent à estimer la politique $\boldsymbol{\pi}$ ainsi que la fonction de valeur d'action $\boldsymbol{Q}$, et cela à l'aide de deux composantes :

- Actor : est un réseau de neurones de paramètres $\boldsymbol{\theta}$ destiné à prendre l'état $\boldsymbol{s}$ en entrée, et produire la distribution d'actions $\boldsymbol{a}$ conditionnelle à $\boldsymbol{s}$: $\boldsymbol{\pi(a|s;\theta)}$. À l'aide de cette réponse, l'agent pourra échantillonner son action et la donner en entrée au Critic.
- Critic : est un réseau de neurones de paramètres $\boldsymbol{w}$ qui prend l'état $\boldsymbol{s}$ et l'action $\boldsymbol{a}$ en entrée pour estimer $\boldsymbol{Q(s,a;w)}$ pour ainsi "Critiquer" le choix de l'action prise par l'Actor.

Les deux réseaux $\boldsymbol{\theta, w}$ sont entraînés en même temps, chacun à l'aide de l'algorithme du gradient. Pour $\boldsymbol{\theta}$, l'équation 3.29 sera utilisée, mais pour $\boldsymbol{w}$, l'équation 3.27 subira un léger changement vu la structure du modèle et l'extrême difficulté à trouver $\mathbf{max}_{a'}\, \boldsymbol{Q(s',a';w)}$ quand l'espace des actions est continu. Pour cela, nous estimons $\mathbf{max}_{a}\, \boldsymbol{Q(s',a)} \approx \boldsymbol{Q(s',a';w)}$ où $\boldsymbol{a'}$ est un échantillon de la distribution $\boldsymbol{\pi(a'|s';\theta)}$. Une fois entraîné, l'Actor $\boldsymbol{\theta}$ sera utilisé afin de prendre les bonnes actions.

3.2.4 Algorithmes

Dans cette partie, vu la quantité d'information et la complexité de chaque modèle, nous présenterons de manière abrégée le fonctionnement de quatre algorithmes décrivant l'état de l'art du RL, faisant partie des algorithmes de type *model-free*. Une description plus détaillée de chaque algorithme peut être trouvée dans l'article concerné.

Deep Deterministic Policy Gradient (DDPG)

DDPG [89] est une méthode Q-learning visant à étendre les actions sur des espaces continus. Elle suit le principe de l'*Actor-Critic*, et sa caractéristique principale est que sa politique est déterministe, cela signifie qu'à la place de renvoyer de manière stochastique une distribution $\boldsymbol{\pi(a|s;\theta)}$, l'Actor enverra de manière déterministe une action $\boldsymbol{a = \pi(s;\theta)}$ et le Critic enverra $\boldsymbol{Q(s,a;w) = Q(s,\pi(s;\theta);w)}$. Optimiser les paramètres des deux réseaux $\boldsymbol{\theta, w}$ revient à utiliser le théorème du gradient sur l'erreur $\boldsymbol{(r + \gamma.Q(s',\pi(s';\theta);w) - Q(s,\pi(s;\theta);w))^2}$ par rapport à $\boldsymbol{w}$ pour le Critic. Afin d'entraîner la politique ($\boldsymbol{\theta}$), l'équation 3.14 permet d'estimer la politique $\boldsymbol{\pi^{k+1}(s)} \in \mathbf{argmax}_{a}\, \boldsymbol{Q^{\pi^k}(s,a)}$ en passant de l'itération $\boldsymbol{k}$ à l'itération $\boldsymbol{k+1}$, $\boldsymbol{Q^{\pi^k}(s,a)}$ étant la fonction de valeur d'action obtenue en utilisant la politique $\boldsymbol{\pi^k}$. Cependant, dans les espaces d'actions continues, la mise à jour de la politique devient problématique, nécessitant une maximisation globale à chaque étape. Une alternative simple et

intéressante sur le plan des calculs consiste à orienter la politique vers le gradient de $\boldsymbol{Q}$, plutôt que de maximiser globalement le $\boldsymbol{Q}$. Plus précisément, pour chaque état visité s, les poids de l'Actor $\boldsymbol{\theta}$ sont mis à jour en proportion du gradient $\boldsymbol{\nabla_\theta Q(s, \pi(s;\theta); w)}$. En appliquant la règle de dérivation en chaîne, une mise à jour de $\boldsymbol{\theta}$ se fera comme suit :

$$\boldsymbol{\theta = \theta + \alpha_\theta . \nabla_\theta \pi(s;\theta) . \nabla_a Q(s, a; w)|_{a=\pi(s;\theta)}} \tag{3.31}$$

Où $\boldsymbol{\alpha_\theta}$ est le taux d'apprentissage pour l'Actor. De plus, pour encourager de l'exploration, un bruit peut être ajouté à l'action prise par l'Actor avant d'être donnée au Critic.

Au-delà de l'entraînement des deux réseaux, afin d'obtenir plus de stabilité de réduire la variance, le DDPG propose la création de deux réseaux Actor et Critic de type *Target Networks* de paramètres $\boldsymbol{\theta', w'}$ qui seront mis à jour doucement par les poids des vrais à l'aide du *soft target network updates* introduit dans [89] sous la forme :

$$\boldsymbol{\theta' \leftarrow \tau . \theta + (1 - \tau) . \theta'} \tag{3.32}$$

$$\boldsymbol{w' \leftarrow \tau . w + (1 - \tau) . w'} \tag{3.33}$$

Advantage Actor-Critic (A2C)

A2C [90] est une approche de type PG, et suit également le principe de l'*Actor-Critic* mais avec quelque différences :

- L'estimation de $\boldsymbol{Q(s, a)}$ ne se fait pas de manière paramétrique (i.e. $\boldsymbol{Q(s, a; w)}$). Nous l'estimons plutôt directement comme suit $\boldsymbol{Q(s_t, a_t) = G_t}$ qui est calculé suivant la loi $\boldsymbol{\pi}$.
- Le Critic obtient en entrée l'état $\boldsymbol{s}$ et produit $\boldsymbol{V(s; w)}$. Cette estimation sera utilisée comme référence dans l'équation 3.30.
- Pour entraîner le Critic, nous utilisons la descente de gradient sur l'erreur des moindres carrés $\boldsymbol{(G - V(s; w))^2}$. Pour l'Actor, l'entraînement se fera suivant l'équation 3.30.

L'un des atouts de A2C est qu'il permet le parallélisme dans l'entraînement de multiples agents. Ces agents sont entraînés en même temps, et leur mise à jour se fait de manière synchronisée. Chaque agent s'entraîne toutefois de manière indépendante des autres, ce qui encourage l'exploration et donc la rapidité de convergence des modèles et leur bonne généralisation tout en réduisant la variance de l'agent.

Trust Region Policy Optimization (TRPO)

Pour améliorer la stabilité de l'entraînement, nous devons éviter les mises à jour des paramètres qui modifient trop la politique à un moment donné. TRPO [91] est une méthode Actor-Critic qui concrétise cette idée en appliquant de l'échantillonnage préférentiel suivi d'une contrainte basée sur la divergence Kullback-Leibler (KL) [92] qui sert à calculer la distance entre deux distributions. La divergence de KL permet dans ce cadre d'éviter que la distribution stochastique de la politique ne change trop entre chaque itération d'où le nom "Trust Region". Pendant que la mise à jour du Critic ($\boldsymbol{w}$) se fait de la même manière que pour A2C, la mise à jour du paramètre $\boldsymbol{\theta}$ du réseau estimant la politique de TRPO se fait comme suit :

$$\boldsymbol{\theta_{k+1}} \in \underset{\boldsymbol{\theta}}{\textbf{argmax}} \quad \boldsymbol{\mathcal{L}(\theta_k, \theta)} \tag{3.34}$$

$$\text{s.c.} \quad \boldsymbol{\bar{D}_{KL}(\theta||\theta_k) \leq \delta} \tag{3.35}$$

Où $\boldsymbol{\mathcal{L}(\theta_k, \theta)}$ mesure combien la politique $\boldsymbol{\pi_\theta}$ est performante relativement à l'ancienne politique $\boldsymbol{\pi_{\theta_k}}$ en générant des séquences à l'aide de cette ancienne politique. $\boldsymbol{\bar{D}_{KL}(\theta||\theta_k)}$ désigne l'espérance de la divergence KL entre les deux politiques sur les états visités par l'ancienne politique :

$$\boldsymbol{\mathcal{L}(\theta_k, \theta)} = \underset{\boldsymbol{s,a\sim\pi_{\theta_k}}}{\mathbb{E}}\left(\frac{\boldsymbol{\pi_\theta(a|s)}}{\boldsymbol{\pi_{\theta_k}(a|s)}}\boldsymbol{A^{\pi_{\theta_k}}(s,a)}\right) \tag{3.36}$$

$$\boldsymbol{\bar{D}_{KL}(\theta||\theta_k)} = \underset{\boldsymbol{s\sim\pi_{\theta_k}}}{\mathbb{E}}\left(\boldsymbol{D_{KL}\left(\pi_\theta(\cdot|s)||\pi_{\theta_k}(\cdot|s)\right)}\right) \tag{3.37}$$

Où $\boldsymbol{\pi_\theta}$ et $\boldsymbol{A^{\pi_{\theta_k}}}$ sont estimés de la même manière qu'avec A2C. Le problème d'optimisation montre que nous ne sommes intéressés que par la recherche d'un point optimal pour $\boldsymbol{\mathcal{L}}$ dans une région de confiance. Les points en dehors de la région de confiance peuvent avoir des récompenses plus élevées, mais nous les ignorerons parce que nous ne pouvons pas faire confiance à la précision de l'amélioration. Afin de résoudre ce problème d'un côté plus pratique, il est utile d'approximer $\boldsymbol{\mathcal{L}(\theta_k, \theta)}$ et $\boldsymbol{\bar{D}_{KL}(\theta||\theta_k)}$ donnant le résultat suivant :

$$\boldsymbol{\theta_{k+1}} = \underset{\boldsymbol{\theta}}{\textbf{argmax}} \quad \boldsymbol{g^T(\theta - \theta_k)} \tag{3.38}$$

$$\text{s.t.} \quad \boldsymbol{\frac{1}{2}(\theta - \theta_k)^T H(\theta - \theta_k) \leq \delta.} \tag{3.39}$$

où $\boldsymbol{g}$ désigne le gradient de $\boldsymbol{\mathcal{L}(\theta_k, \theta)}$ en $\boldsymbol{\theta_k}$, i.e. $\boldsymbol{g = \nabla_\theta \mathcal{L}(\theta_k, \theta)|_{\theta=\theta_k}}$, $\boldsymbol{H}$ désigne la Hessienne de l'espérance de la divergence KL, i.e. $\boldsymbol{H = \nabla^2_\theta \bar{D}_{KL}(\theta||\theta_k)|_{\theta=\theta_k}}$ qui est équivalente à la matrice de l'information de Fisher (FIM) $\boldsymbol{H = F = \mathbb{E}_{s\sim\pi_{\theta_k}}(\nabla_\theta \pi_\theta(\cdot|s).\nabla_\theta \pi_\theta(\cdot|s)^T|_{\theta=\theta_k})}$

selon [91]. Par la suite le problème peut être résolu à travers la dualité Lagrangienne [93] :

$$\boldsymbol{\theta_{k+1}} \in \boldsymbol{\theta_k} + \sqrt{\frac{\boldsymbol{2\delta}}{\boldsymbol{g^T H^{-1} g}}} \boldsymbol{H^{-1} g} \tag{3.40}$$

Où $\boldsymbol{H^{-1}}$ est calculé par le biais de la méthode du gradient conjugué [94].

Actor Critic using Kronecker-Factored Trust Region (ACKTR)

ACKTR [95] est une solution alternative à TRPO qui combine trois techniques distinctes : les méthodes Actor-Critic, l'optimisation des régions de confiance (Trust Regions) pour une amélioration plus cohérente, et la factorisation de Kronecker distribuée (K-FAC) [96] comme méthode d'optimisation de l'Actor et le Critic.

Dans TRPO, l'estimation de $\boldsymbol{H^{-1} = F^{-1}}$ en utilisant la méthode du gradient conjugué nécessite une complexité $\boldsymbol{\Theta(n^2)}$ ce qui rend le processus très difficile quand des réseaux de neurones profonds sont utilisés. Pour corriger ce problème, ACKTR emploie K-FAC qui est une méthode d'optimisation, utilisée dans ce cadre pour approximer l'inverse de FIM, et cela en calculant les mises à jour des poids du réseau de neurones une couche de réseau à la fois. Chaque calcul n'implique que les neurones d'une couche au lieu de tous les neurones du réseau entier. Par conséquent, la complexité du calcul diminue de manière significative. De plus, par rapport à l'optimisation de premier ordre comme la descente de gradient stochastique, ACKTR n'ajoute que seulement 10% à 25% de coût de calcul en plus.

Pour l'entraînement du Critic, le but sera de diminuer l'erreur $\boldsymbol{J(w) = (G - V(s; w))^2}$, soit par une méthode de premier ordre telle que la descente du gradient, ou une méthode de deuxième ordre telle que l'algorithme de Newton. K-FAC est aussi applicable dans ce cas pour calculer l'inverse de $\boldsymbol{J(w)^T . J(w)}$; il sera donc possible de mettre à jour chaque couche une par une pour réduire la complexité du calcul.

3.2.5 Cadre d'étude

Deux cas d'études seront présentés dans ce projet, le but principal étant de réduire le pic de puissance en contrôlant la charge et maintenir le confort (traduit par le contrôle de la température intérieure) des utilisateurs. Le premier consiste à montrer l'impact de l'ajout des prédictions de la charge comme information à l'agent quand le besoin est de gérer la demande et le confort des utilisateurs dans des bâtiments résidentiels. Le deuxième est une étude faite dans le but de construire un algorithme unique généralisé de RL appelé GA2TC capable de bien contrôler la charge et le confort quelle que soit la nature et la taille du bâtiment. Ce

cadre d'étude sera présenté en détails et de manière approfondie dans le chapitre 5.

Pour implémenter ces cadres d'études, nous avons entièrement développé un environnement de *Gym* sur Python de zone thermique [78]. Les algorithmes ont été utilisés par le biais de la librairie Python *Stable-baselines* [97].

CHAPITRE 4 ARTICLE 1: AGGREGATED SHORT TERM LOAD FORECASTING FOR HETEROGENEOUS BUILDINGS USING DEEP LEARNING WITH PEAK ESTIMATION

Authors : Amine Bellahsen and Hanane Dagdougui

Manuscript submitted to : Energy and Buildings, 2020

Abstract : System operations and planning is a crucial aspect of power system management. It aims to maintain the equilibrium between supply and demand of electricity. For many services, consumers have to pay more for electricity in periods of high peak demand. Consequently, if consumers have knowledge about the expected peak load ahead of time, such extra charges may be avoided. Accurate forecasting of energy demand and, therefore, information on expected peak loads will not only help to provide a reliable supply of electricity, but it can also be useful in reducing the cost of electricity at the consumer level. In this paper, we develop a comparative study for aggregated short-term load forecasting using different data strategies and comparing two prediction levels : the first one aims to predict the aggregated load using a whole district data set, while the second one focuses on performing predictions on a lower level then aggregating these predictions at the district level. After finding the best forecasting model and strategy, these accurate predictions will help us predict the percentage of peak over a certain subscribed power in the entire district. For the load prediction, we obtain a mean absolute percentage error between 1.83% and 5.18% depending on the prediction horizon.

Keywords : Short-term load forecasting, Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Mean absolute percentage error (MAPE), Heterogeneous buildings, peak prediction

4.1 Introduction

In the Smart Grid paradigm, it has become essential to perform assistance to consumers/users in their power consumption management using different programs. Some demand response (DR) programs implement the price-based or incentive schemes to encourage changes within the end-uses of electricity [98]. The important aspect of managing power consumption is to reduce electricity consumption during times of high market costs or once the reliability of the network is endangered. Such actions create benefits not only for the electricity provider but also for consumers. To respond appropriately to these DR problems, demand aggregator companies are starting to offer the consumers the possibility to participate in DR activities

using their controllable loads [7], either by reducing loads during specific events, by managing controllable loads in real-time market, or through proposing financial rewards based upon the amount of flexible load provided.

In order to make DR programs operational, reliable load prediction tools and energy management systems must be implemented at the aggregator level. DR of consumers, when accurately gathered and controlled by aggregators, can play an increasing role in the existing wholesale electricity market. In this case, it can support the distribution network operator in managing DR at high-level while exploiting the flexibility potential and help consumers to benefit from rewards or lower energy bills. Load prediction at the aggregator level can better support the management of the load flexibility and transmission system operators to balance between the supply and demand of total energy during peak consumption periods, the aggregator can ask the contracted consumers to provide load flexibility.

Good load forecasting models are primordial to better manage load in day-ahead and in intra-day markets. Recently, with the rapid expansion of smart information and communication technologies (ICT) [13], the conventional way of seeing power systems has shifted to smart grids as these smart infrastructure can generate very high volume and high speed rates beyond the capabilities of traditional power systems. In order to take advantage of this massive amount of generated data, the use of deep learning [15], which is one of the most active approaches in many areas of research, seems to be an essential choice in order to increase the prediction performance and better capture the existing volatility in each consumption profile. In addition, since smarts meters are placed on the endpoint of the network, predictions can even be made at the level of each consumer.

To ensure efficient and reliable electricity distribution for each user and improve reliability of short-term maintenance, the short-term prediction is the most appropriate choice, especially for its importance to allow the optimal planning of the distribution of electrical resources, by minimizing financial risks and ensuring optimal control operation in networked microgrids. It also allows to better detect anomalies on electricity usage pattern [8, 99–102].

The objective being to predict the short-term consumption of each of the different users of the network, then to manage the network from an aggregator perspective, we are constrained to several challenges. System operations and planning becomes more difficult when dealing with buildings of different profiles. In fact, residential buildings do not consume the same way commercial buildings or institutional buildings do, thus they do not share the same profiles. These differences create challenges in the process of constructing precise load prediction models for a district containing many buildings. Moreover, compared to the prediction of the load at the level of the individual users, researches at the level of the aggregator remain

insufficient [16,17] despite its progress with the evolution of the smart grid infrastructures [13].

Many studies have been reported in the literature to solve the problem of short-term load prediction especially with the progress in deep learning [19–21, 103]. A combination of two state-of-the-art time series prediction models has been done in [20] in order to create a multi bidirectional recurrent neural network aiming to create accurate load forecasting results for the short-term. For their experiments, they tested their proposed method in two data sets, outperforming support vector regression (SVR) and neural network algorithms. Authors in [21] proposed a neural network approach for short-term residential load forecasting. A long short-term memory (LSTM) based deep learning forecasting framework has been implemented to address volatility problems regarding the consumption history for an individual household. This approach performance has been compared to multiple strategies like simple feed forward neural networks or K-nearest neighbours. In [103], the authors proposed a novel forecasting method for the predicting electric energy consumption called COSMOS, which combines multiple short-term loaf forecasting models using a stacking ensemble approach. They also proved experimentally COSMOS outperforms many recent machine learning approaches. Kim and al. [104] propose a novel multi short-term load forecasting model based on a combination between recurrent neural networks 1-D convolutional neural networks and inception module. To assess the performance of the proposed model, the authors perform day ahead load forecasting for three distribution industrial complexes South Korea. The forecasting performance of the proposed method was better than other methods such as MLP and SVR.

Before modelling any time series problem, it is crucial to select the variables that are most correlated with the load history in order to make reliable predictions, from historical load variables to external variables such as weather related ones like in [25, 27, 28]. For instance, in [25], authors realised experimental research about the effect of adding temperature and relative humidity on the electricity demand in North California, and demonstrated a decrease in prediction errors. As mentioned earlier, with the progress in smart meters infrastructures, it has become interesting to study load forecasting at an aggregation level using data collected from these smart meters as done in [16, 29, 30, 105]. In [29], authors proposed an LSTM based framework on a real residential smart meter data, in order to forecast the load at an aggregator level, first using the predictions at the individuals level then aggregating them, and second using the aggregated data to make predictions directly. They concluded that the first approach is better. The final goal being using the aggregated predictions to forecast the percentage of peak time over a subscribed power in a day-ahead horizon, several studies have been conducted in [27, 32]. In [27], authors have proposed a dynamical method in order to predict the daily peak load value after focusing primarily on the impact of different multi source data, such as the natural and social factors that lead to load variations like temperature

and holidays. Authors incorporated a framework as follows : For each forecast day, they used random walk with restart algorithm to select similar days to build an adequate training set to finally apply SVR method on it to forecast the daily peak. They ended up outperforming other methods that focused on other features/variables selection approaches.

However, to the best of our knowledge, none of the concerned papers have done a study comparing in one hand time series statistical approaches to get the best performance, and in the other hand leverage the use of the district's data and each of its heterogeneous buildings' data sets to perform load and peak estimation.

In this paper, we aim to build a framework which compares various approaches in order to find the most accurate one to make reliable load prediction at an aggregated level, then make use of these predictions to estimate the average peak over a certain subscribed power. We aim to address the gaps in the literature by addressing three main contributions : (1) use three major deep learning algorithms for time series forecasting in order to compare their performances afterwards while using three prediction horizons : 15 minutes ahead, 2 hours ahead and 24 hours ahead, (2) make heterogeneous load prediction at the consumer level then aggregate these predictions on the district level, and compare them with the predictions made directly on the aggregated level (i.e aggregating the forecasts vs forecasting the aggregator) while exploiting data strategies to select the right historical features and (3) use the best prediction model and strategy to determine their accuracy regarding the percentage of peak over a certain subscribed power.

The remainder of this paper is organised as follows. Sections 4.2 and 4.3 review top three approaches for multi-step ahead prediction and give an explicit description of the used time series prediction algorithms. Section 4.4 presents the case study that is analysed in this paper, followed by the analysis of the historical electricity on both the consumers and the district level. An analysis has been conducted in order to show the importance of selecting the right historical variables. Section 4.5 presents the used deep learning architectures as well as a framework that summarises the building of the predictive models. The key results of load forecasting and peak accuracy estimation are presented in section 4.6 which also presents brief details about the experiments' running time. Section 4.8 concludes the paper with a discussion of future work.

4.2 Multi step ahead times series prediction

For multi step ahead prediction tasks, there are different inference methods among which we find three that are used the most [106] :

- The **recursive method** uses one-step ahead prediction models. At each time step $\boldsymbol{t}$, the output prediction can be implemented as the model input for the next prediction at time step $\boldsymbol{t+1}$. This approach works well but experiences the accumulation of error.
- The **direct method** implements independent models for each time step within the forecast horizon, i.e. $\boldsymbol{T}$ models will be created to forecast outputs at time $\boldsymbol{t+1, t+2, .., t+T}$. The problem is that the predictions are constructed in an independent fashion from each other.
- The **MIMO method** is developed due to the requirement to avoid single-output mapping, that disregards stochastic dependencies among future values and therefore alters the accuracy of the predictions. This approach avoids the error accumulation problem and takes into account the dependence of in the time steps.

In this paper, we will be focusing on the MIMO approach because so far, this approach has been effectively applied to varied real-world multi-step ahead time series forecasting tasks [18, 107, 108], and seems more suited to forecasting load.

4.3 Methodology

This section describes the computational tools on which the proposed work is based. In particular, the concept of feed-forward neural network, recurrent neural network (RNN) and some of its variants : Long short-term memory (LSTM), and sequence to sequence (Seq2Seq) based LSTM will be presented.

4.3.1 Feed Forward Neural Networks (FFNN)

Feed Forward Neural Networks, also known as Multi-Layer Perceptrons (MLPs) [15] are neural network based methods used for the universal function approximation. A MLP is composed of an input layer and an output layer, added to that some hidden layer(s) in between. Part of the hyper parameters of the MLP we find the number of hidden layers $\boldsymbol{l = 1, ..., L}$ and their sizes, which determines the depth of the network and its complexity in terms of neurons.

Given a single input vector $\boldsymbol{\{x_1, ..., x_t\}}$, the multi-layer neural network is then used to output the predicted values at the following time steps $\boldsymbol{\{\hat{y}_{t+1}, ..., \hat{y}_{t+T}\}}$. Specifically, for a certain vector $\boldsymbol{x \in R^t}$ fed to the input of the network, the MLP's computation can be expressed as :

$$\boldsymbol{a_l = W_l^T.h_{l-1} + b_l \quad , \quad l \in \{1, ..., L\}} \tag{4.1}$$

$$\boldsymbol{h_l = \phi_l(a_l)} \tag{4.2}$$

where $\boldsymbol{h_0 = x \in R^t}$ and the output $\boldsymbol{\hat{y}_t = h_L \in R^T}$. The layer $\boldsymbol{l}$ is defined with its weight

matrix $\boldsymbol{W_l} \in \boldsymbol{R^{n_{l-1} \times n_l}}$ and bias vector $\boldsymbol{b_l} \in \boldsymbol{R^{n_l}}$. $\boldsymbol{\phi_l}$ can be any type of non-linearity function. In this paper, the MLP model will be considered as our baseline model.

The output layer dimension matches with the horizon of forecast $\boldsymbol{\{t+1, \ldots, t+T\}}$. the input vector's dimension can depend also on the exogenous variables,and not only on the historical variables. More details will be given in section 4.4.3.

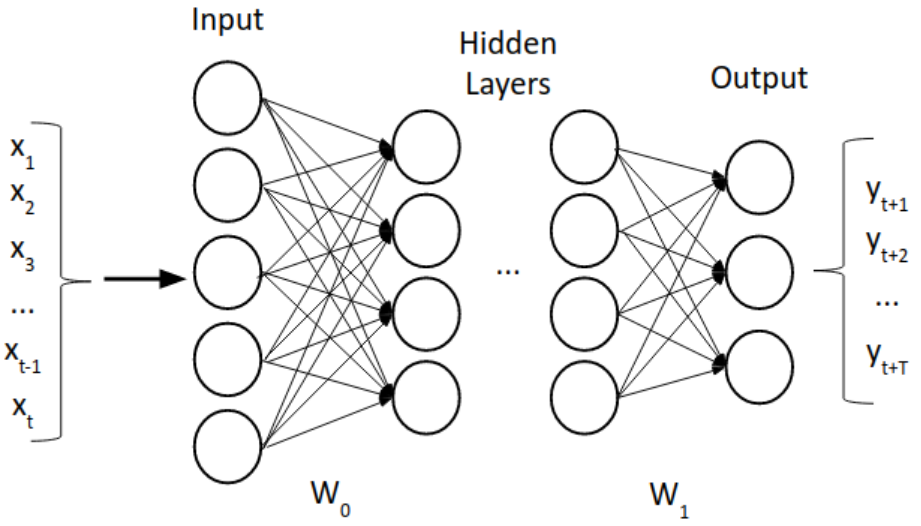

Figure 4.1 Multi layer perceptron

Considering having a training set of composed of input and output vectors $(\boldsymbol{x^i}, \boldsymbol{y^i})$ for $\boldsymbol{i\{\in 1, 2, \ldots, m\}}$, the learning process aims at finding the best possible configuration of the parameters $\boldsymbol{\Theta = (W, b)}$ that will minimise a loss function evaluating the difference between the predicted values $\boldsymbol{\hat{y}^i = f(x^i, \Theta) \in R^T}$ and the ground truth (GT) $\boldsymbol{y^i \in R^T}$.

The mean squared error is a very popular loss function that is used in regression problems and in time series in particular :

$$\boldsymbol{\mathcal{L}(\Theta) = \frac{1}{m} \sum_{i=1}^{m} ||y^i - f(x^i, \Theta)||^2} \tag{4.3}$$

$$\boldsymbol{\hat{\Theta} \in \underset{\Theta}{\operatorname{argmin}}\, \mathcal{L}(\Theta)} \tag{4.4}$$

To minimise this loss function, we will use the Adam optimiser [85], which is one of the most efficient optimisation algorithm designed specifically for training deep neural networks.

4.3.2 Recurrent Neural Networks and Long Short-Term Memory

Traditional feed forward neural networks do not necessarily learn the temporal aspect of the given data because no sequential order is given to the input when fed to the neural network. To overcome such drawback, Recurrent Neural Networks (RNN) were proposed in [15]. They are defined as networks that map the temporal relationship that exists in time series data using feed forward neural networks, in order to capture the temporal component of the problem. In this case, the information relating to the state of the network in a time step is transferred to the next time step. Such feature will permit the persistence of information from an earlier time to learn upcoming new time steps.

RNN Theoretical background

A simple recurrent neural network (SRNN) is also called a vanilla RNN or Elman network.The equations for an SRNN are :

$$\boldsymbol{h_{t+1}} = \boldsymbol{\sigma_h}(\boldsymbol{W_x x_{t+1}} + \boldsymbol{W_h h_t} + \boldsymbol{b_h}) \tag{4.5}$$

$$\boldsymbol{y_{t+1}} = \boldsymbol{\sigma_y}(\boldsymbol{W_y h_{t+1}} + \boldsymbol{b_y}) \tag{4.6}$$

where $\boldsymbol{h_t}$ is the hidden state at time step $\boldsymbol{t}$, computed as a function of the input $\boldsymbol{x_t}$ and the hidden state from the previous time step $\boldsymbol{h_{t-1}}$, using weight parameters $\boldsymbol{W_x, W_h, W_y}$, and bias parameters $\boldsymbol{b_h, b_y}$, hidden-layer activation function $\boldsymbol{\sigma_h}$ (generally $\boldsymbol{\sigma_h = tanh}$) and output activation function $\boldsymbol{\sigma_y}$. $\boldsymbol{y_{t+1}}$ represents the target at time step $\boldsymbol{t+1}$. Figure 4.2 is a schematic representation of a vanilla RNN where 3 time steps are used as input $\boldsymbol{x_{t-1}, x_t, x_{t+1}}$ and 3 time steps are targeted $\boldsymbol{y_{t+2}, y_{t+3}, y_{t+4}}$.

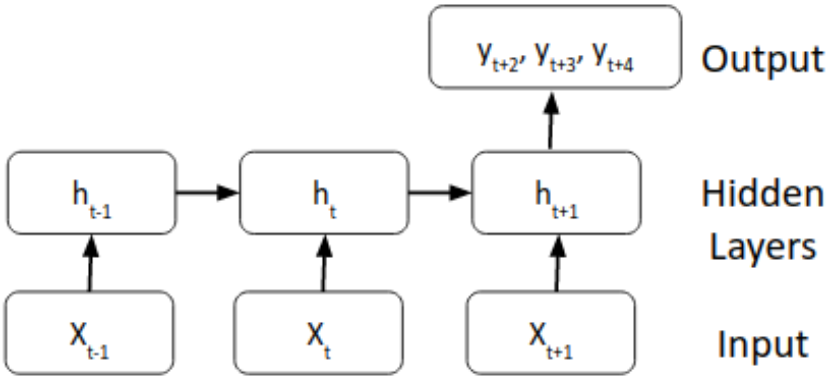

Figure 4.2 Schematic of basic recurrent units

The network shows in figure 4.2 immediately that if t represents the time, then the information is transferred with respect to each time step at a time, thus the temporal aspect is considered. This makes the RNN interesting for studying time series.

Theoretically, with this model, it should be possible to learn any relation between the past and the current time even with long-term dependencies. However, sometimes in practice, RNNs fail to learn these long-term dependencies. The problem, known as the Vanishing Gradient problem [82] is related to some interesting fundamental reasons why it might be difficult to learn from long-term dependencies. In order to overcome this drawback, Long short-term Memory neural network (LSTM) that was introduced by the authors of [83] is a successful approach to this problem. Recently, LSTM is becoming more popular, and it works extremely well on a wide variety of problems from time series forecasting [29] to Image captioning [109].

LSTM

The LSTM networks retain similar structure of the simple recurrent neural network. However the main difference lies in the formation of the inner cell.

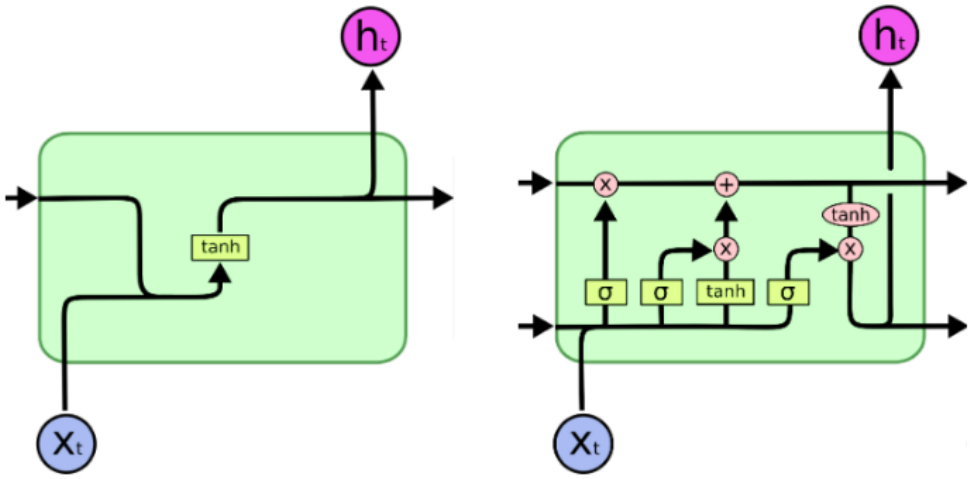

Figure 4.3 Vanilla RNN - LSTM [1]

The novelty in the LSTM architecture is that the inner architecture implements a gate system, which controls the processing of neural information. The aim of gated networks is to control the gradient stream. This feature helps to tackle the vanishing gradient problem. In detail,

the neural calculation includes [1] :

$$\boldsymbol{f_t} = \boldsymbol{\sigma}(\boldsymbol{x_t}.\boldsymbol{U^f} + \boldsymbol{h_{t-1}}.\boldsymbol{W^f}) \tag{4.7}$$

$$\boldsymbol{i_t} = \boldsymbol{\sigma}(\boldsymbol{x_t}.\boldsymbol{U^i} + \boldsymbol{h_{t-1}}.\boldsymbol{W^i}) \tag{4.8}$$

$$\boldsymbol{o_t} = \boldsymbol{\sigma}(\boldsymbol{x_t}.\boldsymbol{U^o} + \boldsymbol{h_{t-1}}.\boldsymbol{W^o}) \tag{4.9}$$

$$\tilde{\boldsymbol{C}}_t = \boldsymbol{tanh}(\boldsymbol{x_t}.\boldsymbol{U^g} + \boldsymbol{h_{t-1}}.\boldsymbol{W^g}) \tag{4.10}$$

$$\boldsymbol{C_t} = \boldsymbol{f_t} \odot \boldsymbol{C_{t-1}} + \boldsymbol{i_t} \odot \tilde{\boldsymbol{C}}_t \tag{4.11}$$

$$\boldsymbol{h_t} = \boldsymbol{tanh}(\boldsymbol{C_t}) \odot \boldsymbol{o_t} \tag{4.12}$$

where $\boldsymbol{W^{i,f,o,g}} \in \boldsymbol{R^{p_H \times p_H}}$ and $\boldsymbol{U^{i,f,o,g}} \in \boldsymbol{R^{p_H \times p}}$ are the weights that the network must adapt and learn, $\odot$ is the Hadamard product. $\boldsymbol{\sigma}$ is the sigmoid function.

The principle of the *gates* $\boldsymbol{i_t}, \boldsymbol{f_t}, \boldsymbol{o_t}$ is that they are used to filter out information from the previous step, information from the current step and then the final information. The purpose is to only keep in memory the important information according to the following explanation :

- $\boldsymbol{f_t}$ (*Forget gate*) : takes a value between zero and one. When it's close to zero, the old state doesn't count for much according to the equation 4.11.
- $\boldsymbol{i_t}$ (*Input gate*) : takes a value between 0 and 1. When it's close to 0, the current state doesn't count for much according to the equation 4.11.
- $\boldsymbol{o_t}$ (*Output gate*) : is used to filter the output of the network according to the equation 4.12.

4.3.3 Seq2Seq with LSTM

Sequence-to-sequence architectures (seq2seq) or encoder-decoder models [84] were originally designed to address the inability of RNNs to produce arbitrary-length output sequences. It was first used in neural machine translation, but it has proved to be the reference in many fields image captioning and is lately well used in time series forecasting [110].

The model consists of 2 parts, encoder and decoder :

- The encoder : a stack of recurrent units (RNN or LSTM cells for better performance) where each of them is given an element of the input. For each element, it collects information and propagates that information forward.
- The decoder : a stack of recurrent units where each predicts an element in the output sequence at a time step t, i.e, there will be $\boldsymbol{T}$ units when the predicted values are for $\boldsymbol{t+1, t+2, ..., t+T}$.

Figure 4.4 is a schematic representation of a Seq2Seq architecture where 3 time steps are used

as input $\boldsymbol{x_{t-1}, x_t, x_{t+1}}$ and 4 time steps are targeted $\boldsymbol{y_{t+2}, y_{t+3}, y_{t+4}, y_{t+5}}$. Seq2Seq models

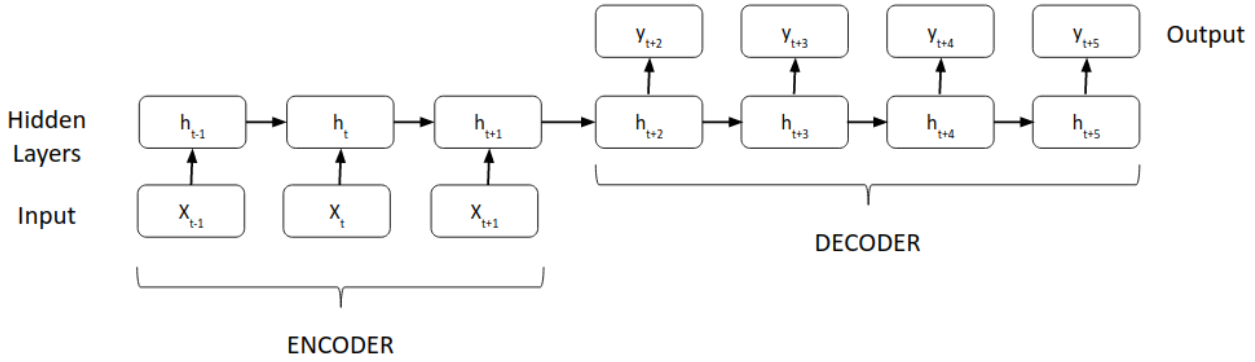

Figure 4.4 Schematic of basic Sequence to Sequence architecture

give this intuition that the temporal aspect is important to be taken, not only for the input, but also for the output. In fact, while its encoder part is similar to the LSTM architecture, the decoder part however brings a novelty in treating the output. In Figure 4.4 $\boldsymbol{y_{t+5}}$ takes the information of the hidden layer at time $\boldsymbol{t+4}$ which takes the information of all the hidden layers that come before temporally.

4.4 Data analysis, processing

4.4.1 Time Series construction

In this study, gathered data from the building energy management system of an urban educational district in Montreal region is used. The district consists of five buildings; it contains residential, commercial, educational and institutional buildings and thus represents an heterogeneous mix of load patterns.

The data includes instantaneous power consumption for the whole district as well as for each building within the district every 15 minutes during 2015 and 2016. Figure 4.5 is a representation of a box plot of the average weekly power consumption, grouped by the 5 buildings. The buildings 1 and 2 refer to an institutional load profile, while building 3 refers to a computing centre. Buildings 4 and 5 contain residential and commercial loads.

It can be observed that various load profiles are present, for instance, the computing center keeps the same trend during weekends and weekdays. Moreover, demands in buildings 1 and 2 have high variations, particularly during weekdays. Significant variability is observed in various buildings which renders the load forecasting a complex task.

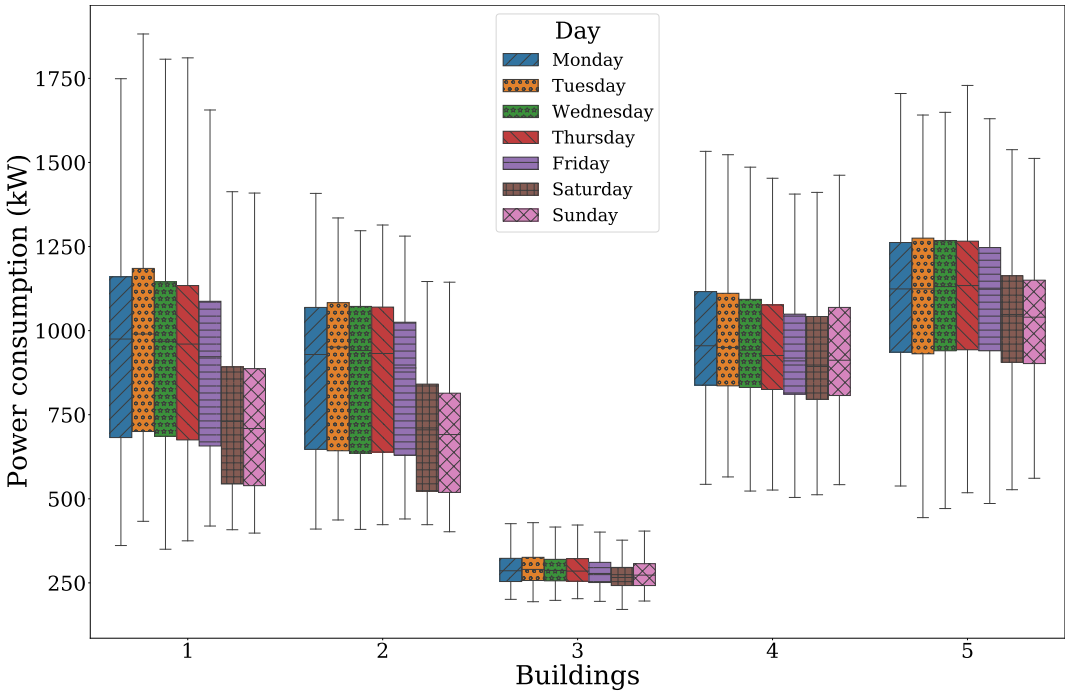

Figure 4.5 Box plot of the weekly average power consumption of the 5 buildings

Since electricity consumption behaviour depends upon several factors such as the variations in temperatures, seasons, relative humidity, days of the week, special holidays, we decided to add external features related to calendar data (important events and holidays, weekend or weekday), meteorological data such as dry-bulb temperature, relative humidity, dew temperature. Adding these external features in the equations have shown great results and improved performances in several load forecasting studies [25].

4.4.2 Data normalisation

The power consumption as well as many variables including the Temperature, not only in the whole district but also in each building, are very stochastic and follow a various value ranges and distributions, thus performing a data normalization strategy becomes necessary before constructing a load forecasting model. It is also useful to normalize the data to fasten the learning process and to make the optimization algorithm converge quickly. The normalization is conducted using the *Mean Scale Transformation* proposed in [111] which can be expressed according to Eq. (4.13) :

$$\boldsymbol{X_{new}} = \frac{\boldsymbol{X}}{\boldsymbol{1 + Mean(X)}} \tag{4.13}$$

4.4.3 Historical inputs

In terms of time series modelling, it is crucial to select the features that will affect the prediction of electricity consumption in order to have the best possible performances. The data includes two types of variables :

- Historical variables : contains the history of the load until the present time.
- External variables : calendar data, temperature, humidity, dew point temperature.

Selecting these historical variables often requires a good knowledge of the data and the domain of the issue we are attempting to solve. That is the reason why in many papers, historical variables are selected using the help of human expertise (HE) [29, 105] where experts use their prior knowledge to select specific lags of power history thanks to their experience in the domain. Another way to select historical variables is to perform technical analysis as done in [18, 22, 112].

In this study, we compare the performance using the human expertise with the performance using a strategy called the *Autocorrelation Analysis*.

Human expertise

Predicting the load consumption at **1** hour ahead might require knowing the load consumption for the last 4 hours, and the load consumption a week before. This incertitude in the selection can consequently influence the model performance. For this part, and as many research papers did from section 4.1, we will implement the load consumption for the last 2 hours and a week before in order to predict different horizons ahead.

Autocorrelation function

In this case, we propose another historical variable selection using the *Autocorrelation Function* (ACF), which specifies the correlation between the observation at the current time spot and the observations at previous time spots. Knowing that each time step corresponds to 15 minutes, the goal here will be to select the past time steps that are most correlated with time step $\boldsymbol{t}$. Figure 4.6 is a representation of the relation between time step t and 4 different previous time steps. The points cluster along a corner to corner line from the bottom-left to the top-right of the plot. This remark demonstrates a positive correlation. Moreover, a strong relationship is implied when the points are tighter into the diagonal line. The point cloud becomes sparser as the correlation value decreases when the lags get larger.

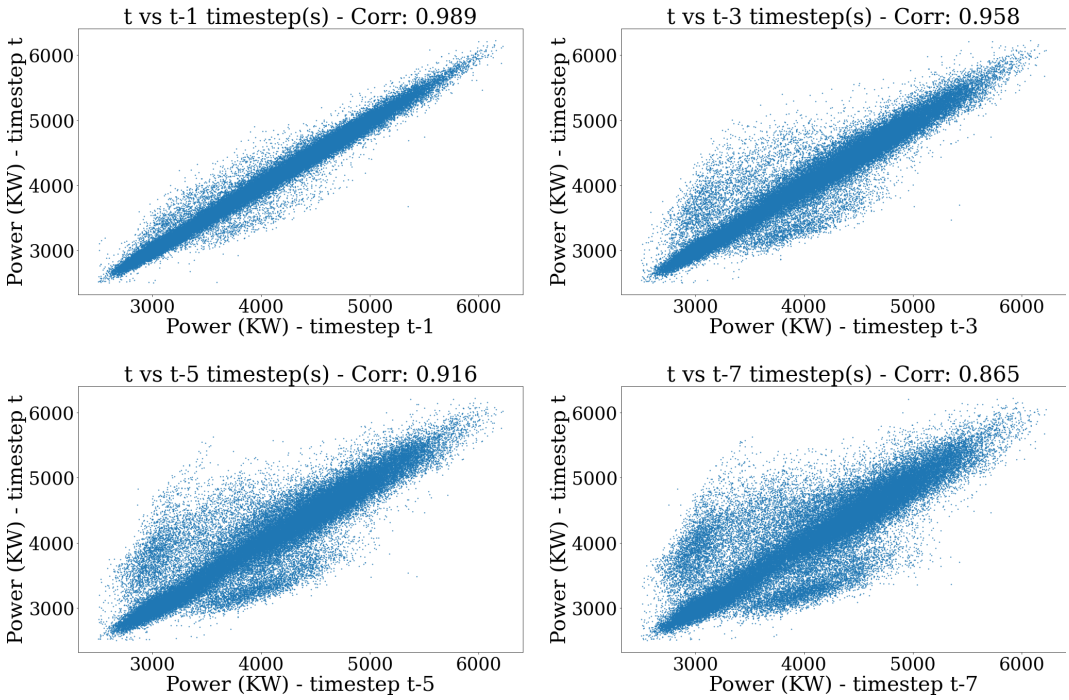

Figure 4.6 Lag plots between time step t and 4 different previous time steps

For more generalisation, an *autocorrelation plot* is made to help find the most correlated time steps with time step $\boldsymbol{t}$. Figure 4.7 represents this autocorrelation plot using the absolute values of these correlations. For the *t vs t-7 days* pointed in red, there is particularly a stronger relationship than most of the other time steps (lags), which indicates that adding the power consumption of 7 days before in our historical inputs might be good strategy.

The 15 best historical variables are selected based on their correlation with time $\boldsymbol{t+1}$. In case the district data is used, here is a list of some of these features : **t, t-15 min, t-7 days, t-8 days**.

4.5 Architectures and Framework

4.5.1 Splitting the data

In any machine learning project, it is important to mention that there are two main steps : Training and Evaluating.

We split our data to a training, a validation and a testing sets with the 60%/20%/20% rule as follows : training set contains data from **08/01/2015** to **08/03/2016**, validation set contains data from **08/03/2016** to **08/08/2016** and test set contains data from **08/08/2016** to **24/12/2016**. The validation set is used to evaluate the model trained on training data and

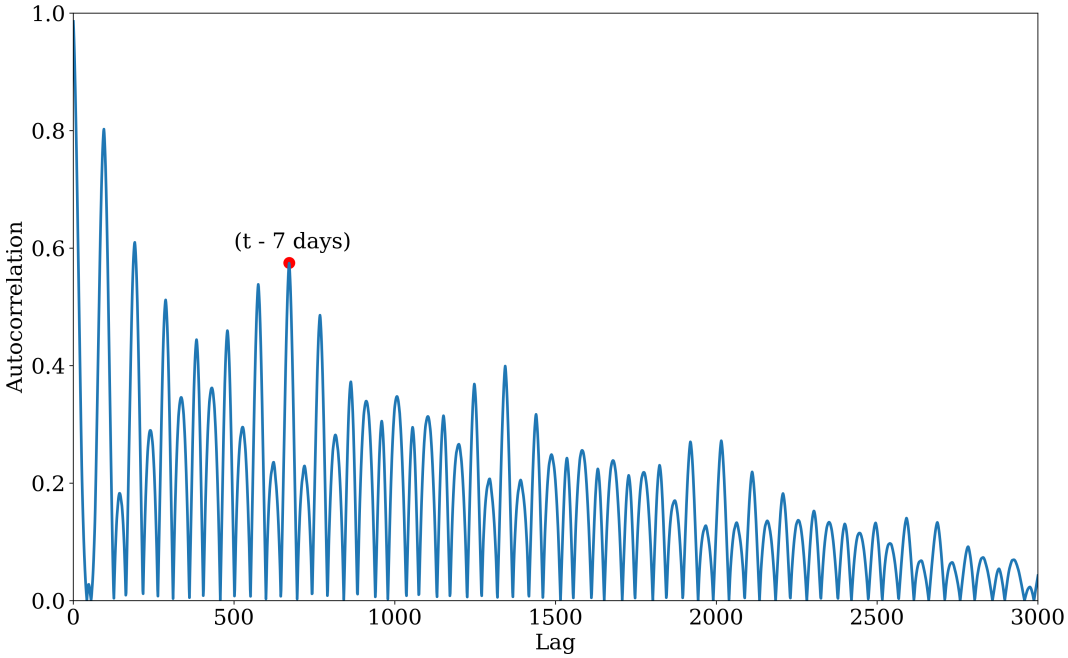

Figure 4.7 Autocorrelation plot : auto correlations for data values at varying time lags

help us find which hyper parameters should give a good performance on the test set.

4.5.2 Model regularisation

Techniques of machine learning sometimes suffer from over-fitting, that results in fitting the training data perfectly which in turns performs badly on the validation and test sets. This study employs a regularisation method called *Early Stopping* [86], which is a generalisation method that quits the training process by controlling the validation loss, before reaching the maximum number of iterations. In such situation, the training process terminates after a specified number of iterations when the error result in the validation set stops decreasing.

4.5.3 Framework

Consider the following framework : We apply this framework on the data set collected from the district. We also apply it on the data set collected from each of the five buildings included in the district. The aim is to compare the predictions performed on the whole district with the sum of the predictions performed on each of the district's building. To do so, we will use the *Mean Absolute Percentage Error* (MAPE) metric :

$$\boldsymbol{MAPE} = \frac{\mathbf{1}}{\boldsymbol{m}} \sum_{t=1}^{m} \left| \frac{\boldsymbol{G_t} - \boldsymbol{P_t}}{\boldsymbol{G_t}} \right| \tag{4.14}$$

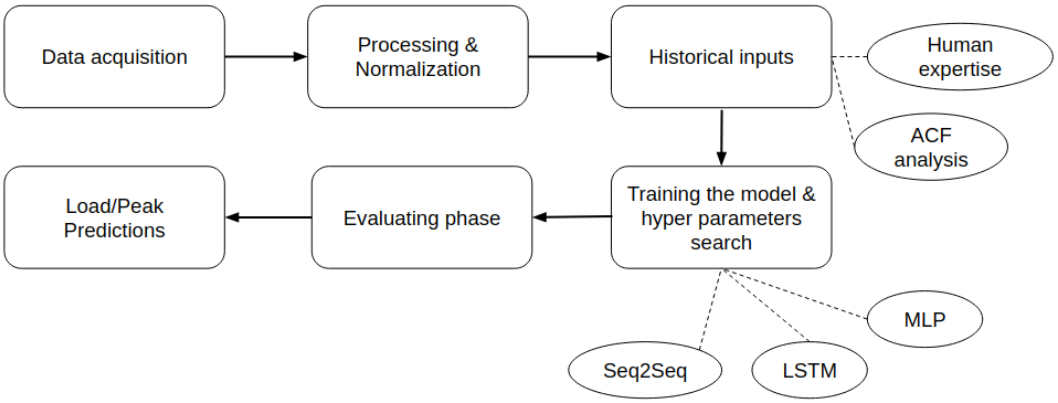

Figure 4.8 Framework from data acquisition to predictions

where $\boldsymbol{G_t}$ corresponds to the ground truth, and $\boldsymbol{P_t}$ corresponds to the predicted value.

Since we are interested in the performance of the models on short-term horizon, we will use the proposed models to predict the power consumption 15 minutes, 2 hours and 24 hours ahead. These predictions will be useful to have an insight to measure how well they are doing regarding estimating the monthly percentage of peak over a subscribed power, in this case 5000 KW.

4.5.4 Selected architectures

In order to find the right hyper parameters for each of the 3 models (MLP, LSTM, Seq2Seq), a random search, which is an iterative approach, is performed as follows :

- Choose randomly a combination a hyper parameters (e.g. number of layers, number of neurons, activation functions) for the model.
- Train the model on the training set.
- Evaluate the model on the validation set, and save its performance.

The selected combination of hyper parameters will be the one which gives the best performance on the validation set.

Each model was trained for 100 epochs, using early stopping. After performing hyper parameter search specifically on the number neurons as well as hidden layers and the used activation functions, we concluded that the best architectures to use are an MLP with 3 hidden layers (*layer 1* : 64 neurons, *layer 2* : 64 neurons, *layer 3* : 32 neurons), then an LSTM with 3 hidden layers (*layer 1* :64 neurons, *layer 2* : 128 neurons, *layer 3* : 32 neurons) and finally a Seq2Seq with 3 hidden layers for the encoder (*layer 1* : 64 neurons, *layer 2* : 64 neurons, *layer 3* : 32 neurons, and 1 hidden layer for the decoder (*layer 4* : 32 neurons).

4.6 Results and discussions

4.6.1 Execution time

The whole process from data acquisition to building and training/testing the models was run on a computer with an Intel Core i7-6500U, processor running at 2.5 GHz using 16 GB of RAM with NVIDIA GEFORCE 940M, running on Ubuntu 18.04.

Due to the complexity of the models regarding their number of layers and neurons, the MLP spent 4h10min in order to train with all combinations of strategies for every horizon while the LSTM spent 25h50min and the Seq2Seq spent 42h55min. Also, the number of epochs for which each model was executed is not always the same du to early stopping. Regarding the testing phase, the execution time is nearly the same for the three models.

4.6.2 Power consumption forecasting

Tables (4.1)-(4.2)-(4.3) present the performance (MAPE) obtained with the proposed models. For each model and for each prediction horizon, the best performance will be in bold with a green colour in the background.

Tableau 4.1 Performance for 15 minutes ahead load prediction

	District data set		Buildings data set	
	HE	ACF	HE	ACF
MLP	2.093 %	2.005 %	2.226 %	**1.873 %**
LSTM	2.087 %	2.127 %	2.002 %	**1.926 %**
Seq2Seq	1.876 %	2.001 %	2.054 %	**1.832 %**

Table 4.1 suggests that the three algorithms have similar performance, whether used for district data set or each building data set, using the human expertise or the ACF analysis. Even though the performance of the MLP is the worst, we are expecting to have larger differences when predicting the power consumption 2 hours or 24 hours ahead. This remark is attributed to the fact that predicting the power consumption 15 minutes ahead requires a single output, while predicting 24 hours ahead requires 96 outputs thus taking the temporal aspect into account will be more important in the 24 hours case. The MLP do not take necessarily that temporal aspect into account. Also, the best performance is for the Seq2Seq algorithm used with the buildings data set and the ACF analysis for the historical inputs.

Tableau 4.2 Performance for 2 hours ahead load prediction

	District data set		Buildings data set	
	HE	ACF	HE	ACF
MLP	5.126 %	3.694 %	5.127 %	**3.322 %**
LSTM	3.283 %	3.298 %	3.304 %	**2.863 %**
Seq2Seq	**3.025 %**	3.618 %	3.115 %	3.197 %

The table 4.2 depicts that the performance of the MLP is the worst and is a bit more far from the performances of the LSTM and Seq2Seq algorithms that share nearly the same MAPEs. The best performance goes to the LSTM using the buildings data set and the ACF analysis for the historical inputs.

Tableau 4.3 Performance for 24 hours ahead load prediction

	District data set		Buildings data set	
	HE	ACF	HE	ACF
MLP	10.288 %	**10.077 %**	10.884	10.380 %
LSTM	5.553 %	5.973 %	5.423 %	**5.181 %**
Seq2Seq	5.908 %	5.845 %	**5.210 %**	5.551 %

It can be seen in table 4.3 that the MLP's MAPE is increasing quickly compared with the other two algorithms. We understand now that taking into account the temporal aspect of the problem plays an important role in building precise models. The LSTM and Seq2Seq algorithms have, again, nearly the same performance. The best performance goes to the LSTM using the buildings data set and the ACF analysis for the historical inputs.

To conclude, the best model and strategy on average are :

- LSTM as the prediction model.
- Predict on each of the buildings, then aggregate.
- Select historical data using ACF for each building.

From now on, we are going to make use of only the selected best model and strategy combination.

Figure 4.9 represents a plot of the prediction versus the ground truth between Monday Aug 8th and Sunday Aug 16th 2016 using the selected model and strategy for the three prediction

horizons. As expected, we get less precision using more steps to predict in the future.

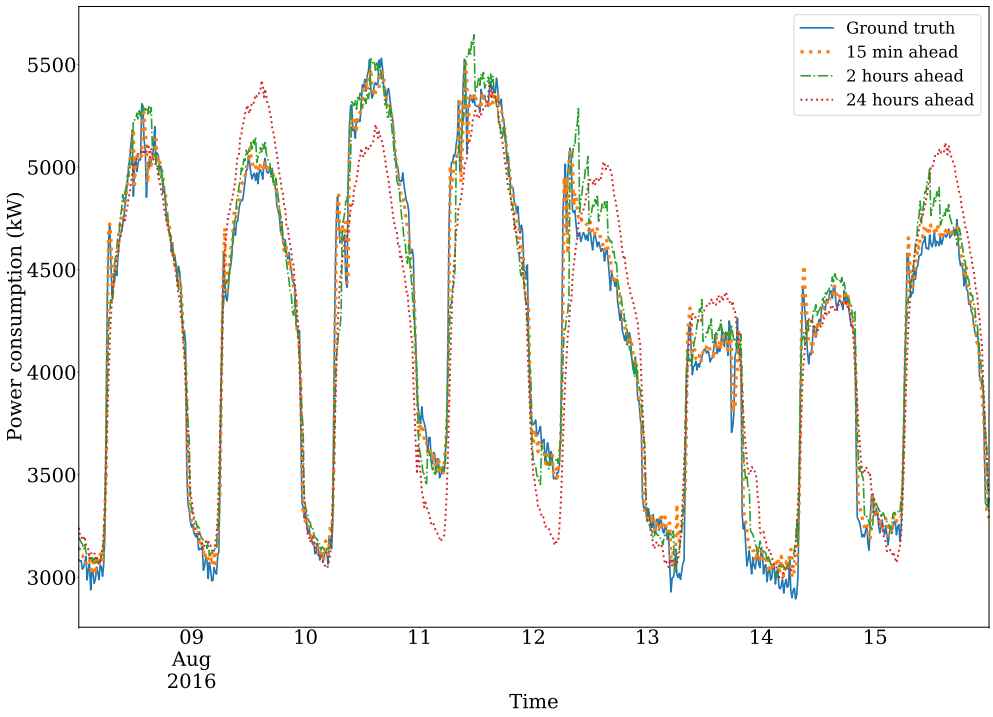

Figure 4.9 Prediction vs Ground truth using LSTM-Buildings data sets-ACF strategy for each of the 15min/2h/24h prediction horizons

The advantage of using the buildings' data set is that since we are training separated models for each of the five buildings, the ACF permits us to select the adequate historical inputs for each of the five buildings as it captures the internal behaviour of each one, which adds a flexibility to our models. Moreover, we can measure the performance (MAPE) of each of these five models, thus give an interpretation to what affects most the performance on the sum of the predictions.

This study confirmed that the models made to predict the power consumption of the buildings 1, 2 and 4 had the worst MAPE performance as in table 4.4. That is normal because institutional and residential buildings know more stochasticity in their power consumption than computing centres due to their dependence on many parameters : the occupancy, the behaviour and habits of each person.

Tableau 4.4 Performance for 24 hours ahead prediction using LSTM - ACF analysis

Building 1	Building 2	Building 3	Building 4	Building 5
9.838 %	8.096 %	5.664 %	9.095 %	7.580 %

Using the selected best model and strategy combination and for each prediction horizon, we perform load forecasting during 2015-2016 obtaining the following box plot representation in Fig.4.10, for which the MAPE(%) was calculated daily then aggregated monthly.

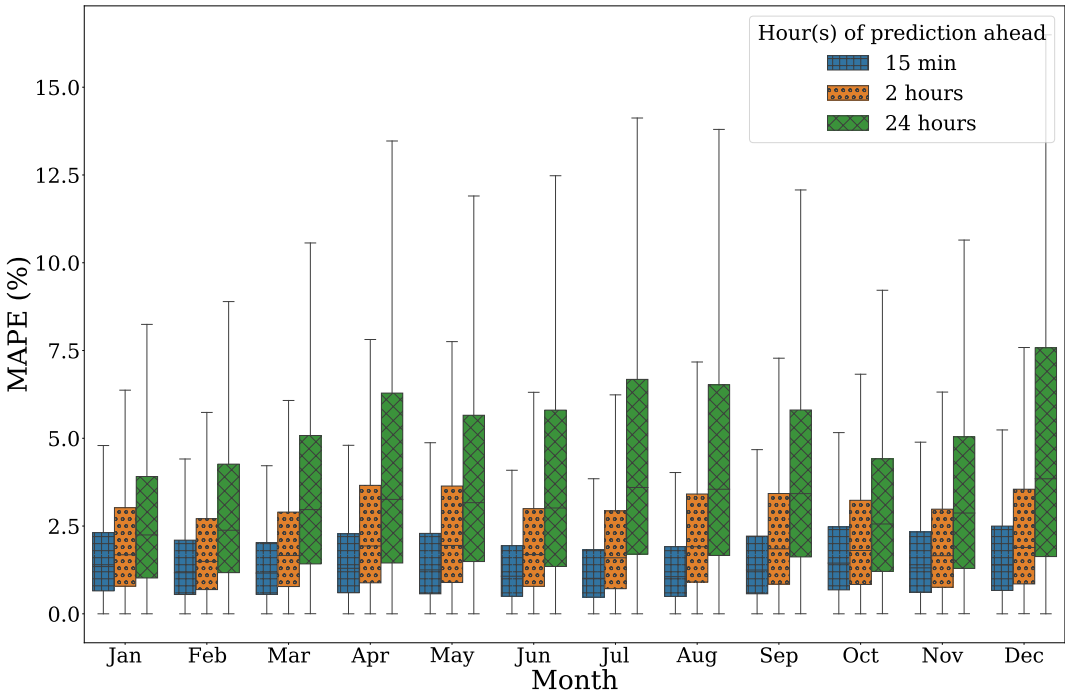

Figure 4.10 MAPE distribution monthly for each of the 3 prediction horizons

On average, the results are reliable especially for the 15 minutes and 2 hours prediction horizons for which the variability of the MAPE is small compared to the 24 hours one. We can see that December suffers the most from a bad model performance regardless of the prediction horizon, and as we can notice, its variability is overall the higher one. That can be explained by having very big events in this month especially at its end like Christmas, which affects directly the power consumption during this month.

4.6.3 Estimation of monthly percentage of peak over a subscribed power

In addition to be performing well, this prediction model can also be used in order to estimate and extract useful information about the peak load. We decide to see how accurate it can be if the task was to predict the percentage of peak times over the subscribed power. For this, we used the forecasts made by the selected model and strategy and calculated the average number of times the prediction values were greater than the subscribed power. And finally we compare it to the same amount related to the ground truth in Fig.4.11.

We see that for the three forecasting horizons, overall results of the percentage of peaks over the subscribed power are similar to the ground truth results. Since the model's performance are accurate as seeing in the section.4.6.2, thus times when power predictions cross the subscribed power are nearly similar to times when power ground truth crosses it, we can conclude that these results are reliable in most cases. However, for the 24 hours ahead prediction they are more accurate for months where the necessity of cooling is important than months where heating is important.

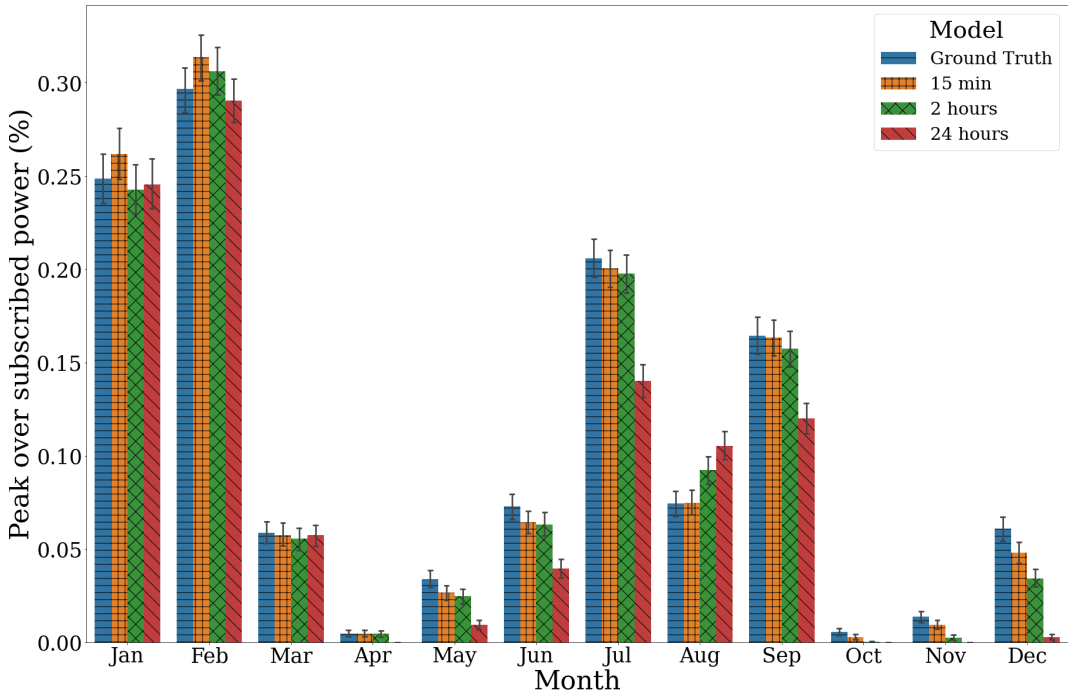

Figure 4.11 Monthly percentage of peaks over 5000 KW - ground truth vs predictions

4.7 Limitations

One of the main challenges of deep learning algorithms is the computational capacities it requires as mentioned in 4.6.1. In fact, since each training depends significantly on the data set, the selected historical features and the different hyper parameters to use, considerable range of combinations should be tested to compare their performances. It is for this reason that the hyper parameters search (number of layers, number of neurons and activation functions) seen in section.4.5.4 is limited and not all the combinations could be tested in this research.

Another limitation concerns the low performance obtained from the 24 hours predictions in section 4.6.3 during Summer. This limitation could be addressed by adding an adaption weight during training the model for summer period.

Furthermore, despite the good performance of our final model, we strongly believe that the data would be much more usable to improve the model in question if we had access to consumption at the consumer level instead of the building level. Indeed, they will serve us to better capture the various behaviors at a lower level and thus make the prediction model more robust to abrupt changes in a consumer's consumption.

4.8 Conclusion and Future work

This work compares the accuracy of various machine learning approaches for aggregated multi-step ahead predictions regarding the short-term electrical load forecasting using heterogeneous data. These forecasts were used to determine their reliability regarding the percentage of peak over a certain subscribed power. Our approaches to time series prediction depend on whether the whole district data set or each of the district's buildings are used, then on historical features extracted from the time series data, and finally on the used deep learning algorithms. Recurrent models proved to produce the best results outperforming the multi-layer perceptron especially when forecasting many time steps ahead. With this particular case where we have heterogeneous buildings, which makes the problem of predicting the aggregated power load more difficult, the best approach is to use the LSTM algorithm on each of the buildings data to predict their load, then sum over these loads. We also concluded that the strategy that works best to select historical variables is to apply the ACF and find the most correlated lags.

Future work should explore demand response in order to assist consumers on their power planning strategies whether it is price based or incentive based demand response. We will also look into reinforcement learning strategies to perform peak shaving on the whole district level depending on each of the buildings behaviour.

Acknowledgement

The authors would like to thank the ÉTS Campus in Montreal for providing the data used for this work. This research was supported by the NSERC-Hydro-Québec-Schneider Electric Industrial Research Chair on Optimization for Smart grids.

CHAPITRE 5 GESTION DU CONFORT ET DE LA PUISSANCE

5.1 Cas d'étude 1 : Contrôle de la température et de la puissance avec les prédictions

Cette étude propose une gestion de l'énergie basée sur le RL dans un groupe de maisons. L'objectif est d'atteindre une réduction maximale de la charge de pointe, tout en contrôlant des charges contrôlées par thermostat et en respectant les limites de température choisies par l'occupant. Des agents RL utiliseront l'état de chaque maison, qui comprend l'historique de la consommation électrique, la température intérieure, la température extérieure et l'humidité afin de trouver la bonne manière de contrôler la puissance. Le but sera de quantifier l'impact de l'ajout des prédictions de la charge électrique comme information à l'agent sur la réduction de la puissance consommée.

Deux scénarios seront considérés :

- **Scénario 1** : consiste à créer un agent capable de contrôler seulement les charges contrôlées par thermostat pour maintenir la température intérieure à une consigne et donc assurer le confort du consommateur.
- **Scénario 2** : consiste à créer un agent capable de contrôler la température en même temps que la puissance consommée. Les prévisions de la charge à consommer par chaque utilisateur seront utilisées afin de pouvoir contrôler cette charge à l'avance.

5.1.1 Modélisation de l'environnement RL

Nous modélisons une maison comme étant une pièce contenant une seule charge contrôlée par thermostat (TCL). Au travers cette étude, nous concevrons un agent qui devra contrôler une à plusieurs maisons indépendantes et identiques.

Vecteur d'état

Nous décidons de munir l'état où se trouvera chaque agent avec les éléments suivants : la température à l'intérieur de chaque maison, la température externe, la puissance agrégée de ces TCLs (systèmes de chauffage, de ventilation et de climatisation), la puissance totale consommée. Pour le scénario 2, une information liée aux prédictions de la puissance totale sera ajoutée au vecteur d'état. Comme indiqué dans [113], nous faisons également l'hypothèse que la consommation des TCLs représente environ 70% de la puissance totale consommée. Les 30% restants (charges non contrôlables) sont calculés en fonction du comportement des

occupants et représentent les autres appareils ménagers. Un vecteur d'état $\boldsymbol{S}$ sera donc défini de la manière suivante :

$$\boldsymbol{S} = [\boldsymbol{T_{TCL_1}}, \ldots, \boldsymbol{T_{TCL_n}}, \boldsymbol{T_{ext}}, \boldsymbol{P_{TCL}}, \boldsymbol{P_{total}}, \boldsymbol{I_{PeakPred}}] \tag{5.1}$$

où $\boldsymbol{T_{TCL_i}}$ représente la température de la maison $\boldsymbol{i}$, $\boldsymbol{n}$ étant le nombre de maisons à contrôler, $\boldsymbol{T_{ext}}$ représente la température extérieure, $\boldsymbol{P_{TCL}}$ la puissance agrégée des TCLs, $\boldsymbol{P_{total}}$ la puissance totale. $\boldsymbol{I_{PeakPred}}$ représente l'information concernant les prédictions et le pic de puissance, et sera employée seulement dans le scénario 2 ; elle sera expliquée plus en détail par la suite.

De plus, pour introduire une hétérogénéité de consommation entre les maisons, nous définissons deux profils de consommation électrique dans lesquels les pics de puissance se produisent le matin et le soir, mais sont légèrement décalés.

Récompense de l'agent

Quand un agent est dans l'état $\boldsymbol{s_t}$, il doit prendre une action continue dans $[\mathbf{-1}, \mathbf{1}]$, qui représente le pourcentage de la puissance maximale du TCL à utiliser. Les valeurs négatives correspondent aux actions de climatisation et les valeurs positives aux actions de chauffage. Ensuite, l'agent se voit attribuer un état $\boldsymbol{s_{t+1}}$, cet état est le résultat de l'équation du modèle thermique RC [114] qui est un modèle simple de premier ordre qui modélise l'évolution de la température à l'intérieur de la pièce. Une récompense $\boldsymbol{r_{t+1}}$ également attribuée à l'agent, est définie selon le scénario concerné. En effet, si nous sommes dans le scénario 1, alors la récompense sera exprimée comme dans [67] avec $\boldsymbol{R_1} = \boldsymbol{\Sigma_{i=1}^{n} R_i}$ où $\boldsymbol{R_i}$ s'exprime comme suit :

$$\boldsymbol{R_i} = \begin{cases} \boldsymbol{e^{-(T_{TCL_i} - T_{target})^2} - 0.1(T_{min} - T_{TCL_i})} \text{ Si } \boldsymbol{T_{TCL_i} < T_{min}} \\ \boldsymbol{e^{-(T_{TCL_i} - T_{target})^2} - 0.1(T_{TCL_i} - T_{max})} \text{ Si } \boldsymbol{T_{TCL_i} > T_{max}} \\ \boldsymbol{e^{-(T_{TCL_i} - T_{target})^2}} \quad \text{Sinon} \end{cases} \tag{5.2}$$

où $\boldsymbol{T_{target}}$ est la température ambiante souhaitée, $\boldsymbol{T_{min}}$ et $\boldsymbol{T_{max}}$ sont les limites de l'intervalle de tolérance de température que le consommateur est supposé fixer. Les résultats de ce premier scénario seront utilisés comme référence pour mesurer les performances de la réduction de la consommation d'énergie dans le deuxième scénario.

Dans le scénario 2, trois parties seront ajoutées :

- Un modèle de prédiction, en l'occurrence le LSTM trouvé dans le chapitre 4, devra être capable de prédire la puissance consommée agrégée parmi les maisons traitées, et

cela durant 2 heures à l'avance.

- La variable $\boldsymbol{I_{PeakPred} = P_{peak} - \max(\text{Predictions})}$ sera ajoutée au vecteur d'état. $\boldsymbol{P_{peak}}$ désigne un seuil souscrit de pic de puissance, $\boldsymbol{\max(\text{Predictions})}$ représente la valeur maximale des prédictions de la consommation totale de puissance pendant les deux heures à venir. $\boldsymbol{I_{PeakPred}}$ est positif lorsque le pic est dépassé, et négatif dans le cas contraire.
- À la récompense reçue par l'agent sera ajouté une récompense liée à la puissance totale consommée afin de réduire cette dernière au maximum, comme exprimée dans [67] :

$$\boldsymbol{R_2} = \begin{cases} \boldsymbol{R_1 - \lambda.\nu P_{total}} & \text{Si } \boldsymbol{P_{total} < P_{peak}} \\ \boldsymbol{R_1 - \nu P_{total}} & \text{Si } \boldsymbol{P_{total} \geq P_{peak}} \end{cases} \tag{5.3}$$

où $\boldsymbol{\nu = 5.10^{-6}}$ est un coefficient permettant d'échelonner la pénalité de puissance avec la récompense de température, et $\boldsymbol{\lambda = 0.7}$ un paramètre permettant de diminuer la pénalité lorsque $\boldsymbol{P_{total}}$ ne dépasse pas $\boldsymbol{P_{peak}}$. Pendant les périodes de pointe, la pénalité de puissance augmente pour intensifier l'effort de l'agent sur la réduction de la consommation de la charge consommée.

5.1.2 Application

La gestion énergétique proposée, basée sur le RL, est appliquée à un groupe de maisons. Des agents sont formés pour maintenir la température aussi près que possible de $\boldsymbol{21°C \pm 1°C}$ dans une, deux, quatre et six maisons sur la base des scénarios 1 et 2. L'implémentation se fera à l'aide de l'algorithme TRPO disponible sur stable-baselines [97]. Ses paramètres sont définis par défaut. L'Actor et le Critic seront des réseaux de neurones à deux couches à 64 neurones chacune. L'entraînement se basera sur les données météorologiques de l'année 2015 à Montréal, et les tests sont effectués en utilisant les données météorologiques d'avril à novembre 2016, pour inclure les périodes d'été et d'hiver. Notez que la période de décembre à mars est délibérément exclue des tests car le dimensionnement du système de chauffage combiné aux températures extérieures extrêmes de cette période oblige l'agent à utiliser toute la puissance de chauffage pour maintenir le confort de l'utilisateur. Enfin, nous considérons les périodes de pointe liées à $\boldsymbol{P_{peak}}$ comme étant les périodes pendant lesquelles la puissance utilisée est supérieure aux top 10% de la consommation de base.

Afin de construire le modèle de prévision de la consommation de la charge, nous générons d'abord le profil de consommation agrégé pour le scénario 1. Après avoir divisé l'ensemble des données et formé les quatre modèles LSTM pour prédire la consommation d'énergie agrégée

pour 2 heures à l'avance, nous évaluons la performance du modèle en utilisant le MAPE où une moyenne de 5,47% pour les quatre modèles est obtenue.

5.1.3 Résultats

Après avoir entraîné l'agent de manière épisodique en utilisant les données météorologiques de 2015 sur chaque scénario, nous l'avons testé sur la période d'avril à novembre 2016.

Pour une raison de simplicité et de représentation, nous allons représenter seulement le résultat de contrôle de 4 maisons. La Figure 5.1 représente une comparaison de la température ambiante dans chacune des 4 maisons suivant chaque scénario pendant la première semaine d'avril 2016 :

Nous remarquons que la température des 4 maisons dans le scénario 1 est parfaitement conservée autour de $\mathbf{21°}\boldsymbol{C}$ ce qui montre la capacité de l'agent de bien maintenir le confort de chaque utilisateur. Dans le scénario 2, l'agent arrive à obtenir de bons résultats, bien que la température des maisons sorte parfois du domaine qui délimite $\mathbf{20°}\boldsymbol{C}$ et $\mathbf{22°}\boldsymbol{C}$, ce qui montre que le contrôle est moins bien que dans le premier scénario. Cela peut être expliqué par la tâche plus complexe que l'agent doit réaliser dans ce scénario, en contrôlant la température et la puissance consommée.

Pour une vision plus globale, le tableau 5.1 représente la moyenne et l'écart type de la température moyenne sur les maisons gérées. On voit que les quatre agents parviennent à bien contrôler la température dans les deux scénarios, et que l'écart type de la température est plus grand dans le deuxième scénario plutôt que le premier. Cela explique et généralise de manière plus globale la conclusion faite à l'égard de la Figure 5.1.

Tableau 5.1 Température moyenne dans les maisons en avril-novembre pour chaque scénario

	1 maison	2 maisons	4 maisons	6 maisons
Scénario 1				
Température moyenne (C)	21.36	21.15	21.23	21.24
Écart type (C)	0.37	0.33	0.60	0.8
Scénario 2				
Température moyenne (C)	21.03	20.66	21.14	20.94
Écart type (C)	1.24	1.2	0.82	1.0

Nous désirons maintenant voir l'effet de l'ajout des prédictions et le contrôle de la puissance sur la puissance totale consommée. Pour cela, nous mettons en place le tableau 5.2 qui compare les deux scénarios pendant avril-novembre, et cela en termes de puissance moyenne consommée, énergie totale, périodes de pointe, et l'énergie consommée pendant ces périodes. La période de pointe est définie par le moment où la puissance consommée est supérieure

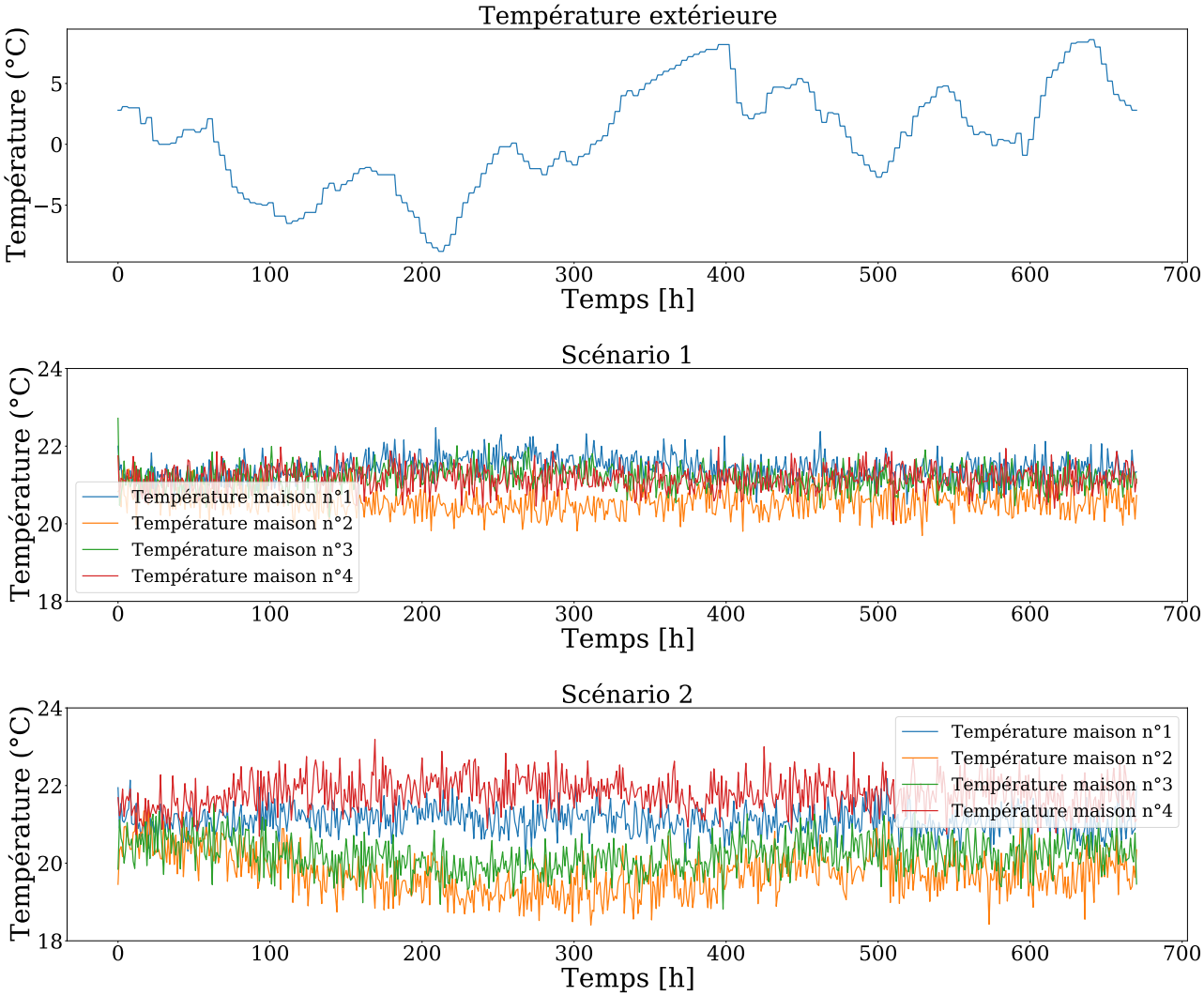

Figure 5.1 Comparaison de la température du scénario 1 et 2 pendant une semaine d'avril

à une consigne. Cette consigne est extraite du profil de puissance agrégé généré à l'aide du scénario 1 et qui définit le seuil de top 10% de la consommation.

Comme prévu, il est clair que l'ajout des prédictions et le contrôle de la puissance permet une baisse de sa consommation. De plus, quand il est question de gérer une seule maison, nous pouvons réduire cette charge à hauteur de 73% pendant les périodes de pointe, qui sont moins fréquentes dans le deuxième scénario. Cependant, l'augmentation du nombre de maisons cause une diminution de l'impact de la gestion en terme de puissance réduite. Cela peut montrer les limites des performances d'un simple agent lorsqu'il est question de contrôler plusieurs instances.

Tableau 5.2 Puissance consommée en avril-novembre pour chaque scénario

	Puissance moyenne consommée (kW)	Énergie totale (kWh)	Périodes de pointe (min)	Énergie consommée en période de pointe (kWh)
1 maison				
Sc. 1	5.463	319922	15225	2788
Sc. 2	5.271 (-3.5%)	308701 (-3.5%)	4275 (-72%)	746 (-73%)
2 maisons				
Sc. 1	10.841	634836	15030	5359
Sc. 2	10.454 (-3.6%)	612181 (-3.6%)	6045 (-60%)	2104 (-61%)
4 maisons				
Sc. 1	21840	1278944	15330	10982
Sc. 2	21677 (-0.7%)	1269360 (-0.7%)	13500 (-12%)	9607 (-13%)
6 maisons				
Sc. 1	32797	1920549	15480	16569
Sc. 2	32425 (-1.1%)	1898727 (-1.1%)	11340 (-27%)	12003 (-28%)

Pour plus de visibilité, représentons également dans la Figure 5.2 une comparaison entre la puissance totale consommée pendant la même semaine d'avril pour chacun des deux scénarios, et cela lorsque l'agent contrôle une seule maison.

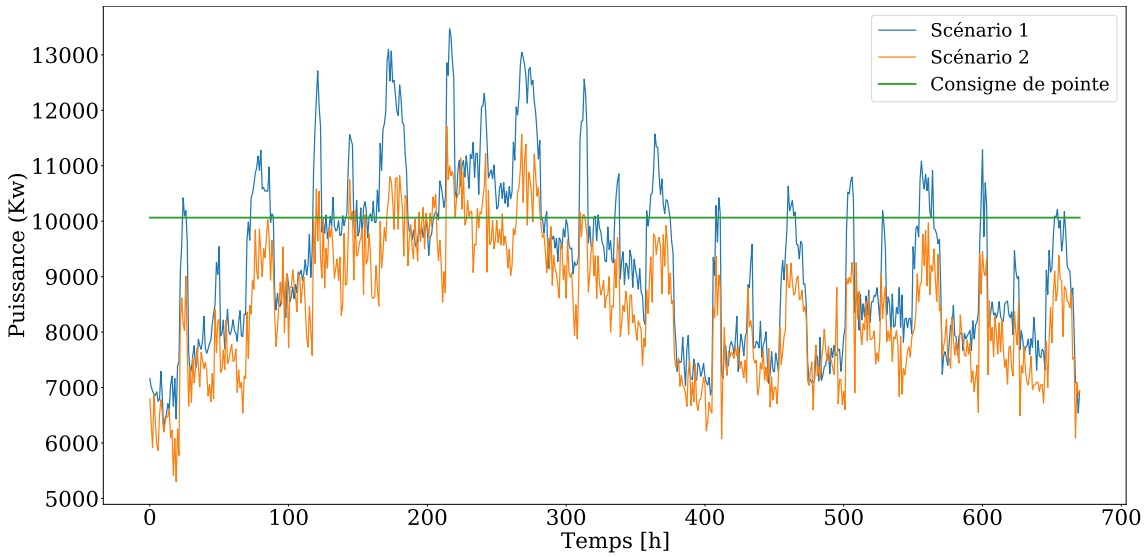

Figure 5.2 Comparaison de la puissance totale consommée du scénario 1 et 2

5.2 Cas d'étude 2 : Contrôle généralisé de la température et de la puissance

Pour l'instant, nous avons pu construire un agent capable d'interpoler son savoir et bien gérer la demande. Cependant, cet agent a été testé sur un environnement similaire à celui observé lors de son entraînement. Cependant, cette généralisation est loin de décrire la réalité car d'une maison à l'autre, beaucoup de paramètres peuvent changer, à savoir le volume de chaque

pièce, la composition des murs, le nombre de murs extérieurs et intérieurs pour chaque pièce, etc.. Idéalement, un agent doit être capable d'extrapoler et de généraliser sa connaissance en dehors des environnements similaires. Cette tâche est particulièrement difficile mais cruciale pour le déploiement des systèmes dans le monde réel car ils doivent être capables de gérer des situations imprévues.

Cette étude porte sur la construction d'un nouvel algorithme de RL qui vise à fournir un contrôle optimal de la température généralisable aux bâtiments résidentiels, à cela un contrôle de la puissance est ajouté. Mis à part le fait de pouvoir extrapoler la connaissance de l'agent dans d'autres environnements, cet algorithme permettra d'éviter de répéter la construction de plusieurs agents sur chaque nouveau bâtiment et donc de rendre plus facile de mettre en application des contrôleurs basés sur le RL dans le monde réel.

Ici également deux scénarios seront considérés :

- **Scénario 1** : consiste à créer un agent généralisé capable de maintenir la température intérieure à une consigne et donc le confort du consommateur.
- **Scénario 2** : consiste à créer un agent généralisé capable de contrôler la température en même temps que la puissance consommée.

5.2.1 Modélisation de l'environnement RL

Afin de pouvoir contrôler tout un bâtiment, il faudra d'abord savoir contrôler chaque pièce à l'intérieur. Afin de simplifier le modèle, on supposera qu'il existe un TCL par pièce rectangulaire, et que chacune est délimitée par des murs et sa longueur et largeur sont entre 4 m et 22 m avec une hauteur entre 2.5 m et 6 m. Ce choix est fait pour balayer un grand ensemble de dimensions de pièces qui peuvent exister.

Afin de pouvoir bien modéliser une pièce, il faut prendre en compte les paramètres qui impactent la température ambiante [115] à savoir principalement : la température ambiante actuelle, le volume de la pièce, la puissance du TCL pour chauffer ou climatiser la pièce, les températures adjacentes à chaque mur et la composition de chaque mur (résistance et capacité thermique). Ici également, la consommation des TCLs représente environ 70% de la puissance totale consommée et les 30% restants représentent les autres appareils ménagers.

Vecteur d'état

Les informations utiles à chaque pièce sont nombreuses et il faut choisir avec soin le contenu des observations de l'agent. En effet, augmenter trop la longueur du vecteur d'état pourra entraîner une détérioration des performances de l'algorithme en raison du fléau de la dimension [116].

Pour cela, le vecteur d'état suivant rassemble les informations essentielles :

$$\boldsymbol{S} = [\boldsymbol{T}, \boldsymbol{T_{out}}, \boldsymbol{P_{hc}}, \boldsymbol{V}, \boldsymbol{\phi_1}, \ldots, \boldsymbol{\phi_n}] \tag{5.4}$$

Où $\boldsymbol{\phi_i} = \frac{\boldsymbol{T - T^i_{adj}}}{\boldsymbol{R^i_{th}}}$ qui représente la puissance perdue à travers le mur $\boldsymbol{i}$. Elle regroupe la température ambiante $\boldsymbol{T}$, la température adjacente $\boldsymbol{T^i_{adj}}$ et la résistance thermique $\boldsymbol{R^i_{th}}$ du mur $\boldsymbol{i}$. $\boldsymbol{T_{out}}$ représente la température extérieure, et $\boldsymbol{P_{hc}}$ est la puissance du TCL pour chauffer ou climatiser la pièce. Ici également, pour passer d'un état à un autre, nous utilisons le modèle Résistance-Capacité (RC) pour modéliser l'évolution de $\boldsymbol{T}$ et $\boldsymbol{P_{hc}}$.

Récompense de l'agent

Comme dans le cas d'étude 1, nous définissons la récompense comme implémentée dans [67] pour le scénario 1 :

$$\boldsymbol{R_1} = \begin{cases} \boldsymbol{exp}(\frac{\boldsymbol{-(T-T_{target})^2}}{\boldsymbol{2}}) - \boldsymbol{\lambda_p}(\boldsymbol{T_{min} - T}) & \text{Si } \boldsymbol{T < T_{min}} \\ \boldsymbol{exp}(\frac{\boldsymbol{-(T-T_{target})^2}}{\boldsymbol{2}}) - \boldsymbol{\lambda_p}(\boldsymbol{T - T_{max}}) & \text{Si } \boldsymbol{T > T_{max}} \\ \boldsymbol{exp}(\frac{\boldsymbol{-(T-T_{target})^2}}{\boldsymbol{2}}) & \text{Sinon} \end{cases} \tag{5.5}$$

Où $\boldsymbol{T_{min} = 20°C}$ et $\boldsymbol{T_{max} = 22°C}$ sont les limites supérieures et inférieures de l'intervalle de température souhaité. Lorsque la température ambiante est en dehors de cet intervalle, l'agent est pénalisé proportionnellement au dépassement, avec une constante de proportionnalité $\boldsymbol{\lambda_p}$. En revanche, l'agent est récompensé lorsque la température est comprise dans l'intervalle, avec une récompense maximale de 1 atteinte lorsque la température ambiante est de $\boldsymbol{T_{target} = 21°C}$. Pour atteindre son objectif, l'agent doit prendre des actions continues dans $\boldsymbol{[-1, 1]}$, qui représente le pourcentage de la puissance maximale du TCL à utiliser. Les valeurs négatives correspondent aux actions de climatisation et les valeurs positives aux actions de chauffage.

Dans le scénario 2, où il est question de réduire en plus la charge, la récompense devra prendre en compte la température et la puissance consommée :

$$\boldsymbol{R_2} = \boldsymbol{R_1} - \boldsymbol{\lambda_Q}\frac{\boldsymbol{P_{hc}}}{\boldsymbol{P_{hc,max}}} \tag{5.6}$$

où $\boldsymbol{P_{hc,max}}$ est la puissance maximale que le TCL peut consommer et $\boldsymbol{\lambda_Q}$ est un coefficient permettant de réguler l'importance de la réduction de puissance. Le problème de l'optimisation est donc bi-objectif et il faut trouver un compromis entre le maintien de la température et la réduction de la consommation d'énergie.

5.2.2 Algorithme : GA2TC

Le processus d'entraînement d'un agent généralisé est en grande partie identique à un entraînement RL classique, durant lequel un agent contrôle les TCLs d'un bâtiment pendant un temps donné pour collecter et maximiser ses récompenses. La durée d'un épisode d'entraînement est donnée par la quantité de données externes disponibles, qui est d'un an dans notre cas. La principale différence ici est que, dans certaines conditions, l'architecture de la zone peut changer d'un épisode à l'autre. L'algorithme 2 présente plus de détails sur la procédure de l'entraînement.

Algorithm 2 GA2TC : Generalized Agent Training for Temperatue Control

```
Algo : un algorithme RL à choisir (ACKTR, TRPO, ...)
params : les hyperparmètres d'Algo
Arch : l'architecture de la pièce initiale sur laquelle l'agent doit s'entraîner
ValidArch : Les architectures de validation
N : nombre d'épisodes
rew_th : seuil de récompense, aide à décider de changer l'architecture au prochain épisode
n_years : nombre d'épisodes maximum à passer sur une architecture de pièce
perf = −∞
procedure GAT(Algo, params, Arch, N = 600, rew_th = 0, n_years = 3)
    Agent ← Algo(Arch, params)
    count ← 0
    for Épisode in 1, ..., N do
        count ← count +1
        Entraîner Agent sur Arch pour un épisode
        p ← performance de l'agent sur Arch
        v ← performance de l'agent sur ValidArch
        if perf ≤ v then
            bestAgent ← Agent
            perf ← v
        end if
        if (rew_th ≤ p) or (count = n_years) then
            Arch = Générer une nouvelle architecture aléatoirement
            count ← 0
        end if
    end for
    return Agent
end procedure
```

Comme c'est décrit dans l'algorithme, ces conditions sont respectées si au bout d'un nombre d'épisodes inférieur à n_{years}, la performance de l'agent sur le contrôle de la pièce en question

dépasse un seuil $\boldsymbol{rew_{th}}$; dans le cas où au bout de $\boldsymbol{n_{years}}$ l'agent n'arrive pas à dépasser ce seuil, le changement aura lieu quand même. Ce changement comprend principalement une nouvelle largeur et une nouvelle hauteur de la pièce, de nouveaux matériaux et types de murs (extérieurs ou intérieurs) ainsi qu'un nouveau profil de consommation des appareils. On peut établir un parallèle entre ce scénario de formation et un jeu à plusieurs niveaux comme CoinRun [77], dans lequel l'agent doit constamment résoudre des problèmes légèrement différents.

En plus de ce changement d'architecture, il faut ajouter une manière de trouver les bons hyperparamètres de l'algorithme que nous allons appliquer, et les paramètres tels que $\boldsymbol{\lambda_p}, \boldsymbol{\lambda_Q}$, $\boldsymbol{rew_{th}}$. Pour cela, nous allons chercher les paramètres capables de maximiser la récompense totale sur des architectures de validation $\boldsymbol{ValidArch}$: 12 architectures sont choisies pour couvrir l'espace des 4 m à 22 m en largeur/longueur et 2.5 m à 6 m en hauteur ainsi que les compositions des murs. Une fois ces paramètres sélectionnés, nous testerons l'agent sur des environnements tests afin d'avoir des performances non biaisées par rapport à la distribution des architectures test. De plus, $\boldsymbol{ValidArch}$ va être appliqué dans le contexte de la méthode de régularisation *Early Stopping* expliquée dans le chapitre 3.1.5 afin de récupérer le modèle qui marche le mieux sur cet ensemble de validation.

L'entraînement se fera à l'aide des données météo de l'année 2015, et le test se fera sur l'hiver et l'été de l'année 2016.

5.2.3 Application et résultats

Afin d'arriver à une bonne performance, un entraînement d'un algorithme RL peut parfois prendre énormément des jours à se réaliser. Vu que nous avons quatre algorithmes à tester, chacun avec une recherche d'hyperparamètres, puis sur deux scénarios, il est utile d'établir un choix stratégique d'applications. Pour cela, nous avons décidé de suivre ces étapes :

1. Comparer les 4 algorithmes dans le cadre du contrôle généralisé de la température seulement, dans des pièces. Une recherche d'hyperparamètres sera incluse.
2. Sélectionner le meilleur algorithme avec ses hyperparamètres.
3. Appliquer cet algorithme généralisé dans le cadre du contrôle de puissance et de température.
4. Comparer cette approche avec l'approche d'un agent simple entraîné sur une seule architecture.
5. Appliquer cette approche sur un bâtiment.

Il est à noter que l'implémentation d'ACKTR et A2C permet d'exécuter plusieurs agents

de manière synchronisée et en parallèle, chaque agent s'entraînera sur ses architectures indépendamment des autres agents, jusqu'au moment de faire la mise à jour de poids des réseaux, où tous partageront les mêmes poids. Ce nombre d'agents sera défini par $\boldsymbol{n_{workers}}$.

Contrôle de la température seulement

Pour cette partie, nous allons d'abord choisir au préalable 8 architectures tests telle que leurs propriétés couvrent l'espace de possibilités d'architectures. Nous entraînons par la suite les 4 algorithmes en tenant compte du fait que les architectures d'entraînement doivent différer de ceux du test. La recherche d'hyperparamètres nous permet de conclure que :

- TRPO : la meilleure configuration est de mettre les paramètres par défaut, à savoir ceux de l'implémentation stable-baselines [97].
- DDPG : $\boldsymbol{rew_{th} = 0.5}$, $\boldsymbol{\lambda_p = 0.1}$,. Les autres paramètres restent par défaut.
- ACKTR : $\boldsymbol{rew_{th} = 0.25}$, $\boldsymbol{\lambda_p = 0.2}$, $\boldsymbol{n_{workers} = 40}$. Les autres paramètres restent par défaut.
- A2C : $\boldsymbol{rew_{th} = 0.9}$, $\boldsymbol{\lambda_p = 0.1}$, $\boldsymbol{n_{workers} = 60}$. Les autres paramètres restent par défaut.

La Figure 5.3 les boîtes à moustaches de températures générées par chaque algorithme sur l'ensemble des tests.

Chaque test est réalisé sur 1250 heures de simulation qui regroupent la période hivernale et la période estivale. Les deux lignes noires horizontales indiquent l'intervalle de température souhaité. Les volumes des architectures de test sont triés de manière que Arch 1 soit le plus petit et Arch 8 le plus grand.

On peut remarquer que l'agent DDPG se comporte bien à chaque test, sauf aux tests 5 et 6, où l'écart est élevé et les températures souvent hors de l'intervalle souhaité. Les résultats des méthodes ACKTR et A2C sont plus cohérents sur l'ensemble des tests avec de bonnes récompenses moyennes (0.67 et 0.64) et il semble que cela soit lié au $\boldsymbol{n_{workers}}$ qui servent au calcul parallèle. En effet, chaque *worker* génère des échantillons de différentes architectures en même temps, de sorte que l'agent voit une partie beaucoup plus grande de l'espace d'observation à chaque itération, ce qui rend l'exploration spatiale globale plus pertinente car l'agent voit un nombre beaucoup plus important d'architecture pendant la formation. Cela rend également la formation ACKTR et A2C beaucoup plus efficace en termes de temps.

Le modèle choisi sera donc ACKTR et sera utilisé par la suite pour les prochaines parties.

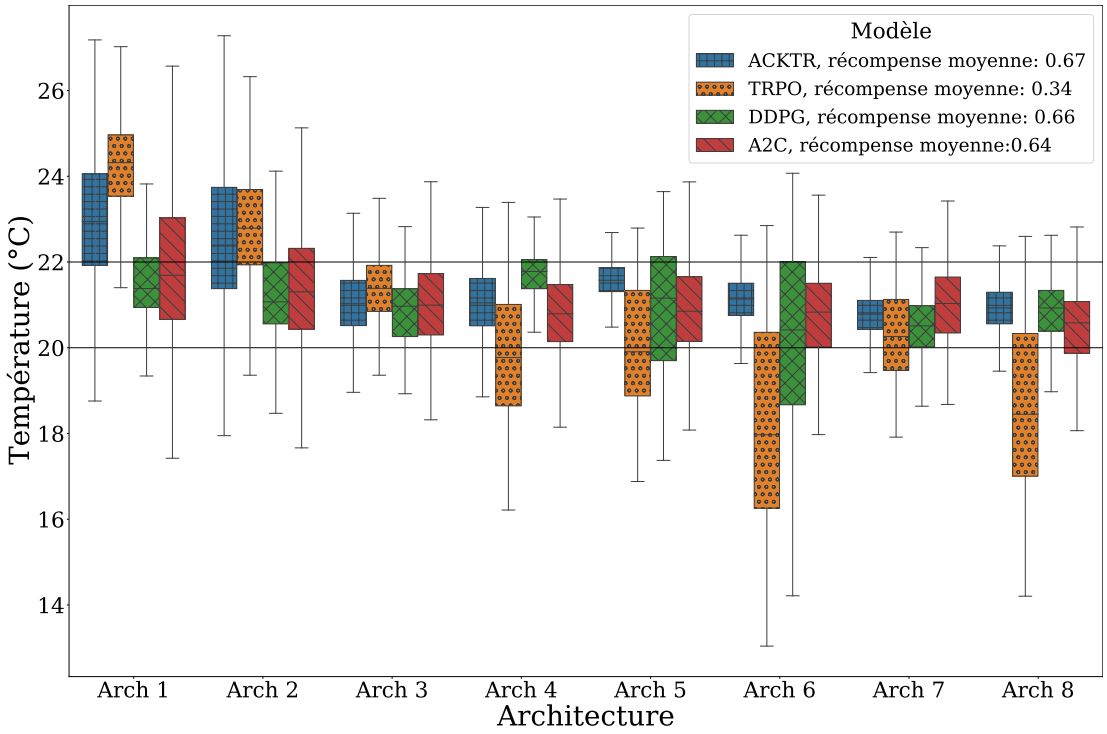

Figure 5.3 Box plot des quatre modèles de sortie de température, testés dans huit environnements

Contrôle de la température et de la puissance

Nous entraînons maintenant le modèle ACKTR afin de contrôler la température et la puissance avec l'algorithme GA2TC et cela sur les mêmes 8 environnements test. $\boldsymbol{\lambda_Q}$ fait également l'objet d'une recherche par validation, en prenant des valeurs entre 0.1 et 1 : nous trouvons que $\boldsymbol{\lambda_Q = 0.7}$ est celle qui donne le meilleur modèle en validation. Le tableau 5.3 montre une comparaison des résultats obtenus en termes de température moyenne dans chaque scénario sur les architectures de test, puis l'impact du contrôle de la charge sur la réduction de l'énergie moyenne consommée.

À chaque test, l'agent réduit la puissance consommée tout en maintenant la température autour de $\boldsymbol{21°C}$. Il est intéressant de noter que certaines des plus grandes réductions de consommation se sont accompagnées d'un meilleur contrôle, comme sur les architectures tests 1 et 3 où les écarts de température sont plus faibles.

Efficacité de l'approche - contrôle de température seulement

Pour prouver l'efficacité de la méthode GA2TC, nous comparons les performances des agents ACKTR avec et sans GA2TC. Pour cela, nous avons entraîné de nouveaux agents sur chacune

Tableau 5.3 ACKTR : Température ambiante et consommation d'énergie obtenues avec le meilleur agent RL généralisé avec et sans l'objectif de réduction de puissance

	Scénario 1	Scénario 2	
Architecture de test	Temp. ($°\boldsymbol{C}$)	Temp. ($°\boldsymbol{C}$)	Réduction d'énergie
Arch 1	21.9 $\pm$ 1.3	20.5 $\pm$ 1.1	-15.8%
Arch 2	21.1 $\pm$ 1.7	20.0 $\pm$ 1.2	-10.6%
Arch 3	21.3 $\pm$ 0.9	20.5 $\pm$ 0.8	-11.0%
Arch 4	20.9 $\pm$ 2.0	20.2 $\pm$ 1.6	-8.56%
Arch 5	21.2 $\pm$ 0.5	20.9 $\pm$ 0.5	-6.77%
Arch 6	20.7 $\pm$ 0.6	20.3 $\pm$ 0.8	-6.66%
Arch 7	21.2 $\pm$ 1.0	20.7 $\pm$ 0.6	-6.31%
Arch 8	20.8 $\pm$ 2.3	20.4 $\pm$ 1.5	-4.63%

des 12 architectures de manière que chaque agent 'spécialisé' soit spécifiquement entraîné sur une seule et unique architecture. Le but sera de comparer la performance de ces agents spécialisés avec celle de notre agent GA2TC quand il est question de généraliser sur plusieurs architectures. Pour rendre le processus plus représentatif, nous avons utilisé 12 architectures test à la place de 8, ce qui engendrera l'entraînement de 12 agents, chacun sur une architecture. Une fois entraînés, nous testons ces agents sur toutes les architectures test, et comparons les performances obtenues avec celles de l'agent GA2TC. La Figure 5.4 représente des box plots de comparaison entre notre agent ACKTR-GA2TC et l'agent ACKTR entraîné sur l'Arch 6 seulement.

Comme on le remarque, l'agent spécialisé a bien géré la température sur l'architecture sur laquelle il a été entraîné : Arch 6. Cependant, le contrôle de la température se détériore rapidement lorsque le volume de la pièce diminue pour atteindre des températures médianes éloignées de la cible. Le contrôle de la température est meilleur pour les grandes pièces. Néanmoins, le contrôle commence à se détériorer sur les Arch 11 et Arch 12 car les températures s'étendent en dehors de l'intervalle souhaité (voir les lignes de consignes horizontal). D'autre part, l'agent généralisé reste bien adapté à l'ensemble de la des architectures test, ce qui prouve l'efficacité de la méthode GA2TC. De plus, le profil de température obtenu par l'agent généralisé sur l'Env6 est très similaire à celui de l'agent spécialisé. Cela montre la qualité du contrôle de la température atteint par l'agent généralisé.

Contrôle dans un bâtiment

Dans cette section, nous appliquons le modèle ACKTR-GA2TC pour contrôler la température des bâtiments multizones. Les deux architectures de bâtiments présentées à la Figure 5.5

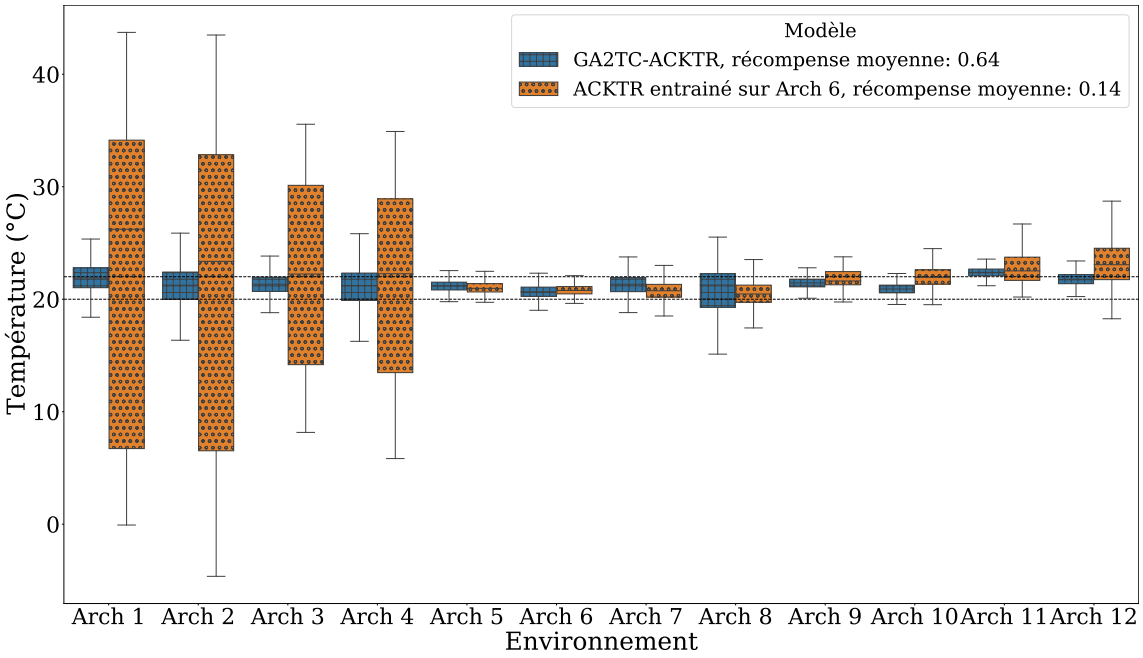

Figure 5.4 Box plots de GA2TC vs Agent spécialisé entraîné sur Arch 6

regroupe des petites et des grandes pièces, pour chacune desquelles l'agent généralisé est aura pour objectif à la fois de contrôler la température et de consommer le moins d'énergie possible. Notez que cet agent n'a reçu aucun entraînement supplémentaire ni aucune information préalable sur le bâtiment qu'il doit maintenant contrôler.

Comme on peut le voir sur les chiffres de la Figure 5.6 dans les deux bâtiments, les agents parviennent à contrôler la température des pièces près de l'objectif de $\mathbf{21°}\boldsymbol{C}$. Cela montre comment, en utilisant la méthode GA2TC pour former un agent généralisé, on peut contrôler des bâtiments inconnus sans avoir besoin de former spécifiquement de nouveaux agents et donc sans avoir besoin de développer la modélisation thermique du bâtiment.

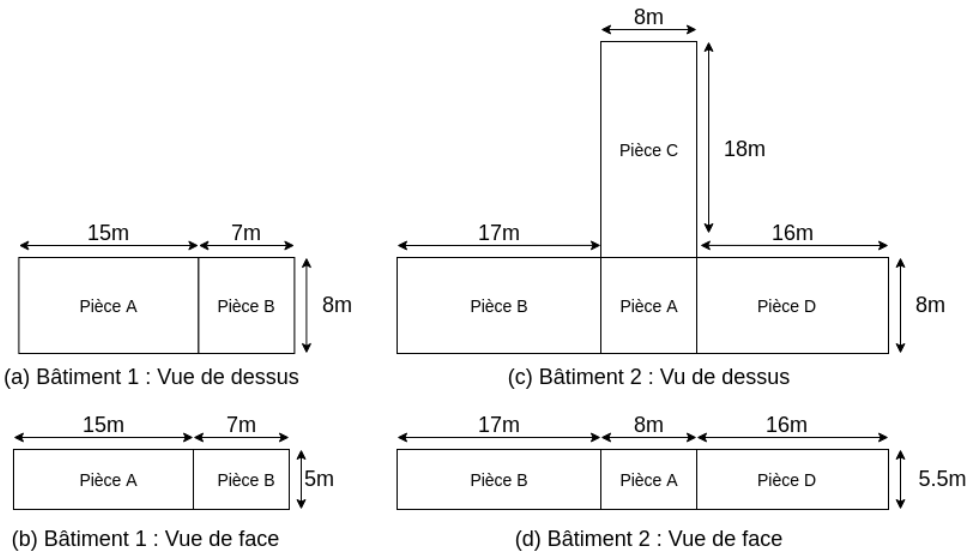

Figure 5.5 Dimensions des deux bâtiments

Cela montre comment, en utilisant la méthode GA2TC pour former un agent généralisé, on peut contrôler des bâtiments inconnus sans avoir besoin de former spécifiquement de nouveaux agents et donc sans avoir besoin de développer la modélisation thermique du bâtiment.

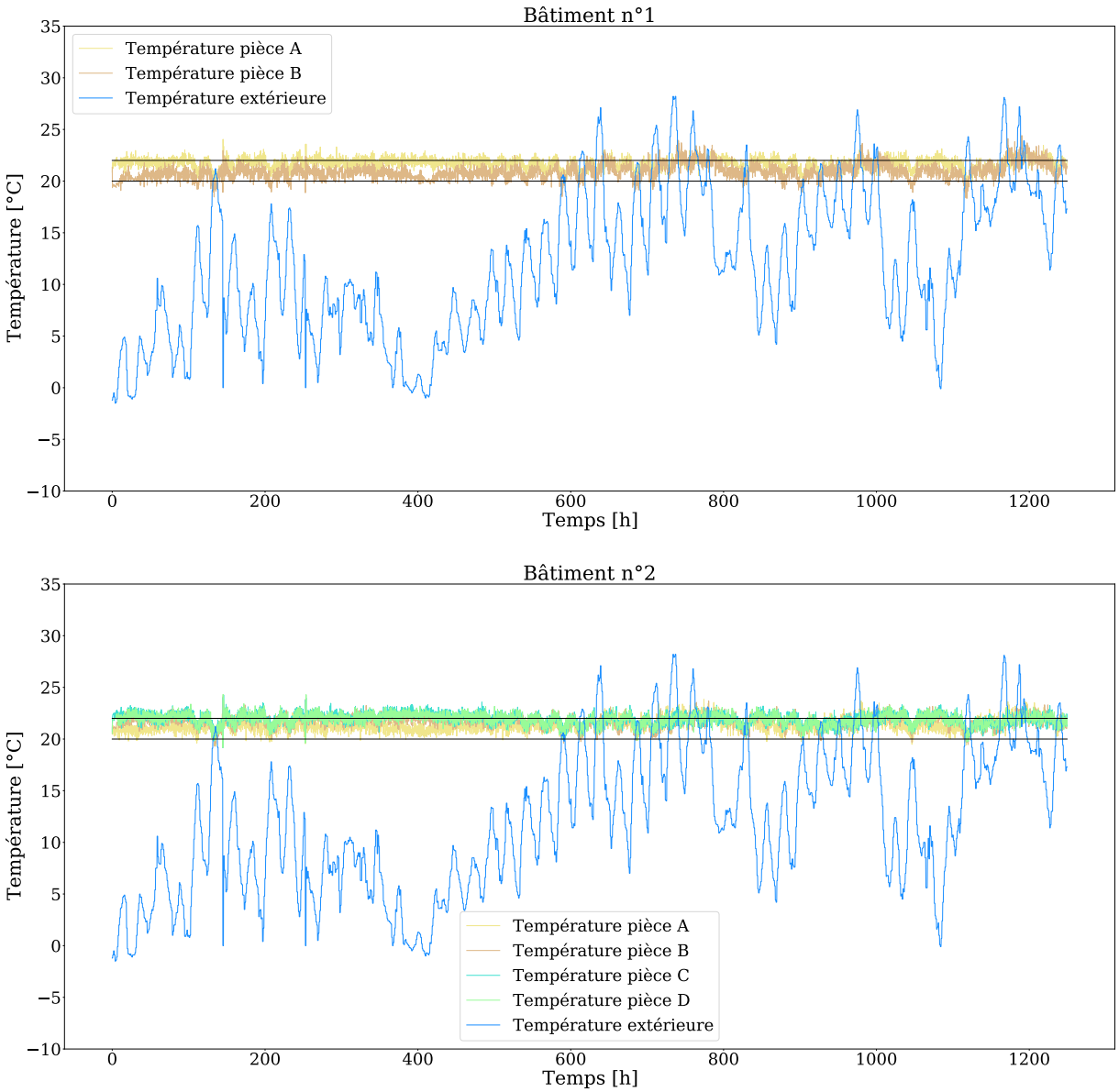

Figure 5.6 Évolution de la température dans les deux bâtiments

CHAPITRE 6 DISCUSSION GÉNÉRALE

Ce chapitre dresse le portrait des résultats rapportés au sein de cette recherche. Nous discuterons des limitations et des pistes d'amélioration observées dans le chapitre suivant.

Le premier volet du projet présente un article qui expose une étude comparative de diverses approches d'apprentissage profond pour des prévisions agrégées de la charge électrique à court terme en utilisant des données hétérogènes. Ces prévisions ont été utilisées pour déterminer leur fiabilité concernant le pourcentage de pointe sur une certaine puissance souscrite.

Tout d'abord, nos approches de la prévision des séries chronologiques dépendent principalement :

- Du type de données utilisées : les données agrégées du district ou de chacun des bâtiments du district.
- Des caractéristiques historiques extraites des données des séries chronologiques.
- Des algorithmes d'apprentissage profond utilisés.

Nous avons conclu que la stratégie la plus efficace pour sélectionner les variables historiques consiste à appliquer l'ACF. De plus, les modèles récurrents se sont avérés les plus performants par rapport au MLP, surtout lorsqu'il s'agit de prévoir la charge de nombreux pas de temps à l'avance. Dans ce cas particulier où nous avons des bâtiments hétérogènes, ce qui rend le problème de la prévision de la charge agrégée plus difficile, nous avons conclu que la meilleure approche est d'utiliser l'algorithme LSTM sur les données de chaque bâtiment pour prévoir leur charge, puis de faire la somme de ces charges. L'avantage avec cette approche est que la sélection de variables historiques devient dynamique car elle dépend du bâtiment non pas du campus, ce qui permet au modèle de mieux détecter la volatilité existante dans les données de chaque bâtiment. Finalement, Les performances de notre approche ont été mesurées, on obtient un MAPE compris entre 1,83% et 5,18% selon l'horizon de prédiction.

Le deuxième volet du projet porte sur l'implémentation de méthodes décrivant l'état de l'art en terme du contrôle de l'énergie électrique, également basées sur l'apprentissage par renforcement afin d'avoir un bon maintien de la température intérieure de bâtiments résidentiels, tout en réduisant la consommation d'énergie, et cela particulièrement pendant les périodes de pointe. Deux cas d'études y sont présentés :

- le premier cas présente un travail dont le but est de voir comment donner à un agent RL des informations concernant la future consommation de puissance agrégée de plusieurs maisons peut impacter sa performance sur le contrôle. Le résultat est satisfaisant car nous pouvons

réduire jusqu'à 73% d'énergie consommée pendant les périodes de pointes tout en ayant un contrôle optimal de la température dans chaque maison.

- Le deuxième cas d'étude présente une étude approfondie réalisée pour la première fois afin d'examiner la possibilité de développer un agent RL qui peut fournir un contrôle optimal de la température dans un bâtiment donné sans prendre en compte des informations préalables telles que son volume ou le nombre de murs extérieurs, tout en réduisant la puissance consommée du bâtiment. Tout d'abord, nous avons comparé 4 algorithmes RL dans le but de maintenir le confort des consommateurs pour constater que l'ACKTR constitue la meilleure adéquation avec notre méthode d'entraînement. Nous avons ensuite effectué une analyse sur la capacité de l'agent à réduire la puissance consommée tout en maintenant la température en utilisant le même modèle ACKTR, en obtenant jusqu'à 15.8% de réduction d'énergie consommée avec un très bon maintien du confort. De plus, nous démontrons l'efficacité de notre méthode d'entraînement GA2TC en comparant les performances de notre agent généralisé avec celles d'agents entraînés sur une architecture unique. Par la suite, cette étude montre que le contrôle peut être directement étendu à un bâtiment entier, indépendamment de sa dimension et des caractéristiques des nouveaux murs.

CHAPITRE 7 CONCLUSION ET RECOMMANDATIONS

Le chapitre 7 clôture l'ensemble de ce mémoire dont l'objectif est de créer un cadre de prédiction et de contrôle de la charge électrique en maximisant le confort des consommateurs. Pour ce faire, l'étude est orientée autour de deux principaux objectifs :

- Développer une étude comparative de méthodes d'apprentissage profond pour la prévision à court terme de la charge agrégée pour des bâtiments hétérogènes en apportant une estimation du pic de puissance. De plus, l'étude propose diverses stratégies pour capturer l'aspect comportemental des consommateurs dans les bâtiments.
- Proposer des modèles basés sur l'apprentissage par renforcement afin de pourvoir contrôler la charge électrique en maintenant le confort thermique dans multiple maisons résidentiels. Ensuite, un nouvel algorithme généralisé appelé GA2TC propose une manière d'établir ce contrôle dans des ménages en dépit de leurs architectures.

Cette conclusion énonce les principales limitations associées aux méthodes employées et leurs applications en suivant l'ordre des études réalisées, pour finalement proposer de nouvelles perspectives de recherche.

7.1 Prédiction de la charge agrégée à court terme

Trois algorithmes d'apprentissage profond sont utilisés afin de prédire la charge à court terme. Mis à part leur entraînement qui tend à trouver les bons poids de chaque architecture, chacun de ces modèles possède une multitude d'hyperparamètres à trouver : le nombre de neurones, le nombre de couches cachées, la manière de sélectionner les variables, le traitement des données, l'algorithme d'optimisation etc.. De ce fait, nous sommes limités dans l'exploration de ces hyperparamètres et devons restreindre l'espace de recherche.

Par ailleurs, malgré la bonne performance de notre modèle final, nous pensons fortement que les données seraient beaucoup plus exploitables pour améliorer le modèle en question si nous avions accès à la consommation au niveau du consommateur au lieu du bâtiment. En effet, ils vont nous servir à mieux capturer les divers comportements à un plus bas niveau et donc rendre le modèle de prédiction plus robuste face aux changements brusques dans la consommation d'un consommateur. De plus, il a aussi été prouvé expérimentalement [117] que la performance des prédictions agrégées devient meilleure lorsque ces prédictions se font à un plus bas niveau (e.g. au niveau du consommateur au lieu du bâtiment).

7.2 Gestion de la charge et contrôle de la température

Pour les deux cas d'études, quelques limitations reviennent de la même manière que pour le chapitre 7.1. En effet, les algorithmes RL sont connus pour prendre beaucoup de temps à entraîner, et la modélisation de l'environnement, en termes de vecteur d'état et de récompense, joue un important rôle dans la performance de l'agent. De plus, l'un de leurs inconvénients majeurs est que la recherche d'hyperparamètres, à savoir l'architecture de l'Actor et du Critic dans DDPG, le nombre d'épisodes etc., est très importante pour obtenir de bons résultats. Aussi, il est important de noter que le modèle thermique RC utilisé dans les deux cas est un modèle très simple de premier ordre, et donc ne reflète pas forcément la dynamique thermique dans le monde réel. De plus, les radiations qui à leur tour font évoluer la température ambiante plus rapidement ne sont pas prises en compte. Pour améliorer cette modélisation, deux cas de figures sont envisageables : implémenter une modélisation thermique plus compliquée de deuxième ou troisième ordre, ou bien utiliser un programme de simulation énergétique des bâtiments comme *EnergyPlus*. Un autre aspect important est que nos agents sont spécialisés dans la gestion des bâtiments résidentiels, où le comportement quotidien des consommateurs a été modélisé de manière générale par un pic de consommation d'électricité le matin, un autre le soir et une activité aléatoire le midi, ce qui est loin de représenter le comportement de tous les consommateurs de bâtiments résidentiels. Pour pallier cela, avoir accès à la consommation résidentielle réelle serait une solution qui permettrait à l'agent d'élargir son espace d'exploration et donc être plus robuste quand une fois testé dans un cadre réel. Dans la même optique, l'une des améliorations à ajouter est le contrôle d'autres types de bâtiments tels que les bâtiments commerciaux où institutionnels, où le comportement des consommateurs peut être totalement différent. Aussi, un aspect très important dans ce cadre de maîtrise et que nous nous sommes penchés sur le contrôle du chauffage et de la climatisation que par le biais de TCLs. Une amélioration possible est d'étendre ce contrôle en l'appliquant sur tous les appareils contrôlables de la maison.

Le premier cas d'étude porte sur le développement d'un seul agent qui a comme mission la bonne gestion de la charge de la température dans 1 à 6 maisons. Malgré ses bonnes performances, nous remarquons qu'en terme de réduction d'énergie, l'agent n'arrive pas à reproduire les mêmes performances quand il s'agit de gérer plusieurs maisons. En effet, avec une seule maison à gérer, l'agent est capable de réduire à 73% l'énergie consommée pendant les périodes de pointes. Alors qu'avec 6 maisons à gérer, seulement 28% de réduction d'énergie est réalisée pendant ces mêmes périodes. Cela montre les limites d'un seul agent et suggère l'adoption d'une stratégie multi-agents à la place.

Pour le deuxième cas d'étude, un problème non abordé avec cette approche de généralisation

est que nous n'avons pas utilisé les prédictions des futures consommations de puissance comme une information à donner à l'agent. Cela revient au fait que l'agent est entraîné sur différentes pièces et par conséquence si on devait fournir les données de prédictions à cet agent, on devrait avoir accès aux prédictions de chaque pièce que l'agent devra contrôler, et cela en dépit de l'architecture de la pièce en question. Or, nos modèles de prévisions ne sont pas forcément faits pour généraliser sur toutes les architectures. Il faut donc trouver une manière à ce qu'on puisse construire un modèle de prédiction généralisé également.

RÉFÉRENCES

[1] C. Olah, "Understanding lstm networks," 2015. [En ligne]. Disponible : http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[2] J. Tyndall, *Heat : a mode of motion*. D. Appleton, 1881.

[3] H. Rodhe, R. Charlson et E. Crawford, "Svante arrhenius and the greenhouse effect," *Ambio*, p. 2–5, 1997.

[4] [En ligne]. Disponible : https://www.theguardian.com/environment/2019/may/06/human-society-under-urgent-threat-loss-earth-natural-life-un-report

[5] [En ligne]. Disponible : https://www.nytimes.com/interactive/2018/08/01/magazine/climate-change-losing-earth.html

[6] [En ligne]. Disponible : https://www.climate.gov/news-features/understanding-climate/climate-change-global-sea-level

[7] A. M. Carreiro, H. M. Jorge et C. H. Antunes, "Energy management systems aggregators : A literature survey," *Renewable and Sustainable Energy Reviews*, vol. 73, p. 1160–1172, 2017.

[8] A. Dedinec *et al.*, "Deep belief network based electricity load forecasting : An analysis of macedonian case," *Energy*, vol. 115, p. 1688–1700, 2016.

[9] M. E. El-Hawary, "The smart grid—state-of-the-art and future trends," *Electric Power Components and Systems*, vol. 42, n°. 3-4, p. 239–250, 2014.

[10] S. Khatiri-Doost et M. Amirahmadi, "Peak shaving and power losses minimization by coordination of plug-in electric vehicles charging and discharging in smart grids," dans *2017 IEEE International Conference on Environment and Electrical Engineering and 2017 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe)*. IEEE, 2017, p. 1–5.

[11] Hydro-Québec. Using energy more wisely in cold weather. [En ligne]. Disponible : http://www.hydroquebec.com/residential/customer-space/electricity-use/winter-electricity-consumption.html

[12] S.-Y. Son *et al.*, "Feature selection for daily peak load forecasting using a neuro-fuzzy system," *Multimedia Tools and applications*, vol. 74, n°. 7, p. 2321–2336, 2015.

[13] B. P. Hayes, J. K. Gruber et M. Prodanovic, "Multi-nodal short-term energy forecasting using smart meter data," *IET Generation, Transmission & Distribution*, vol. 12, n°. 12, p. 2988–2994, 2018.

[14] Hydro-Québec. Consumers webpage about smart meters. [En ligne]. Disponible : http://www.hydroquebec.com/residentiel/espace-clients/compte-et-facture/releve-compteur.html

[15] I. Goodfellow, Y. Bengio et A. Courville, *Deep learning*. MIT press, 2016.

[16] H. Chen *et al.*, "Day-ahead aggregated load forecasting based on two-terminal sparse coding and deep neural network fusion," *Electric Power Systems Research*, vol. 177, p. 105987, 2019.

[17] S. Sreekumar, K. C. Sharma et R. Bhakar, "Gumbel copula based aggregated net load forecasting for modern power systems," *IET Generation, Transmission & Distribution*, vol. 12, n°. 19, p. 4348–4358, 2018.

[18] C. Fan *et al.*, "Assessment of deep recurrent neural network-based strategies for short-term building energy predictions," *Applied energy*, vol. 236, p. 700–710, 2019.

[19] M. Tan *et al.*, "Ultra-short-term industrial power demand forecasting using lstm based hybrid ensemble learning," *IEEE Transactions on Power Systems*, 2019.

[20] X. Tang *et al.*, "Short-term power load forecasting based on multi-layer bidirectional recurrent neural network," *IET Generation, Transmission & Distribution*, vol. 13, n°. 17, p. 3847–3854, 2019.

[21] W. Kong *et al.*, "Short-term residential load forecasting based on resident behaviour learning," *IEEE Transactions on Power Systems*, vol. 33, n°. 1, p. 1087–1088, 2017.

[22] M. A. Rahman, A. Zubair *et al.*, "Electric load forecasting with hourly precision using long short-term memory networks," dans *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*. IEEE, 2019, p. 1–6.

[23] N. Somu, G. R. MR et K. Ramamritham, "A hybrid model for building energy consumption forecasting using long short term memory networks," *Applied Energy*, vol. 261, p. 114131, 2020.

[24] L. Wen, K. Zhou et S. Yang, "Load demand forecasting of residential buildings using a deep learning model," *Electric Power Systems Research*, vol. 179, p. 106073, 2020.

[25] J. Xie *et al.*, "Relative humidity for load forecasting models," *IEEE Transactions on Smart Grid*, vol. 9, n°. 1, p. 191–198, 2016.

[26] Q. Wu *et al.*, "Short-term load forecasting support vector machine algorithm based on multi-source heterogeneous fusion of load factors," *Automation of Electric Power Systems*, vol. 40, n°. 15, p. 67–72, 2016.

[27] P. Zeng et M. Jin, "Peak load forecasting based on multi-source data and day-to-day topological network," *IET Generation, Transmission & Distribution*, vol. 12, n°. 6, p. 1374–1381, 2017.

[28] Y. Wang *et al.*, "Conditional residual modeling for probabilistic load forecasting," *IEEE Transactions on Power Systems*, vol. 33, n°. 6, p. 7327–7330, 2018.

[29] W. Kong *et al.*, "Short-term residential load forecasting based on lstm recurrent neural network," *IEEE Transactions on Smart Grid*, vol. 10, n°. 1, p. 841–851, 2017.

[30] Y. Wang *et al.*, "An ensemble forecasting method for the aggregated load with subprofiles," *IEEE Transactions on Smart Grid*, vol. 9, n°. 4, p. 3906–3908, 2018.

[31] J. Nagi *et al.*, "A computational intelligence scheme for the prediction of the daily peak load," *Applied Soft Computing*, vol. 11, n°. 8, p. 4773–4788, 2011.

[32] Z. Yu *et al.*, "Deep learning for daily peak load forecasting–a novel gated recurrent neural network combining dynamic time warping," *IEEE Access*, vol. 7, p. 17 184–17 194, 2019.

[33] M. Rekik *et al.*, "Geographic routing protocol for the deployment of virtual power plant within the smart grid," *Sustainable Cities and Society*, vol. 25, p. 39–48, 2016.

[34] I. Rahman *et al.*, "Swarm intelligence-based smart energy allocation strategy for charging stations of plug-in hybrid electric vehicles," *Mathematical Problems in Engineering*, vol. 2015, 2015.

[35] S. Rahim *et al.*, "Exploiting heuristic algorithms to efficiently utilize energy management controllers with renewable energy sources," *Energy and Buildings*, vol. 129, p. 452–470, 2016.

[36] T. Niknam *et al.*, "A new approach based on ant algorithm for volt/var control in distribution network considering distributed generation," 2005.

[37] M. Kezunovic, "Smart fault location for smart grids," *IEEE transactions on smart grid*, vol. 2, n°. 1, p. 11–22, 2011.

[38] D. Huynh et L. Ho, "Improved pso algorithm based optimal operation in power systems integrating solar and wind energy sources," *International Journal of Energy, Information and Communications*, vol. 7, p. 9–20, 04 2016.

[39] A. Arabali *et al.*, "Genetic-algorithm-based optimization approach for energy management," *IEEE Transactions on Power Delivery*, vol. 28, n°. 1, p. 162–170, 2012.

[40] P. H. Shaikh *et al.*, "A review on optimized control systems for building energy and comfort management of smart sustainable buildings," *Renewable and Sustainable Energy Reviews*, vol. 34, p. 409–429, 2014.

[41] J. H. Holland, "Genetic algorithms," *Scientific american*, vol. 267, n°. 1, p. 66–73, 1992.

[42] J. Kennedy et R. Eberhart, "Particle swarm optimization," dans *Proceedings of ICNN'95-International Conference on Neural Networks*, vol. 4. IEEE, 1995, p. 1942–1948.

[43] R. S. Sutton et A. G. Barto, *Reinforcement Learning : An Introduction*, 2e éd. The MIT Press, 2018.

[44] A. Galbazar, S. Ali et D. Kim, "Optimization approach for energy saving and comfortable space using aco in building," *International Journal of Smart Home*, vol. 10, p. 47–56, 04 2016.

[45] J. R. Vázquez-Canteli et Z. Nagy, "Reinforcement learning for demand response : A review of algorithms and modeling techniques," *Applied Energy*, vol. 235, p. 1072 – 1089, 2019. [En ligne]. Disponible : http://www.sciencedirect.com/science/article/pii/S0306261918317082

[46] R. Lu et S. H. Hong, "Incentive-based demand response for smart grid with reinforcement learning and deep neural network," *Applied Energy*, vol. 236, p. 937 – 949, 2019. [En ligne]. Disponible : http://www.sciencedirect.com/science/article/pii/S0306261918318798

[47] A. Afram et F. Janabi-Sharifi, "Theory and applications of hvac control systems–a review of model predictive control (mpc)," *Building and Environment*, vol. 72, p. 343–355, 2014.

[48] J. Bai, S. Wang et X. Zhang, "Development of an adaptive smith predictor-based self-tuning pi controller for an hvac system in a test room," *Energy and Buildings*, vol. 40, no. 12, p. 2244–2252, 2008.

[49] J. Bai et X. Zhang, "A new adaptive pi controller and its application in hvac systems," *Energy Conversion and Management*, vol. 48, no. 4, p. 1043–1054, 2007.

[50] M. R. Kulkarni et F. Hong, "Energy optimal control of a residential space-conditioning system based on sensible heat transfer modeling," *Building and Environment*, vol. 39, no. 1, p. 31–38, 2004.

[51] Y. Zong *et al.*, "Model predictive control for smart buildings to provide the demand side flexibility in the multi-carrier energy context : Current status, pros and cons, feasibility and barriers," *Energy Procedia*, vol. 158, p. 3026–3031, 2019.

[52] Y. Yan *et al.*, "Adaptive optimal control model for building cooling and heating sources," *Energy and Buildings*, vol. 40, no. 8, p. 1394–1401, 2008.

[53] B. Dong, "Non-linear optimal controller design for building hvac systems," dans *2010 IEEE International Conference on Control Applications*, 2010, p. 210–215.

[54] J. Nishiguchi, T. Konda et R. Dazai, "Data-driven optimal control for building energy conservation," dans *Proceedings of SICE Annual Conference 2010*, 2010, p. 116–120.

[55] X. Wei *et al.*, "Multi-objective optimization of the hvac (heating, ventilation, and air conditioning) system performance," *Energy*, vol. 83, p. 294–306, 2015.

[56] A. Garnier *et al.*, "Low computational cost technique for predictive management of thermal comfort in non-residential buildings," *Journal of Process Control*, vol. 24, n°. 6, p. 750–762, 2014.

[57] A. Kusiak, M. Li et F. Tang, "Modeling and optimization of hvac energy consumption," *Applied Energy*, vol. 87, n°. 10, p. 3092–3102, 2010.

[58] M. Aydinalp-Koksal et V. I. Ugursal, "Comparison of neural network, conditional demand analysis, and engineering approaches for modeling end-use energy consumption in the residential sector," *Applied Energy*, vol. 85, n°. 4, p. 271–296, 2008.

[59] A. E. Ben-Nakhi et M. A. Mahmoud, "Energy conservation in buildings through efficient a/c control using neural networks," *Applied Energy*, vol. 73, n°. 1, p. 5–23, 2002.

[60] S. Soyguder, M. Karakose et H. Alli, "Design and simulation of self-tuning pid-type fuzzy adaptive control for an expert hvac system," *Expert systems with applications*, vol. 36, n°. 3, p. 4566–4573, 2009.

[61] A. Pal et R. Mudi, "Self-tuning fuzzy pi controller and its application to hvac system," *International Journal of Computational Cognition*, vol. 6, 01 2008.

[62] T. Wei, Y. Wang et Q. Zhu, "Deep reinforcement learning for building hvac control," dans *Proceedings of the 54th Annual Design Automation Conference 2017*, 2017, p. 1–6.

[63] B. J. Claessens *et al.*, "Model-free control of thermostatically controlled loads connected to a district heating network," *Energy and Buildings*, vol. 159, p. 1 – 10, 2018. [En ligne]. Disponible : http://www.sciencedirect.com/science/article/pii/S0378778817303353

[64] Q. Wei *et al.*, "Optimal self-learning battery control in smart residential grids by iterative q-learning algorithm," dans *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, Dec 2014, p. 1–7.

[65] S. Bahrami, V. W. S. Wong et J. Huang, "An online learning algorithm for demand response in smart grid," *IEEE Transactions on Smart Grid*, vol. 9, n°. 5, p. 4712–4725, Sep. 2018.

[66] B. Li et L. Xia, "A multi-grid reinforcement learning method for energy conservation and comfort of hvac in buildings," dans *2015 IEEE International Conference on Automation Science and Engineering (CASE)*, Aug 2015, p. 444–449.

[67] T. Moriyama *et al.*, *Reinforcement Learning Testbed for Power-Consumption Optimization : 18th Asia Simulation Conference, AsiaSim 2018, Kyoto, Japan, October 27–29, 2018, Proceedings*, 10 2018, p. 45–59.

[68] Z. Zhang *et al.*, "Whole building energy model for hvac optimal control : A practical framework based on deep reinforcement learning," *Energy and*

Buildings, vol. 199, p. 472 – 490, 2019. [En ligne]. Disponible : http://www.sciencedirect.com/science/article/pii/S0378778818330858

[69] G. T. Costanzo *et al.*, "Experimental analysis of data-driven control for a building heating system," *Sustainable Energy, Grids and Networks*, vol. 6, p. 81–90, 2016.

[70] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, n°. 7540, p. 529–533, 2015.

[71] D. Silver *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, n°. 7587, p. 484, 2016.

[72] B. M. Lake *et al.*, "Building machines that learn and think like people," *Behavioral and brain sciences*, vol. 40, 2017.

[73] G. Marcus, "Deep learning : A critical appraisal," *arXiv preprint arXiv :1801.00631*, 2018.

[74] K. Kansky *et al.*, "Schema networks : Zero-shot transfer with a generative causal model of intuitive physics," dans *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, p. 1809–1818.

[75] A. Rajeswaran *et al.*, "Epopt : Learning robust neural network policies using model ensembles," *arXiv preprint arXiv :1610.01283*, 2016.

[76] Y. Duan *et al.*, "Rl$^{\mathbf{2}}$: Fast reinforcement learning via slow reinforcement learning," 2016.

[77] L. Harries *et al.*, "Mazeexplorer : A customisable 3d benchmark for assessing generalisation in reinforcement learning," dans *2019 IEEE Conference on Games (CoG)*. IEEE, 2019, p. 1–4.

[78] G. Brockman *et al.*, "Openai gym," 2016.

[79] K. Cobbe *et al.*, "Quantifying generalization in reinforcement learning," *arXiv preprint arXiv :1812.02341*, 2018.

[80] M. Leshno *et al.*, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function," *Neural networks*, vol. 6, n°. 6, p. 861–867, 1993.

[81] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, n°. 4, p. 303–314, 1989.

[82] R. Pascanu, T. Mikolov et Y. Bengio, "On the difficulty of training recurrent neural networks," dans *International conference on machine learning*, 2013, p. 1310–1318.

[83] S. Hochreiter et J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, n°. 8, p. 1735–1780, 1997.

[84] I. Sutskever, O. Vinyals et Q. V. Le, "Sequence to sequence learning with neural networks," dans *Advances in neural information processing systems*, 2014, p. 3104–3112.

[85] D. P. Kingma et J. Ba, "Adam : A method for stochastic optimization," *arXiv preprint arXiv :1412.6980*, 2014.

[86] L. Prechelt, "Early stopping-but when ?" dans *Neural Networks : Tricks of the trade*. Springer, 1998, p. 55–69.

[87] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[88] C. J. Watkins et P. Dayan, "Q-learning," *Machine learning*, vol. 8, n°. 3-4, p. 279–292, 1992.

[89] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv :1509.02971*, 2015.

[90] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," dans *International conference on machine learning*, 2016, p. 1928–1937.

[91] J. Schulman *et al.*, "Trust region policy optimization," dans *International conference on machine learning*, 2015, p. 1889–1897.

[92] S. Kullback et R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, n°. 1, p. 79–86, 1951.

[93] S. Boyd, S. P. Boyd et L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[94] Wikipedia contributors, "Conjugate gradient method — Wikipedia, the free encyclopedia," 2020, [Online ; accessed 17-June-2020]. [En ligne]. Disponible : https://en.wikipedia.org/w/index.php?title=Conjugate_gradient_method&oldid=962856150

[95] Y. Wu *et al.*, "Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation," dans *Advances in neural information processing systems*, 2017, p. 5279–5288.

[96] J. Martens et R. Grosse, "Optimizing neural networks with kronecker-factored approximate curvature," dans *International conference on machine learning*, 2015, p. 2408–2417.

[97] A. Hill *et al.*, "Stable baselines," https://github.com/hill-a/stable-baselines, 2018.

[98] R. Lu et S. H. Hong, "Incentive-based demand response for smart grid with reinforcement learning and deep neural network," *Applied energy*, vol. 236, p. 937–949, 2019.

[99] A. Ouammi, H. Dagdougui et R. Sacile, "Optimal control of power flows and energy local storages in a network of microgrids modeled as a system of systems," *IEEE Transactions on Control Systems Technology*, vol. 23, n°. 1, p. 128–138, 2014.

[100] C. Bersani *et al.*, “Distributed robust control of the power flows in a team of cooperating microgrids,” *IEEE Transactions on Control Systems Technology*, vol. 25, n°. 4, p. 1473–1479, 2016.

[101] H. Dagdougui, A. Ouammi et R. Sacile, “Optimal control of a network of power microgrids using the pontryagin’s minimum principle,” *IEEE Transactions on Control Systems Technology*, vol. 22, n°. 5, p. 1942–1948, 2014.

[102] H. Dagdougui et R. Sacile, “Decentralized control of the power flows in a network of smart microgrids modeled as a team of cooperative agents,” *IEEE Transactions on Control Systems Technology*, vol. 22, n°. 2, p. 510–519, 2013.

[103] J. Moon *et al.*, “Combination of short-term load forecasting models based on a stacking ensemble approach,” *Energy and Buildings*, p. 109921, 2020.

[104] J. Kim *et al.*, “Recurrent inception convolution neural network for multi short-term load forecasting,” *Energy and Buildings*, vol. 194, p. 328–341, 2019.

[105] J. Ponoćko et J. V. Milanović, “Forecasting demand flexibility of aggregated residential load using smart meter data,” *IEEE Transactions on Power Systems*, vol. 33, n°. 5, p. 5446–5455, 2018.

[106] S. B. Taieb *et al.*, “A review and comparison of strategies for multi-step ahead time series forecasting based on the nn5 forecasting competition,” *Expert systems with applications*, vol. 39, n°. 8, p. 7067–7083, 2012.

[107] A. Motamedi, H. Zareipour et W. D. Rosehart, “Electricity price and demand forecasting in smart grids,” *IEEE Transactions on Smart Grid*, vol. 3, n°. 2, p. 664–674, 2012.

[108] M. Askari et F. Keynia, “Mid-term electricity load forecasting by a new composite method based on optimal learning mlp algorithm,” *IET Generation, Transmission & Distribution*, vol. 14, n°. 5, p. 845–852, 2020.

[109] O. Vinyals *et al.*, “Show and tell : A neural image caption generator,” dans *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, p. 3156–3164.

[110] A. Gasparin, S. Lukovic et C. Alippi, “Deep learning for time series forecasting : The electric load case,” *arXiv preprint arXiv :1907.09207*, 2019.

[111] K. Bandara *et al.*, “Sales demand forecast in e-commerce using a long short-term memory neural network methodology,” *arXiv preprint arXiv :1901.04028*, 2019.

[112] C. Heghedus, A. Chakravorty et C. Rong, “Energy load forecasting using deep learning,” dans *2018 IEEE International Conference on Energy Internet (ICEI)*. IEEE, 2018, p. 146–151.

[113] N. R. Canada, “Energy efficiency trends in canada–1990 to 2013,” 2013.

[114] R. E. Hedegaard et S. Petersen, “Evaluation of grey-box model parameter estimates intended for thermal characterization of buildings,” *Energy Procedia*, vol. 132, p. 982 – 987, 2017, 11th Nordic Symposium on Building Physics, NSB2017, 11-14 June 2017, Trondheim, Norway. [En ligne]. Disponible : http://www.sciencedirect.com/science/article/pii/S1876610217348397

[115] X. Li et J. Wen, “Review of building energy modeling for control and operation,” *Renewable and Sustainable Energy Reviews*, vol. 37, p. 517 – 537, 2014.

[116] R. Bellman, “Dynamic programming,” *Science*, vol. 153, nº. 3731, p. 34–37, 1966.

[117] A. Marinescu *et al.*, “Residential electrical demand forecasting in very small scale : An evaluation of forecasting methods,” dans *2013 2nd International Workshop on Software Engineering Challenges for the Smart Grid (SE4SG)*. IEEE, 2013, p. 25–32.