HAL open science

Statistical learning methods combining the Bayesian approach and deep learning

Honorine Royer

▶ To cite this version:

Honorine Royer. Statistical learning methods combining the Bayesian approach and deep learning. General Mathematics [math.GM]. Nantes Université, 2022. English. NNT : 2022NANU4056 . tel-03876730v2

HAL Id: tel-03876730

https://theses.hal.science/tel-03876730v2

Submitted on 27 Mar 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE DE DOCTORAT DE

NANTES UNIVERSITÉ

ÉCOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Mathématiques et leurs Interactions*

Par
Honorine ROYER

Méthodes d'apprentissage statistique mêlant approche bayésienne et deep learning

Thèse présentée et soutenue à Laboratoire de Mathématiques Jean Leray, le 03/11/2022
Unité de recherche : Laboratoire de Mathématiques Jean Leray (LMJL)

Rapporteurs avant soutenance :

Badih GHATTAS	Professeur d'université - Université d'Aix Marseille
Nicolas W. HENTGARTNER	Professeur - Los Alamos National Laboratory

Composition du Jury :
Attention, en cas d'absence d'un des membres du Jury le jour de la soutenance, la composition du jury doit être revue pour s'assurer qu'elle est conforme et devra être répercutée sur la couverture de thèse

Président :	Bertrand MICHEL	Professeur d'université - ECN
Examinateurs :	Badih GHATTAS	Professeur d'université - Université d'Aix Marseille
	Sophie ANCELET	Chercheur - IRSN
	Yannig GOUDE	PAST - Université Paris Saclay et Chercheur senior - EDF R&D
	Jairo CUGLIARI	MC - Université Lyon Lumière 2
Dir. de thèse :	Anne PHILIPPE	Professeur d'université - LMJL, Université de Nantes

Invité(s) :

Laurent BOZZI	Ingénieur de recherche - EDF R&D
Philippe CHARPENTIER	Ingénieur de recherche - EDF R&D

Acknowledgement

"There is nothing like looking, if you want to find something. You certainly usually find something, if you look, but it is not always quite the something you were after."

J. R. R. Tolkien, **The Hobbit**

Je voudrais tout d'abord remercier mes encadrants Laurent Bozzi, Philippe Charpentier et Anne Philippe. Un merci tout particulier à Anne. C'est grâce à tes conseils avisés que ce manuscrit a pris forme.

Je tiens ensuite à remercier à Badih Ghattas et Nicolas Hentgartner d'avoir accepté d'être rapporteurs de cette thèse. J'ai pu, grâce à vos remarques enrichir mon manuscrit. Je souhaites également remercier Sophie Ancelet, Jairo Cugliari, Yannig Goude et Bertrand Michel d'avoir accepté d'en faire partie de mon jury de thèse.

Merci beaucoup à toutes les personnes qui m'ont aiguillé, lors de toutes mes démarches administratives chez EDF : Sylvie, France, Nathalie et Danielle, mais aussi au labo : Stéphanie, Anais, Béatrice et évidemment Brigitte.

J'ai une pensée pour tous mes anciens collègues du département Séquoia de la R&D d'EDF. En particulier, merci à Valérie, Claire, Kafia, Sylvie, Pierre C, Blandine, Salvatore, Adelaïde, Jérémy, Benjamin, Clément, Alexandra, Pierre S, Alain, Bruno, Bérénice, Manel et j'en oublie beaucoup d'autres. Merci pour votre accueil et votre bienveillance. Je remercie également mes anciens cobureau pour leur accueil à chacune de mes venues au LMJL. Merci à vous Claire, Arthur et Jean-Michel !

Je tiens également à remercier profondément mes professeurs qui m'ont guidée sur le chemin de la statistique. J'ai une pensée particulière pour Hermine Biermé qui m'a fait aimer cette discipline, mais aussi pour El Mostapha Qannari et Evelyne Vigneau auprès de qui j'ai beaucoup appris.

La thèse est un long chemin, pas toujours tranquille mais tout à fait passionnant. Sur un plan plus personnel je souhaite remercier les personnes qui m'ont suivie et soutenue

depuis le début : Annick, Hynd, Hakima, François, Brigitte, Bernard, Lise, Mos et Mourad. Merci à vous tous pour votre gentillesse et vos conseils toujours bienvenus.

Mes remerciements vont évidemment à tous mes amis chers : Pierre, Thaïs, Léa, Max, Ayoub, Isa, Emily et tous les autres, vous vous reconnaîtrez. Merci de votre soutien indéfectible et d'avoir été la quand j'avais besoin de souffler. Merci à Fufu, Popo et Cricri, pour tous les énigmes que nous avons résolues ensemble et toutes les prochaines qui nous attendent !

Merci à Gilles et Berengere pour leur soutien et leur bienveillance, ils m'ont été d'une grande aide. Je tiens évidemment à remercier mes parents qui m'ont donnée le goût des sciences. Vos encouragements et votre confiance me suivent à chaque instant. Merci à mon grand frère et à Nathou, Bennozh Doue deoc'h !

Thibaud, merci pour ta grande confiance, ta générosité d'esprit et ta sérénité. Cette aventure n'aurait pas été la même sans ta présence, ton soutien inconditionnel et ta sagesse au quotidien. J'en suis profondément reconnaissante et je te remercie du fond du coeur pour tout.

Table of Contents

Acknowledgement **3**

Résumé **9**

1 Introduction **17**

1.1 Context and industrial stakes 17

1.1.1 Individual consumption data 18

1.2 Modeling approaches 19

1.3 Deep learning methods and neural networks 19

1.3.1 History on deep learning and neural networks 19

1.3.2 Deep learning for supervised learning 20

1.3.3 Unsupervised deep learning 21

1.3.4 Deep learning for time series analysis 22

1.4 Bayesian methods 24

1.4.1 Bayesian statistics 24

1.4.2 Bayesian neural networks 25

1.4.3 Posterior approximation 26

1.4.4 Issues in Bayesian deep learning 27

1.5 Thesis organization 27

1.5.1 Methodology of estimation of a multi-target regression model in high dimension 28

1.5.2 Application of the methodology of estimation of a multi-target regression problem: the case of predicting individual load curves of non residential customers 28

1.5.3 Bayesian transfer learning for panel data 29

1.5.4 Application of Bayesian transfer for panel data: end of month electrical consumption forecasting for residential customers 30

1.5.5 Classifying new customers into homogeneous groups with deep learning 31

2 Methodology of estimation of a multi-target regression model in high dimension 33

2.1 Introduction 33

2.2 Notations 35

2.3 Methodology for estimating a multi-target regression model 36

2.3.1 Multitarget nonlinear regression in high dimension (MNR) 36

2.3.2 Encoding, nonlinear regression and reconstruction(ENR-R) 38

2.4 Prediction under constraint 46

2.4.1 Prediction in a catalog 46

2.4.2 Encoding and nonlinear regression (ENR) 47

2.5 Probabilistic prediction intervals using the Bayesian posterior predictive distribution 47

2.6 Conclusion 52

3 Application of the methodology of estimation of a multi-target regression problem: the case of predicting individual load curves of non residential customers 55

3.1 Introduction 55

3.2 Prediction of a load curve with industrial stakes 59

3.2.1 Solar loss function 59

3.2.2 Prediction under industrial constraint 61

3.2.3 Diversity of the catalog 63

3.2.4 Adaptation of the modeling frameworks to the industrial constraint 63

3.3 Application to industrial data 64

3.3.1 Dataset 64

3.3.2 Transfer Learning - fine-tuning 66

3.3.3 Dimensionality reduction 68

3.3.4 MNR framework 72

3.3.5 Comparison of the two methods of prediction in the ENR-R and ENR frameworks 73

3.4 Probabilistic prediction intervals using the Bayesian posterior predictive distribution : numerical results 75

3.5 Conclusion and future research 79

4 Bayesian transfer learning for panel data 81
4.1 Introduction . . . 81
4.2 Methodology . . . 83
4.2.1 Weakly informative approach . . . 83
4.2.2 Informative approach . . . 86
4.2.3 Bayesian forecasting . . . 89
4.2.4 Summary of the methodology . . . 90
4.3 Application of the methodology to simulated data . . . 91
4.3.1 Polynomial regression . . . 91
4.3.2 Auto-regressive model . . . 101
4.3.3 Hierarchical Poisson model . . . 104
4.4 Conclusion and discussion . . . 107
4.A Polynomial regression . . . 107
4.A.1 Scenario n°3: some or all of the parameters vary, $\sigma = 4$ . . . 107
4.B Auto-regressive model . . . 107
4.C Hierarchical Poisson model . . . 107

5 Application of Bayesian transfer for panel data: end of month electrical consumption forecasting for residential customers 115
5.1 Introduction . . . 115
5.2 Description of the dataset . . . 116
5.3 Notations . . . 116
5.4 Application to end-of-month electrical consumption forecasting . . . 117
5.4.1 Weakly informative approach . . . 118
5.4.2 Informative approach . . . 120
5.4.3 Forecasting the end-of-month electrical consumption of individual customers . . . 121
5.4.4 Results . . . 122
5.5 Conclusion . . . 127

6 Classifying new customers into homogeneous groups with deep learning 131
6.1 Introduction . . . 131
6.2 Notations . . . 133
6.3 Dimensionality reduction . . . 134
6.3.1 Dense layers . . . 135

6.3.2 1D convolutions layers 135
6.4 Clustering 136
6.4.1 Hierarchical agglomerative clustering (HAC) 136
6.4.2 K-Means and K-Medoids clustering 137
6.4.3 Selection of the number of cluster 138
6.5 Affecting a new customer to the clusters 140
6.6 Application to real data 143
6.6.1 Description of the dataset 143
6.6.2 Evaluation error for dimensionality reduction 145
6.6.3 Selection of the number of clusters and vizualisation 146
6.6.4 Quality of the affectation to the clusters 150
6.7 Weighting predictions using the softmax scores 156
6.8 Conclusion and perspectives 159

Conclusion **165**

Bibliography **169**

Résumé

Contexte et enjeux industriels

Les fournisseurs d'énergie comme EDF font face à de nombreux nouveaux défis du fait des nombreuses transformations qu'a connu le marché de l'électricité français ces dernières années.

D'une part, en accord avec les régulations européennes concernant le marché de l'énergie, le marché de l'électricité français est ouvert à la concurrence depuis 2007. Cette ouverture concerne à la fois les clients résidentiels et non résidentiels. Cela a conduit à l'apparition de fournisseurs d'énergie alternatifs qui se sont positionnés sur ce marché et ont lancé de nombreuses offres de fourniture et des services concurrentiels. Cette concurrence amène de nouveaux défis pour les fournisseurs historiques en Europe tels qu'EDF.

D'autre part, EDF se doit d'innover au vu des transformations sociétales récentes liées au déploiement des compteurs communicants Linky, à l'essort des véhicules électriques, au besoin croissant de décarboner les consommations énergétiques ainsi qu'aux transformations digitales notamment liées à la relation client.

Avec plus de 35 millions de clients, à la fois particuliers et non résidentiels (industries, collectivités, etc.), EDF est le premier fournisseur d'électricité en France. Pour toutes ces catégories de clients, EDF souhaite fournir et développer de nouvelles offres de marché innovantes et des services en lien avec la fourniture d'énergie. Pour ce faire, EDF développe des services individualisés pour les besoins de ses clients à l'aide de leurs données et de méthodes d'apprentissage statistique.

Cependant, dans un contexte concurrentiel accru, où EDF souhaite conserver ses clients historiques et gagner de nouvelles part de marché (par exemple pour des offres d'autoconsommation ou des offres de fourniture pour les véhicules électriques), le besoin de développer des modèles statistiques performants est croissant.

Ces dernières années, il y a eu un essort de l'usage des méthodes d'apprentissage profond et d'apprentissage statistique. Le principal écueil de ces modèles statistiques repose sur le besoin de large volumes de données en vue de leur apprentissage. Cela pose problème pour de nombreuses raisons, parmi lesquelles :

— Pour de nouveaux clients, l'historique de données disponibles est souvent très court, voire inexistant. Cela peut poser problème lors de l'apprentissage de modèles statistiques du fait du manque d'information disponible.

— Les entreprises ayant recourt à la collecte de données à caractère personnel doivent être en conformité avec le Réglement Général sur la Protection des Données (RGPD, pour plus d'informations, se référer à European Commission (2016)). Cela a un impact sur la collecte et la durée de conservation de données à caractère personnel à plusieurs égards. Par exemple, un client peut consentir à la collecte de ses données de courbes de charge soit au pas demi-horaire soit au pas journalier. Cela influe sur la volumétrie des données collectées. Il est aussi important de noter que les données de consommation collectées ont une durée de conservation limitée. Un non respect du RGPD est passible de poursuites judiciaires.

— Du fait de sa position en tant qu'opérateur historique sur les marché de l'électricité, EDF n'a pas le droit d'utiliser les données de ses clients au Tarif Réglementé de Vente (offre historique de fourniture) pour construire des offres de marché. Cela constituerait un abus de position dominante et est aussi passible de poursuites judiciaires.

— Certains comportements de consommation d'électricité restent sous-représentés parmi les clients français: par exemple, la possession d'un climatiseur reste mineure.

— Il existe d'autres situations dans lesquelles un manque de données 'client' peut apparaître: c'est le cas notamment lors de l'emménagement d'un client. En effet, la collecte des données n'est pas immédiate et est effective plusieurs jours après l'emménagemen, ce qui complique l'usage éventuel de l'historique de données.

Pour toutes ces raisons, l'approche bayésienne semble être une alternative intéressante pour pallier le manque de données. Le paradigme bayésien établit que lors de l'apprentissage d'un modèle statistique, toute l'information ne provient pas des données à dispostion. La connaissance a priori, obtenue indépendamment des observations, est intégrée au modèle au travers de l'usage de lois de probabilités sur les paramètres du modèle. L'information a priori résume l'information à disposition avant apprentissage du modèle sur les observations. L'estimation repose sur la loi a posteriori i.e. la loi des paramètres sachant les observations. Au travers de cette loi a posteriori, il est possible de récupérer des estimations ponctuelles, des intervalles de crédibilité et les lois prédictives a posteriori de nouvelles observations.

Dans cette thèse, nous nous concentrons sur les méthodes d'apprentissage statistique combinant approche bayésienne et apprentissage profond dans le but d'estimer et de prédire les consommations individuelles de clients ayant une petite volumétrie de données disponible. L'idée essentielle est d'utiliser la connaissance issue de clients avec un plus large volume de données et de la transférer à de nouveaux clients en situation d'historique court.

Données de consommation individuelle

Dans ce manuscrit, nous utilisons principalement des données comme décrites ci-dessous :

— Les courbes de charges individuelles : les séries temporelles de consommation, collectées au pas demi-horaire ou journalier.

— Les informations de facturation : par exemple, les ratios de consommations heures pleines heures creuses pour les clients non résidentiels

— Les caractéristiques du foyer des clients résidentiels (e.g. la possession d'un système de chauffage électrique)

— La température collectée à la station météolorogique la plus proche des clients considérés, aux même dates et pas de temps que les courbes de charge.

Modélisation

On note X la courbe de charge individuelle pour un client donné et V les variables explicatives disponibles. La plupart des services liés à la consommation fournis par EDF reposent sur l'estimation de la courbe de charge individuelle X à l'aide des variables V et d'une modèle de régression de ce type :

$$X = f(V)$$

Dans cette thèse, nous présentons des méthodologies fondées sur l'utilisation de méthodes bayésienne combinées à du deep learning pour estimer la fonction de régression f.

Les méthodologies développées sont appliquées à des données réelles afin de répondre à des probématiques industrielles spécifiques pour EDF.

Dans ce manuscrit, nous mélons l'approche bayésienne et le deep learning de deux façons:

— Au travers de l'usage de réseaux de neurones bayésiens, qui sont des réseaux de neurones pour lesquels des lois a priori sont placées sur les poids des réseaux.

— Dans une modélisation en deux parties, où le deep learning est utilisé d'abord pour la réduction de dimension avant clustering. Puis la modélisation bayésienne est utilisée sur les clusters construits (cette fois-ci avec des modèles qui ne sont pas des réseaux de neurones).

Organisation du manuscrit

Nous présentons ci-dessous brièvement les différents chapitres constituant ce manuscrit. Le dernier chapitre de cette thèse est dédié à la conclusion générale de ces travaux et aux perspectives pouvant en résulter.

Chapitre 2 : Méthodologie d'estimation d'un modèle de régression à plusieurs sorties en grande dimension

Dans le Chapitre 2, nous présentons une méthodologie pour l'estimation d'un modèle de régression à plusieurs sorties dit "multi-target" en grande dimension. Cette méthodologie est appliquée au Chapitre 3 à la problématique d'estimation et de prédiction de courbes de charges électriques annuelles au pas demi-horaire de clients non résidentiels.

Trois stratégies de modélisation sont décrites, dont deux reposent sur la réduction de dimension par un modèle de deep learning. La première stratégie repose sur l'estimation directe du modèle de régression "multi-target" en grande dimension. Pour la deuxième stratégie, après réduction de la dimension, le modèle est estimé dans l'espace en dimension réduite (aussi appelé espace latent). La troisième stratégie, quant à elle, diffère de la deuxième parce qu'elle a recourt à de la reconstruction. Si la méthode de réduction de la dimension le permet, après estimation du modèle dans l'espace latent, la reconstruction permet d'obtenir une estimation dans l'espace en grande dimension initial.

Selon la stratégie considérée, le modèle de régression "multi-target" est estimé soit avec des réseaux de neurones profonds, des réseaux de neurones bayésiens ou des "deep Gaussian processes".

Nous décrivons deux façons de prédire la courbe d'un nouveau client, après estimation du modèle de régression. Les modèles bayésiens, que sont que les réseaux de neurones bayésiens et les "deep Gaussian processes", permettent de construire des intervalles de

prédictions à partir de la loi a posteriori. Les intervalles de prédiction sont construits suivant les deux méthodes de prédiction décrites.

Chapitre 3 : Application de la méthodologie pour l'estimation d'un modèle de régression multi-target regression : cas de la prédiction de courbes de charge individuelles de clients non residentiels

Le Chapitre 3 est dédié à l'application de la méthodologie présentée au Chapitre 2.

Le cas d'usage industriel est lié à la prédiction de courbes de charge individuelles de clients non résidentiels. Pour ce faire, nous souhaitons n'utiliser que les informations de facturation des clients à disposition pour estimer et prédire les séries temporelles de consommations de nouveaux clients.

Les trois stratégies de modélisation présentées dans le Chapitre 2 sont adaptées ici à des données réelles.

Le jeu de données considéré contient deux sous catégories de clients non résidentiels. Elles consistent en des PME et de grands sites industriels. Les PME sont sous-représentées dans le jeu de données. Les deux sous-populations partagent à la fois des similarités en termes de forme de courbe de charge, mais des différences subsistent notamment sur leurs plages d'heures creuses. Pour tenir compte à la fois des similarités et des différences, nous utilisons du "fine-tuning" pour l'apprentissage des modèles et évaluons son effet sur leurs performances.

Deux contraintes industrielles sont intégrées à cette étude :

— La courbe de charge prédite doit appartenir à un catalogue de courbes de charge de clients existants.

— Les consommations pendant les heures d'ensolleilemment sont mises en exergue, car cette application industrielle est liée à la problématique du dimensionnement d'installation photo-voltaïque de clients non résidentiels.

Les performances prédictives obtenues avec les trois stratégies sont comparées et nous regardons également les intervalles de prédiction issus des lois a posteriori des modèles bayésiens.

Chapitre 4 : Apprentissage par transfert bayésien pour des données de panel

Dans le Chapitre 4, nous introduisons une méthodologie d'apprentissage par transfert bayésien adaptée des travaux de Launay et al. (2015), pour le cas des données de panel.

Nous souhaitons transférer l'information à un nouvel individu en situation d'historique court, à partir de ce qui a été appris sur un panel comprenant plusieurs individus avec un historique de données long.

La méthodologie se décline en deux étapes. D'abord, un modèle hiérarchique bayésien est appris, avec des lois a priori faiblement informatives, sur des données de panel avec un historique long. Les hyperparamètres du modèle représentent le comportement global du panel. Ce sont les lois a posteriori de ces hyperparamètres qui nous intéressent, car nous souhaitons transférer l'information contenue dans leurs lois a posteriori à un modèle adapté pour un nouvel individu. Plus précisément, nous récupérons les moyennes et matrices de variance-covariance a posterori et les intégrons dans les lois a priori du modèle bayésien du nouvel individu.

Cette approche informative constitue la seconde étape de la méthodologie. Des lois normales sont utilisées comme loi a priori pour les paramètres du nouveau modèle, et l'information est transférée via les moyennes et covariance de ces lois a priori. Des hyperparamètres modélisant la similarité entre le nouvel individu et le panel initial sont intégrées aux lois a priori. Ces hyperparamètres permettent de tenir compte de la similarité ou des différences éventuelles entre le nouvel individu et le panel.

La méthodologie est évaluée sur des données simulées, issues de Launay et al. (2015). Nous appliquons la méthodologie à trois situations : la régression polynomiale, le modèle autorégressif et le modèle de Poisson hiérarchique.

Chapitre 5 : Application de l'apprentissage par transfert bayésien à la prévision de consommation à la fin du mois pour des clients residentiels

Le Chapitre 5 est dédié à l'application de l'apprentissage par transfert bayésien, décrite au Chapitre 4, à des données réelles.

Nous traitons un cas d'usage industriel sur la prévision de la consommation à la fin du mois pour des clients résidentiels. Dans ce chapitre, la méthodologie et les modèles sont

testés sur un sous ensemble du jeux de données issu de la CER Commission for Energy Regulation (2012). La construction de ce sous ensemble fait l'objet du Chapitre 6.

Le but est de transférer l'information apprise sur des clients avec un long historique de consommation à des clients avec un historique court.

Pour l'approche faiblement informative, nous estimons deux modèles sur un panel de clients, dont on dispose pour chacun d'une courbe de charge de longueur 365 jours. Les variables explicatives intégrées aux deux modèles sont : la température extérieure enregistrée au jour t, $t = 1, \ldots, T$, une variable binaire indiquant la présence d'un chauffage électrique pour l'individu i du panel, une variable binaire indiquant la présence d'un système d'eau chaude sanitaire électrique pour le foyer i, $i = 1, \ldots, n$. Le second modèle contient une variable supplémentaire : la consommation individuelle du jour précédent $X_{t-1}^{(i)}$. Les deux modèles ont des hyperparamètres suivant des lois a priori faiblement informatives. Après inférence des modèles, on extrait les moyennes et covariances a posteriori.

Pour l'approche informative, on considère des modèles individuels, contenant les mêmes variables explicatives, et on transfère l'information des moyennes et covariances a posteriori dans les loi a priori normales comme dans le Chapitre 4.

Les modèles sont appris pour l'approche informative en faisant varier la longueur de l'historique des données des nouveaux clients. Nous obtenons des prévisions pour chaque consommation à chaque pas de temps de l'horizon considéré. Les prévisions journalières sont alors sommées pour obtenir une prévision pour la consommation à la fin du mois.

Nous évaluons les performances des modèles en comparant l'approche informative et son equivalent non informatif (modèle sans transfert). Plusieurs indicateurs sont évalués : l'erreur entre l'estimation ponctuelle et la vraie valeur de consommation à la fin du mois, la longueur des intervalles de prédiction et la probabilité que la vraie valeur soit contenue dans l'intervalle.

Chapitre 6 : Classification de nouveaux clients dans des groupes homogènes avec du deep learning

Dans le Chapitre 5, nous décrivons une application du transfer learning bayésien à la prévision de consommation à la fin du mois de clients résidentiels. Nous appliquons la méthodologie à un sous ensemble du jeu de donnée issu de Commission for Energy Regulation (2012). Dans le Chapitre 6, nous décrivons la méthodologie développée pour construire le sous groupe de clients utilisé au Chapitre 5.

Plusieurs étapes constituent cette méthodologie.

D'abord, des autoencoder profonds sont utilisés pour réduire la dimension de courbes de charge annuelles. Deux types de couches sont considérées : les couches Dense et les couches 1D-convolutives. Nous comparons les performances de ces deux types de couches pour les données considérées en termes d'erreur de reconstruction.

Ensuite, des méthodes de clustering sont appliquées à l'espace latent issu du meilleur autoencoder. Trois méthodes sont comparées : K-means, la classification ascendante hiérarchique et l'algorithme des K-médoïdes. Le nombre de clusters optimal est choisi à l'aide de différents critères de sélection. La méthode de choix semble être les K-médoïdes de par sa robustesse aux individus atypiques.

Puis, après construction des classes, nous présentons un modèle de réseau de neurones à double entrée permettant d'affecter un nouvel individu à une de ces classes. La qualité d'affectation est évaluée selon plusieurs métriques

Enfin, nous adaptons la méthodologie de prévision de consommation à la fin du mois du Chapitre 5 pour pallier le problème des individus mal classés dans les clusters. Nous utilisons les scores de la fonction softmax en sortie du réseau d'affectation pour construire des probabilités d'appartenance aux différentes classes. Ces probabilités servent à pondérer les prédictions obtenues pour chaque cluster. On obtient alors un mélange de prédicteurs et les résultats pour la prévision à la fin du mois associés. Ces résultats sont comparés à ceux obtenus dans le Chapitre 5, lorsque la prévision est réalisée à cluster fixé.

Chapter 1

Introduction

1.1 Context and industrial stakes

In the recent years, the French utility market has undergone several transformations, that bring new challenges for utility companies such as EDF.

On the one hand, in compliance with EU regulations on energy, the French electricity market is opened to competition. Both residential and non residential customers are concerned by this opening. This has lead alternative utility companies to emerge and offer new and competitive services and energy-supply offers. As competition grows, this brings out new challenges for historical utility companies like EDF.

On the other hand, societal transformations linked with the deployment of smart meters, electrical vehicles, a need to de-carbonize energy consumptions and digital transformations compel EDF to innovate.

With over 35 million customers, both residential and non residential (e.g. businesses or regional authorities), EDF is the lead energy provider in France. EDF wishes to provide innovative and competitive services to its various range of customers. To do so, EDF develops individual data-driven services tailored to the customer's needs.

However, in a context of increased competition, where the leader wishes to retain historical customers and gain new customers (for instance, with solar panel sizing offers, or electric vehicle supply offers), the need to develop efficient statistical models is growing.

In recent years, there has been an expanding interest in machine learning and deep learning-based methods. The issue being the large volumetry of data required for the training of such statistical models. This is a challenging problem for the following reasons:

— For new customers, the volumetry of data is rather small, lacking sometimes depending on the type of data needed. This complicates the use of statistical learning models.

— Companies collecting personal data need to comply with the General Data Pro-

tection Regulation (GDPR, see European Commission (2016)). This impacts the collection and conservation of personal data in several ways. For instance The customer can either consent to have its consumption data collected at half-hourly period or at daily period. This has an impact on the amount of data collected. It is also worthy to note that the consumption data collected has a duration limitation of conservation. Failing to comply is liable to prosecution.

— Because of its historical position as a leader in energy supply, EDF must not use the data of its historical customers to build new supply offers. This would constitute an abuse of a dominant position and is liable to prosecution.

— Some electrical consumption behaviors remain underrepresented among French customers: for instance, the possession of an air conditioning is still minor.

— Other situations may involve a lack of customer data: when a customer moves in a new location, it takes several days to collect data, making it challenging to exploit its historical data.

Because of these reasons, the Bayesian approach appears to be an interesting alternative to cope with the lack of data. The Bayesian paradigm postulates that the information for training a statistical model does not come solely from the observations available. Prior information independent from the observations is integrated to the model through the use of probability distribution set on the parameters. The prior information summarizes the information available prior to training the model using the data at hand. The estimation relies on the posterior distribution i.e. the distribution of the parameters given the observations. Through the posterior distribution, we can get point estimates, as well as credibility intervals and the posterior predictive distribution of new data points.

In this thesis, we focus on mixing Bayesian methods and deep learning to estimate and forecast the individual consumption of customers with a small volumetry of data. The main idea is to use information available on customers with longer historical data and transferring it to new customers with shorter historical data.

1.1.1 Individual consumption data

In this manuscript, we use mostly the following types of individual customer data:

— Individual load curves: the consumption time series at half-hourly period or daily period, depending on the data collected.

— Billing information: for instance peak and off peaks hours consumption ratios for non residential customers.

— Information relative to the customers' household and behavior for residential customers (e.g. the possession of electrical heating systems).

— The temperature collected, near the customers considered, at the same timestep as the load curves.

1.2 Modeling approaches

Let X denote the individual load curve of any customer and V the available explanatory variable. Most consumption based services provided by EDF rely on estimating the individual load curve from the variable, using a regression model as such:

$$X = f(V) \tag{1.1}$$

In this thesis, we present methodologies based on Bayesian methods combined with deep learning to estimate the regression function f.

The methodologies developed are applied to real world data for specific industrial applications for EDF.

The Bayesian approach is mixed with deep learning in the following ways:

— Through the use of Bayesian neural networks, which are neural networks with prior distribution set on their weights.

— Being a two step approach, deep learning is used for dimensionality reduction prior to clustering and Bayesian models (not neural networks) are used on the ensuing clusters.

1.3 Deep learning methods and neural networks

1.3.1 History on deep learning and neural networks

Neural networks, in their earliest form, have been introduced by McCulloch and Pitts (1943). They are constituted of artificial neurons that are mathematical representations of a biological neurons. They paved the way for the perceptron of Rosenblatt (1958), a

linear acyclic neural network, with only one output and fully connected inputs, designed for pattern recognition.

The perceptron, however, presents a severe limitation: it is only applicable to linear problems, as highlighted by Minsky and Papert (1969).

The Hopfield network, introduced by Hopfield (1982), allowed for a renewal of interest for neural networks. Rumelhart et al. (1986) bring out the use of the gradient backpropagation algorithm for training of multilayer perceptrons, first introduced by Werbos (1974). This algorithm allows to learn nonlinear problems with neural networks.

At the same time, LeCun (1985) independently focuses on the backpropagation algorithm. The algorithm is applied to the case of image recognition by LeCun et al. (1989). Those are the premises of convolutional neural networks for computer vision, further developed for handwritten digit recognition in LeCun et al. (1998).

Hornik et al. (1989) demonstrate the capacities of multilayer neural networks as universal approximators of any continuous function.

Despite those advances, it remained difficult to train multilayered neural networks. In Hinton et al. (2006), the authors propose an algorithm to alleviate this issue. This is the beginning of deep learning methods.

With the advent of big data and the increase of computational power with GPUs, training deep learning models has been facilitated. They have notably been gaining traction since the paper of Krizhevsky et al. (2012), where convolutional neural networks have been applied succesfully for the ImageNet competition for image recognition.

Nowadays, deep neural networks are widely used in many domains of application, as we will see in the following sections.

1.3.2 Deep learning for supervised learning

Supervised learning relates to modeling approaches that map an input into its related output. Tasks such as regression, binary and multi-class classification are supervised learning tasks. Nowadays, deep neural networks are widely used to solve supervised learning tasks. Because of the flexibility of their architectures as well as their scalability to large datasets, they are well suited for numerous industrial applications.

Deep Neural networks are mostly applied to solve multiclass-classification problems. In image recognition for instance, Simonyan and Zisserman (2015) use a very deep convolutional approach to the classification of images. He et al. (2016) have developed the ResNet architecture for the purpose of image recognition. Graves et al. (2013) develop a

deep recurrent method for phoneme recognition. Richard et al. (2019) propose a convolutional DenseNet for the classification of time series. Fawaz et al. (2019) propose a survey of deep learning approaches for time series classification, among which fully convolutional neural networks and residual neural networks. Ziat (2017) propose a convolutional and a reccurent-convolutional approach to the task of classification of time series.

Merkel et al. (2018) apply a deep neural network regression framework to short term load forecasting of natural gas.

1.3.3 Unsupervised deep learning

Deep neural networks can also be considered for unsupervised learning tasks such as dimensionality reduction and clustering.

Dimensionality reduction methods based on deep learning can be conducted using autoencoders. Originally used as a denoising method for noisy data (see for instance LeCun (1987), Gallinari et al. (1987)), autoencoders are nowadays seen as the Principal Component Analysis deep learning counterpart. They are able, through the stacking of hidden layers, to learn and encode nonlinearities of the input data. They are constituted of two blocks: encoding and decoding. Encoding reduces the dimension of the inputs into a lower dimensional space, whereas decoding reconstructs from the encoded space the input into its original dimension. Encoding and decoding are built using multiple layers, that can be of different kinds such as Fully connected (Dense) layers, convolutional layers, recurrents layers etc. Goodfellow et al. (2016) give some examples of the variety of autoencoders architectures. They are effective and flexible tools for dimensionality reduction and feature representation of high dimensional data. Richard et al. (2020) apply a 1D convolutional autoencoder-based approach to dimensionality reduction of electrical load curves. In Kong et al. (2020b), a stacked autoencoder is used to reduce the dimension of electrical load curve. The Seq2Seq model, developed by Sutskever et al. (2014), applies a Recurrent encoder-decoder architecture to translation tasks.

Another type of neural network designed for unsupervised learning is Self-Organizing Maps (SOM), introduced by Kohonen (1995). When feeding input data to SOM, the algorithm assesses which of the randomly positioned neurons are closest to each data point. This is done iteratively until the position of the neurons of the network has stabilized. Because of this structure, SOM are able to learn nonlinear patterns in high dimension. They may be applied to reduce the dimension of highly dimensional input. This is the case for instance in Yu et al. (2015), where the authors adapt a SOM-based method for

dimensionality reduction in the case of risk-based fault detection. The main drawback of SOM is that they are computationally expensive.

Dimensionality reduction by means of deep models can also be achieved using Deep Belief Networks (DBN) (see Hinton (2009)). DBN are compositions of multiple hidden layers made of Restricted Boltzmann Machines (RBM). RBM, introduced by Smolensky (1986), are generative neural networks for unsupervised learning, that are able to learn the probability distribution of the input data. The stacking of RBM layers allows to build DBN. Arsa et al. (2016) propose a DBN model for reducing the dimension of hyperspectral image in the context of image classification.

Clustering from deep learning methods has also been subject to recent developments. In image recognition for instance, the FaceNet algorithm developed by Schroff et al. (2015), is a deep convolutional neural network for facial recognition and clustering. Aljalbout et al. (2018) introduced a convolutionnal autoencoder-based method for clustering. Min et al. (2018) provides a survey concerning deep learning approaches for clustering. Notably, many clustering approaches rely on autoencoders, variational autoencoders or generative adversarial networks. Suo et al. (2018) proposed a deep belief network and C-means clustering method for the analysis of brain genes. Wang et al. (2004) propose a SOM based approach for clustering time series. Fortuin et al. (2019) use a deep learning model combining a SOM and a Variational autoencoder for the purpose of clustering time series.

Some deep learning approaches combine dimensionality reduction models and clustering. For instance, Madiraju et al. (2018) propose an encoder-decoder based model for dimensionality reduction for temporal clustering, where the encoder contains 1D convolutions as well as LSTM layers. Similarly, Ma et al. (2019) use a Recurrent encoder-decoder architecture combined with K-Means clustering for time series data.

1.3.4 Deep learning for time series analysis

Time series analysis is a core topic of this manuscript as data applications in the energy sector make use of time series such as electrical and natural gas load curves, temperature recordings, electrical production etc. Time series data are also encountered in many areas outside of the energy sector. Examples include: in finance for the analysis and forecasting of stock market indexes, in economics for the analysis of real gross domestic products of countries, in logistics for the forecasting of road traffic, in ecology for the analysis of the quantity of chemicals in the atmosphere etc.

Since time series data are ubiquitous in various domains, many approaches have been

considered. For deep learning methods, classical architectures of neural networks such as the multilayer perceptron (see Hippert et al. (2001)), are not the most appropriate as they do not take the temporal dependence into account. Hence, specific architectures have been developed, in particular for forecasting tasks. Those neural networks are generally referred to as recurrent neural networks. Among those, the Hopfield neural networks, introduced by Hopfield (1982) are somehow a precursor of recent recurrent neural networks. In Hopfield networks, all the neurons are connected to each other and are characterized by their state. The state of a given neuron at time t depends of its state at the previous instant. Ideally, they converge towards a state similar to the initial state. Application of Hopfield network include time series prediction. For instance, Guang (2013) applies a Hopfield network coupled with wavelet decomposition to predict network traffic.

Recently, Long-Short Term Memory (LSTM) neural networks, introduced by Hochreiter and Schmidhuber (1997), have been successfully applied in the domain of speech recognition (see Graves et al. (2013)). This success is due to the structure of the units that constitute LSTM neural networks. Those units are names memory cell and allow to retain the previous information throughout the neural network. They also allow to forget this information, when it becomes necessary by means of "forget gates" in the memory cells. LSTM neural networks have also been applied to short term individual residential load forecasting (see Kong et al. (2017) and Kong et al. (2019)).

Other recent advances on time series analysis using recurrent neural networks include Gated Recurrent Units (GRU) neural networks introduced by Cho et al. (2014). GRU neural networks can be seen as an alternative to LSTM neural networks, as they are designed in a similar manner. In fact, they are constituted of cells, similar to the memory cells of the LSTM, allowing to keep past information. The cells contain "reset gates" and "update gates", which enables the networks to store relevant information and filter the rest. Notably, GRU neural networks have been applied to the forecasting of short term photovoltaic production by Wang et al. (2018). Tao et al. (2019) apply a GRU-based convolutional neural network to the forecasting of air pollution.

Another architecture of recurrent neural network is the Echo State Network (ESN), introduced by Jaeger and Haas (2004). The ESN's singularity lies in their main hidden layer named the reservoir. The reservoir is a dynamic system where the neuron are sparsely connected to each other. This input of the ESN is connected to the reservoir in an aleatoric and unalterable way, whereas the output and the reservoir are connected to each other and the weights of those connections are learned and updated during the

training of the network. Because of this specific structure, the ESN is able to learn specific temporal patterns. ESN have been applied by Deihimi and Showkati (2012) to short term load forecasting of a country. Hu et al. (2020) apply a deep ESN to forecasting annual energy consumption and wind power generation.

Time series can also be modeled with non recurrent neural networks. Some methods couple classical approaches such as ARIMA with deep neural networks. Qin et al. (2017) propose a combined approach between ARIMA and deep belief networks to forecasting the occurrence of red tides. Triebe et al. (2019) introduce the AR-Net, a neural network that mimics an autoregressive process, for time series forecasting. Other approaches include the combination of the wavelet transform and convolutional neural networks to the forecasting of wind power generation. Lang et al. (2020) develop a 1D convolutional neural network to short term electricity load forecasting. Merkel et al. (2018) apply non recurrent deep neural networks to short term load forecasting of natural gas. In recent years, forecasting high-multidimensional time series has been the subject of numerous competitions such as the M4 and M5 competitions (see Makridakis et al. (2020) and Makridakis et al. (2022)). These events allowed for machine learning based methods and hybrid models to be applied to real world data.

1.4 Bayesian methods

1.4.1 Bayesian statistics

Let $X = (X_1, \ldots, X_n)$ denote a sample and $\mathcal{M} = \{P_\theta, \theta \in \Theta\}$ a parametric model.

Bayesian inference relies on the Bayesian paradigm: not all the information of a statistical model comes from the observations. Specifically, the parameter θ is seen as a random variable, that follows a certain probability distribution. This differs from the traditional frequentist statistical point of view where θ is deterministic.

The prior distribution of θ, denoted $\pi(\cdot)$, reflects the prior information available on the parameter before inference of the model with the data X.

The data X_i, $i = 1, \ldots, n$ are independent conditionally to the parameter θ.

Bayesian inference consists in finding the probability distribution of θ given the observations X. This distribution is named the posterior distribution. The posterior is expressed

using Bayes rule:

$$\pi(\theta|X) = \frac{\pi(\theta) f_X(X|\theta)}{\int_\Theta \pi(\theta) f_X(X|\theta) d\theta}, \tag{1.2}$$

where $f_X(X|\theta)$ is the likelihood of the model.

In practice, the denominator of Equation (1.2), does not have an analytical form. The marginal is estimated using Markov Chain Monte Carlo (MCMC) methods (see for instance Robert and Casella (2005).

For an extensive presentation on Bayesian methods, refer to Robert et al. (2007)

1.4.2 Bayesian neural networks

Bayesian deep learning is an area of deep learning that focuses on incorporating Bayesian modeling and inference to deep neural networks.

Since neural networks are parametric models, adopting a Bayesian approach involves putting prior distribution on the parameters and inference relies on finding the posterior distribution.

The combination of Bayesian methods and neural networks can be found in literature as early as the 1990's Neal (1996) and MacKay et al. (1994). Those methods have gained renewed interest in the mid 2010's, some focusing on dropout (Gal and Ghahramani (2016), Gal et al. (2017), Kendall et al. (2017), Nalisnick et al. (2018)), others on learning methods for posterior inference (Blundell et al. (2015), Hernàndez-Lobato and Adams (2015), Kingma and Welling (2014), Kingma et al. (2015), Wen et al. (2018)). Other aspects of Bayesian neural nets (BNN) are explored such as prior distributions (Nalisnick ((2018), Vladimirova et al. (2019)), and applications of BNN (Kendall and Gal (2017), Polson et al. (2017)).

Dropout, a regularization technique, has been used to provide uncertainty estimates with neural networks' predictions by Gal and Ghahramani (2016). They provide theoretical results, showing that the use of dropout in a neural network provides a Bayesian approximation equivalent to a Gaussian process.

For more details on Bayesian deep learning and inference, see Gal (2016).

Bayesian analysis applied to neural networks has been developed at first for "shallow" neural networks (with one hidden layer). This is notably the case in MacKay (1992) and Neal (1995). In a Bayesian setup, the neural networks parameters follow some prior distribution and the goal is to infer the posterior distribution of those parameters.

In Neal (1995), the author proposes the Hamiltonian Monte Carlo method to perform inference of the posterior distribution of the parameters.

Bayesian neural networks, compared to their non Bayesian counterparts, offer several advantages. Specifically, some drawbacks of regular neural networks include:

— The need of a large sample size for training limits their application to big databases. On small dataset, neural networks tend to overfit.

— Biases inherently present in databases are not taken into account during training of neural networks. This has an impact on predictive performances.

— The lack of interpretability of neural network is also a major drawback for many industrial applications.

— Neural networks do not naturally model uncertainty. This complicates the evaluation of the level of confidence given to the predictions.

Bayesian neural networks naturally provide uncertainty estimates associated with their predictions, making them an interesting alternative to regular neural networks.

1.4.3 Posterior approximation

As in non deep learning Bayesian methods, inference relies on finding the posterior distribution of the parameters given the data. This is a challenging task, as the marginal is usually intractable. Yet, the posterior distribution is necessary to obtain credible intervals and predicting new data points.

Methods to find the posterior distribution in Bayesian neural networks are an active subject of research. Due to the computational cost of Bayesian inference, methods based on variational inference are often favored over MCMC based methods such as Hamiltonian Monte Carlo.

Variational based methods are notably used in Blundell et al. (2015), where the proposed learning algorithm (Bayes by Backprop) of Bayesian neural network relies on minimization of the evidence lower bound.

Variational inference has the advantage of being scalable to large datasets and high dimensional parameter spaces.

A further description of the method is given in 2.3.2. For more details on variational inference, see Blei et al. (2017).

1.4.4 Issues in Bayesian deep learning

In practice, several limitations remain when sampling from the posterior distribution:

— Bayesian methods are known to be computationally costly and take more tim to train than their non Bayesian counterparts.

— Traditional MCMC methods are very efficient since they are asymptotically exact. However, they lack scalability in high dimensional parameter spaces and when the sample size is large. This complicates their usage in deep learning settings.

— For the posterior approximation in Bayesian neural networks, methods of choice are usually based on variational inference. They are known to be less precise than MCMC methods.

— The Bayesian paradigm allows to integrate prior knowledge on the parameters before inference, through the prior distribution. This is a difficult task due to the nature of neural networks parameters.

1.5 Thesis organization

This Section is dedicated to the description of the main aspects covered during the thesis. We provide in the following sections a brief summary for each chapter of this manuscript. Section 1.5.1 introduces the deep learning-based methodology, presented in Chapter 2 to tackle multi-target regression problems. An application of the methodology to the estimation of annual load curves of non residential customers is developed in Chapter 3, this is summarized in Section 1.5.2. Section 1.5.3 describes the methodology further developed in Chapter 4, where a Bayesian approach for transfer learning of panel data is shown. In Section 1.5.4, we briefly discuss the subject of Chapter 5, where an application of this methodology to the case of end-of-month consumption forecasting of a subgroup of residential customers is developed. Section 1.5.5 summarizes the content of Chapter 6, containing a deep-learning approach for the construction of the subgroup of residential customers used in Chapter 5.

The last Chapter of this manuscript is dedicated to the general conclusion and perspectives of the work developed herein.

1.5.1 Methodology of estimation of a multi-target regression model in high dimension

In Chapter 2, we present a methodology of estimating a multi-target regression model in high dimension. The methodology is applied in Chapter 3 to the estimation and prediction of annual electrical load curves of non residential customers. Three strategies are detailed, two of which rely on dimensionality reduction by means of deep learning. The first strategy relies on the direct estimation of the model in high dimension. For the second strategy, after dimensionality reduction, the model is estimated in the lower dimensional space. The third strategy differs from the second because after the estimation in low dimension, if the method allows for reconstruction, we can get an estimation in the original dimensional space. Depending on the strategy considered, the multi-target regression model is estimated using deep neural networks, Bayesian neural networks or deep Gaussian processes.

Two possibilities are described to carry out the prediction of a new customer's curve, after the estimation of the regression model. The Bayesian methods allow the construction of prediction intervals ensuing from the posterior distribution. Again, there are two possibilities for the determination of those prediction intervals, related to the two described prediction methods.

1.5.2 Application of the methodology of estimation of a multi-target regression problem: the case of predicting individual load curves of non residential customers

Chapter 3 is dedicated to the application of the methodology presented in Chapter 2.

The industrial use-case relates to the prediction of individual annual load curves of non residential customers. The goal is to use only billing information of customers to estimate and predict the consumption time series of new customers.

The three strategies of estimation presented in Chapter 2 are adapted here to real world data.

The considered dataset comprises two subgroups of non residential customers. They consist in small and medium businesses (SMB) and large industrial sites. The SMB customers are under represented in the dataset. The two sub-populations share similarities in terms of shape of load curves but differences such as their peak hour range remain.

To account for both similarities and differences, we use fine tuning for the training of the models and assess whether this has an effect on their performances.

Two industrial constraints are integrated to the application:

— The prediction of the load curve of a new customer is taken from a catalog of existing customers

— An emphasis is put on the consumption during hours of sunlight, because this industrial application is related to solar-panel sizing of industrial customers.

The three modeling frameworks' predictive performances are compared. We also investigate the prediction intervals obtained from the Bayesian posterior distribution.

1.5.3 Bayesian transfer learning for panel data

In Chapter 4, we introduce a methodology of Bayesian transfer learning for panel data, adapted from the work of Launay et al. (2015).

We aim to transfer information to a new individual with short historical data, from what has been learned from a panel of several individuals with long historical data.

The methodology consists in a two step approach. Firstly, a Bayesian hierarchical model is trained, with weakly informative priors, on the panel data with long historical data. Hyperparameters are set to take into account the general behavior of the panel. We are interested in the posterior distributions of those hyperparameters, as they contain the information to be transferred to the new individual. Specifically, we get the posterior mean and posterior covariances and integrate them to the prior distribution of the parameters of the model for a new individual. This informative approach is the second step of the methodology. Gaussian priors are taken for the parameters, and the information is transferred on the mean and covariances of those priors. Hyperparameters of similarities are integrated to the Gaussian prior to take into account the potential similarities and differences between the new individual and the original panel data.

The methodology is evaluated on simulated data, taken from Launay et al. (2015). Three situations are considered: a polynomial regression, an auto regression and a hierachical Poisson model.

1.5.4 Application of Bayesian transfer for panel data: end of month electrical consumption forecasting for residential customers

Chapter 5 is dedicated to the application of Bayesian transfer learning for panel data, as described in Chapter 4, to real world data.

We focus on the industrial use-case of end-of-month consumption forecasting for new customers. In this Chapter, the methodology and models are evaluated and assessed on a specific subset of the CER dataset Commission for Energy Regulation (2012). The construction of this subset is the object of Chapter 6.

The goal is to transfer information gained on customers with longer historical consumption to customers with a short historical.

For the weakly informative approach, we estimate two models on a panel, where the historical consumption available for each individual is of 365 days. The explanatory variables integrated to both models are: the outside temperature recorded for the day t, $t = 1, \ldots, T$, the binary variable indicating the presence of electric heating for the individual i and the binary variable indicating the presence of an electric sanitary hot water system in the household i, $i = 1, \ldots, n$. The second model has an additional variable, which is the individual lagged consumption of the previous day $X_{t-1}^{(i)}$. Both models have hyperparameters, where weakly informative priors are taken. after the inference of the models we extract the posterior mean and covariances.

For the informative approach, we consider individual models, with the same variables as before, and we transfer the information of the the posterior mean and covariances to the Gaussian priors as is done in Chapter 4.

We train the models for the informative approach on different length of historical data and we forecast the values of consumptions at each timestep of the horizon. We then sum the forecasts to obtain the end-of-month consumption forecasting value.

We evaluate the performances of the models, considering the informative approach and the uninformative counterpart. Several indicators are examined: the error between the point estimate and the real value of the end-of-month consumption, the length of prediction intervals and the coverage probabilities.

1.5.5 Classifying new customers into homogeneous groups with deep learning

In Chapter 5, we describe an application of Bayesian transfer learning to forecasting the end-of-month consumption of residential customers. We apply the methodology to a subset of the CER data Commission for Energy Regulation (2012). In Chapter 6, we describe the methodology designed to build the subgroup used in Chapter 5.

The methodology consists in several steps.

Firstly, dimensionality reduction is done on annual load curves at a daily period using deep autoencoders. We consider two types of layers: Dense and 1D-convolutional layers and we compare the performances of both types, in terms of error of reconstruction.

Secondly, clustering methods are applied to the latent space from the best autoencoder. Three methods are compared: K-Means, HAC and K-medoids. The optimal number of clusters for the three methods is chosen using various criteria. The best method of clustering appears to be K-medoids, as it is more robust to atypical individuals.

Once the clusters have been set up, we describe how to assign a new individual to a class, using a double input neural network. We evaluate the quality of the model using different metrics.

Finally, we adapt the methodology of forecasting the end-of-month consumption as in Chapter 5, to alleviate the issue of missclassified new individuals. We use the softmax scores of the assignment neural network to build probabilities that an individual belongs to each of the classes. The probabilities are used to weigh the prediction we obtain for each class. Therefore, we have a mixture prediction and associated results that we compare to the results obtained in Chapter 5, for the fixed cluster setting.

Chapter 2

Methodology of estimation of a multi-target regression model in high dimension

2.1 Introduction

This chapter and the following are combined into an article under review in Applied Energy.

Modeling directly a customer's individual load curve using explanatory variables (e.g. billing information or household characteristics) amounts to estimating a multi-target regression model, where the response variable is multi-dimensional:

$$\mathbb{E}((X_1, \ldots, X_t)|V) = f(V).$$

The outcomes $X_1, \ldots, X_t$ are dependent from each other, as we are dealing with time series. The forecast is obtained with the features V, without historical data. This differs from the classical approach to time series forecasting where:

$$\mathbb{E}(X_{t+1}, \ldots, X_{t+s}|X_1, \ldots, X_t) = f(V, X_1, \ldots, X_t),$$

the forecasts are obtained using past data.

In this Chapter, we describe a general methodology for multi-target regression and estimation based on three strategies. We wish to predict X from V, and ideally we would do so by directly estimating the multi-target regression model. However, due to the high dimension of X, we propose two different strategies based on dimensionality reduction. In those strategies, what is modeled is the reduced space of the curves. Afterwards, we consider several strategies to predict X.

Multi-target or multi-output regression problems are encountered in many real-world applications. They have been the subject of tremendous interest in statistical learning, before the rise of deep learning methods. For instance, Segal and Xiao (2011) propose multivariate random forests to tackle multiple outcome settings. In chemometrics for instance, Burnham et al. (1999) consider a multivariate latent variable regression framework in the context of chemical compounds measures. A multi-target regression framework is proposed by Li et al. (2017) for the prediction of drug efficacy. Aras and Aras (2004) propose a multi-target framework for forecasting the demand of natural gas. Multi-target regression can be estimated from statistical models such as Support Vector Regression (Zhang et al. (2012), Vazquez and Walter (2003)), Gaussian Process (GP) Regression (Liu et al. (2018), Cai et al. (2020)), Partial Least Square (PLS) Regression (Golmohammadi (2009), Barbon Junior et al. (2020)) etc. Additional examples of multi-output regression approaches can be found in Borchani et al. (2015). Multi-output regression is a subfield of multi-task learning, for more information on the subject, refer to Xu et al. (2019).

Deep learning methods are particularly suited to tackle multi-target regression problems. Helmiriawan and Al-Ars (2019) propose a deep learning-based approach to tackle the multi-target regression problem of predictive maintenance in oil refineries. An LSTM-RNN approach is developed by Sun et al. (2017) for multi-target regression in the context of speech enhancement. Kong et al. (2020a) propose a deep learning model to tackle zero-inflated multitarget regression problems, applied to the case of estimating the distribution of animal species. Narayanan et al. (2021) propose a deep learning model based on the VGG-16 model (Simonyan and Zisserman (2015)) to tackle the multi-target regression problem of predicting pasture biomass percentage.

Data in high dimension complicate multi-target learning tasks. To remedy this, it might be necessary to integrate a dimensionality reduction method prior to the regression task. In Barbon Junior et al. (2020), a PLS based approach is used for dimensionality reduction. Deep learning methods can also be considered for dimensionality reduction prior to regression. A framework proposed by Renganathan et al. (2021) is based on an encoder-decoder architecture combined with GP-regression, for wind turbines modeling.

Being able to predict and forecast without the availability of historical data is crucial in some domains. For instance, Driver and Alemi (1995) apply a Bayesian method to obtain forecasts without historical data, based on expert opinions, to the case of medical malpractice litigation. In this context, no historical data is available on the patients having

experienced medical mishap that causes them to take legal action. van Steenbergen and Mes (2020) focus on forecasting the demand of new products (lacking historical sales data) and develop a method combining K-Means, Random Forests and Quantile Regression Forest to do so. Thomassey and Happiette (2007) propose a neural network based approach hybriding Self-Organizing Maps and Probabilistic neural networks (based on the Naive Bayes classifier) to estimate the sale profiles of new apparel, for which no historical sales are available.

The chapter is organized as follows. Section 2.2 specifies the notations used for the methodology. In Section 2.3, two possibilities are considered to estimate the multi-target regression model between the load curves and the features: the model is either directly estimated using neural networks or via an additional step based on dimensionality reduction and reconstruction. The model is estimated in the lower dimensional space using either deep neural networks, Bayesian neural networks and deep Gaussian processes. Furthermore, we show how prediction of a new load curve with known features ensues from the two strategies. Section 2.4 describes an alternative to predict a new customer's load curve by using a catalog of existing customers, as well as the loss function designed to put an emphasis on certain time periods. In Section 2.5, we describe the construction of probabilistic prediction intervals using the Bayesian posterior predictive distribution ensuing from Bayesian neural networks. Section 2.6 is dedicated to conclusion and follow-up work.

2.2 Notations

Hereinafter, we use the following notations :

- X, V and I denote to the random variables corresponding to the time series (for instance a load curve as in Chapter 3), the features (predictors) and the reduced representation of the time series respectively.
- m is the number of individual electrical load curves of length n.
- n is the dimension of X and we have $dim(X) \gg dim(V)$.
- The samples $\mathcal{X} = (\mathbf{X}_k)_{1 \leq k \leq m}$, $\mathcal{V} = (\mathbf{V}_k)_{1 \leq k \leq m}$ and $\mathcal{I} = (\mathbf{I}_k)_{1 \leq k \leq m}$ represent respectively the database of individual load curves, features and reduced load curves.
- $\mathbf{X}_{new}$, $\mathbf{V}_{new}$ refer to the unobserved individual load curve and features of a new customer to predict.

2.3 Methodology for estimating a multi-target regression model

In a multitarget regression setting, estimating a model in high dimension is often challenging due to the high dimensionality of the target (in fact, here the dimension of the target is much larger than that of the features). Two possibilities are considered to conduct the estimation. The first relies on the direct estimation of the model whereas the other depends on a dimensionality reduction intermediate step. The model is estimated in the reduced space and the estimation in the reduced space is reconstructed, provided that the dimensionality reduction method allows it. In both cases, neural networks are suitable candidates for the estimation.

2.3.1 Multitarget nonlinear regression in high dimension (MNR)

Assume $dim(X) \gg dim(V)$, the goal is to model the time series X in high dimension using the features V.

In a multitarget nonlinear regression problem, the best theoretical predictor, in terms of quadratic error in $\mathbb{L}^2$, is the conditional expectation:

$$\mathbb{E}(X|V) = g(V), \tag{2.1}$$

where the function g is unknown. In practice, a modeling step is required to estimate g, in a parametric or non parametric framework.

Let $\hat{g}$ be an estimator of g. If a set of features $\mathbf{V}_{new}$ is known, we predict the load curve $\mathbf{X}_{new}$ with:

$$\hat{\mathbf{X}}_{new} = \hat{g}(\mathbf{V}_{new}).$$

The diagram in Figure 2.1 summarizes the different steps of this approach.

Due do the dimension of the problem, neural networks are good candidates to approximate g (see Goodfellow et al. (2016) for a review on the variety of deep learning methods). We use neural networks built with dense layers only, which means that each unit of a given layer is fully connected to each unit of the next layer.

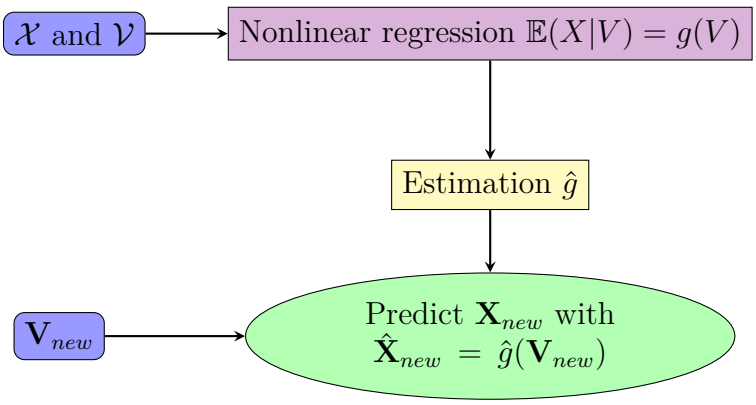


Figure 2.1 – Outline of the MNR framework (Section 2.3.1), blue boxes specify the inputs (variables) of the models, pink boxes the processes/models, yellow boxes the estimations and green boxes the predictions.

We approximate g by a neural network with i hidden layers of the following form:

$$\mathbf{h}_1 = \sigma(\mathbf{W}_1^T \cdot V + \mathbf{b}_1), \tag{2.2}$$

$$\vdots$$

$$\mathbf{h}_i = \sigma(\mathbf{W}_i^T \cdot \mathbf{h}_{i-1} + \mathbf{b}_i), \tag{2.3}$$

$$X = \sigma(\mathbf{W}_{i+1}^T \cdot \mathbf{h}_i + \mathbf{b}_{i+1}), \tag{2.4}$$

where $\mathbf{h}_p,\ 1 \leq p \leq i$ are the outputs of the pth hidden layer of the neural network, $\mathbf{W}_p$ and $\mathbf{b}_p$, $1 \leq p \leq i+1$ the weights and biaises of the pth layer respectively. The function σ is the Rectified Linear Unit (ReLU) activation function:

$$\sigma(x) = \max(0, x), \text{for all } x \in \mathbb{R} \tag{2.5}$$

Theorem 1 (Universal approximation theorem Pinkus (1999)) *Let $\sigma \in C(\mathbb{R})$, a continuous function, then $\mathcal{M}(\sigma) = \{\sum_{i=1}^{r} c_i \sigma(w^i \dot{x} - b_i), w^i \in \mathbb{R}^d, c_i, b_i \in \mathbb{R}\}$, for some fixed r, is dense in $C(\mathbb{R}^n)$ for the uniform convergence on a compact, if and only if σ is not a polynomial.*

Remark 1 *The ReLU activation function has many advantages compared to other functions such as the Hyperbolic tangent TanH, such as better convergence and computational efficiency, hence most neural networks used in this study are build with the ReLU function.*

The usual loss functions used for training are the root mean squared error or the

MAE, a special case of the error specified in (2.17). We adapt the loss function to give more emphasis to specific periods of time of the curve. This aspect is further explored in Chapter 3.

Remark 2 *To use neural networks as presented here, all the times series X have to be on the same time window, without any missing values.*

2.3.2 Encoding, nonlinear regression and reconstruction(ENR-R)

In our regression problem, the dimension of the curves X is much larger than the dimension of the features V. Estimating a model in such high dimension, even using neural networks is challenging because of complexity issues, optimization stability and the necessity of large volumes of data.

It can be beneficial to consider reducing the dimension of the target X prior to estimating them in a reduced space using the features V.

We present here another possibility of estimation of the curves, based on dimensionality reduction and reconstruction. They are referred to respectively as encoding and decoding.

Firstly, dimensionality reduction is applied to the load curves X.

The next step, after dimensionality reduction, is to model the resulting reduced representation I with the nonlinear multi-target regression model:

$$\mathbb{E}(I|V) = f(V). \tag{2.6}$$

Let $\hat{f}$ be an estimator of f, and for a new customer $\mathbf{V}_{new}$, we can predict the load curve of the new customer in the reduced space :

$$\hat{\mathbf{I}}_{new} = \hat{f}(\mathbf{V}_{new}). \tag{2.7}$$

Encoding

Several methods exist for dimensionality reduction. Among these models the discrete wavelet transform is well suited for time series (see for instance Guan et al. (2012), Cugliari et al. (2016) and Auder et al. (2018)). Neural networks also provide statistical tools for dimensionality reduction, notably with autoencoders. They can be seen as the deep learning counterpart of Principal Component Analysis (PCA), because of their ability to

perform dimensionality reduction (see Hinton and Salakhutdinov (2006)), though unlike PCA, they can capture nonlinearities in the data.

Autoencoders

Autoencoders have a "diabolo" shape (see Figure 2.2) that takes an input and aims to reconstruct it through various nonlinear transformations.

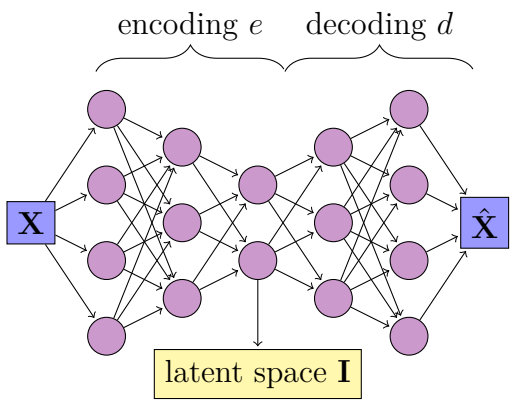


Figure 2.2 – The autoencoder take X as input and reconstruct it through various nonlinear operations, notably by reducing the dimension of the input through the encoding phase and using the reduced space I to obtain the reconstruction X through the decoding phase. Thus it has the same number of input units as output units.

The autoencoder takes X as input and reconstructs it through various nonlinear operations, notably by reducing the dimension of the input through the encoding phase and using the reduced space I to obtain the reconstruction X through the decoding phase. To summarize, we have:

$$I = e(X) \tag{2.8}$$

$$X = d(I), \tag{2.9}$$

where e and d are respectively the encoding and decoding functions, that can be composed of several layers. I is also referred to as the latent or encoded space.

As for the neural networks described in Section 2.3.1, the autoencoder is also trained using a specific loss function.

Discrete Wavelet Transform

Wavelets have interesting properties, one of which being able to reconstruct the original signal through the inverse discrete wavelet transform. Hence this allows for

direct comparison between the autoencoder and the wavelet transform where the encoding can be seen as a similar process to the discrete wavelet transform and is used to reduce the dimension of the input data, whereas the decoding and the inverse discrete wavelet transform use the reduced dimension to reconstruct the original data. The discrete wavelet transform consists in a hierarchical representation of the initial signal, for instance time series, all generated through dilation and translation operations. Considering the time series $X(t)$, it can be expressed as:

$$X(t) = \sum_j \sum_{k \in \mathbb{Z}} DWT(j,k)\psi_k^j(t), \tag{2.10}$$

where $DWT(j,k)$ is the discrete wavelet transform. $\psi_k^j(t)$ is a function dependent on the mother wavelet $\Psi(\dot{})$ as such:

$$\psi_k^j(t) = \Psi(\frac{t-k}{j}).$$

The mother wavelet $\Psi(\dot{})$ is an oscillating $\mathbb{L}^2$ function of zero mean.

Example 1 *A classical example of the mother wavelet $\Psi(\dot{})$ is the Haar wavelet:*

$$\Psi(t) = \begin{cases} 1 & \textit{if } 0 \leq t < \frac{1}{2} \\ -1 & \textit{if } \frac{1}{2} \leq t < 1 \\ 0 & \textit{otherwise} \end{cases}$$

Thus, the Haar function is defined as:

$$\psi_k^j(t) = 2^{j/2}\Psi(2^j - k),\ t \in \mathbb{R}$$

for $j, k \in \mathbb{Z}$.

The developed expression in (2.10) is truncated to estimate a finite number of coefficients. The truncated discrete wavelet transform allows to represent the signal $X(t)$ in a lower dimensional space.

Both methods of dimensionality reduction allow to the reconstruct the time series using its estimated reduced representation. Reconstruction gives a way of predicting the load curve.

Estimation of f

The function f is estimated from the $(\mathbf{I}_k, \mathbf{V}_k)_{1 \leq k \leq m}$ observations using models such as neural networks, Bayesian neural networks and deep Gaussian processes.

Residual neural networks

Aside from the neural networks described in Section 2.3.1, some of the models to estimate f employ more complex architecture, by using skip connections (also called residual connections) introduced by He et al. (2016).

Skip connections are designed to alleviate the vanishing gradient issue arising during training of deep neural networks. They allow information to flow through successive layers by skipping intermediate ones.

Recalling the notations introduced earlier when defining the neural network model, if we wish to introduce a skip connection between the first layer in (2.2) and all the $i-2$ intermediate layers before the $i$eth hidden layer, we add the outputs $\mathbf{h}_1$ and $\mathbf{h}_{i-1}$, and feed them as inputs to the Dense layer in (2.3). This example of residual connection is depicted in Figure 2.3.

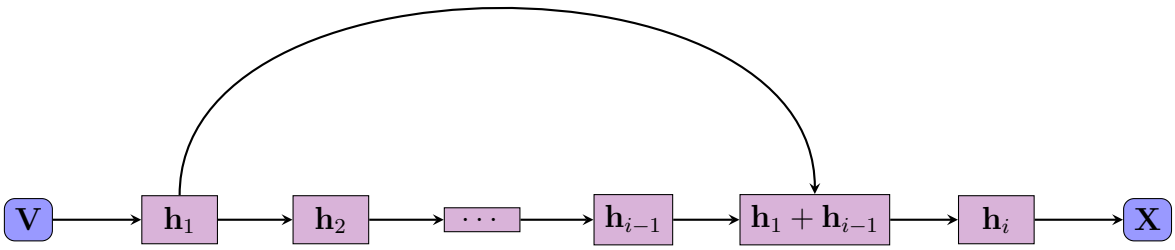


Figure 2.3 – Diagram of a neural network with a residual connection between the outputs of the first layer and of the $i-1$eth hidden layer, using the notations in (2.2) to (2.3).

Bayesian neural networks

In Bayesian neural networks, the parameters, namely weights and biases, as in (2.2) and (2.4), follow a prior distribution. Bayesian neural networks, specifically Bayesian LSTM have been used in the context of load forecasting (see Sun et al. (2020) and Yang et al. (2020)). However those neural networks are designed to deal with time series and require historical data to be efficiently trained. As we focus on estimating a regression model without historical data, LSTM layers are not suitable.

We apply Bayesian neural networks to estimate the function f defined in (2.6) using the features V. the Bayesian neural networks used here are built with Dense layers as we do not apply them directly to the time series but their reduced representation.

In a Bayesian neural network, the networks parameters, namely weights and biases, as introduced in (2.2) and (2.4) follow some probability distribution called the prior distribution.

Recalling the notations introduced in (2.2) and (2.4), in our Bayesian framework, the parameters $\mathbf{W}$ are stochastic and follow the prior distribution $p(\mathbf{W})$. Generally, the prior distribution expresses the prior belief one has over the distribution of the parameters of the model. In a deep neural network setting, it remains complicated to incorporate prior knowledge to the prior distribution of the network parameters, hence, the chosen prior for the neural networks weights in our study is the standard multivariate normal distribution $\mathbf{W} \sim \mathcal{N}(0, \mathrm{Id})$, where Id is the identity matrix. Here the biaises are set to 0.

In Bayesian analysis, the inference relies on finding the posterior distribution, which is the conditional distribution of the parameters given the observed data. The posterior distribution we want to approximate is defined using Bayes theorem:

$$p(\mathbf{W}|I, V) = \frac{p(\mathbf{W})p(I|V, \mathbf{W})}{p(I|V)} \tag{2.11}$$

the goal is to find the posterior distribution of the parameters given the observable data.

The main issue in Bayesian neural networks resides in the approximation of the posterior due to the large number of parameters and non linearity of deep neural networks. Another issue is the fact that the likelihood does not have an analytical form. The normalization constant $p(I|V)$ is intractable and is approximated using various methods.

Bayesian neural networks have some benefits over regular neural networks as they provide uncertainty estimates. Another perk of using Bayesian neural networks is that they naturally encompass regularization and model selection. We are also able to build probabilistic prediction intervals using them, this is developed further in the chapter (see Section 2.5).

deep Gaussian processes

Gaussian processes (GP) are stochastic processes widely used in machine learning specifically as prior distributions in the context of regression, in a Bayesian setting. For an extended review of the use of Gaussian processes in machine learning refer to Rasmussen and Williams (2006).

Definition 1 *A real valued random process* $(f(\mathbf{V}_1), \ldots, f(\mathbf{V}_m))$ *is a Gaussian process if all its finite dimensional distributions (fidi) are Gaussian. The Gaussian process can be solely defined by its mean* μ *and its covariance* k*.*

Recalling our problem of nonlinear regression in (2.6), applying the Gaussian process regression, which is a Bayesian nonparametric method, in that case means to place a GP prior over the space of regression functions f:

$$(f(\mathbf{V}_1), \ldots, f(\mathbf{V}_m)) \sim \mathcal{N}(0, \Sigma), \text{ where } \Sigma_{i,j} = k(\mathbf{V}_i, \mathbf{V}_j). \tag{2.12}$$

This Gaussian process is solely defined by its covariance function and we have $k(\mathbf{V}_i, \mathbf{V}_j) = \mathbb{E}((\mathbf{V}_i - \mathbb{E}(\mathbf{V}_i))(\mathbf{V}_j - \mathbb{E}(\mathbf{V}_j)))$.

We wish to estimate the posterior distribution. However, when using a GP prior, this requires the inversion of the covariance matrix, the complexity of the problem is then $O(N^3)$, hence making it difficult to use Gaussian processes when the dimension N is large. To counter this complexity issue, methods have been developped such as the sparse GP (Titsias (2009), Hensman et al. (2013)), the approximation of the posterior is found using variational inference.

Remark 3 *Many links between Bayesian neural networks and Gaussian processes have been studied throughout literature (see Neal (1996), and more recently Matthews et al. (2018) and Novak et al. (2019)).*

Deep Gaussian processes, introduced by Damianou and Lawrence (2013), are a class of deep models, built similarly to a deep neural network, as layers of Gaussian

process regressions are stacked. An i hidden layers DGP model can be defined as:

$$\mathbf{h}_1 = f_1(V) + \varepsilon_1, \tag{2.13}$$
$$\mathbf{h}_2 = f_2(\mathbf{h}_1) + \varepsilon_2,$$
$$\vdots$$
$$\mathbf{h}_i = f_i(\mathbf{h}_{i-1}) + \varepsilon_i, \tag{2.14}$$
$$X = f_{i+1}(\mathbf{h}_i) + \varepsilon_{i+1}, , \tag{2.15}$$

where $f_1 \sim GP(0, k^1(V,V))$, $f_2 \sim GP(0, k^2(\mathbf{h}_1, \mathbf{h}_1))$, $f_i \sim GP(0, \Sigma(k^i, \mathbf{h}_{i-1})$ and $\Sigma(k^i, \mathbf{h}_{i-1}) = \mathbb{E}((k^i - \mathbb{E}(k^i))(\mathbf{h}_{i-1} - \mathbb{E}(\mathbf{h}_{i-1})))$.

Posterior distribution in DGP can be approximated using sparse methods with variational inference.

Reconstruction and prediction

Once the regression function f is estimated with either of the methods aforementioned, we focus on reconstructing a predicted curve using its prediction in the reduced space $\hat{\mathbf{I}}_{new}$.

In some cases, the dimensionality reduction method allows for the reconstruction of the original load curve, back into its initial space, by estimating a function denoted r (d in the case of the autoencoder). Using an estimation of r and $\hat{\mathbf{I}}_{new}$, we are able to reconstruct a prediction of the load curve of a new customer.

The predicted load curve $\hat{\mathbf{X}}_{new}$ of a new customer, is then obtained as:

$$\hat{\mathbf{X}}_{new} = \hat{r}(\hat{\mathbf{I}}_{new}). \tag{2.16}$$

This modeling strategy, named ENR-R for Encoding Nonlinear Regression and Reconstruction, differs from the previous approach because of its dependency on a dimensionality reduction method.

Remark 4 *If we had used PCA to reduce the dimension of the curves, as it is a linear transformation, we would have:*

$$\mathbb{E}(g(I)|V) = g(\mathbb{E}(i|V))$$

. This relation is not true in the case of autoencoders. This is due to error terms in encoding and decoding.

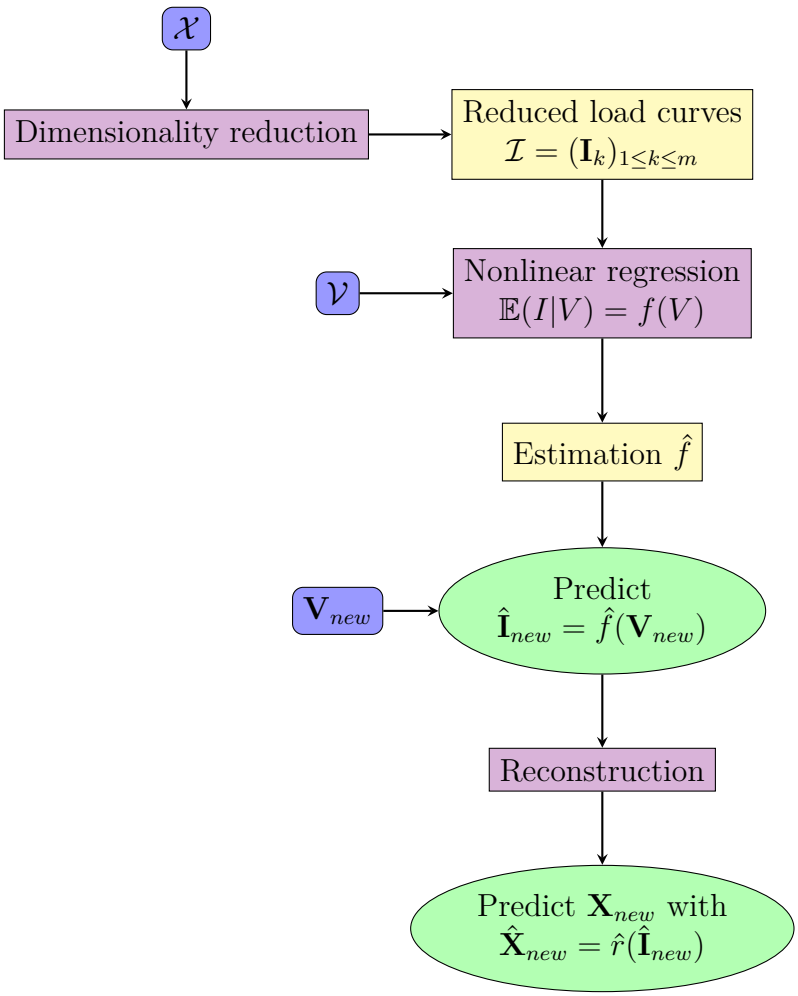


Figure 2.4 – Outline of the ENR-R strategy (Section 2.3.2), blue boxes specify the inputs (variables) of the models, pink boxes the processes/models, yellow boxes the estimations and green boxes the predictions.

When reducing the dimension of the curve, we loose the initial target and instead have a new model: where we estimate I (the latent space) using the features V.

Ideally, we would predict the curve X from V directly, and because of the high dimension of X, we have to consider an alternative.

Our approach has two steps: - first reducing the dimension of X - estimating the model of I using V

Because, we use autoencoder, as they are nonlinear, at the second step, we have already lost the optimality, and we focus on estimating the new target I. Afterwards, we focus on how to get back to the initial space to predict X.

The diagram in Figure 2.4 details the ENR-R modeling framework.

2.4 Prediction under constraint

An alternative to predicting directly using the estimated models described in Section 2.3 is to conduct the prediction is a limited space.

2.4.1 Prediction in a catalog

For some applications, it can be necessary to seek the prediction of a new individual in a finite restricted space. This is usual for instance in text generation, where the prediction is taken from a dictionary of existing words. In Chapter 3, we present an application where the load curve of a new customer is carried out in a catalog of existing customers. This is done to integrate an industrial constraint into the application.

Let $\mathcal{I}$ denote such a catalog, library or dictionary in a reduced space, obtained from the catalog in the initial space. For a new individual for which a prediction $\hat{\mathbf{I}}_{new}$ is available, to integrate the constraint, we seek its nearest neighbor in the catalog $\mathcal{I}$. $\mathcal{I}$ contains m individuals of dimension q.

We define $\mathcal{E}$ a distance on $\mathcal{X}$ as such:

$$\mathcal{E}(Y, \hat{Y}) = \sum_{i=1}^{q} w_i |Y_i - \hat{Y}_i|, \tag{2.17}$$

where Y and $\hat{Y}$ are respectively the target and its prediction, both of size q. Here, w_i are weights, allowing to give more or less importance to specific indexes $1 \leq i \leq q$.

Remark 5 *When predicting a time series, it can be interesting to give more or less weight to certain time periods. This is further developed in Chapter 3.*

It is worth noting that, giving the same importance to each time period (i.e. $w_i = \frac{1}{q}$, $1 \leq i \leq q$) amounts to consider the mean absolute error as the error of choice.

Proposition 1 *The best prediction of the new individual load curve given the constraint is $\mathbf{X}_{\hat{k}_I}$ where*

$$\hat{k}_I = \underset{1 \leq k \leq m}{\operatorname{argmin}} \, \mathcal{E}(\mathbf{I}_k, \hat{\mathbf{I}}_{new}). \tag{2.18}$$

<u>Proof:</u>

- **Existence**

  If we wish to conduct the prediction of the new individual given the constraint, for each element of the catalog, we calculate:

$$\mathcal{E}(Z, \hat{\mathbf{I}}_{new}), \text{ with } Z \in \mathcal{I} \tag{2.19}$$

  The catalog is finite, thus justifying the existence of a minimal value of $\mathcal{E}(Z, \hat{\mathbf{I}}_{new})$ for a certain Z.

- **Unicity**

  If the minimum is not reached in a unique point, we arbitrarily choose the smallest value of k. □

Remark 6 *If a prediction of the complete load curve $\hat{\mathbf{X}}_{new}$ is available as in both strategies in Section 2.3, it is also possible to seek the nearest neighbor in the original catalog of curves in high dimension $\mathcal{X}$. This is further developed in Chapter 3.*

2.4.2 Encoding and nonlinear regression (ENR)

Using $\hat{\mathbf{I}}_{new}$, we can find the nearest neighbor's in the catalog of reduced load curves and then we predict the load curve with $\mathbf{X}_{\hat{k}_I}$, where $\hat{k}_I$ is defined in (2.18). This approach is named ENR for Encoding and Nonlinear Regression and represented in Figure 2.5.

The ENR modeling strategy is represented in Figure 2.5.

Specifically, the dimensionality reduction (encoding) step and the nonlinear regression on the encoded load curve space in (2.6) are the same in both ENR-R and ENR. The ENR-R framework only differs from the ENR one because instead of directly searching for the prediction in the catalog of reduced load curves, we first reconstruct the signal from the predicted load curve in the reduced space and then, we seek the prediction in the catalog of load curves in the original space as detailed in Section 3.2.2.

2.5 Probabilistic prediction intervals using the Bayesian posterior predictive distribution

In this section, we construct credible intervals on predicted curves. Assume that Bayesian neural networks are used for the estimation of the nonlinear regression model

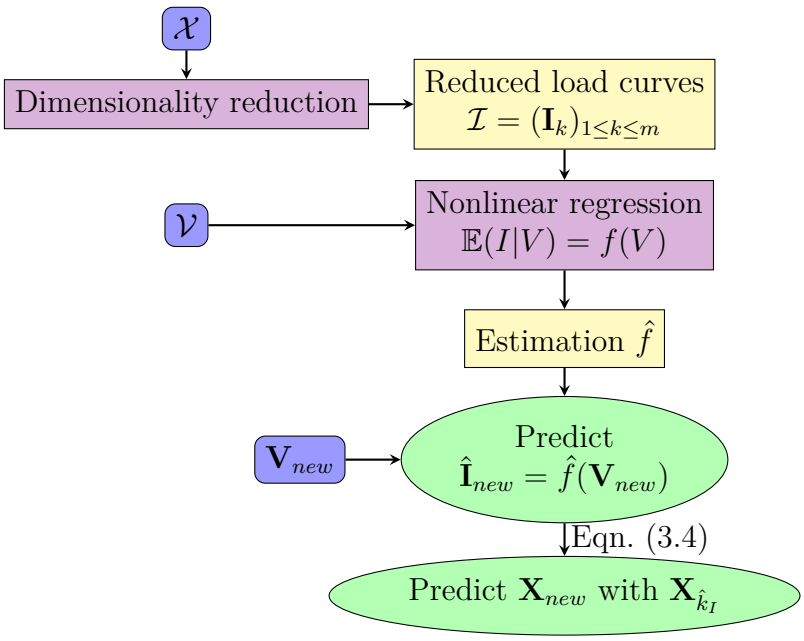


Figure 2.5 – Outline of the ENR research framework (Section 2.4.2), blue boxes specify the inputs (variables) of the models, pink boxes the processes/models, yellow boxes the estimations and green boxes the predictions.

in (2.6). In a neural network, the parameters $\mathbf{W}$ lack interpretability, the benefit of the Bayesian approach does not appear through building informative prior. However, this approach allows to build a predictive distribution and credible intervals. From the posterior distribution of the parameters, given by:

$$p(\mathbf{W}|\mathcal{I}, \mathcal{V}) = \frac{\exp(-\frac{1}{2}\mathbf{W}^T\mathbf{W})p(\mathcal{I}|\mathbf{W}, \mathcal{V})}{p(\mathcal{I}|\mathcal{V})}, \tag{2.20}$$

Where $\mathcal{V} = (\mathbf{V}_k)_{1 \leq k \leq m}$ and $\mathcal{I} = (\mathbf{I}_k)_{1 \leq k \leq m}$ represent respectively the database of features and reduced load curves. For a new customer defined by its billing information $\mathbf{V}_{new}$, we can make predictions using the posterior predictive distribution.

Proposition 2 *The predictive distribution of the reduced load curve* $\mathbf{I}_{new}$ *can be written as:*

$$p(\mathbf{I}_{new}|\mathcal{I}, \mathcal{V}, \mathbf{V}_{new}) = \int_{\mathcal{W}} p(\mathbf{I}_{new}|W, \mathbf{V}_{new})p(W|\mathcal{I}, \mathcal{V})\, dW, \tag{2.21}$$

where $p(\mathbf{I}_{new}|W, \mathbf{V}_{new})$ *is the likelihood of the regression model in the latent space.*

Proof: The predictive distribution is the conditional distribution of $\mathbf{I}_{new}$ given $\mathcal{I}$. We have:

$$\begin{aligned} p(\mathbf{I}_{new}|\mathcal{I},\mathcal{V},\mathbf{V}_{new}) &= \int_{\mathcal{W}} p(\mathbf{I}_{new},W|\mathcal{I},\mathbf{V}_{new},\mathcal{V})\,dW, \\ &= \int_{\mathcal{W}} p(\mathbf{I}_{new}|W,\mathcal{I},\mathbf{V}_{new},\mathcal{V})p(W|\mathcal{I},\mathbf{V}_{new},\mathcal{V})\,dW. \end{aligned}$$

As the features are supposed independent,

$$p(\mathbf{I}_{new}|W,\mathcal{I},\mathbf{V}_{new},\mathcal{V}) = p(\mathbf{I}_{new}|W,\mathbf{V}_{new})$$

$p(W|\mathcal{I},\mathbf{V}_{new},\mathcal{V})$ is the posterior distribution, it does not depend on $\mathbf{V}_{new}$, thus:

$$p(W|\mathcal{I},\mathbf{V}_{new},\mathcal{V}) = p(W|\mathcal{I},\mathcal{V}).\square$$

We build two typee of probabilistic prediction for the load curve:

1. **Prediction by decoding.**

   The predicted curve is:

   $$\mathbf{X}_{new} = \hat{d}(\mathbf{I}_{new}), \tag{2.22}$$

   where $\hat{d}$ is an estimator of the decoding function d of the autoencoder defined in (2.9).

   The predictive distribution of $\mathbf{X}_{new}$ is obtained through a variable change:

   $$\mathbb{E}(h(\mathbf{X}_{new})|\mathcal{I},\mathcal{V},\mathbf{V}_{new}) = \int_{\mathbb{R}} h(\hat{d}(i))p(i|\mathcal{I},\mathcal{V},\mathbf{V}_{new})\,di, \tag{2.23}$$

   where h is an arbitrary measurable positive function, $i \in \mathbb{R}$, and $p(i|\mathcal{I},\mathcal{V},\mathbf{V}_{new})$ is defined in (2.21).

   For fixed time $t = 1,\ldots,n$, we can calculate prediction intervals for $\mathbf{X}_{new}(t)$ from the quantiles of the predictive distribution defined in (2.23).

2. **Prediction in the catalog.**

   As described in (3.4), the predicted curve is of the form

   $$X_{new} = X_{k_{\mathbf{I}_{new}}},$$

where $k_{\mathbf{I}_{new}}$ is a random variable taking value in $\{1, \ldots, m\}$. The predictive distribution of $k_{\mathbf{I}_{new}}$ is given by

$$\begin{aligned}
\mathbb{P}(k_{\mathbf{I}_{new}} = j | \mathcal{I}, \mathcal{V}, \mathbf{V}_{new}) & & (2.24)\\
&= \mathbb{P}(\operatorname*{argmin}_{1 \leq k \leq m} \mathcal{E}_{MAE}(\mathbf{I}_k, \mathbf{I}_{new}) = j | \mathcal{I}, \mathcal{V}, \mathbf{V}_{new}) & (2.25)\\
&= \int_{\{i \,:\, \operatorname{argmin}_{1 \leq k \leq m} \mathcal{E}_{MAE}(\mathbf{I}_k, i) = j\}} p(i | \mathcal{I}, \mathcal{V}, \mathbf{V}_{new})\, di. \\
&=: p_j
\end{aligned}$$

where $p(i|\mathcal{I}, \mathcal{V}, \mathbf{V}_{new})$ is defined in (2.21). For fixed time $t = 1, \ldots, n$, we can calculate prediction intervals for $\mathbf{X}_{new}(t)$ from the predictive distribution of $k_{\mathbf{I}_{new}}$.

Let q_β^{new} be the β-quantiles of the discrete distribution of support $\{X_j(t), j \in \{1, \ldots, m\}\}$ and probability $\{p_j, j \in \{1, \ldots, m\}\}$, defined in 2.26.

All intervals of the form $[q_\beta^{new}(t), q_{1-\alpha+\beta}^{new}(t)]$, with $\beta \in [0, \alpha]$ are intervals of level $1 - \alpha$.

$$\mathbb{P}(\mathbf{X}_{new}(t) \in [q_\beta^{new}(t), q_{1-\alpha+\beta}^{new}(t)] | \mathcal{I}, \mathcal{V}, \mathbf{V}_{new}) = 1 - \alpha.$$

We select the shortest interval by taking

$$\beta^{opt} = \operatorname*{argmin}_{\beta \in [0;\alpha]} |q_{1-\alpha+\beta}^{new}(t) - q_\beta^{new}(t)|$$

Simulation from the predictive distribution

We apply variational inference to approximate the posterior distribution. One interesting aspect of this method is that is does not necessitate to approximate the normalization constant $p(I, V)$ which is intractable in (2.20).

Considering a family of distribution $\mathcal{Q}$, e.g. Gaussian distributions or a product of Gaussian distributions. Variational methods rely on the minimization of the Kullback-Leibler divergence between the real posterior distribution in (2.20) and a distribution $q(\mathbf{W}) \in \mathcal{Q}$ which has an analytical expression. Hence, we seek the optimal $q^\star$:

$$q^\star(\mathbf{W}) = \operatorname*{argmin}_{q(\mathbf{W}) \in \mathcal{Q}} \; K(q(\mathbf{W}), p(\mathbf{W}|I, V)). \tag{2.26}$$

The Kullback-Leibler divergence between $q(\mathbf{W})$ and $p(\mathbf{W}|I, V)$ can be expressed as:

$$\begin{aligned} K(q(\mathbf{W}), p(\mathbf{W}|I, V)) &= \mathbb{E}(\log(\frac{q(\mathbf{W})}{p(\mathbf{W}|I, V)})) \\ &= \mathbb{E}(\log(q(\mathbf{W}))) - \mathbb{E}(\log(p(\mathbf{W}|I, V))) \\ &= \mathbb{E}(\log(q(\mathbf{W}))) - \mathbb{E}(\log(p(\mathbf{W}, I|V))) + \mathbb{E}(\log(p(I|V))). \end{aligned}$$

Minimizing the Kullback-Leibler divergence, in (2.26), is equivalent to maximizing the Evidence Lower Bound (ELBO) which is:

$$L(\mathbb{W}) = \mathbb{E}(\log(p(\mathbf{W}, I, V))) - E(\log(q(\mathbf{W}))). \tag{2.27}$$

$$L(\mathbf{W}) = \int q(\mathbf{W}) \log(p(I|V, \mathbf{W}))\, d\mathbf{W} - \int q(\mathbf{W}) \log(\frac{q(\mathbf{W})}{p(\mathbf{W})})\, d\mathbf{W}. \tag{2.28}$$

The ELBO, defined in (2.28), serves as the loss function for the training of our Bayesian neural networks. For an exhaustive review on variational inference, see Blei et al. (2017).

The Flipout estimator, introduced by Wen et al. (2018), is then used for the minimization of the negative ELBO.
The algorithm in **??** details the process to obtain the probabilistic prediction intervals.

Algorithm: Calculate probabilistic prediction intervals

Inputs: sample size J;

Instructions:

- Sample weights $(\mathbf{W}_j)_{1\leq j\leq J}$ from $p(\mathbf{W}|\mathcal{I},\mathcal{V})$
- Generate $(\mathbf{I}^{pos}_{new_j})_{1\leq j\leq I}$, $\mathbf{I}^{pos}_{new_j} \sim p(\mathbf{I}_{new}|(\mathbf{W}_j)_{1\leq j\leq J},\mathcal{I},\mathcal{V},\mathbf{V}_{new})$

if Prediction by decoding **then**

- decode the realizations $\mathbf{I}^{pos}_{new_j}$ using the autoencoder:

$$\hat{\mathbf{X}}^{pos}_{new_j} = \hat{d}(\mathbf{I}^{pos}_{new_j}),\ \forall\ j\{1,\ldots,J\}$$

- For each value of t, $t = 1,\ldots,n$, compute the empirical quantiles of the sample $(\hat{\mathbf{X}}^{pos}_{new_j}(t))_{j\in\{1,\ldots,J\}}$. ;

if Prediction in the catalog **then**

- calculate the index of the nearest neighbor of $\mathbf{I}^{pos}_{new_j}$:

$$k^{pos}_{I_j} = \underset{1\leq k\leq m}{\operatorname{argmin}}\, \mathcal{E}(\mathbf{I}_k, \mathbf{I}^{pos}_{new_j}),\ \forall\ j \in \{1,\ldots,J\}$$

- approximate the posterior distribution of p_k by taking the empirical distribution $\hat{p}_k$ of sample k^{pos}_I supported by $\mathcal{K} \subset \{1,\ldots,m\}$, the set of values taken by the sample $(k_{I_j})_{j\in\{1,\ldots,J\}}$ (i.e. the sample of indexes without repetitions);
- For $t = 1,\ldots,n$ compute the quantiles of the discrete distribution with support $\{X_k(t), k \in \mathcal{K}\}$ and the probabilities $(\hat{p}_k)_{k\in\mathcal{K}}$;

2.6 Conclusion

We show how to estimate a load curve in high dimension from available features. Our approaches are based on neural networks either for direct estimation, or for dimensionality reduction and estimation in the reduced space. More generally, we propose three possible strategies based on deep learning to estimate multi-target regression problems, when the dependent variable dimension is very large.

The prediction of a new individual, when features are known, is either direct or taken from a catalog of existing individuals.

Bayesian neural networks, used to estimate the curve in the reduced space, allow to build probabilistic prediction intervals.

An application of the methodology developed here is described in Chapter 3 for pre-

dicting non residential customers load curves. We adapt the three modeling frameworks to integrate industrial constraints. We apply and compare diverse methods in each strategy, and evaluate them on real data from EDF. We also describe how to extend the methodology of predicting in a catalog of existing curves to all the strategies.

The MNR modeling strategy is estimated with neural networks with dense layers. A potential drawback is that the time dimension is not taken into account with dense layers. An alternative, which has not been tested in this methodology, could be to substitute the dense layers with layers more suitable to deal with that, such as 1D-convolution layers (see Zhao et al. (2017) for an application to time series classification). In Chapter 6, we show an application of dimensionality reduction using a 1D-convolutional autoencoder applied to residential customers' load curves.

Chapter 3

Application of the methodology of estimation of a multi-target regression problem: the case of predicting individual load curves of non residential customers

3.1 Introduction

This chapter provides an application of the methodology presented in the previous chapter.

The opening of the French electricity market and the wide expansion of smart meters lead energy companies like Électricité de France (EDF) to innovate and provide new customer services. They rely mostly on individual load curve analysis, made possible by the deployment of smart meters, and are developed using statistical methods applied to customer data. Individual load curve analysis pertains to various statistical methods from time series analysis for load forecasting, to dimensionality reduction for clustering and feature extraction. A detailed survey on smart meter data analysis and its application can be found in Wang et al. (2019).

In this Chapter, we show an application of the methodology presented in Chapter 2, to load curve modeling of non residential customers (typically, businesses such as retail stores, farming and insurance companies etc). Load modeling for non residential customers is challenging due to their diversity. In fact some have seasonal activity while others are active throughout the whole year. Yet, being able to model the consumption of non residential customers is crucial. Some industrial application rely directly on this aspect:

- load profiling of non-metered customers to correctly estimate the sourcing costs on wholesale energy markets.
- sizing of potential photo-voltaic installations which relies on the estimation of the customer's consumption during daytime.

Most studies on load analysis and modeling focus on residential customers. For instance, Teeraratkul et al. (2018) apply dynamic time warping to build clusters of residential load curves with similar shapes, and use those clusters to encode portions of the load curves. They also propose a Markov model to predict the cluster of the load curve for the next 24 hours. Fitzpatrick et al. (2020) propose a clustering method on residential load curves in order to perform supervised classification of consumers from survey data.

Additionally, the majority of available studies deal with short term load forecasting. Kong et al. (2019) develop a Long Short-Term Memory (LSTM) neural network to that end. Auder et al. (2018) present a bottom-up short term load forecasting approach, meaning that, from a set of individual load curves, forecasting is done by aggregating the consumption of the given set. Amarasinghe et al. (2017) use convolutional neural networks to perform load forecasting of a single residential household.

Load modeling on non residential customers and industrial buildings is also a topic of interest. Recent studies on the subject include bottom-up heat and electric forecasting. For instance, this approach is developed by Lindberg et al. (2019) for non residential buildings in Norway. A survey of methods applied for load forecasting for commercial buildings can be found in Yildiz et al. (2017). Vaghefi et al. (2015) propose a clustering method to identify industrial load patterns. New load profiles are subsequently classified into the obtained clusters.

Deep learning methods have been gaining traction in many fields of application including computer vision (Krizhevsky et al. (2012)) and speech recognition (Graves et al. (2013)). Application also include load curve analysis, as shown in Shi et al. (2018), where a recurrent neural network is used for residential load curve forecasting. Recently, in He (2017), a deep neural network is used to forecast the aggregated load curve of a Chinese city. In Yang et al. (2020) and in Sun et al. (2020), a Bayesian LSTM is used for forecasting. Varga et al. (2015) apply stacked sparse autoencoders to encode load curves and to classify load profiles using a locality hashing sensitive method.

Probabilistic load forecasting, which relates to providing uncertainty estimates with the forecasts, has also been the subject of recent studies. A detailed survey is presented in Hong and Fan (2016). Shepero et al. (2018) develop Gaussian process and lognor-

mal process-based methods for probabilistic load forecasting. Chen et al. (2019) uses MC dropout, introduced by Gal and Ghahramani (2016), on a deep residual network to perform probabilistic load forecasting. In Yang et al. (2020) a Bayesian neural network with LSTM layers is applied to forecast individual load curves. A Bayesian LSTM is also used in Sun et al. (2020), in the case of day-ahead forecasting of the aggregated load of residential household using photo-voltaic generation as input of the network. Raza et al. (2018) apply neural network ensemble and Bayesian model averaging to predict solar output of photo-voltaic sites. A review of probabilistic forecasting for photo-voltaic production and consumption is given in van der Meer et al. (2018).

The high dimension of load curve data complicates many tasks for analysis. To remedy this, it might be necessary to reduce the dimension of the curves prior to other tasks such as regression or clustering. For instance, Motlagh et al. (2019) reduce the dimension of load curves with a Principal Component Analysis-based method before clustering residential customers. In Cho et al. (2015), Singular Value Decomposition is used to reduce dimension prior to electrical short-term load forecasting. Auder et al. (2018) adopt a wavelet-based approach to reduce the dimension of the curves in order to cluster them.

One of the underlying objective is to correctly estimate and predict for non residential customer their consumption during hours of sunlight, so as to size their potential photo-voltaic installations. Solar power generation has been the subject of numerous studies due to its recent growth, as it is low-carbon and cost of production are decreasing.

In this paper, we focus on non residential customers, typically, businesses such as retail stores, farming and insurance companies etc . Load modeling for non residential customers is challenging due to the large variety of consumers, some having seasonal activity while others are active throughout the whole year. Recent studies on non residential customers' load modeling include a bottom-up heat and electric forecasting. For instance, this approach is developed by Lindberg et al. (2019) for non residential buildings in Norway. A survey of methods applied for load forecasting for commercial buildings can be found in Yildiz et al. (2017).

Often, the need to predict a subgroup's behavior arises. This subgroup can be underrepresented in the dataset at hand and similar to the prevailing subgroup but differing in some aspects. For instance, if we consider a dataset containing small and medium businesses (SMB) and large industrial sites, where the SMB are underrepresented compared to the large customers. If the sub-population of interest is the SMB, we can still use both sub-populations (SMB and larger businesses) to predict the behavior of the SMB. We

detail subsequently an approach of transfer learning dealing with a similar scenario, in the context of electrical load curve prediction.

Subsequently, the following aspects are considered:

- Estimating a multi-target regression model to explain the individual load curves using only billing information. Three modeling frameworks are developed to that end. One is based on directly estimating the model in high dimension with neural networks.

  The two others are based on reducing the dimension of the load curve space, and the reduced load curve is modeled from the billing information using deep models such as residual neural networks, Bayesian neural networks and deep Gaussian processes.

  For dimensionality reduction, we compare an autoencoder and the Discrete Wavelet Transform. Both methods allow for reconstruction of the load curve into the original space. The two frameworks involving dimensionality reduction differ on this aspect: one requires reconstructing the curve to conduct prediction while the other does not.
- Predicting the load curve of a new individual in a catalog of existing load curves only using the available billing information.

  We do so to integrate an industrial constraint to this study
- Evaluating the quality of the modeling strategies by giving more emphasis on certain time periods. We introduce a loss function for evaluation or optimization of the models, that penalizes hours out of hours of sunlight.

  We use this loss function to integrate an industrial constraint to our study. However we could imagine other scenarios where penalizing certain periods of time can be relevant for prediction.
- Transferring information when the subgroup of interest is under-represented in the whole population.
- Building prediction intervals on individual load curves. From the Bayesian neural networks, we obtain prediction intervals. Two methods for building the intervals are described. One of them is based on the reconstruction properties of autoencoder.

Section 3.2 describes how prediction of a new customer's load curve is carried out in the catalog of existing customers, as well as the loss function designed to put an emphasis on certain time periods. Section 3.2.4 outlines the three modeling frameworks applied to first estimate the relationship between the load curves and the billing variables and

then predict the curve for a new individual for which billing information are known. The frameworks are adapted from those described in Chapter 2, two of which include an additional prediction step in the catalog of existing customers. Results, technical details and a brief description of the dataset are specified in Section 3.3. Section 2.5 describes the probabilistic prediction intervals using the Bayesian posterior predictive distribution ensuing from the Bayesian neural networks obtained for this application. Section 3.5 is dedicated to the conclusion and perspectives.

Hereinafter, we use the same notations as in Chapter 2.

3.2 Prediction of a load curve with industrial stakes

The main purpose is to predict a new customer's load curve $\mathbf{X}_{new}$ using only its billing information, also referred to as features, $\mathbf{V}_{new}$.

For the purpose of this study, two strong industrial constraints are imposed:

- An emphasis is put on correctly predicting high consumption phases during hours of sunlight,
- the prediction of $\mathbf{X}_{new}$ must belong to the catalog of existing clients $\mathcal{X}$.

Remark 7 *Those stakes are taken into account in the context of a specific industrial application. However, they can easily be generalized to other setups.*

For instance, predicting in a catalog can be relevant in other domains (e.g. text generation).

As for giving more emphasis to specific period of times when working with time series, it can be adapted for example to the prediction of occupancy rate of public transports.

3.2.1 Solar loss function

An industrial stake is to correctly predict high consumption phases during the day. Specifically, we want to predict electrical consumption during hours of sunlight. This industrial constraint can be interesting to consider, for example, for sizing the potential photovoltaic installations of a new customer. In Figure 3.1, areas of interest are illustrated for one non residential customer's load curve. Consumption peaks also occur during the night, yet we are interested in consumption peaks during the day as they match solar power production peaks.

Considering Y and $\hat{Y}$ are respectively a load curve and its prediction, both of size n. The mean absolute error (MAE):

$$\mathcal{E}_{MAE}(Y, \hat{Y}) = \frac{1}{n} \sum_{i=1}^{n} |Y_i - \hat{Y}_i|, \tag{3.1}$$

gives the same weight to the errors independently of the instant.

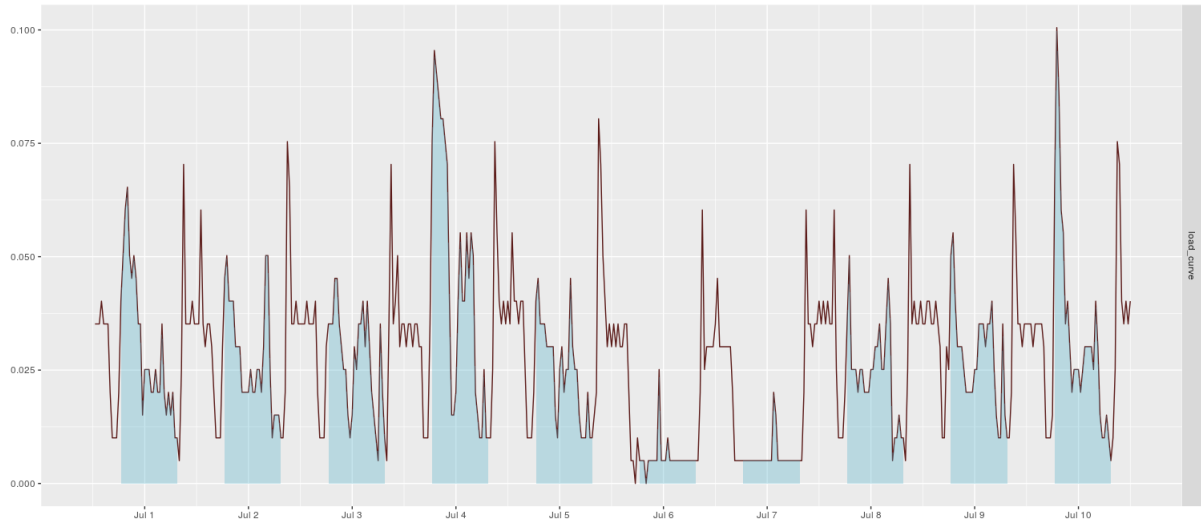


Figure 3.1 – Load curve of one customer for ten days (brown), areas highlighted (blue) show the hours of sunlight, typically between 6 a.m. and 20 p.m. .

We build a weighted loss function where the weights are adapted to take into account hours of sunshine. Our approach consists in using solar power generation data collected from several solar power plants at half hourly period over a year. The weights are then obtained by aggregating and normalizing these curves, so that their sum equals 1. These weights $(w_i^{sol})_{1 \leq i \leq n}$, named solar weights, are such that $w_i^{sol} \in [0, 1]$ and $\sum_{i=1}^{n} w_i^{sol} = 1$. We define the weighted loss function named solar MAE by:

$$\mathcal{E}_{sol}(Y, \hat{Y}) = \sum_{i=1}^{n} w_i^{sol} |Y_i - \hat{Y}_i|, \tag{3.2}$$

As illustrated in Figure 3.2, the weights thus obtained are equal to zero at night and strictly greater than zero during certain periods of the day, generally between 9 a.m. and 6 p.m. in the winter and between 6 a.m. and 10 p.m. in the summer. They reach a maximum during the summer, as sunlight is at its highest.

The solar MAE, introduced in (3.2), is used subsequently as the loss function to

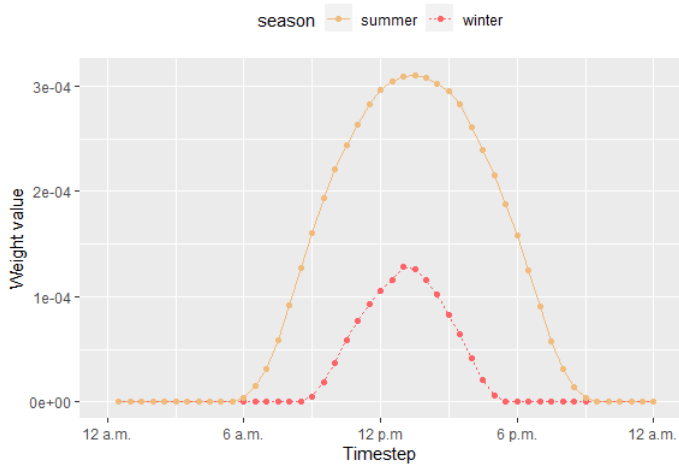


Figure 3.2 – Solar weights for January 1 [Red] versus July 1 [Yellow]

optimize during training of some of our models such as neural networks (see Section 2.3.1) and autoencoders (see Section 2.4.2). It is also used as our evaluation method to compare all the modeling schemes.

3.2.2 Prediction under industrial constraint

Recall that we want the predicted load curves to belong to the catalog of load curves $\mathcal{X}$. To solve this constraint, we consider the following scenarios:

- the forecasting method provides a predicted load curve that does not belong to $\mathcal{X}$,
- the prediction is calculated after reducing the dimension of the load curves.

Such forecasting methods are described in Section 2.3, in Chapter 2.

Direct prediction of the load curve X

Assuming that a forecasting method provides $\hat{\mathbf{X}}_{new}$, a predicted load curve of $\mathbf{X}_{new}$,(see Sections 2.3.1 and 2.3.2), we search for a nearest neighbor in the library of existing load curves $\mathcal{X} = (\mathbf{X}_k)_{1 \leq k \leq m}$ using the predicted load curve.

We minimize the solar MAE, defined in (3.2):

$$\mathcal{E}_{sol}(Z, \hat{\mathbf{X}}_{new}), \text{ with } Z \in \mathcal{X}$$

Since $\mathcal{X}$ is finite ($|\mathcal{X}| = m$), the minimum always exists, but not necessarily unique. However, for all considered forecasting method, $\hat{\mathbf{X}}_{new}$ is random vector with continuous distribution. And so, the minimum is unique almost surely. Therefore, the argmin is well defined almost surely.

We denote:

$$\hat{k}_X := \underset{1 \leq k \leq m}{\operatorname{argmin}} \, \mathcal{E}_{sol}(\mathbf{X}_k, \hat{\mathbf{X}}_{new}). \tag{3.3}$$

Finally, the predicted load curve obtained from $\hat{X}_{new}$ is $\mathbf{X}_{\hat{k}_X}$.

Prediction of the load curve in a reduced space

Due to the large dimension of the curves, it is often challenging to directly estimate and predict the load curves. A classical approach consists in reducing their dimension. Then, we use statistical methods to predict the reduced representation of $\mathbf{X}_{new}$ denoted $\hat{\mathbf{I}}_{new}$ (see Section 2.4.2).

We denote $\mathcal{I} = (\mathbf{I}_k)_{1 \leq k \leq m}$ the reduced representation of the catalog of existing curves $\mathcal{X} = (\mathbf{X}_k)_{1 \leq k \leq m}$. We search for the nearest neighbor's load curve in the catalog of reduced load curves $\mathcal{I}$. We minimize the MAE, defined in (3.1):

$$\mathcal{E}_{MAE}(Z, \hat{\mathbf{I}}_{new}), \text{ with } Z \in \mathcal{I}$$

Since $\mathcal{I}$ is also finite ($|\mathcal{I}| = m$), the same argument ensures that the argmin is well defined almost surely.

Remark 8 *For the choice of the distance in the latent space, we take $\mathcal{E}_{MAE}$. We could have considered taking a weighted distance, using the an encoding of the solar weights w_i^{sol}, however, in practice this method does not outperform the one with the MAE.*

We denote

$$\hat{k}_I = \underset{1 \leq k \leq m}{\operatorname{argmin}} \, \mathcal{E}_{MAE}(\mathbf{I}_k, \hat{\mathbf{I}}_{new}) \tag{3.4}$$

Then, the load curve of the new customer $\mathbf{X}_{new}$ is predicted by $\mathbf{X}_{\hat{k}_I}$.

Remark 9 *This second approach does not take into account the constraint coming from the solar MAE. If we can reconstruct* $\hat{\mathbf{X}}_{new}$ *from* $\hat{\mathbf{I}}_{new}$*, then we have*

$$\mathcal{E}_{sol}(\hat{\mathbf{X}}_{new}, \mathbf{X}_{\hat{k}_X}) \leq \mathcal{E}_{sol}(\hat{\mathbf{X}}_{new}, \mathbf{X}_{\hat{k}_I})$$

3.2.3 Diversity of the catalog

When seeking the nearest neighbor, whether using (3.3) or (3.4), the quality of prediction is strongly dependent on the quality and the diversity of customers available in the catalog.

One way to evaluate the quality of the prediction is to calculate the distance between $\mathbf{X}_{new}$ and the catalog i.e.:

$$\min_{1 \leq k \leq m} \mathcal{E}_{sol}(\mathbf{X}_k, \mathbf{X}_{new}). \tag{3.5}$$

This is its optimal theoretical prediction error under the industrial constraint. The quality of the prediction can be calculated on the testing data subset.

On the analyzed dataset in Section 3.3.1, the optimal solar MAE obtained has a median of 0.296 and a mean of 0.339. This error is irreducible considering that it depends on the database available. It can be seen as an indicator of the diversity of the database and how much individuals are represented among it.

3.2.4 Adaptation of the modeling frameworks to the industrial constraint

Recalling the modeling framework introduced in Chapter 2, we adapt each of them to the industrial constraint described here.

The MNR and ENR-R modeling strategies both provide $\hat{\mathbf{X}}_{new}$, a predicted load curve of $\mathbf{X}_{new}$. The final prediction is taken from the catalog of existing load curves as seen in (3.3). Both modeling strategies are outlined in Figure 3.3 and Figure 3.4 respectively.

The ENR modeling strategy remains the same as described in Chapter 2, as the prediction is conducted in the latent space and a direct prediction $\hat{\mathbf{X}}_{new}$ is therefore unavailable.

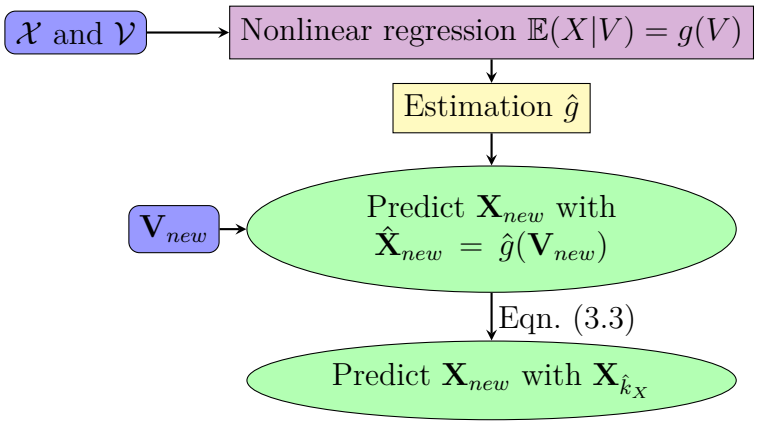


Figure 3.3 – Outline of the MNR framework (Section 2.3.1), adapted for the industrial constraint on prediction (Section 3.2.2). Blue boxes specify the inputs (variables) of the models, pink boxes the processes/models, yellow boxes the estimations and green boxes the predictions.

3.3 Application to industrial data

All of our experiments are conducted with the R software (R Core Team (2019)). Specifically, we use the wavelets package (Aldrich (2020)) to apply the Discrete Wavelet Transform for dimensionality reduction. For the autoencoders and the non Bayesian neural networks we use Tensorflow (Abadi et al. (2015)) and Tensorflow Probability (Dillon et al. (2017) and Tran et al. (2019)) for the Bayesian neural networks and deep Gaussian processes.

3.3.1 Dataset

As said in Section 3.1, the dataset contains individual load curves of non residential customers and the related customers' billing information.

The load curves are measured over a year, at half hourly period, meaning that for each customer in the database, their load curve contains $n = 17472$ datapoints. The billing information contain a mix of categorical and continuous variables, such as peak and off peak hours ratio for each month of the year and the French Classification of Economic Activities which specifies the business activity of the customers.

The database contains approximately 200 features after one hot encoding of the categorical variables. Billing information are normalized using min-max scaling, so that theirs values lie between 0 and 1. Scaling the data is essential before using neural networks as

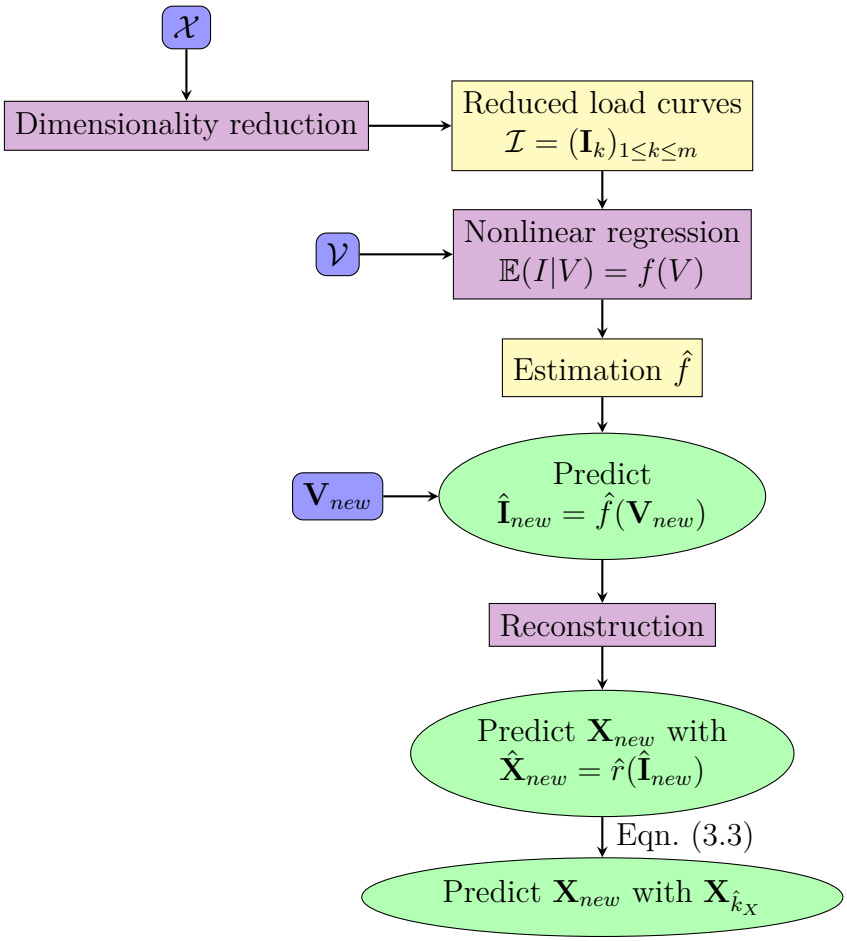


Figure 3.4 – Outline of the ENR-R strategy (Section 2.3.2), adapted for the industrial constraint on prediction (Section 3.2.2). Blue boxes specify the inputs (variables) of the models, pink boxes the processes/models, yellow boxes the estimations and green boxes the predictions.

it allows faster convergence of the algorithm.

We want to predict full load curves, at half hourly period over a year, for a specific group of non residential customers. Those customers are segmented according to their contract power in two categories:

- $C4$ customers correspond to power between 37 and 250kVA,
- $C2$ customers correspond to power over 250kVA.

For all the customers, the database contains billing information (e.g. business sector) and historical consumption data.

The subgroup of interest is the $C4$ customers and we want to predict those cus-

tomers' load curves, using only their billing information. A major difficulty is their under-representation in the data-base. In fact, the $C4$ customers represent 7% of the total of the dataset, while the remaining 93% are $C2$ customers.

To remedy this lack of observation, we take advantage of their resemblance with the $C2$ customers.

While $C2$ and $C4$ customers share many similarities, some differences subsist, such as off-peak hours span for the two segments. Since $C4$ are underrepresented, if the training data is constituted of both categories, disparities will not be taken into account. We propose a strategy based on transfer learning to solve this problem. The $C2$ customers are used to learn the models initially and $C4$ customers in the training dataset are used for transfer learning on those pretrained models.

We have a strong industrial stake, which is to predict the load curves of the $C4$ customers by real load curves taken from a catalog of existing $C2$ as well as $C4$ customers. The load curves constituting the catalog are all scaled since we are not interested in the consumption volume here rather than the load curve profile.

Both categories of customers, $C2$ and $C4$, have very heterogeneous consumption curves as can be seen in Figure 3.5. Our study is based on the assumption that both categories have similar consumer behaviors, however seeing Figure 3.5, the original load curves (in brown) have very different shapes. Weighting those load curves by the solar weighted described in Section 3.2 allows the load curves to show less disparities as seen on the weighted load curves displayed in yellow in Figure 3.5.

Except for Figure 3.5, where we show both the original load curves and the weighted load curves using the solar weights described in Section 3.2, hereinafter, we display only the weighted load curves, in accordance with the industrial stake.

3.3.2 Transfer Learning - fine-tuning

Transfer learning pertains to a variety of methods aiming at improving performances when learning a certain task, the target, using knowledge gained from learning a related task the source. Surveys on transfer learning can be found in Torrey and Shavlik (2010) and Pan and Yang (2009).

The use of transfer learning in this study is motivated by the fact that $C2$ and $C4$ share some similarities, such as the shape of some of the load curves, their main difference being the off peak hours period.

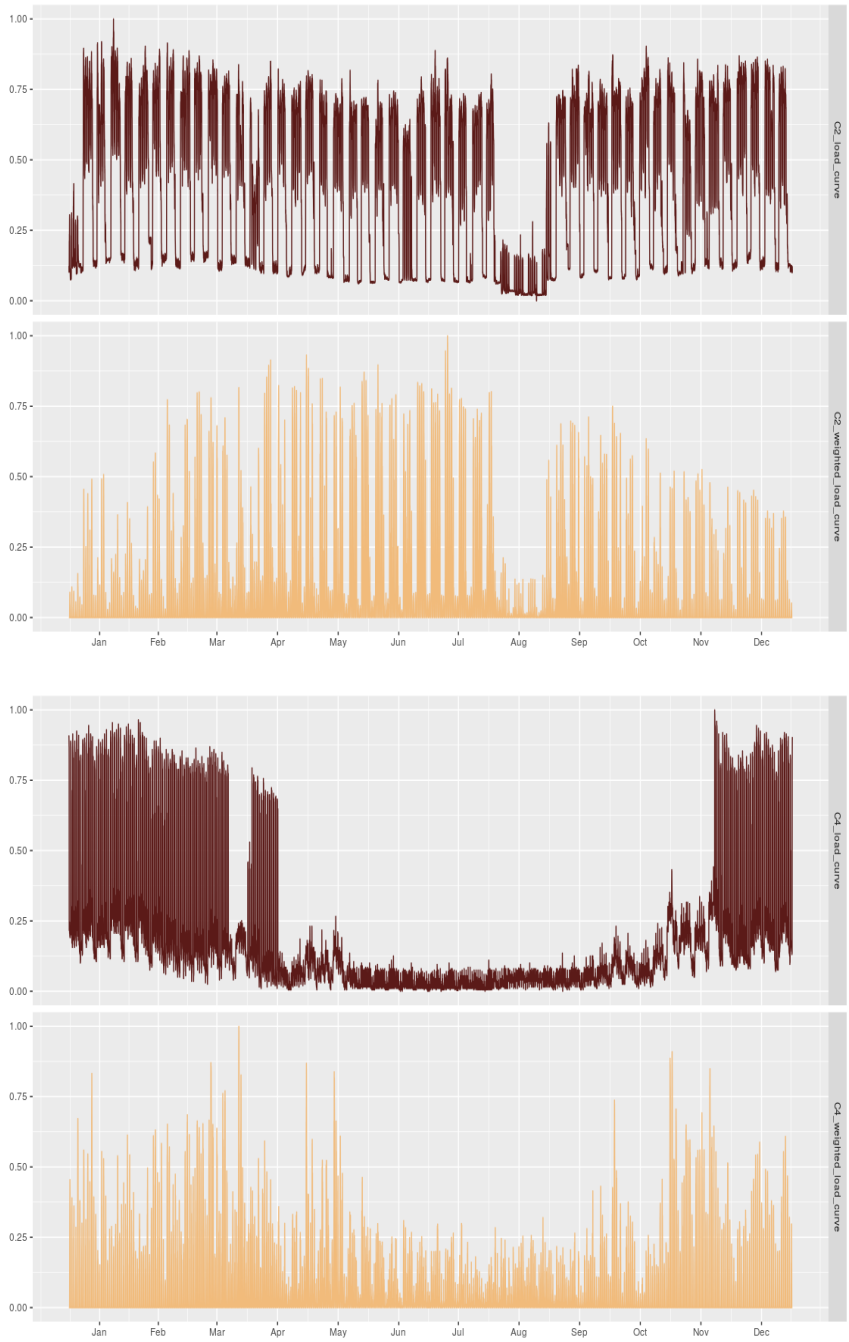


Figure 3.5 – The original load curves of one $C2$ customer [Top] and $C4$ customer [Bottom] are displayed in brown. In yellow, the customer's load curve weighted by the solar weights for the $C2$ customer [Top] and the $C4$ customer [Bottom].

In our setting the source domain consists of $C2$ customers while the target domain contains only $C4$ customers.

- Learning the models on only the $C2$ customers and testing them on the $C4$ customers does not work well because, despite the many similarities between the two categories, some differences remain.
- Learning the models on a mix of $C2$ and $C4$ customers does not work either, because the size of the $C4$ subset is too small to have an impact on training.
- Learning the models only on the $C4$ customers is also inadequate due to the size of the dataset. In fact, training on too little data results in unstable optimization and thus leads to poor results. Another drawback of learning models on a small dataset is overfitting, because the number of parameters to estimate is much larger than the number of observations available for this estimation.

To solve this problem, we propose the use of a transfer learning method. Specifically, we focus on fine-tuning, a special case of transfer learning. It consists in using pretrained weights of a neural network, trained on the source domain, as weight initializers when learning the aforementioned network on the target domain. The initialization of the weight is very informative in this approach because it allows the model to benefit from the similarities of the two domains. An application of fine-tuning to an autoencoder is given in Hinton and Salakhutdinov (2006).

The methodology we adapt is the following:

1. **Train** : training is done in two steps.

   we first train the models on the subgroup of $C2$ consumers. The $C4$ subgroup (7% of the complete dataset) is split so that a subset of those $C4$ customers is used as a training subset to continue training the pre-trained models.

2. **Test** : the remaining $C4$ customers constitute the testing subset, distinct from the $C4$ training subset. All the strategies are tested on those remaining $C4$ customer.

3.3.3 Dimensionality reduction

We implement two methods of dimensionality reduction. The first method uses an autoencoder build with eight Dense layers, where layers 1 to 4 constitute the encoding part, while the decoding part is made of layers 5 to 8. The detailed architecture of the autoencoder is displayed in Table 3.1.

The second method uses the discrete wavelet transform. It consists in a hierarchical representation of the initial signal, for instance time series, all generated through dilation

and translation operations. Considering the time series $X(t)$, it can be expressed as in (2.10).

All the models tested take the original load curves as inputs.

Table 3.1 – Architecture of the chosen autoencoder, used in both the ENR-R and ENR methodology schemes described in Section 2.3

Architecture of the neural network			
Layers	Type	Number of hidden units	Activation function
Hidden 1	Dense	1000	ReLU
Hidden 2	Dense	500	ReLU
Hidden 3	Dense	100	ReLU
Hidden 4	Dense	30	ReLU
Hidden 5	Dense	100	ReLU
Hidden 6	Dense	500	ReLU
Hidden 7	Dense	1000	ReLU
Output 8	Dense	17472	Linear
Optimisation / Training			
Loss	Optimizer	Number of epochs	Batchsize
Solar MAE	Adam	100	64

Remark 10 *Due to the ReLU activation function, the latent space* $\mathbf{I}$ *obtained the encoding step of the autoencoder contains several zero inflated variables. In the first and second framework, those columns are removed before the modeling of the reduced load curve space. Hence, the activation function has a thresholding role, it is also relevant to note that, when using the hyperbolic tangent instead, a similar behaviour is observed, as some columns are not zero valued but constant close to zero.*

Remark 11 *The Dense layers, used for the autoencoder, do not take into account the temporal dependency of the load curves. However, the use of the solar weights allows to integrate some temporal structure into solar MAE, as they put an emphasis on certain periods of the day, over the entire curve.*

The four following dimensionality reduction methods are trained on the $C2$ load curves and tested on the $C4$ testing subset:

- The autoencoder trained with $\mathcal{E}_{sol}$, as described in Table 3.1, without transfer learning using the $C4$ training subset of load curves.

- The autoencoder trained with $\mathcal{E}_{sol}$, as described in Table 3.1, with transfer learning using the $C4$ training subset of load curves.
- An autoencoder with the same architecture as in Table 3.1, but trained with $\mathcal{E}_{MAE}$ on the $C2$ original load curves.
- The Daubechies 4 Wavelet Transform, with 10 levels of decomposition trained on the $C2$ original load curves.

Remark 12 *The discrete wavelet transform can also be applied on the load curves weighted by the solar weights, however, we obtain higher errors in this case, thus we do not present the results associated here.*

Remark 13 *In this setup, we decompose the load curves* 2^{10} *times with the discrete wavelet transform, thus obtaining reduced load curves of length* 32*. Reconstructing the time series is done using (*(2.10)*) and setting coefficients at lower decomposition levels to* 0*.*

Table 3.2 – $\mathcal{E}_{sol}(\hat{\mathbf{X}}_{new}, \mathbf{X}_{new})$ obtained through the reconstruction of the load curve by the various autoencoders and the discrete wavelet transform on the $C4$ testing subset and on the $C2$ training subset. The minimum median and mean errors for each subset are signaled with an asterisk.

$C4$ testing subset		
	Median	Mean
Autoencoder trained with $\mathcal{E}_{sol}$, without fine tuning	0.239	0.258
Autoencoder trained with $\mathcal{E}_{sol}$, with fine tuning	0.223*	0.248*
Autoencoder trained with $\mathcal{E}_{MAE}$	0.282	0.304
Wavelets	0.496	0.538
$C2$ subset		
	Median	Mean
Autoencoder trained with $\mathcal{E}_{sol}$, without fine tuning	0.153*	0.184*
Autoencoder trained with $\mathcal{E}_{sol}$, with fine tuning	0.178	0.217
Autoencoder trained with $\mathcal{E}_{MAE}$	0.193	0.231
Wavelets	0.382	0.480

Results of the dimensionality reduction are given in Table 3.2. On both the $C2$ training and $C4$ testing subset, the autoencoders' performances are better than the Discrete Wavelet transform ones. Performing transfer learning on the autoencoder trained with the solar MAE seems to improve results on the $C4$ subset but deteriorates them on the $C2$ subset.

On the left of Figure 3.6, we display an example of $C4$ customer's weighted load curve (in yellow) against its reconstruction using the autoencoder trained with the solar MAE and fine-tuning and using the Discrete Wavelet transform. The shape of the weighted reconstruction from the autoencoder resembles heavily the customer's weighted load curve, while the shape of the weighted DWT reconstruction is quite different from the curve.

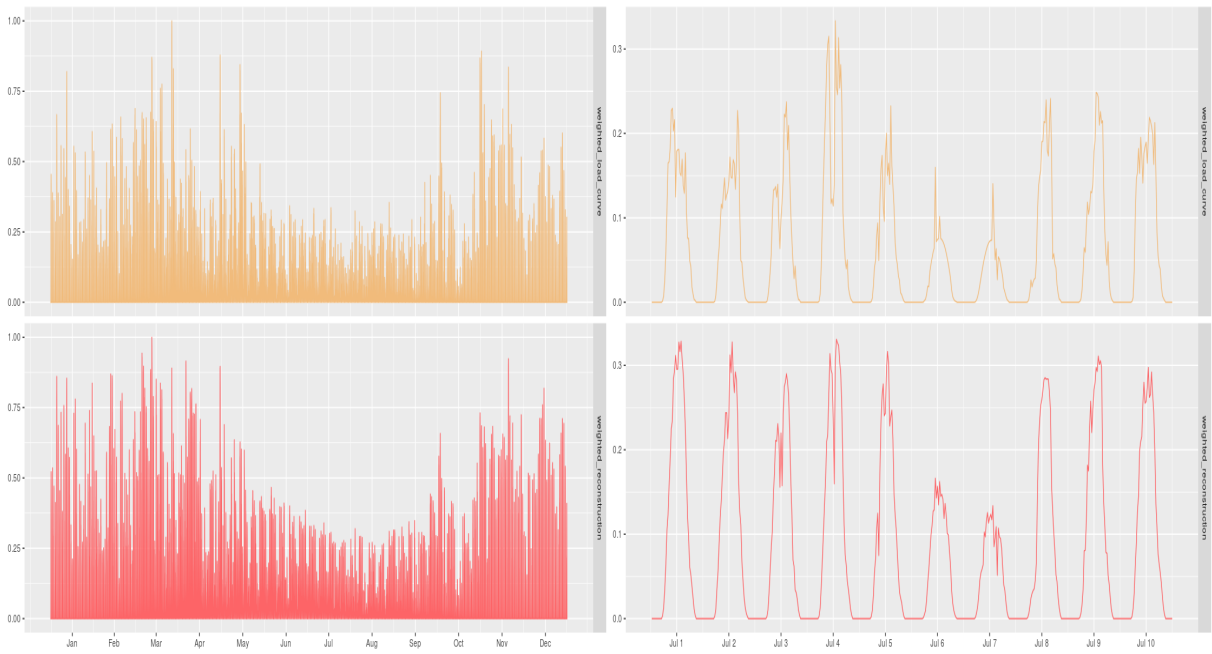


Figure 3.6 – Load curve weighted by the solar weights [Yellow] of one $C4$ customer and its weighted reconstruction [Red] using the solar autoencoder with transfer learning. The load curve and both reconstruction are displayed for the whole year [Left] and zoomed for ten days in July [Right].

Zooming on the load curve for the ten first days of July (on the right of Figure 3.6), we can see that the reconstruction provided by the Discrete Wavelet Transform has a larger amplitude than both the weighted load curves (in yellow) and the reconstruction obtained from the autoencoder.

We choose to model the latent space obtained with autoencoder trained with $\mathcal{E}_{sol}$ with transfer learning for both the ENR-R and ENR frameworks, and the results presented

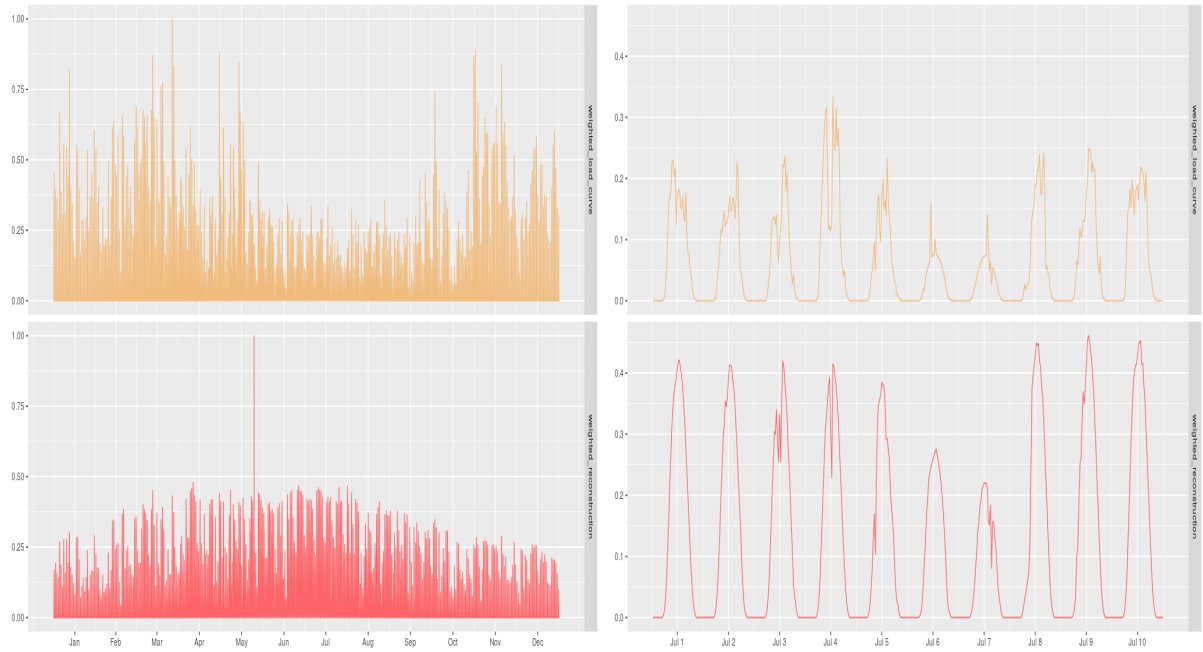


Figure 3.7 – Load curve weighted by the solar weights [Yellow] of one $C4$ customer and its weighted reconstruction [Red] using the Discrete Wavelet Transform [Bottom]. The load curve and both reconstruction are displayed for the whole year [Left] and zoomed for ten days in July [Right].

further in the paper are obtained using the latent space from this autoencoder.

Remark 14 *The solar MAE is not taken into account when applying the DWT, thus performances obtained from this method are overall less satisfactory.*

3.3.4 MNR framework

The models used for this modeling strategy, described in Section 2.3.1, are the four feed-forward neural networks named $\mathbf{NN}_i\mathbf{MNR}$, where $i \in \{1, 2, 4, 6\}$ is the number of hidden layers.

Results obtained with the MNR modeling scheme are presented in Table 3.3. Those results highlight the utility of transfer learning as, for each of the four neural networks tested, the median and average solar MAE are undoubtedly improved by the addition of fine-tuning.

The level of improvement differs for the four models tested as the best model without fine-tuning, $\mathbf{NN}_4\mathbf{MNR}$, yields the worst results after fine-tuning.

Table 3.3 – $\mathcal{E}_{sol}(\mathbf{X}_{new}, \mathbf{X}_{\hat{k}_X})$ obtained with the models tested for the MNR scheme on the $C4$ testing subset. Here as well, the minimum median and mean errors for each case are signaled with an asterisk.

Without fine-tuning		
	Median	Mean
$\mathbf{NN}_1$MNR	1.642	1.644
$\mathbf{NN}_2$MNR	1.700	1.654
$\mathbf{NN}_4$MNR	1.532*	1.465*
$\mathbf{NN}_6$MNR	1.542	1.476
With fine-tuning		
	Median	Mean
$\mathbf{NN}_1$MNR	0.553	0.722
$\mathbf{NN}_2$MNR	0.609	0.702
$\mathbf{NN}_4$MNR	0.668	1.101
$\mathbf{NN}_6$MNR	0.547*	0.648*

After using fine-tuning, the **$\mathbf{NN}_6$MNR**, is the model yielding the best results, with an improvement of 99.5% of the median solar MAE.

3.3.5 Comparison of the two methods of prediction in the ENR-R and ENR frameworks

For both the ENR-R and ENR frameworks, the estimation of the reduced load curves is made using the same models. The difference between the two strategies lies in the prediction method. We use the prediction method described in Section 3.2.2 for the ENR-R scheme, where we rely on the reconstruction of the load curve to find the nearest neighbor whereas the ENR scheme's prediction method relies on the reduced load curve to find the nearest neighbor, as shown in Section 3.2.2.

The estimation of the reduced load curve using the billing variables, which both of the ENR-R and ENR frameworks have in common, can be done two ways: frequentist and Bayesian.

For the frequentist modeling, we use the following neural networks, with deterministic parameters.

- **$\mathbf{NN}_1$**: a one hidden layer neural network similar to the one defined in (2.2) and (2.4), where the inputs are **V** the clients features, and the outputs are **I** the reduced load curve space.

- $\mathbf{RN}_1$: a nine hidden layers neural network, built with residual connections as shown in Figure 2.3.

As for the Bayesian modeling, we use the subsequent Bayesian models:

- $\mathbf{BayesNN}_1$: a one hidden layer Bayesian neural network, the Bayesian counterpart of $\mathbf{NN}_1$.
- $\mathbf{BayesRN}_1$: the Bayesian counterpart of $\mathbf{RN}_1$, here all the layers and residual connections are Bayesian.
- $\mathbf{DGP}_2$: a two layers deep Gaussian process.

Remark 15 *In a deep neural network setting, it remains complicated to incorporate prior knowledge to the prior distribution of the network parameters, hence, the chosen prior for the bayesian neural networks' weights in our study is the standard multivariate Gaussian distribution* $\mathbf{W}_p \sim \mathcal{N}(0, \mathrm{Id})$*, where* Id *is the identity matrix. Here the biaises* $\mathbf{b}_p$ *are set to* 0.

In Table 3.4, we display the errors $\mathcal{E}_{sol}(\mathbf{X}_{new}, \mathbf{X}_{\hat{k}_X})$ obtained with the five models listed above, tested for both the autoencoder, trained with the solar MAE $\mathcal{E}_{sol}$ and fine-tuning on the $C4$ training subset, and wavelets for reference.

All the results presented here have to be put in perspective with Section 3.2.3, because all of them are dependent on the diversity of load curves available in the catalog.

For the ENR-R prediction method, fine-tuning improves the performances of all the models, except the $\mathbf{DGP}_2$ applied on the latent space obtained from the autoencoder. Overall, using this prediction method gives out high errors, considering that all of them are above 0.50.

Conversely, fine-tuning degrades results for all of the models with the prediction method applied to the ENR modeling strategy. The Bayesian neural network $\mathbf{BayesRN}_1$ without fine-tuning in the ENR sheme gives out the lowest solar MAE.

Table 3.5 displays the solar MAE obtained with the reconstructed load curves $\mathcal{E}_{sol}(\mathbf{X}_{new}, \hat{\mathbf{X}}_{new})$ in the ENR-R scheme for the $C4$ testing subset. They are much lower than in Table 3.4 for this framework. Here as well, the Bayesian neural network $\mathbf{BayesRN}_1$, this time with fine-tuning applied to the latent space obtained from the autoencoder with transfer learning, provides the lowest reconstruction error.

We can use the estimations provided by the Bayesian neural network $\mathbf{BayesRN}_1$ to build prediction intervals from the approximated posterior predictive distribution. Results obtained are the focus of Section 3.4.

Table 3.4 – $\mathcal{E}_{sol}(\mathbf{X}_{new}, \mathbf{X}_{\hat{k}_X})$ obtained with the models tested for the ENR-R scheme [Left] and $\mathcal{E}_{sol}(\mathbf{X}_{new}, \mathbf{X}_{\hat{k}_I})$ obtained with the models tested for the ENR scheme [Right]. All the results displayed relate to the $C4$ testing subset. Here as well, the minimum median and mean errors for each case are signaled with an asterisk.

Dimensionality reduction method: Autoencoder								
	ENR-R $\mathcal{E}_{sol}(\mathbf{X}_{new}, \mathbf{X}_{\hat{k}_X})$				**ENR** $\mathcal{E}_{sol}(\mathbf{X}_{new}, \mathbf{X}_{\hat{k}_I})$			
	Without fine-tuning		With fine-tuning		Without fine-tuning		With fine-tuning	
	Median	Mean	Median	Mean	Median	Mean	Median	Mean
$\mathbf{NN}_1$	0.755	0.847	0.570*	0.623*	0.422	0.456	0.491	0.539
$\mathbf{RN}_1$	0.792	0.896	0.575	0.754	0.427	0.465	0.502	0.560
$\mathbf{BayesNN}_1$	1.997	1.656	0.633	0.682	0.431	0.462	0.466*	0.499*
$\mathbf{BayesRN}_1$	0.751	0.943	0.611	0.652	0.409*	0.451*	0.503	0.559
$\mathbf{DGP}_2$	0.685*	0.842*	0.894	0.915	0.451	0.490	0.984	1.004
Dimensionality reduction method: Wavelets								
	ENR-R $\mathcal{E}_{sol}(\mathbf{X}_{new}, \mathbf{X}_{\hat{k}_X})$				**ENR** $\mathcal{E}_{sol}(\mathbf{X}_{new}, \mathbf{X}_{\hat{k}_I})$			
	Without fine-tuning		With fine-tuning		Without fine-tuning		With fine-tuning	
	Median	Mean	Median	Mean	Median	Mean	Median	Mean
$\mathbf{NN}_1$	0.535	0.576	0.524	0.566	0.632	0.890	0.560	0.632
$\mathbf{RN}_1$	0.530	0.590	0.524	0.572	0.750	0.815	0.653	0.688
$\mathbf{BayesNN}_1$	0.535	0.576	0.535	0.576	0.748	0.839	0.791	0.980
$\mathbf{BayesRN}_1$	0.535	0.576	0.524	0.565*	0.592	0.649	0.612*	0.679*
$\mathbf{DGP}_2$	0.524*	0.565*	0.523*	0.566	0.489*	0.540*	0.792	0.790

3.4 Probabilistic prediction intervals using the Bayesian posterior predictive distribution : numerical results

As seen in Table 3.4 and Table 3.5, the Bayesian neural network **BayesRN1** gives out both the lowest errors $\mathcal{E}_{sol}(\mathbf{X}_{new}, \mathbf{X}_{\hat{k}_I})$ for the ENR scheme, and $\mathcal{E}_{sol}(\mathbf{X}_{new}, \hat{\mathbf{X}}_{new})$ for the ENR-R scheme. Hence, in this Section, we present the prediction intervals obtained from the sample generated by the approximated predictive posterior distribution of the **BayesRN1** model. We have $n = 17472$ the length of each load curve and $J = 1000$ the size of the sample.

In Figure 3.8, we display the boxplots relating to the proportion of 80% prediction intervals containing the true value of the load curve, for each customer's load curve, obtained for both strategies. Boxplots on the left relate to the intervals obtained from the

Table 3.5 – Solar MAE $\mathcal{E}_{sol}(\mathbf{X}_{new}, \hat{\mathbf{X}}_{new})$ obtained with the models tested for the ENR-R scheme on the $C4$ testing subset. Here as well, the minimum median and mean errors for each case are signaled with an asterisk.

Dimensionality reduction method: Autoencoder				
	Without fine-tuning		With fine-tuning	
	Median	Mean	Median	Mean
$\mathbf{NN}_1$	0.624	0.687	0.504	0.547
$\mathbf{RN}_1$	0.670	0.723	0.467	0.527
$\mathbf{BayesNN}_1$	0.519*	0.583*	0.493	0.552
$\mathbf{BayesRN}_1$	0.572	0.639	0.464*	0.526*
$\mathbf{DGP}_2$	0.540	0.590	0.760	0.812
Dimensionality reduction method: Wavelets				
	Without fine-tuning		With fine-tuning	
	Median	Mean	Median	Mean
$\mathbf{NN}_1$	0.559	0.613	0.560	0.617
$\mathbf{RN}_1$	0.530	0.590	0.523	0.572
$\mathbf{BayesNN}_1$	0.533	0.591	0.532	0.589
$\mathbf{BayesRN}_1$	0.514*	0.568	0.533	0.577
$\mathbf{DGP}_2$	0.521	0.565*	0.522*	0.565*

realizations of the sample from the **BayesRN1** model, with and without fine-tuning, by applying the first strategy. Overall, For most clients, the majority of the real load curve is in the prediction intervals obtained with this first strategy, as the median is between 0.85 and 0.90 with and without fine-tuning the **BayesRN1** model. The results shows a slight deterioration when using sample from the **BayesRN1** model with fine-tuning (in pink), this is consistent with the observations made from Table 3.4. As for the second strategy to obtain prediction intervals, which relies on decoding the sample from the **BayesRN1** model, with and without fine-tuning, when reconstructing $\hat{\mathbf{X}}^{pos}_{new_j}$, $j \in 1, \ldots, J$ from $\hat{\mathbf{I}}^{pos}_{new_j}$ obtained from the estimation of the **BayesRN1** model, which is the model with the lowest error $\mathcal{E}_{sol}(\mathbf{X}_{new}, \hat{\mathbf{X}}_{new})$ (see Table 3.5), we calculate for each customer, at each time step $1 \leq t \leq n$, the 80% prediction intervals. Boxplots of the proportion of prediction intervals containing the true value of the load curve, for each customer's load curve are shown on the right on Figure 3.8. Here the median lies between 0.55 and 0.60 with and without fine-tuning the **BayesRN1** model, and the results shows a small improvement when using the realizations of the samplefrom the **BayesRN1** model with fine-tuning (in pink), which is consistent with the observations made from Table 3.5. Looking solely at the proportion of data points of the real curve contained within the intervals, the first

strategy seems to be more efficient than the second one.

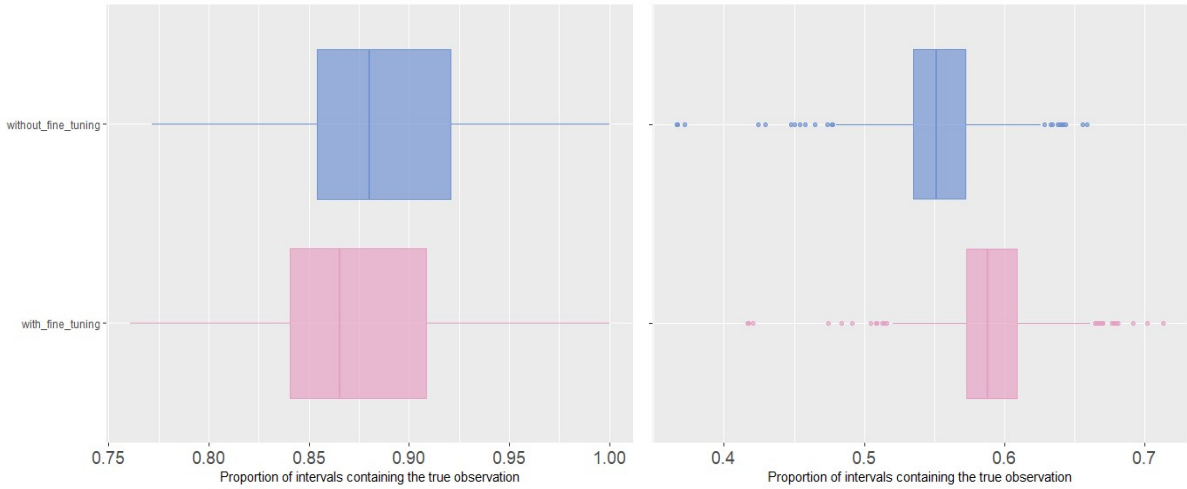


Figure 3.8 – Boxplot of the proportions of the 80% prediction intervals containing the true observation on the $C4$ test subset, obtained with the two strategies. From the sample of the posterior predictive distribution of the **BayesRN1** model (with fine-tuning [Pink] and without fine-tuning [Blue]), [Left] Intervals obtained from the discrete distribution on $\hat{k}_I$ and [Right] from decoding the sample.

However, when we look at the average width of intervals obtained with both methods, which is displayed in Figure 3.9, we can see that intervals obtained with the first strategy are wider, as the median width is between 0.25 and 0.30 whereas it is between 0.17 and 0.20 for the intervals obtained with decoding the sample. Overall, the first method, while having a proportion of intervals containing the true value of the curves larger than the second, also provides intervals that are wider and may be imprecise.

As for the intervals obtained from the first method, they are plotted in Figures 3.10 for the first ten days of January. The intervals obtained for both cases (with and without fine-tuning) are quite wide, seem to follow the shape of the curve better, but may be imprecise. This illustrates what is seen on the average width of intervals displayed on the left of Figure 3.9. The strategy of seeking nearest neighbor to build prediction interval might not be adapted, and is unnecessary in this context as we can free ourselves from this industrial constraint.

Figures 3.11 show the load curve of one customer respectively for the first ten days of January. The prediction intervals obtained by decoding the sample of **BayesRN1** are respectively in blue for the model without fine-tuning and in purple for the model with fine-tuning. The intervals obtained from **BayesRN1** with fine-tuning are slightly smaller

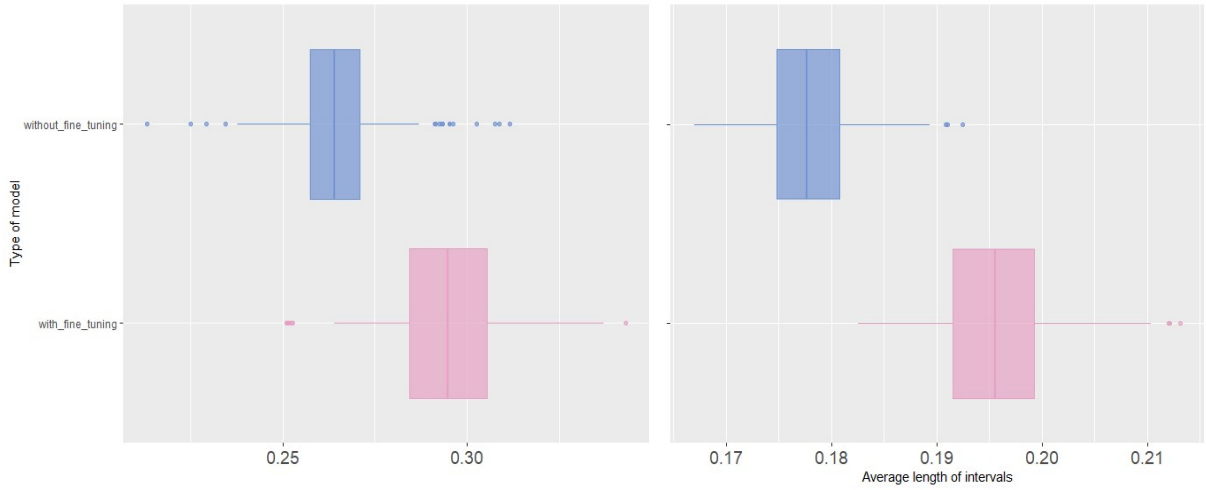


Figure 3.9 – Boxplot of the average interval width of the 80% prediction intervals on the $C4$ test subset, obtained with the two strategies. From the sample of the posterior predictive distribution of the **BayesRN1** model (with fine-tuning [Pink] and without fine-tuning [Blue]), [Left] intervals obtained from the discrete distribution on $\hat{k}_I$ and [Right] from decoding the sample.

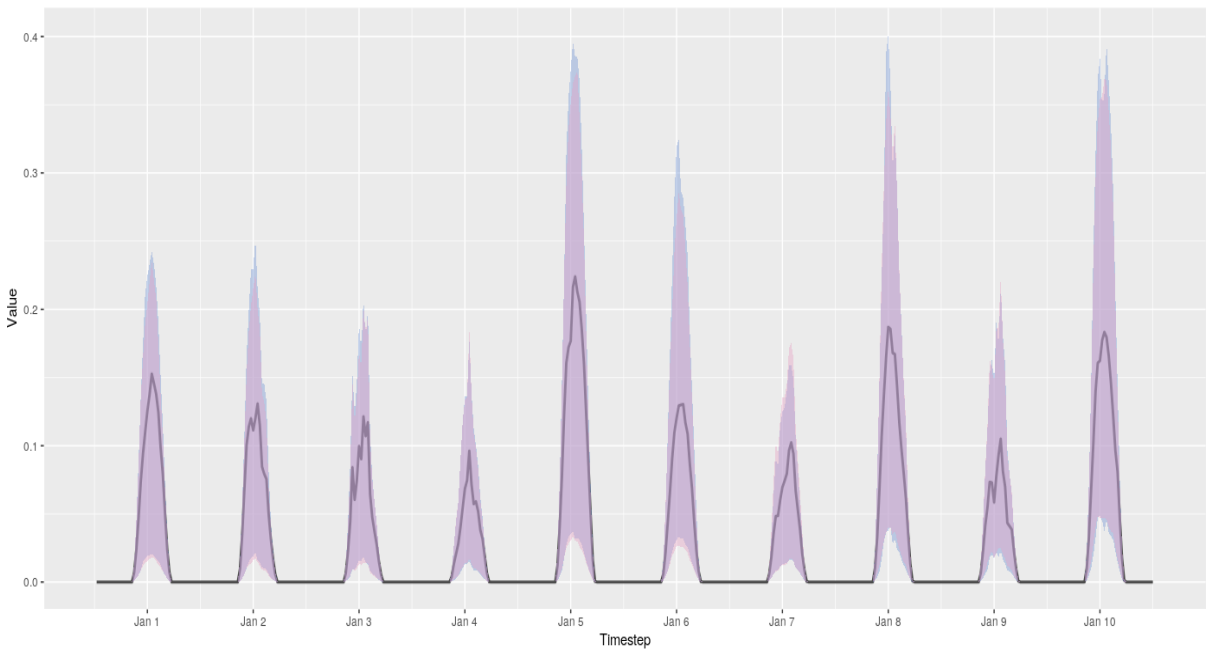


Figure 3.10 – Weighted load curve (black) for the first ten days of January and the prediction intervals associated. In blue the intervals obtained with the discrete distribution on $\hat{k}_I$ from the **BayesRN1** model without fine-tuning, in purple, the intervals obtained from the **BayesRN1** model with fine-tuning.

than without it. Despite not capturing the consumption peaks as well as the previous method, intervals obtained with this method are much smaller. This gives interesting perspectives for future research and application to load modeling.

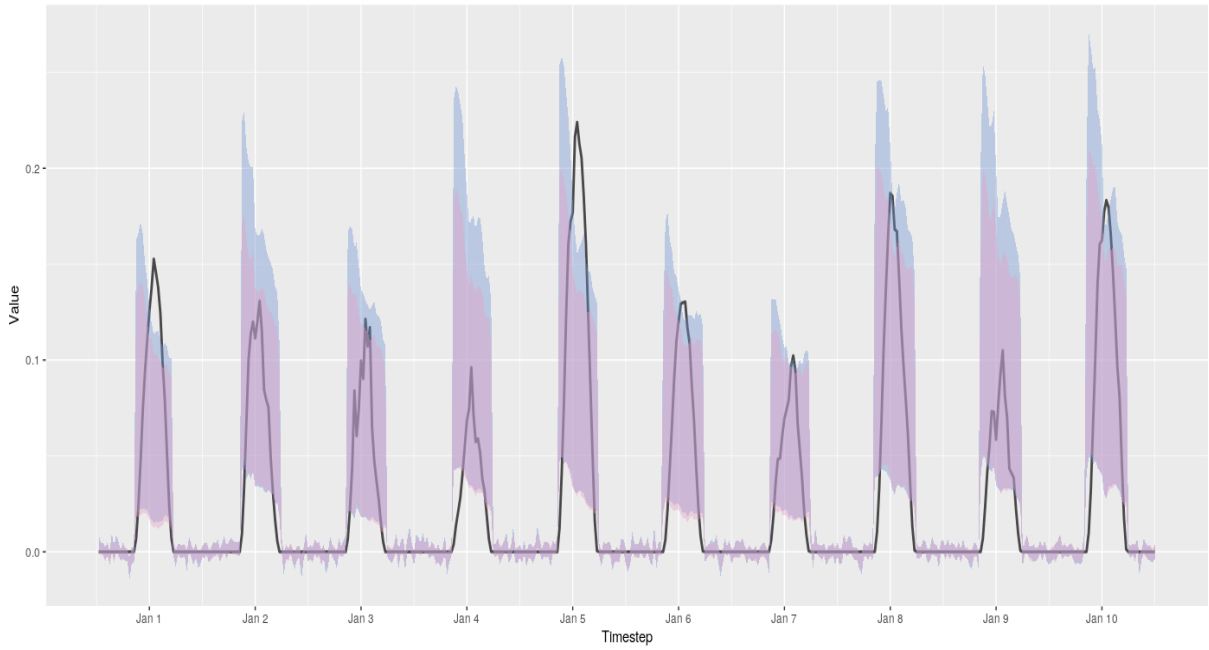


Figure 3.11 – Weighted load curve (black) for the first ten days of January and the prediction intervals associated. In blue the intervals obtained with the decoded sample from the **BayesRN1** model without fine-tuning, in purple, the intervals obtained with the decoded sample from the **BayesRN1** model with fine-tuning.

3.5 Conclusion and future research

We have introduced three different modeling strategies designed to predict the electrical load curves of non residential customers. All of our approaches use fine-tuning, because of the assumption that the two groups of customers share some similarities but are not exactly the same type of consumers.

On the autoencoder, fine-tuning does improve the performances of the model on the $C4$ test subset. For the MNR framework as well, all of the feedforward neural networks' performances benefit from this approach. For the ENR-R and ENR frameworks however, fine-tuning has a positive effect when reconstructing the curve, for all models except $\mathbf{DGP}_2$, but deteriorates slightly the performances of all the models tested when we work

on the reduced load curves. Yet, the ENR framework, out of the three tested is the one that gives out the best results. Specifically, the Bayesian neural network $\mathbf{BayesRN}_1$ trained without fine-tuning on the latent space obtained from the autoencoder trained with fine-tuning is the model that performs the best.

The autoencoders considered are built with Dense layers, thus not taking the time dependency of each load curves into account. Future research may include adapting 1D-convolutional autoencoders to that end. It would be interesting to add the transfer learning step described here to the convolutional autoencoders.

The Bayesian neural network $\mathbf{BayesRN}_1$ model also allows us to build prediction intervals using its approximated predictive posterior distribution. Two types of intervals are thus obtained following either the ENR-R or the ENR scheme. Overall, the intervals obtained from the discrete distribution on $\hat{k}_I$ give large prediction intervals which may be too imprecise, thus for future work, we may prefer the second approach that decodes the sample from the posterior predictive distribution as they are smaller.

The overall quality of the performances obtained is to be put in perspective with the database available whether in diversity of the individuals or the billing information at hand. Since the models only rely on billing information, in the considered features, we lack information on some characteristics of the customers and the buildings. For instance, we do not have information concerning the customers' range of consumption during hours of sunlight. Adding variables relating to that may be a way to improve overall performances of the three frameworks. It might be interesting to include variables such as the dimensions of the building, the geographical localization or meteorological information, if they are available. The additional features may be beneficial for potential industrial application such as sizing of photovoltaic installations.

Future research may include adapting the MNR scheme with Bayesian neural networks, to see whether those methods could improve performances, as well as obtaining prediction intervals on the load curves.

In the MNR modeling framework as well, it would be interesting to see why fine-tuning improves each of the models differently. As for the ENR-R and ENR modeling strategies, one could be interested in understanding why transfer learning deteriorates results when estimating the reduced load curves using the customers' features.

Chapter 4

Bayesian transfer learning for panel data

4.1 Introduction

Panel data analysis and modeling are frequently encountered in socio-economic and marketing applications. The specificity of panel data rests on its structure: observations (quantitative or qualitative) are collected over a period of time for several individuals. Thus, panel data analysis pertains to analyzing the attributes of individuals over a period of time. Panel data arise frequently in economics, for examples on the subject refer to Baltagi et al. (2008).

A common issue, arising in industrial applications such as forecasting of individual residential consumption, is the lack of historical data to forecast the behavior of new customers. One way to overcome this issue is to use information and knowledge on existing customers and apply it to new customers. This can be seen as a transfer learning situation where the existing customers constitute the source domain and the new customers of interest the target domain. Transfer learning can be an efficient way to predict behavior of new customers. It has the advantage of reducing training time of models for new observations, and requires a smaller amount of training data when transferring on the source domain to learn the source task.

We propose an approach based on a Bayesian hierarchical model for panel data applied to forecasting of individual residential consumption. Firstly, the hierarchical model is trained with weakly informative prior distribution, on customers with long-term historical data. Secondly, we recover the posterior distribution, and use it to build an informative prior for the Bayesian model trained on a new customer, with short-term historical data. Finally we focus on forecasting the consumption of the new customer, using the posterior distribution recovered after the second step.

Transfer learning generally refers to the context of learning one task and use the

knowledge gained to learn another task. For a review on transfer learning methods in machine learning see Torrey and Shavlik (2010). Transfer learning can also be considered in a Bayesian setting. For instance, Raina et al. (2006) use a Bayesian approach of transfer learning to construct informative priors in the context of binary text classification. In Yu et al. (2005), the authors approach is a Gaussian processes-based hierarchical Bayesian framework for transfer learning. Marx et al. (2005) propose a Bayesian logistic regression for transfer learning. A hierarchical Bayesian transfer approach is developed by Wilson et al. (2012) for reinforcement learning. Recent approaches for transfer learning based on Bayesian modeling include Finn et al. (2018) where a Bayesian hierarchical model is used for few-shot learning.

Our approach is related to the one developed by Launay et al. (2015), based on building an informative hierarchical prior using a long dataset and using it for the estimation of a model on a shorter dataset of interest. We extend the methodology proposed by Launay et al. (2015) to the case of the individual load forecasting of a new customer and panel data. This is a specific case of Bayesian transfer learning applied to panel data modeling.

Galharret and Philippe (2019) also adapt the methodology developed by Launay et al. (2015) to the construction of informative priors in the case of mediation analysis.

Bayesian hierarchical models induce flexibility to modeling because of their multi-leveled structure. Bayesian hierarchical approaches are particularly useful when we lack prior knowledge on the parameter of interest. Bayesian hierarchical models are particularly useful when dealing with complex data. The upper levels of the hierarchy (the hyperparameters) can express the knowledge available, if any, on the global behavior of the population studied.

Because of those advantages, Bayesian hierarchical modeling is encountered in many domains. This is the case in Dyrrdal et al. (2015), where the hierarchical-based approach is applied to the modeling of heavy rainfall. Lionetti et al. (2019) usse hierarchical models for Sensory Processing Sensitivity. In Zhao et al. (2019), a Bayesian hierarchical framework is developed for human activity recognition. Fu et al. (2020) propose such an approach to traffic crash estimation using video data as well as crash data. A hierarchical Bayesian model is presented in Meager (2019) for the impact of microcredit expansions. Papoutsis et al. (2020) adapt a multi-level Bayesian approach to the context of predicting the driver flow for carpooling services.

In the context of panel data, Bayesian hierarchical models are quite appropriate, as they allow to model the relation between all the individuals considered by means of

shared parameters. In Feller and Gelman (2015), the authors discuss the use of Bayesian hierarchical models for causal inference. Congdon (2010) provides details on Bayesian hierarchical models and applications, notably for panel data. Recently, Lee and Bateman (2021) applied a Bayesian hierarchical approach for modeling the demand of organic coffee, using consumer panel data.

Remark 16 *A common issue in a Bayesian setting is the approximation of the posterior distribution due its intractability. This can be achieved numerically using MCMC algorithms.*

The following aspects are subsequently addressed:

— A weakly informative hierarchical Bayesian regression model is adapted for panel data.

— An informative Bayesian regression model is developed for a new panelist with shorter historical data.

— Forecasting of new data points of the individual is conducted using the posterior predictive distribution, extracted after inference of the informative model.

In Section 4.2, we propose a methodology of transfer learning using a Bayesian approach, in the case of panel data analysis. Section 4.3 shows an application of the methodology to three simulated datasets based on the following models: the polynomial regression model, an auto§regressive model and the hierarchical Poisson model. Additional results of the simulation are detailed in the appendix. Section 4.4 is dedicated to the conclusion and discussion of this Chapter.

4.2 Methodology

In this section we describe our methodology based on a Bayesian approach for transfer learning for panel data analysis.

4.2.1 Weakly informative approach

Let n denote the number of individuals considered for which we collect T observations denoted $X_t^{(i)}$ and $V_t^{(i)}$, $i = 1, \ldots, n$ and $t = 1, \ldots, T$, where $X_t^{(i)}$ are the variables of interest and $V_t^{(i)}$ the explanatory variables.

Time series can be viewed as a special case of panel data, for example, $X_t^{(i)}$ can be the electrical consumption of the i-eth customer at time t and $V_t^{(i)}$ customer's features (e.g. the type of heating system, a binary variable equal to 1, if the customer has an electrical heating system and 0 otherwise). Here n is the size of a subset of customers and T the length of historical data available for the subset considered.

We consider the following parametric model of panel data to estimate the relationship between $X_t^{(i)}$, the dependent variable and $V_t^{(i)}$ the predictors:

$$X_t^{(i)} \quad = \quad f^1(\theta^i, V_t^{(i)}) + \epsilon_t^{(i)}, \; i = 1, \ldots, n$$

where $\epsilon_t^{(i)}$ is the individual noise (i) at time t and where θ_i are the individual parameters and f^1 is a parametric function. The goal is to estimate each individual parameters θ^i, $i = 1, \ldots, n$, of the model. For convenience, we denote $\mathcal{D}^1 = (X_t^{(i)}, V_t^{(i)})_{\substack{i=1,\ldots,n \\ t=1,\ldots,T}}$ the observations used to estimate θ^i, $i = 1, \ldots, n$.

We assume that $\theta^1, \ldots, \theta^n$ are i.i.d. with a common distribution depending on some parameter $\theta \in \{Theta\}$. In a Bayesian approach, θ is assumed to be random.

We denote the prior of each θ^i given θ as such:

$$\theta^i|\theta \quad \sim \quad \pi(\theta^i|\theta) \tag{4.1}$$

The parameter θ is common to all the individuals $i = 1, \ldots, n$ and follows a weakly informative prior distribution $\pi_1(\theta)$ also know as the hyperprior.

$$\theta \quad \sim \quad \pi_1(\theta) \tag{4.2}$$

Example 2 *A classical example is to assume that:*

$$\theta^i \quad = \quad \theta + \lambda^i,$$

where $(\lambda^1, \ldots, \lambda^n)$ *are i.i.d. with zero mean. In this case,* θ *can be viewed as a central value of* $(\theta^1, \ldots, \theta^n)$ *since* $\mathbb{E}(\theta^i|\theta) = \theta$.

Remark 17 π_1 *is taken weakly informative here on* θ *for regularization. The goal is to keep the parameters within reasonable range when inferring the model. Usually, the prior distribution considered is a Gaussian probability distribution with large variance.*

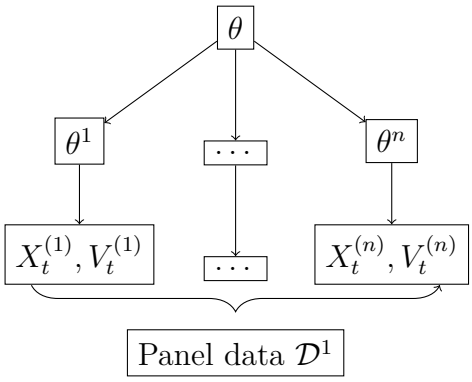


Figure 4.1 – Directed acyclic graph of the hierarchical model for the panel data $\mathcal{D}^1$

Bayesian inference relies on finding the posterior distribution of the parameters given the observations.

Using Bayes's theorem, the joint posterior distribution of the parameters of the model is expressed as follows:

$$\pi_1(\theta^1, \ldots, \theta^n, \theta|\mathcal{D}^1) \quad \propto \quad \pi_1(\theta)\pi(\theta^1|\theta)\ldots\pi(\theta^n|\theta)p(\mathcal{D}_X^1|\theta^1, \ldots, \theta^n, \mathcal{D}_V^1), \tag{4.3}$$

with $\mathcal{D}_X^1 = (X_t^i)_{i=1,\ldots,n,t=1,\ldots,T}$, and $\mathcal{D}_V^1 = (V_t^i)_{i=1,\ldots,n,t=1,\ldots,T}$. $p(\mathcal{D}_X^1|\theta^1, \ldots, \theta^n, \mathcal{D}_V^1)$ is the likelihood of the model:

$$p(\mathcal{D}_X^1|\theta^1, \ldots, \theta^n, \mathcal{D}_V^1) \quad = \quad \prod_{i=1}^{n} p((X_t^i)_{t=1,\ldots,T}|\theta^i, (V_t^i)_{t=1,\ldots,T})$$

In Figure 4.1, we show the Directed acyclic graph (DAG) of the hierarchical model. This represents graphically the conditional independence of each θ^i given θ. This property is important as it allows to express the joint posterior distribution as presented in (4.3).

We make the hypothesis that a new individual will have a similar behavior to the other individuals. Thus we want to transfer the information learned on the panelists to the new individual using θ.

For transfer learning, we are interested in the θ parameter, and specifically the moments of its marginal posterior distribution. θ is the parameter common to all individuals, thus for a new individual, our goal is to use θ and its posterior distribution's moments, as a way to incorporate knowledge into the prior for the parameters of the model of the individual.

The marginal posterior distribution of θ, given the observations, is obtained by integrating the joint posterior over each of the parameters $\theta^i \in \Theta$:

$$\pi_1(\theta|\mathcal{D}^1) \quad = \quad \int_\Theta \pi_1(\lambda^1, \ldots, \lambda^n, \theta|\mathcal{D}^1)\, d\lambda^1 \ldots d\lambda^n, \tag{4.4}$$

where $\pi_1(\cdot|\mathcal{D}^1)$ is defined in 4.3.

The Directly acyclic graph (DAG), shown in Figure 4.1, summarizes the hierarchy of the model. The upper level of the hierarchical model is constituted of the parameter θ, common to all individuals. This parameter symbolizes the global similarity of all the individuals. Therefore, θ is connected on the graph to all the individuals parameters θ^i, $i = 1, \ldots, n$. The lower level of the hierarchy consists in the observed panel data: $X_t^{(i)}$ and $V_t^{(i)}$, $i = 1, \ldots, n$ and $t = 1, \ldots, T$. The DAG shows the conditional independence between individuals given the hyperparameter θ.

4.2.2 Informative approach

After inferring the marginal posterior distribution, specified in Equation (4.4), we have access to its moments. Let $\mu_p^{\mathcal{D}^1}$ the p-eth moment of $\pi_1(\theta|\mathcal{D}^1)$ (e.g. the mean or variance if $p = 1$ or $p = 2$ respectively).

Let X_t^* the dependent variable and V_t^* the predictors of a new individual, $t \in \{t_1, \ldots, t_2\}$, where $t_2 - t_1 < T$. The length of historical data available for the new individual is shorter than the one of the panelists.

We want to estimate the relationship between X_t^* and V_t^*. We consider the following regression model:

$$X_t^* \quad = \quad f^1(\tilde{\theta}, V_t^*) + \epsilon_t^*, \tag{4.5}$$

We adopt a Bayesian approach to estimate the parameter $\tilde{\theta}$ using the historical data of the new individual.

We want to transfer the information gathered from the previous step into the estimation of the model. Thus we use an informative prior for the parameter $\tilde{\theta}$ using the posterior mean $\mu_1^{\mathcal{D}^1}$ and posterior variance $\Sigma^{\mathcal{D}^1}$. Specifically, we want the prior distribution on $\tilde{\theta}$ to

have the following properties:

$$\mathbb{E}(\tilde{\theta}) = \mathbb{E}(\theta|\mathcal{D}^1) = \mu_1^{\mathcal{D}^1} \tag{4.6}$$

and

$$Cov(\tilde{\theta}) \quad = \quad Var(\theta|\mathcal{D}^1) \tag{4.7}$$

We assume that $\tilde{\theta}$ is very close to θ as it would be the case for an average individual in the panel. To loosen up the strong assumption of closeness between the new individual and the panel, we introduce hyperparameters to the model. As in Launay et al. (2015), we assume that there exists parameters of similarity denoted here k and l. They symbolize the similarity between the data of the new individual and the historical data $\mathcal{D}^1$. Let $\pi_2(\cdot)$ denote the prior distribution of $\tilde{\theta}$:

$$\tilde{\theta}|k, l \quad \sim \quad \pi_2(\tilde{\theta}|k, l),$$

Remark 18 *For convenience, the operator K is diagonal and defined by:*

$$K = diag(k).$$

The parameters k follow a prior distribution, denoted $p(k)$, centered around 1. This prior distribution allows for flexibility in the model depending on how similar the new individual is to the panelists.

Following the Gaussian strategy introduced by Launay et al. (2015), the hierarchical prior is as follows:

$$\tilde{\theta}|k, l \quad \sim \quad \mathcal{N}(K\mu_1^{\mathcal{D}^1}, l^{-1}\Sigma^{\mathcal{D}^1}), \tag{4.8}$$

and

$$k|q, r \quad \sim \quad \mathcal{N}(q(1, \ldots, 1,)', r^{-1}Id). \tag{4.9}$$

The priors of q, r and l are denoted respectively $p(q)$, $p(r)$ and $p(l)$, and are taken as:

$$\begin{aligned} q &\sim \mathcal{N}(1, \sigma_q^2), \\ r &\sim \mathcal{G}(a_r, b_r), \\ l &\sim \mathcal{G}(a_l, b_l). \end{aligned}$$

The informative prior thus has the following form:

$$\pi_2(\tilde{\theta}, k, q, r, l) \sim \pi_2(\tilde{\theta}|k, l)p(k|q, r)p(q)p(r)p(l). \tag{4.10}$$

the prior $\pi_2(\tilde{\theta}, k, q, r, l)$ has the following properties, as defined in Launay et al. (2015):

Property 1 *$\pi_2(\tilde{\theta}) = \int \pi_2(\tilde{\theta}, k, q, r, l)\, dk\, dq\, dr\, dl$ admits a moment of order 1 equal to $\mu_1^{\mathcal{D}^1}$.*

Property 2 *Assume that $a_r > 1$ and $a_l > 1$. Under suitable conditions of regularity:*

1. *the correlation of the prior $\pi_2(\tilde{\theta})$ is equivalent to the correlation of the posterior $\pi_1(\cdot|\mathcal{D}^1)$ as T goes to infinity.*
2. *For a fixed T, the correlation of the prior $\pi_2(\tilde{\theta})$ is equivalent to the correlation of the posterior $\pi_1(\cdot|\mathcal{D}^1)$ as σ_q^2 and $\mathbb{E}(r^{-1})$ go to 0.*

The posterior distribution of the parameter $\tilde{\theta}$, given the observations X_t^* and V_t^* is obtained with Bayes' theorem:

$$\pi_2(\tilde{\theta}, k, q, r, l|X^*, V^*) \propto \pi_2(\tilde{\theta}, k, q, r, l)p(X^*|V^*, \tilde{\theta}) \tag{4.11}$$

$$\propto \pi_2(\tilde{\theta}|k, l)p(k|q, r)p(q)p(r)p(l)p(X^*, V^*|\tilde{\theta}). \tag{4.12}$$

where $p(X^*|V^*, \tilde{\theta})$ is the likelihood of the model.

The marginal posterior of $\tilde{\theta}$ is obtained by integrating the joint posterior in Equation (4.12) over the parameters (k, q, r, l).

The DAG of the hierarchical model for the informative approach applied to the new individual with short historical data is displayed in Figure 4.2. The upper levels of the hierarchy are constituted of the hyperparameters q and r of k. The middle of the hierarchy contains the parameters k and l above of the parameter of interest $\tilde{\theta}$. The observations of the new individual $X_t^{(*)}, V_t^{(*)}$ constitute the lower level of the hierarchy.

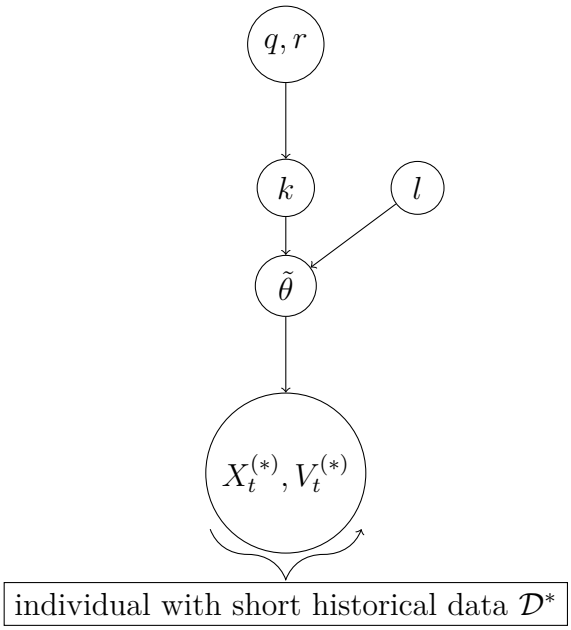


Figure 4.2 – Directed acyclic graph of the hierarchical model for the new individual $\mathcal{D}^*$

4.2.3 Bayesian forecasting

The Bayesian approach for forecasting retains some advantages over its frequentist counterpart. It allows us to get probabilistic forecasts using the predictive distribution and thus obtain credible intervals.

We wish to forecast the values of X_t^* for $t = t_2+1, \ldots, t_2+h$, where h is the forecasting horizon. We consider that the predictors V_t^* for $t_2 + 1 \leq t \leq t_2 + h$ are known.

The posterior distribution of $\tilde{\theta}$ is essential when we want to predict a new data point. Let $\mathcal{D}^*_{past} = (X_t^*, V_t^*)_{t_1 \leq t \leq t_2}$ denote the historical data of the individual (i.e. the training data used for the estimation of the model in Equation (4.5))

The distribution of a new data point X_t^*, given the past, is:

$$\begin{aligned}
p(X_t^*|\mathcal{D}^*_{past}, V_t^*) &= \int p(X_t^*, \tilde{\theta}, k, q, r, l|\mathcal{D}^*_{past}, V_t^*)\, d\tilde{\theta}\, dk\, dq\, dr\, dl \\
&= \int p(X_t^*|\tilde{\theta}, k, q, r, l, \mathcal{D}^*_{past}, V_t^*)\pi_2(\tilde{\theta}, k, q, r, l|V^*, \mathcal{D}^*_{past})\, d\tilde{\theta}\, dk\, dq\, dr\, dl \\
&= \int p(X_t^*|\tilde{\theta}, V_t^*)\pi_2(\tilde{\theta}, k, q, r, l|\mathcal{D}^*_{past})\, d\tilde{\theta}\, dk\, dq\, dr\, dl \\
&= \int_{\tilde{\Theta}} p(X_t^*|\tilde{\theta}, V_t^*)\pi_2(\tilde{\theta}|\mathcal{D}^*_{past})\, d\tilde{\theta}. \qquad (4.13)
\end{aligned}$$

The distribution of a new data point X_t^*, in Equation (4.13), can provide point esti-

mates as well as intervals.

A common point estimate is the Maximum A Posteriori (MAP) which is the value $\hat{X}^*_{MAP\ t}$ that maximizes the distribution:

$$\hat{X}^*_{MAP\ t} \quad = \quad \text{argmax}_{X^*_t}(p(X^*_t|\mathcal{D}^*_{past}, V^*_t)).$$

4.2.4 Summary of the methodology

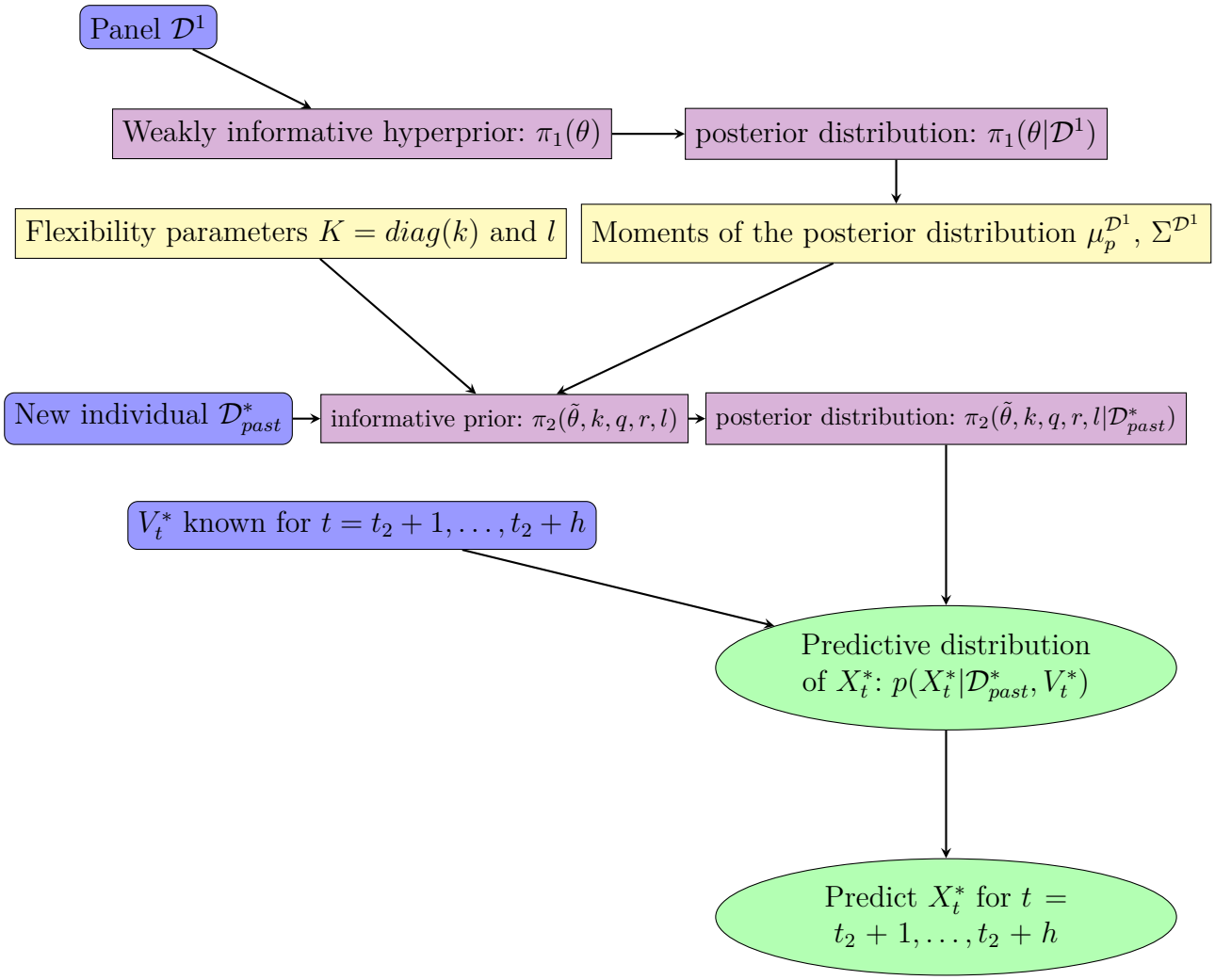


Figure 4.3 – Outline of the methodology of transfer learning for panel data. Information is extracted from estimation on the panel data to build informative prior for a new individual. The prediction of a new data point of the individual relies on the posterior distribution $\pi_2(\tilde{\theta}, k|\mathcal{D}^*_{past})$.

The complete methodology of transfer learning for panel data, based on a Bayesian approach, is summarized in Figure 4.3.

4.3 Application of the methodology to simulated data

We show an application to simulated datasets of the methodology presented in Section 4.2. The method is implemented using the `R` software(R Core Team (2019)) and the Stan software (Stan Development Team (2020), Stan Development Team (2018)).

Hereinafter, we adapt the examples developed in Launay et al. (2015) to the case of panel data analysis.

4.3.1 Polynomial regression

As in Launay et al. (2015), we consider a polynomial regression of order $p \geq 0$, which is adapted to the case of panel data:

$$y_t^{(i)} = \sum_{k=0}^{p} \beta_k^{(i)} (x_t^{(i)})^k + \epsilon_t^{(i)}, \tag{4.14}$$

where $\epsilon_t^{(i)} \sim \mathcal{N}(0, \sigma^{(i)2})$ are the individual errors of individual $i = 1, \ldots, n$ at time $t = 1, \ldots, T$. Here, $\sigma^{(i)} = \sigma = 1$. Each $x_t^{(i)}$ are simulated from the uniform distribution $\mathcal{U}[-1, 1]$.

We first simulate panel data according to the model in (4.14) to learn a model as described in Section 4.2. For this purpose, we denote $\beta^{(i)} = (\beta_0^{(i)}, \ldots, \beta_p^{(i)})$ the parameters of individual i and $\beta^h = (\beta_0^h, \cdots, \beta_p^h)$, the set of parameters common to all the individuals of the panel.

We take $p = 4$ the order of the polynomial and $\beta^h = (2, -1, 3, 1, 2)$. We consider the two following situations for the simulation of the panel.

— **First situation:** we simulate $y_t^{(i)}$ with $\beta^{(i)} = \beta^h$ for $i = 1, \ldots, n$. In this case, we simulate $n = 50$ series of length $T = 200$ according to the model in (4.14) and the parameters $\beta^{(i)}$. This means that the individual parameters $\beta^{(i)}$ are the same.

— **Second situation:** we simulate $y_t^{(i)}$ with $\beta^{(i)} = \beta^h + \eta^{(i)}$ for $i = 1, \ldots, n$, where $\eta^{(i)} \sim \mathcal{N}(0, \sigma')$. Here, a small noise is added to each parameter $\beta^{(i)}$.

For both situations, we learn a Bayesian hierarchical regression model on the simulated panels. We set the priors of each parameters as such:

$$\beta^{(i)}|\mu_{\beta^h}, \sigma_{\beta^h} \quad \sim \quad \mathcal{N}(\beta^h, \Sigma) \tag{4.15}$$

where $\Sigma = diag(\Sigma_0, \ldots, \Sigma_4)$ and

$$\begin{aligned} \beta_k^h &\sim \mathcal{N}(0, 5), \\ \Sigma_k &\sim \mathcal{HC}(0, 2.5), \text{for } k = 0, \ldots, 4. \end{aligned}$$

where $\mathcal{HC}(0, 2.5)$ is the Half-Cauchy distribution such that its density is as such:

$$f_y(0; 2.5) = \begin{cases} \frac{2}{2.5\pi} \frac{1}{1+\frac{y^2}{(2.5)^2}} & \text{if } y \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Once the model has been trained on the panel data for both situation, we extract the posterior mean $\mu_{\hat{\beta^h}}$ and posterior covariance matrix $\sigma_{\hat{\beta^h}}$. To evaluate the transfer strategy, we consider several scenarios of simulation of a new individual we denote y_t^*.

Considering the model in (4.14), we simulate an individual series of length $T = 10$ for each of the following scenarios:

- **Scenario n°1 - ideal case:** we simulate y_t^* with $\beta^* = \beta^h$ and $\sigma^* = 1$.
- **Scenario n°2 - some or all of the parameters vary and $\sigma^* = 1$:**
  (a) We simulate y_t^* with $\beta^* = \rho\beta^h$ and $\sigma^* = 1$, where $\rho = 0.5$.
  (b) We simulate y_t^* with $\beta^* = (\rho\beta_0^h, \rho\beta_1^h, \beta_2^h, \beta_3^h, \beta_4^h)$ and $\sigma^* = 1$, where $\rho = 0.5$.
- **Scenario n°3 - some or all of the parameters vary and $\sigma^* = 4$:**
  (a) We simulate y_t^* with $\beta^* = \beta^h$ and $\sigma^* = 4$.
  (b) We simulate y_t^* with $\beta^* = \rho\beta^h$ and $\sigma^* = 4$, where $\rho = 0.5$.
  (c) We simulate y_t^* with $\beta^* = (\rho\beta_0^h, \rho\beta_1^h, \beta_2^h, \beta_3^h, \beta_4^h)$ and $\sigma^* = 4$, where $\rho = 0.5$.

For all the scenarios described above we train on the shorter series the following models:

— **Model without transfer (MWT):** the Bayesian regression model with weakly informative priors on β^*. Specifically, we take $\beta_k^* \sim \mathcal{N}(0, 5)$, and $k = 0, \ldots, 4$.

— **Model with exact transfer (MTE):** the Bayesian regression model with informative priors where $\beta^* \sim \mathcal{N}(\mu_{\hat{\beta^h}}, \sigma_{\hat{\beta^h}})$.

— **Model with flexible transfer (MTF):** the Bayesian regression model with informative priors and flexibility parameters. We have $\beta^* \sim \mathcal{N}(K \cdot \mu_{\hat{\beta^h}}, l^{-1}\sigma_{\hat{\beta^h}})$, where $K = diag(k)$.

The informative prior is constructed following the Gaussian strategy developed in Launay et al. (2015). Specifically the hyperparameters k and l are taken as in Launay et al. (2015):

$$k|q, r \sim \mathcal{N}(q(1, \ldots, 1)', r^{-1}Id), \tag{4.16}$$

$$q \sim \mathcal{N}(1, \sigma_q), \tag{4.17}$$

$$r \sim \mathcal{G}(a_l, b_l), \tag{4.18}$$

$$l \sim \mathcal{G}(a_r, b_r), \tag{4.19}$$

where $a_l = b_l = 10^{-3}$ and $a_r = b_r = 10^{-6}$.

Each simulation scenario is repeated a 100 times and we evaluate the performances of the models using several metrics. Specifically, we look at the following information:

— the error $\mathcal{E}$ between the MAP estimates of the parameters denoted $\hat{\beta}^h$ and the real value of the parameters β^h defined by:

$$\mathcal{E}(\beta^h, \hat{\beta}^h) = \frac{|\beta^h - \hat{\beta}^h|}{|\beta^h|}. \tag{4.20}$$

— The posterior variance of $\pi(\beta^h|y_t^*)$.

— The Highest Posterior Density (HPD) regions of level $1 - \alpha$:

$$\mathcal{R}^{\alpha}_{HPD} = \{\lambda, \pi(\lambda|D) \geq h_\alpha\}, \tag{4.21}$$

for $h_\alpha > 0$ such that $\mathbb{P}(\lambda \in \mathcal{R}^{\alpha}_{HPD}) = 1 - \alpha$

— The coverage probability that the real value of the parameters β^h belong to the HPD regions over each repetition.

Scenario n°1: ideal case

We first focus on the case where the data y_t^*, on which the information is transferred, has the same parameters as the initial panel data. Recall that when simulating the panel

data we consider two situations : either the parameters $\beta^{(i)}$ are identical, or a small noise is added.

The boxplots of errors $\mathcal{E}(\beta^h, \hat{\beta}^h)$, for all the replications are displayed at the top of Figure 4.4 and Figure 4.5, for the first and second situation respectively. Both models with transfer (MTE and MTF) show lower error than the model without transfer (MTF) on the top left of Figure 4.4 and Figure 4.5. This shows the interest of this approach. There is clear gain to transferring information previously learned.

Furthermore, when looking at the posterior variance of each β^h (bottom of Figure 4.4 and Figure 4.5), we see that both MTE and MTF have lower posterior variance for each parameters over all the replications. For both situations, the posterior variance for the model with flexible parameters MTF is slightly more dispersed than the model with exact transfer MTE. Looking both at the errors and the posterior variance, we see that the two models with transfer outperform the model without transfer.

The boxplots of errors and variances obtained with MTE and MTF for the first situation (see Figure 4.4) are quite small when compared to the results obtained with the second situation, this is due to the fact that the posterior distribution transferred have very low variances. The first situation is not realistic in the sense that all there is no noise in the panel data, and this lack of noise is transferred to the model for the new individual. In this scenario, the individual considered has the same parameters as the panel. Thus the informative models, especially the model without flexible parameters, display very satisfying results (in terms of error). Hereinafter, we do not consider the first situation, and focus solely on the second situation, where the information transferred comes from the noisy data.

Figure 4.6 shows the length of HPD regions for each parameters (top) and the probability that over the 100 replications (bottom), the real value of the parameters are contained within the HPD regions. The model MWT shows lower coverage probability for all the parameters compared to the other two models as well as larger HPD regions. The informative approaches, in this ideal case, are both more efficient. The MTE approach differs only slightly from the MTF approach when looking at those indicators.

Scenario n°2: some or all of the parameters vary, $\sigma = 1$

In this scenario, two cases are considered:

(a) $\beta^* = 0.5\beta^h$ and $\sigma^* = 1$.

(b) $\beta^* = (0.5\beta_0^h, 0.5\beta_1^h, \beta_2^h, \beta_3^h, \beta_4^h)$ and $\sigma^* = 1$.

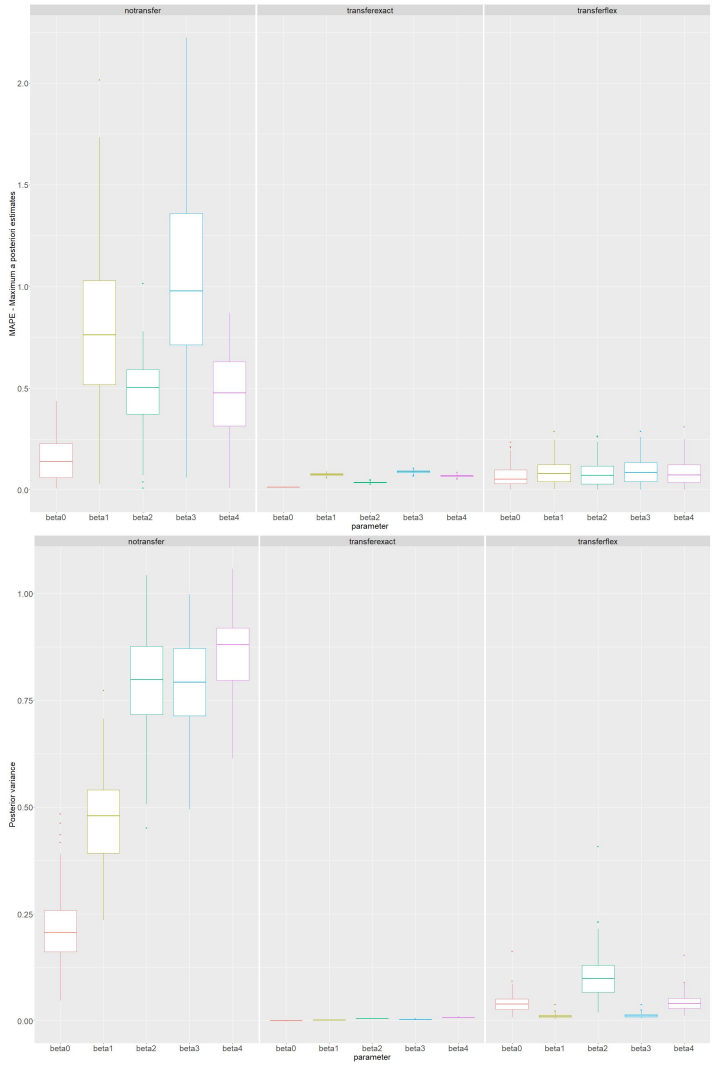


Figure 4.4 – **Polynomial regression - First scenario** - $\mathcal{E}(\beta^h, \hat{\beta}^h)$ is obtained for all the parameters [top]. The posterior variance boxplots for all the parameters [bottom]. The results are displayed for the three models MWT [left], MTE [center] and MTF [right], for the first situation: all the $\beta^{(i)}$ are identical.

For both cases, the errors are displayed in Figure 4.7. The boxplots of errors obtained with MWT are larger than thoses obtained with MTF for both cases. For case (a), where all the parameters differs, the exact transfer approach (MTE) performs badly, results are comparable with the weakly informative approach. This shows the interest in using similarity parameters, as the MTF approach outperforms the others. For case (b), only the first two parameters are different, this has an impact on the error results, where the first two boxplots for both informative approaches are larger than the three others. Both

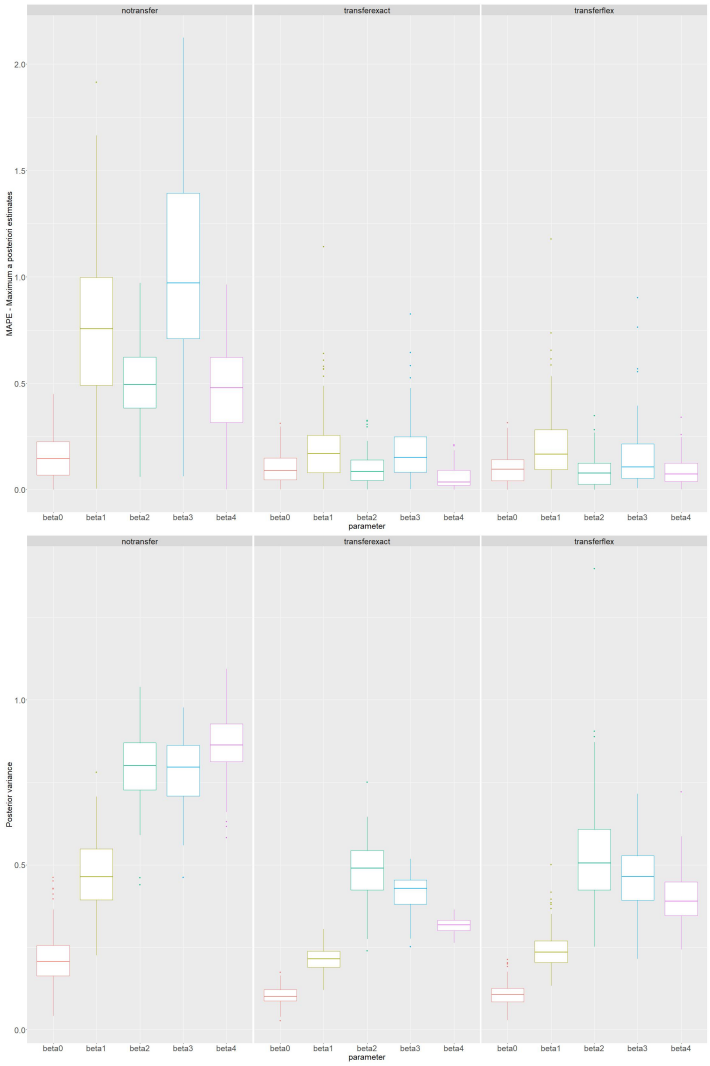


Figure 4.5 – **Polynomial regression - First scenario** - $\mathcal{E}(\beta^h, \hat{\beta}^h)$ is obtained for all the parameters [top]. The posterior variance boxplots for all the parameters [bottom]. The results are displayed for the three models MWT [left], MTE [center] and MTF [right], for the second situation: an individual noise is added to the $\beta^{(i)}$.

MTE and MTF outperform MWT in this case.

The posterior variance are shown at the bottom of Figure 4.7. For case (a), MTF has lower errors, with lower variance, making it the best model out of the three, when looking at both indicators. For case (b), the posterior variance obtained with MWT is much larger than both informative approaches.

In Figure 4.8, we show the length of HPD regions and the coverage probability for both cases. For case (a), the length of HPD regions from MTE and MTF is smaller

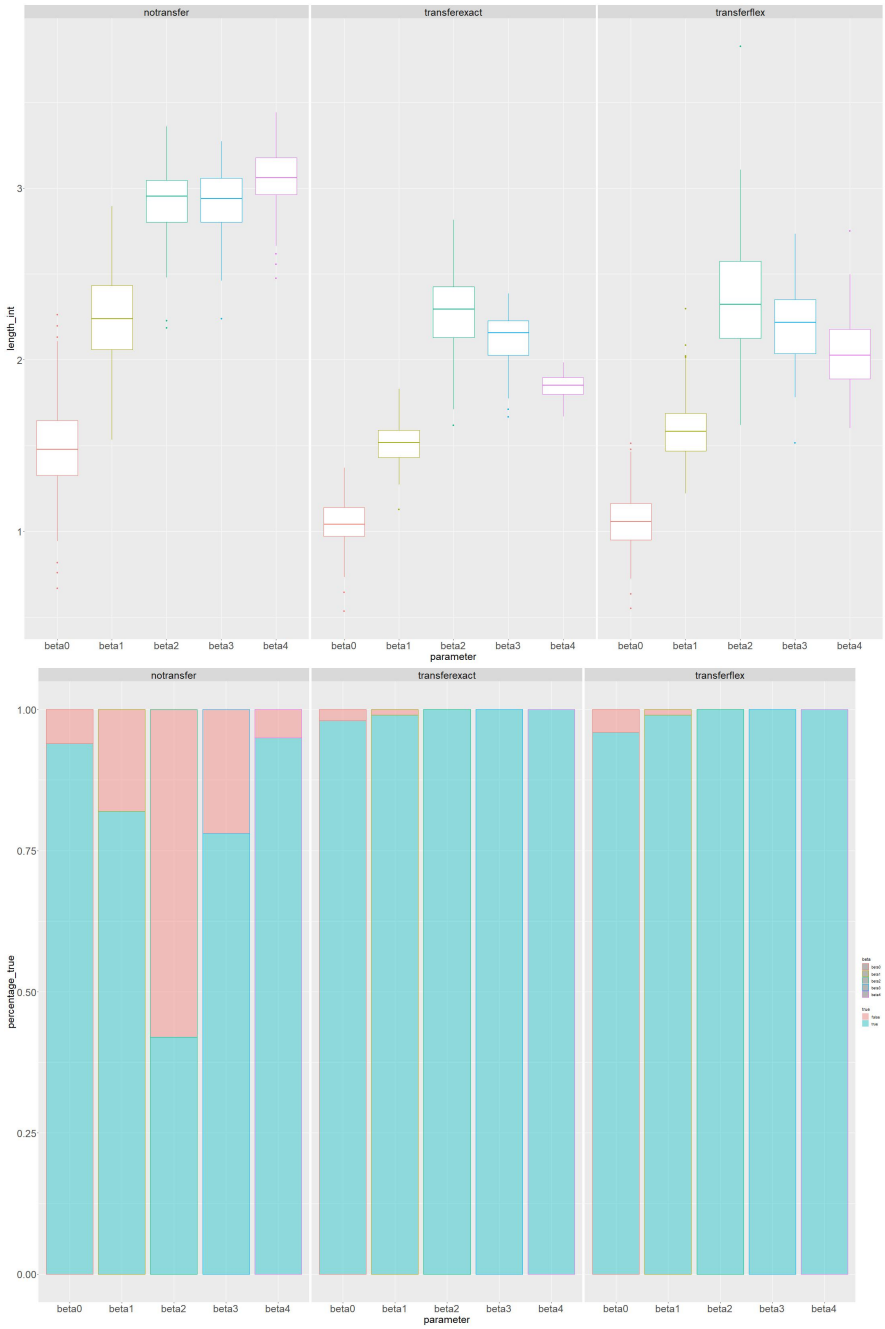
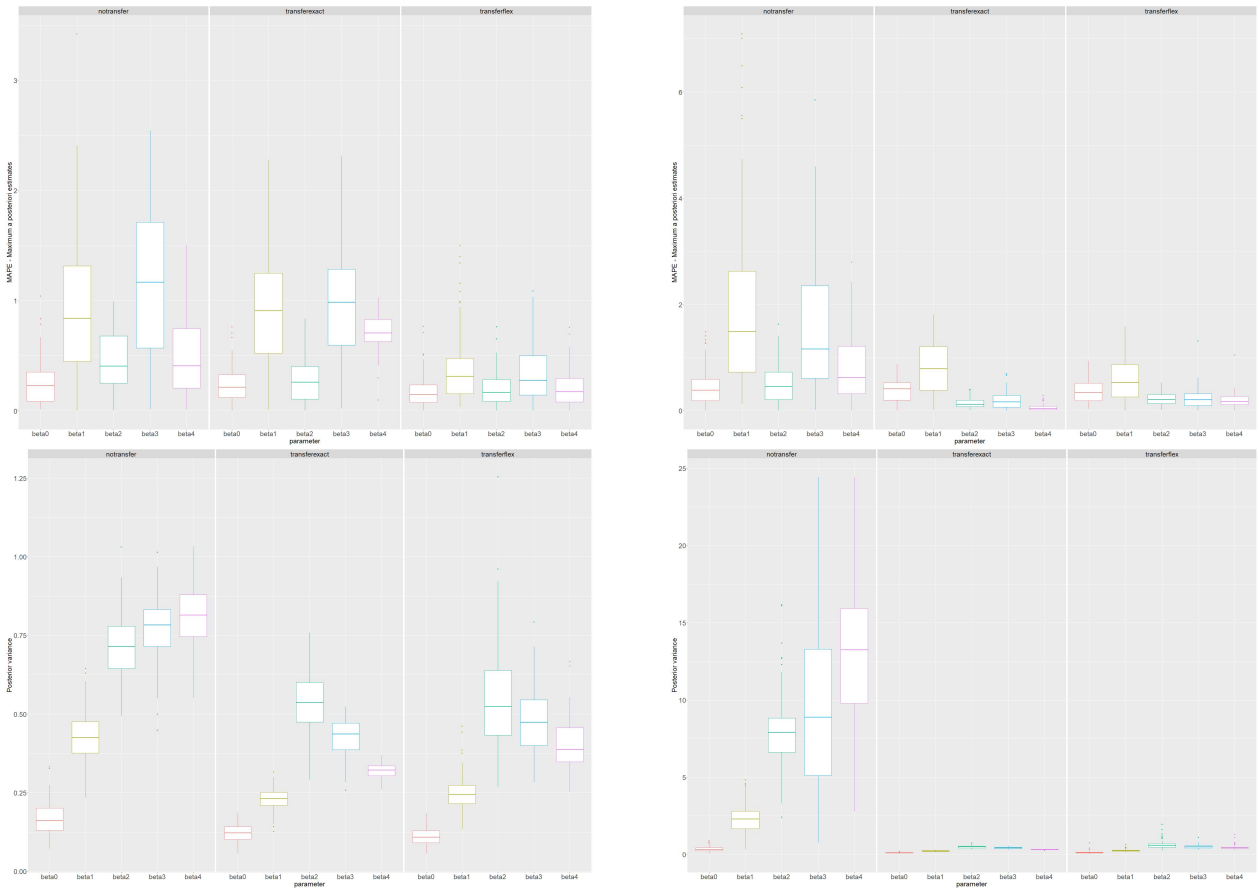


Figure 4.6 – **Polynomial regression - First scenario** - [top] Length of HPD regions for each parameters. [bottom] Coverage probability over the 100 replications is shown in blue. The results are displayed for the three models MWT [left], MTE [center] and MTF [right]. The second situation is considered here.

than with MWT. However, the coverage probability obtained with MTF in this case, is higher than the other approaches. For case (b), the coverage probabilities for all the



(a) (b)

Figure 4.7 – **Polynomial regression - Second scenario** - $\mathcal{E}(\beta^h, \hat{\beta}^h)$ is obtained for all the parameters [top]. The posterior variance boxplots for all the parameters [bottom]. For each panel, the results are displayed for the three models MWT [left], MTE [center] and MTF [right], for the second situation: an individual noise is added to the $\beta^{(i)}$.

parameters obtained with all MWT is higher than the informative approaches, but the length of HPD regions is also much higher. Thus the regions are more imprecise. For the informative approaches, the coverage probabilities are globally high for all parameters except the first, this is consistent with the error results seen in Figure 4.7. For the first two parameters, MTF has slighlty higher coverage probability with similar length of HPD regions. Overall, the MTF approach seems to be more efficient than the MTE approach, when the parameters differ.

(a) (b)

Figure 4.8 – **Polynomial regression - Second scenario** - [top] Length of HPD regions for each parameters. [bottom] Coverage probability over the 100 replications is shown in blue. The results are displayed for the three models MWT [left], MTE [center] and MTF [right], for each panel. The second situation is considered here.

Scenario n°3: some or all of the parameters vary, $\sigma = 4$

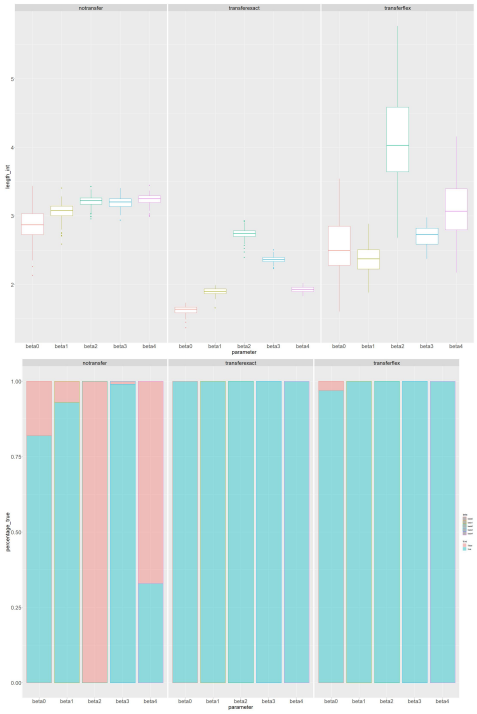
In this scenario, the three following cases are considered:

(a) $\beta^* = \beta^h$ and $\sigma^* = 4$.

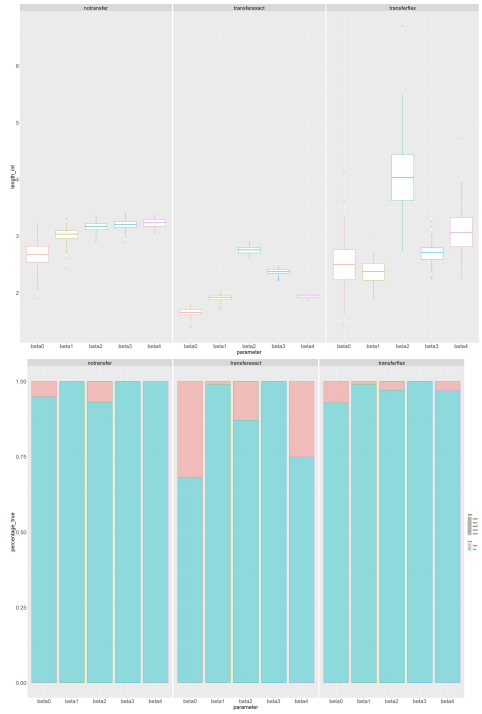
(b) $\beta^* = 0.5\beta^h$ and $\sigma^* = 4$.

(c) $\beta^* = (0.5\beta_0^h, 0.5\beta_1^h, \beta_2^h, \beta_3^h, \beta_4^h)$ and $\sigma^* = 4$.

We display the length of HPD regions and coverage probability in Figure 4.9 for the three cases. For case (a), the coverage probabilities obtained with MWT are much lower than MTE and MTF. The length of HPD regions is also quite high, though not as high as with MTF. Here, the best model is MTE. For case (b), MTE performs badly compared



(a)



(b)

(c)

Figure 4.9 – **Polynomial regression - Third scenario** - For each panel, [top] length of HPD regions for each parameters, [bottom] coverage probability over the 100 replications is shown in blue. The results are displayed for the three models MWT [left], MTE [center] and MTF [right], for each panel.

to the other approaches. Here, the weakly informative approach performs the best, when looking at both the coverage probabilities and the length of HPD regions. For case (c), looking only at coverage probabilities, MWT seems to perform better than the informative approaches. However, the length of HPD regions is much higher than with the informative approaches. MTE performs baldly for the first parameter in terms of coverage probability. Overall, the best results are obtained with MTF.

MTF seems to outperform MWT most of the times, and is robust to noisy data and change in parameters. This shows the interest of Bayesian transfer learning, in the context of short historical data. Even when the shorter series is not too similar to the initial panel data, a gain in observed, with the addition of flexibility parameters.

MTE, the model with exact transfer, performs well when the parameters of the new individual are very close to the parameters of the panel data (see Figure 4.5 and Figure 4.15a). When the some or all of the parameters are different, the results obtained with MTE are deteriorated (see Figure 4.7a and Figure 4.15b). Hereinafter, we do not consider this model.

The error between the Maximum a posteriori estimates and the true value of the parameters obtained for the third scenario can be found in Appendix 4.A.

4.3.2 Auto-regressive model

We consider the following $AR(p)$ model for panel data:

$$y_t^{(i)} = \sum_{k=1}^{p} \beta_k^{(i)} y_{t-k}^{(i)} + \epsilon_t^{(i)}, \tag{4.22}$$

where $\epsilon_t^{(i)} \sim \mathcal{N}(0, \sigma)$ is the individual noise of each individual series $i = 1, \ldots, n$ of length $t = 1, \ldots, T$ and $\sigma = 1$.

As in Section 4.3.1, we learn the $AR(p)$ model on the simulated panel data, with a weakly informative prior on the hyperparameters as described in Section 4.2.

For the simulation of the panel, we denote $\beta^{(i)} = (\beta_1^{(i)}, \ldots, \beta_p^{(i)})$ the parameters of individual i and $\beta^h = (\beta_1^h, \cdots, \beta_p^h)$, the set of parameters common to all the individuals of the panel.

We simulate the panel data according to the model in (4.22) taking the same set of parameters as in Launay et al. (2015). Specifically, each individual series is simulated with $\beta^{(i)} = \beta^h + \eta^{(i)}$ for $i = 1, \ldots, n$, where $\eta^{(i)} \sim \mathcal{N}(0, \sigma')$ and $\beta^h = (1.7, -0.23, -0.833, 0.3528)$,

$i = 1, \ldots, n$ and $p = 4$. The length of each individual series of the panel is $T = 400$. Similarly to the Bayesian hierarchical regression in Section 4.3.1, we are interested in the posterior estimates of the upper levels of the hierarchical model denoted $\hat{\mu_{\beta^h}}$ and $\hat{\sigma_{\beta^h}}$.

For the transfer strategy, we simulate an individual series denoted y_t^* according to the model in (4.22). For the choice of parameters, we consider the following scenarios:

1. we simulate $y_t^* =: y_t^{*1}$ with $\beta^* = \beta^h$ and $\sigma^* = 1$.
2. We simulate $y_t^* = y_t^{*1} + \eta_1$ with $\eta_1 \sim \mathcal{N}(0, \sigma^*)$ and $\sigma^* = 4$.
3. We simulate $y_t^* =: y_t^{*2}$ with $\beta^* = \rho\beta^h$ and $\sigma^* = 1$, where $\rho = 0.5$.
4. We simulate $y_t^* = y_t^{*2} + \eta_2$with $\eta_2 \sim \mathcal{N}(0, \sigma^*)$ and $\sigma^* = 4$.

We simulate series of length $T = 10, 30, 100$ for each of the scenarios described. Each scenario of simulation is replicated a 100 times and results are aggregated over all the replications. The results presented in this section focus on forecasting, when length of historical data available is $T = 10$ and the forecasting horizon is $h_f = 10$ points.

The modelling strategy of transfer on y_t^* is the same as in Section 4.3.1. To evaluate the forecasting, we compute the Mean Absolute Percentage Error (MAPE) defined by:

$$MAPE(y_t^*, \hat{y_t}^*) = \frac{1}{h_f} \sum_{t=1}^{h_f} |\frac{y_t^* - \hat{y_t}^*}{y_t^*}|.$$

The MAPE of the forecast for the two first scenarios are displayed in Figure 4.10. We see that, when the noise is small, there is a clear benefit to applying transfer whereas performances are degraded when the data is noisy.

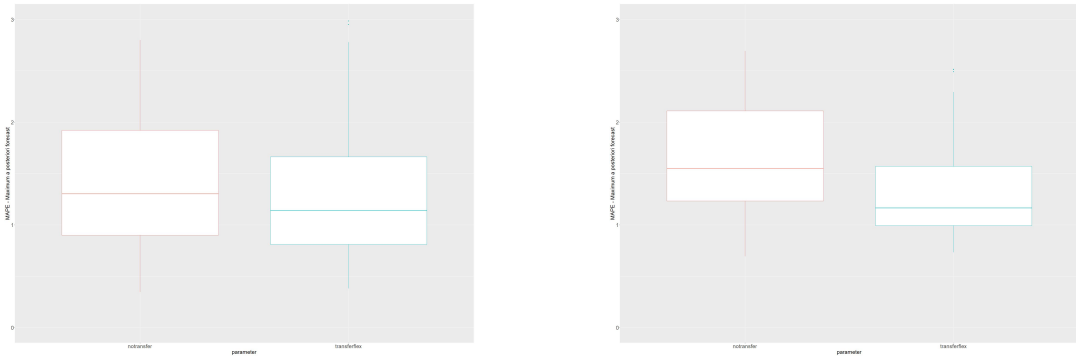
The MAPE of the forecast for the two last scenarios are displayed in Figure 4.11. In this case, the set of parameters differs a lot from the initial parameters of the panel data. THe series simulated are quite different due to that. We see that, when the noise is small, applying transfer to the autoregressive model displays lower errors whereas performances are degraded when the data is noisy ($\sigma^* = 4$).

Now looking at the length of prediction intervals and coverage probabilities for the first two scenarios in Figure 4.12, we see that, the coverage probabilities is lower for the transfer case. The prediction intervals are also quite smaller.

Finally, in Figure 4.13, we show the length of prediction intervals and related coverage probabilities for the last two scenarios. Again, the coverage probabilities are lower with the transfer case, as well as the length of prediction intervals. For the third scenario, when the noise is small, we see that though coverage probability is slightly small in the transfer

(a) $\beta^* = \beta^h$ and $\sigma^* = 1$ (b) $\beta^* = \beta^h$ and $\sigma^* = 4$

Figure 4.10 – **Autoregressive model** - For both panels, Average MAPE between the Maximum a posteriori and real value of the forecast for $T = 10$ points of historical data, The results are displayed for the two models MWT [left] and MTF [right], for each panel.

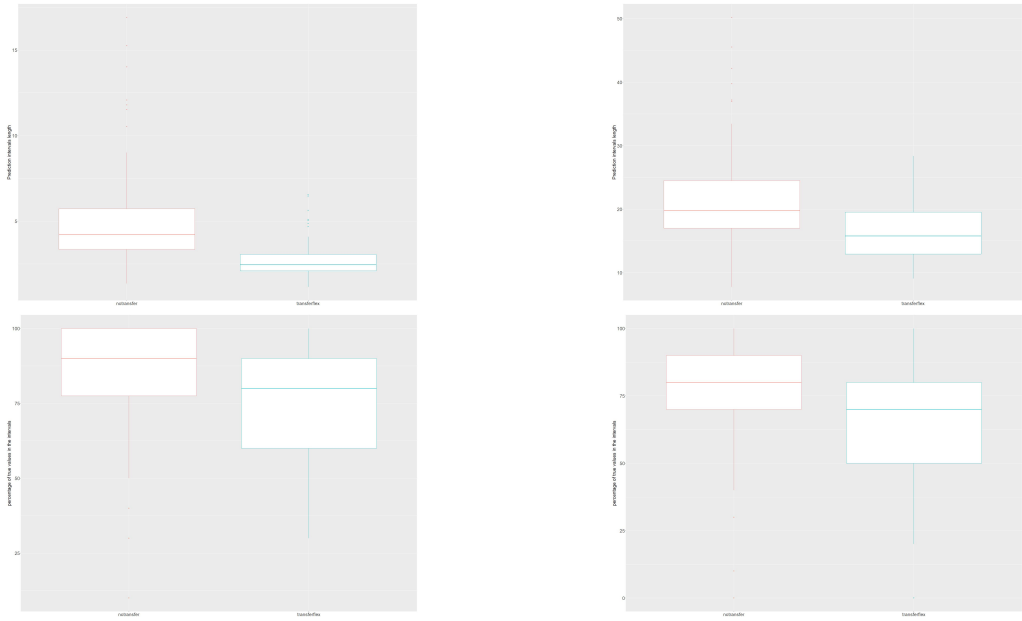


(a) $\beta^* = \rho\beta^h$ and $\sigma^* = 1$ (b) $\beta^* = \rho\beta^h$ and $\sigma^* = 4$

Figure 4.11 – **Autoregressive model** - For both panels, Average MAPE between the Maximum a posteriori and real value of the forecast for $T = 10$ points of historical data, The results are displayed for the two models MWT [left] and MTF [right], for each panel.

case, the difference between the two models is small, in terms of coverage probability. However, the length of prediction intervals is much lower in the transfer situation. The transferring strategy seems to provide interesting results, even when the new individual is not as close as the initial panel. However, when the data is too noisy, the benefit of transfer is a little mitigated.

Additional results obtained for varying length of historical data $T = 30$ and $T = 100$ are available in Appendix 4.B.



(a) $\beta^* = \beta^h$ and $\sigma^* = 1$ (b) $\beta^* = \beta^h$ and $\sigma^* = 4$

Figure 4.12 – **Autoregressive model** - For both panels, [top] Average length of prediction intervals for $T = 10$ points of historical data, [bottom] Percentage of values contained within the prediction intervals over the 100 replications (in blue). The results are displayed for the two models MWT [left] and MTF [right], for each panel.

4.3.3 Hierarchical Poisson model

We simulate n individuals series length T for the panel data according to the following hierarchical Poisson model:

$$y_t^{(i)} \sim \mathcal{P}(\lambda_i), \tag{4.23}$$

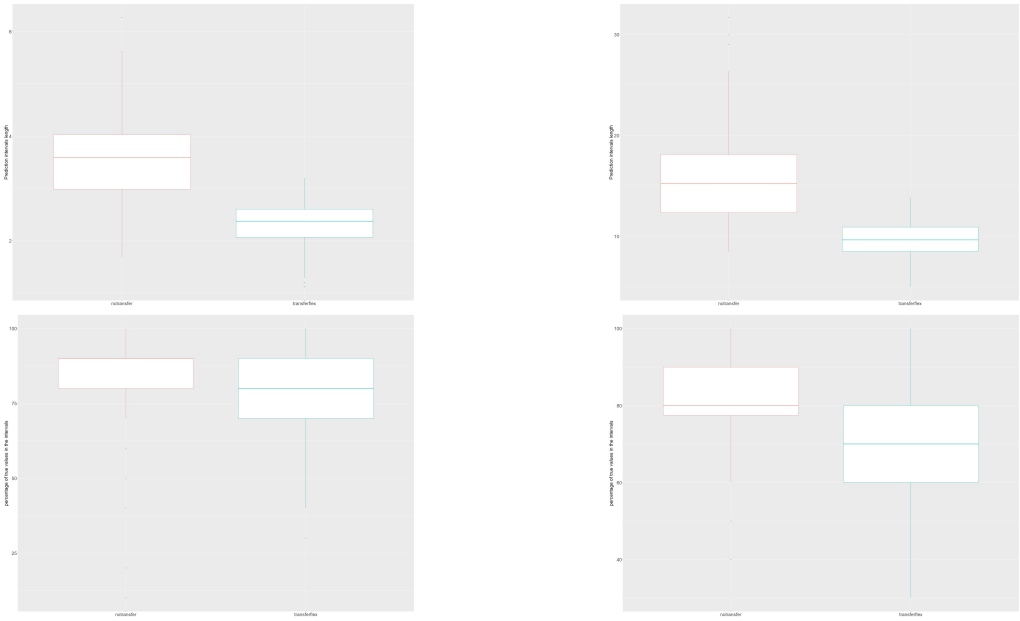
$$\lambda_i \sim \Gamma(a_i, b_i), \tag{4.24}$$

$$a_i \sim \mathcal{E}(1/a), \tag{4.25}$$

$$b_i \sim \mathcal{E}(1/b), \tag{4.26}$$

$$a \sim \Gamma(1, 1), \tag{4.27}$$

$$b \sim \Gamma(1, 1), \tag{4.28}$$



(a) $\beta^* = \rho\beta^h$ and $\sigma^* = 1$ (b) $\beta^* = \rho\beta^h$ and $\sigma^* = 4$

Figure 4.13 – **Autoregressive model** - For both panels, [top] Average length of prediction intervals for $T = 10$ points of historical data, [bottom] Percentage of values contained within the prediction intervals over the 100 replications (in blue). The results are displayed for the two models MWT [left] and MTF [right], for each panel.

where $i = 1, \ldots, n$ and $t = 1, \ldots, T$. Each individual series i has a set of individual parameters a_i and b_i. The parameters common to all the individuals are a and b and we wish to transfer the information of the posterior distribution of those parameters to new individuals with shorter historical data. The length of each individual series in the panel is $T = 5000$ and the number of individuals in the panel is $n = 50$.

We are in a univariate setting here and the information transferred is the posterior mean and posterior variance of parameters a and b. We denote $\hat{\mu}_a$ and $\hat{\mu}_b$ the posterior means of hyperparameters a and b respectively. The posterior variances of a and b are $\hat{\sigma}_a$ and $\hat{\sigma}_b$ respectively.

For the simulation of the new individual with short historical data $\mathcal{D}^*$, we consider the following cases:

(a) **ideal case:** $a = b = 1$,

(b) $a = \frac{9}{4}$ and $b = \frac{3}{2}$,

(c) $a = 4$ and $b = 2$.

The length of the simulation for the short data is $T = 25$ points. Each scenario of simu-

lation is replicated a 100 times.

For the informative approach, we choose the following priors on a and b:

$$\begin{aligned} a &\sim \mathcal{G}(\alpha_a, \beta_a), \\ b &\sim \mathcal{G}(\alpha_b, \beta_b), \end{aligned}$$

where

$$\begin{aligned} \frac{\alpha_a}{\beta_a} &= K \cdot \hat{\mu}_a, \\ \frac{\alpha_b}{\beta_b} &= K \cdot \hat{\mu}_b, \\ \frac{\alpha_a}{\beta_a^2} &= l^{-1} \cdot \hat{\sigma}_a, \\ \frac{\alpha_b}{\beta_b^2} &= l^{-1} \cdot \hat{\sigma}_b, \end{aligned}$$

Remark 19 *We set gamma priors on the parameters here because we have positive constraints on $a > 0$ and $b > 0$. Using truncated normal distributions here is not recommended. With the Gaussian strategy of Launay et al. (2015), transfer of information (mean and covariance) is done on a complete distribution, and truncation has an effect on the moments of this new distribution.*

We display the length of HPD regions and coverage probabilities obtained with the three scenarios of simulation for the shorter dataset in Figure 4.14. The ideal case (a) shows that MTF performs well compared to MWT (see Figure 4.14a). Coverage probabilities are very high with and without transfer, but the length of HPD regions with transfer is much lower. When the new individual is very similar to th original panel, the transferring srategy is quite efficient.

For scenario (b), we see in Figure 4.14b, that coverage probabilities are high without transfer, but the length of HPD regions are very large. Transfer degrades the coverage probability for parameter a, but improves is for parameter b, while the length of the HPD region is much smaller.

For scenario (c), we see similar results in Figure 4.14c. The coverage probability with transfer for parameter (b) is very high and very low for parameter (a), while the length of HPD regions are quite small.

Overall, in this univariate situation, when the new individual differs from the original panel, the transferring strategy's performances are mitigated.

The MAPE between the Maximum a posteriori estimates and the true value of the parameters obtained for all the scenarios can be found in Appendix 4.C.

4.4 Conclusion and discussion

We have proposed a methodology for transfer learning on panel data using a Bayesian approach. The methodology is adapted from the proposed methodology of Bayesian transfer Launay et al. (2015) where the longer historical data consists in only one time series.

We extend the methodology to the case of panel data. The first part of the method consists in training a weakly informative Bayesian hierarchical model on panel data with longer historical data. Posterior mean and covariances of the hyperparameters (the global behavior of the panel) are used in the second part of the method to transfer information to a new individual with shorter historical data.

We show applications of the methodology to three simulation situations. For polynomial regression, we see that there is a clear gain in using the transfer learning strategy. For the autoregressive model, we see that when the new individual is not too noisy, the transfer learning strategy remains relevant. For the hierarchical Poisson model, the performances of transfer are mitigated when the new individual differs too much from the panel. As for this last situation, we transfer the variance and not the correlation between the parameters, this may explain why the results obtained are less satisfying.

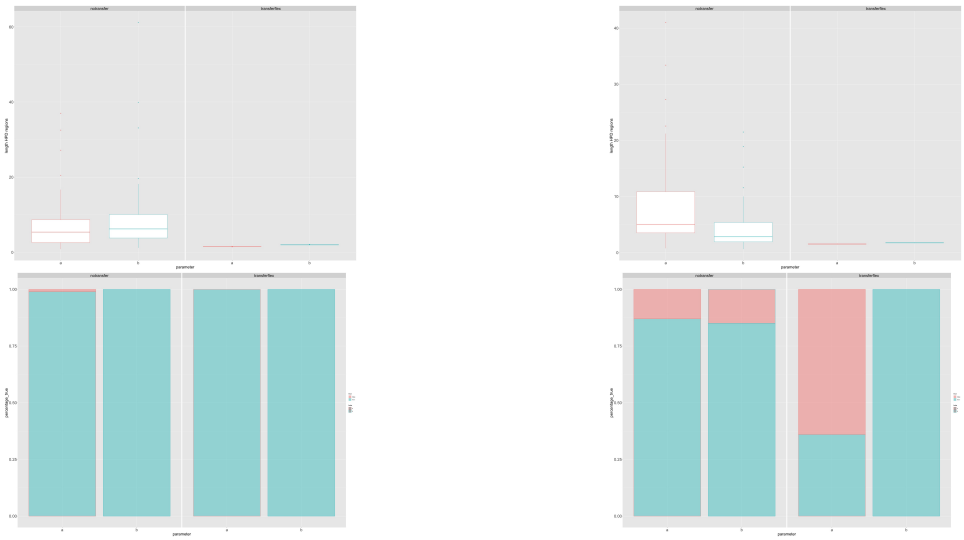
In Chapter 5, we apply this methodology to real world data and the case of forecasting the end-of-month consumption of a new customer, with short historical, using the information extracted from a panel of customers with longer historical data.

4.A Polynomial regression

4.A.1 Scenario n°3: some or all of the parameters vary, $\sigma = 4$

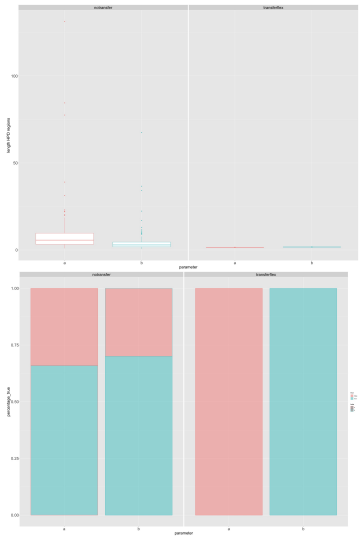
4.B Auto-regressive model

4.C Hierarchical Poisson model



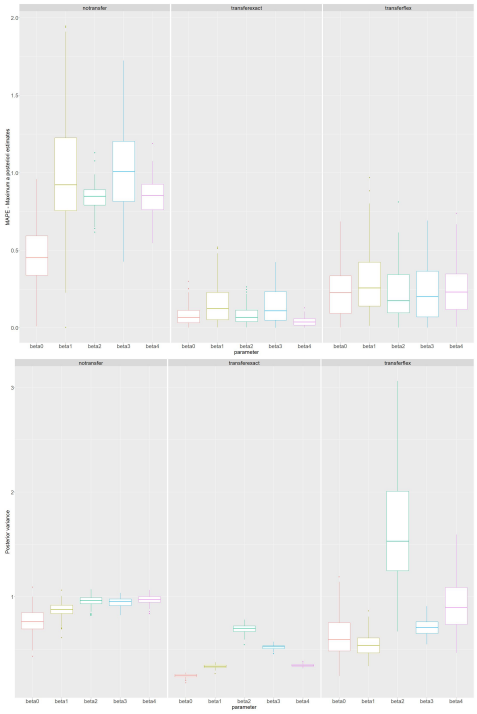
(a)

(b)

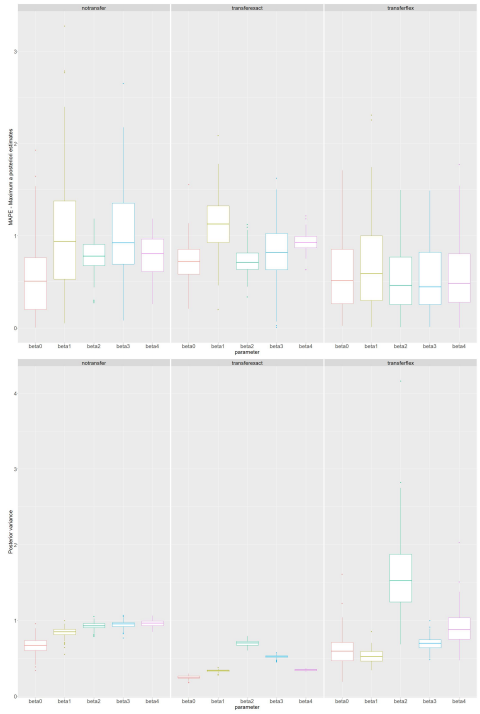


(c)

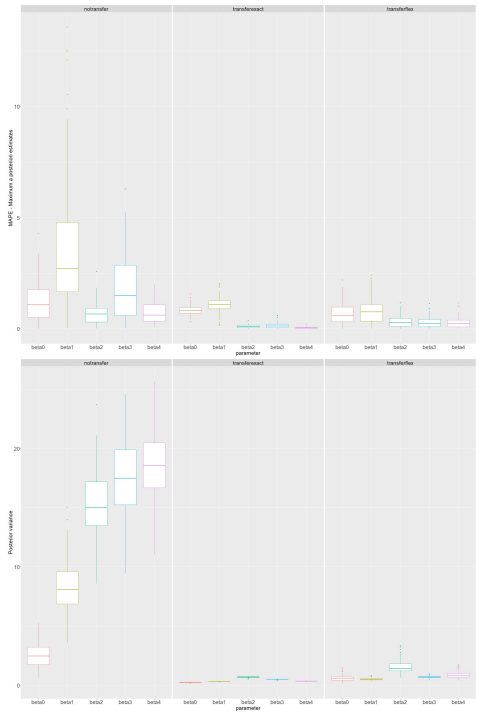
Figure 4.14 – **Hierarchical Poisson model** - For each panel, [top] length of HPD regions for each parameters, [bottom] coverage probability (in blue) over the 100 replications. The results are displayed for the three models MWT [left] and MTF [right], for each panel. For each model, the results are shown for parameters a and b side by side.



(a)

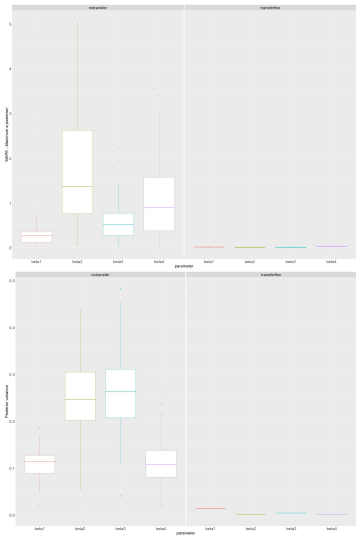


(b)

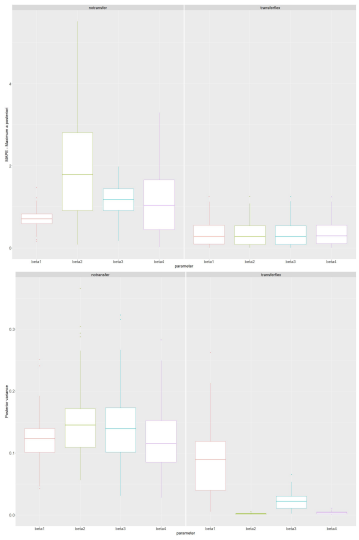


(c)

Figure 4.15 – **Polynomial regression - Third scenario** - $\mathcal{E}(\beta^h, \hat{\beta}^h)$ is obtained for all the parameters [top]. The posterior variance boxplots for all the parameters [bottom]. The results are displayed for the three models MWT [left], MTE [center] and MTF [right], for all three panels.

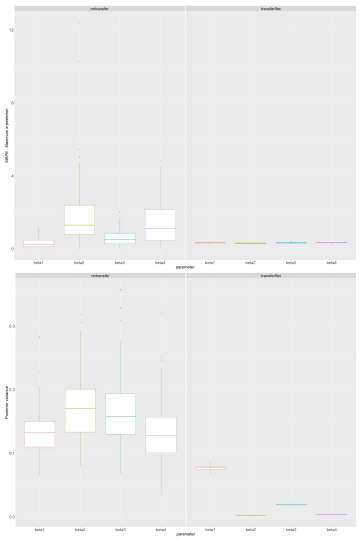


(a) $\beta^* = \beta^h$ and $\sigma^* = 1$

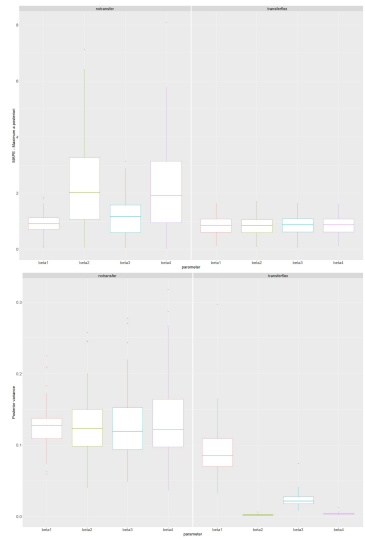


(b) $\beta^* = \beta^h$ and $\sigma^* = 4$

Figure 4.16 – **Autoregressive model** - Length of historical: $T = 10$. [top] $\mathcal{E}(\beta^h, \hat{\beta}^h)$ is obtained for all the parameters. [bottom] The posterior variance boxplots for all the parameters. The results are displayed for the two models MWT [left] and MTF [right], for all panels. For all models, the results are shown for all the parameters.

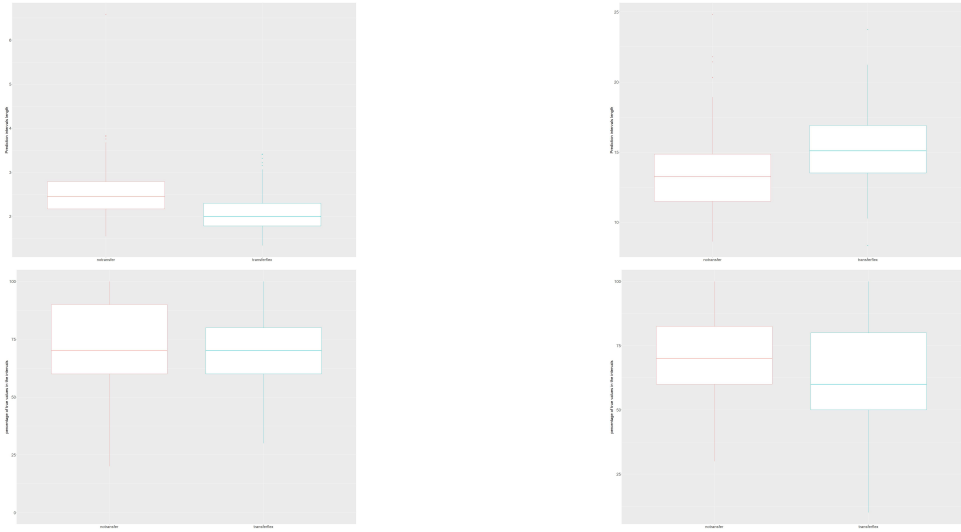


(a) $\beta^* = \rho\beta^h$ and $\sigma^* = 1$



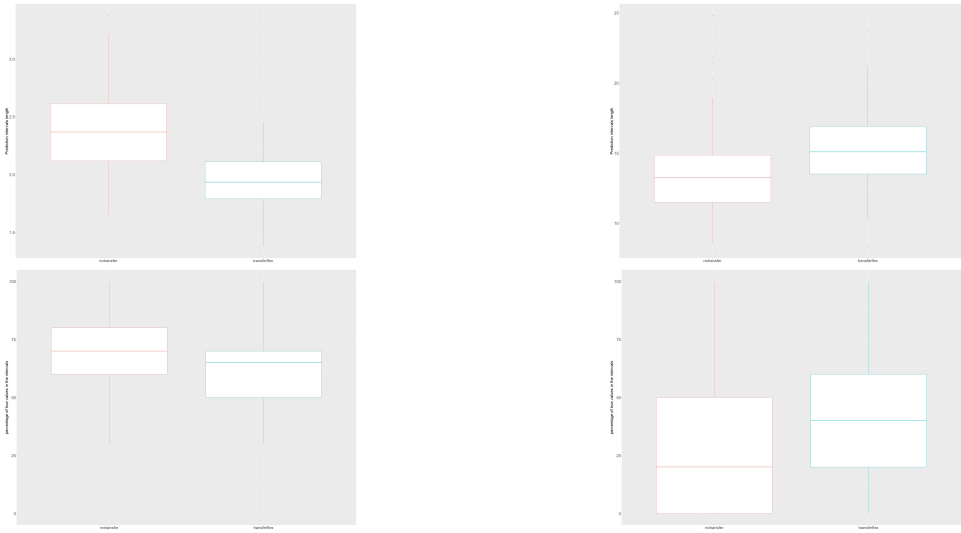
(b) $\beta^* = \rho\beta^h$ and $\sigma^* = 4$

Figure 4.17 – **Autoregressive model** - Length of historical: $T = 10$. [top] $\mathcal{E}(\beta^h, \hat{\beta}^h)$ is obtained for all the parameters. [bottom] The posterior variance boxplots for all the parameters. The results are displayed for the two models MWT [left] and MTF [right], for all panels. For all models, the results are shown for all the parameters.



(a) $\beta^* = \beta^h$ and $\sigma^* = 1$ (b) $\beta^* = \beta^h$ and $\sigma^* = 4$

Figure 4.18 – **Autoregressive model** - For both panels, [top] Average length of prediction intervals for $T = 30$ points of historical data, [bottom] Percentage of values contained within the prediction intervals over the 100 replications. The results are displayed for the two models MWT [left] and MTF [right], for each panel.

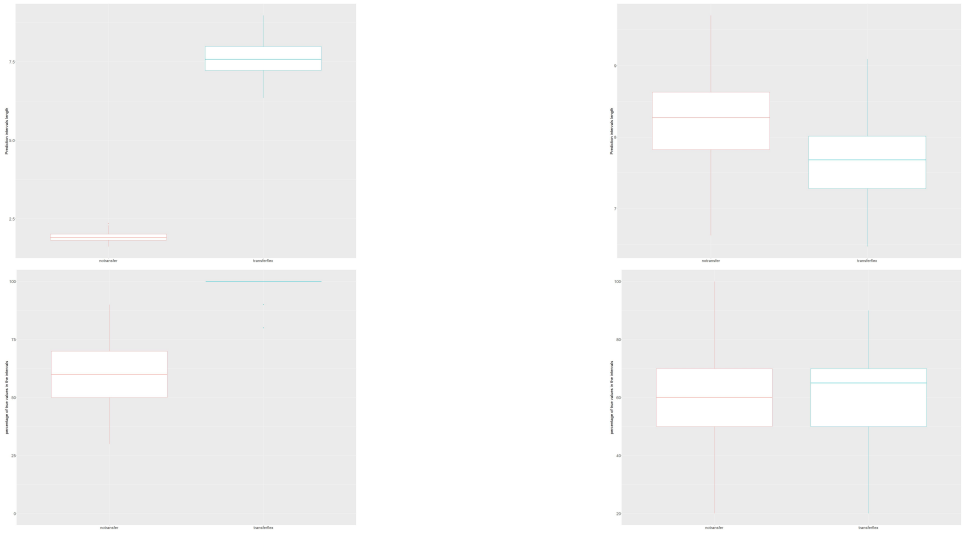


(a) $\beta^* = \rho\beta^h$ and $\sigma^* = 1$ (b) $\beta^* = \rho\beta^h$ and $\sigma^* = 4$

Figure 4.19 – **Autoregressive model** - For both panels, [top] Average length of prediction intervals for $T = 30$ points of historical data, [bottom] Percentage of values contained within the prediction intervals over the 100 replications. The results are displayed for the two models MWT [left] and MTF [right], for each panel.

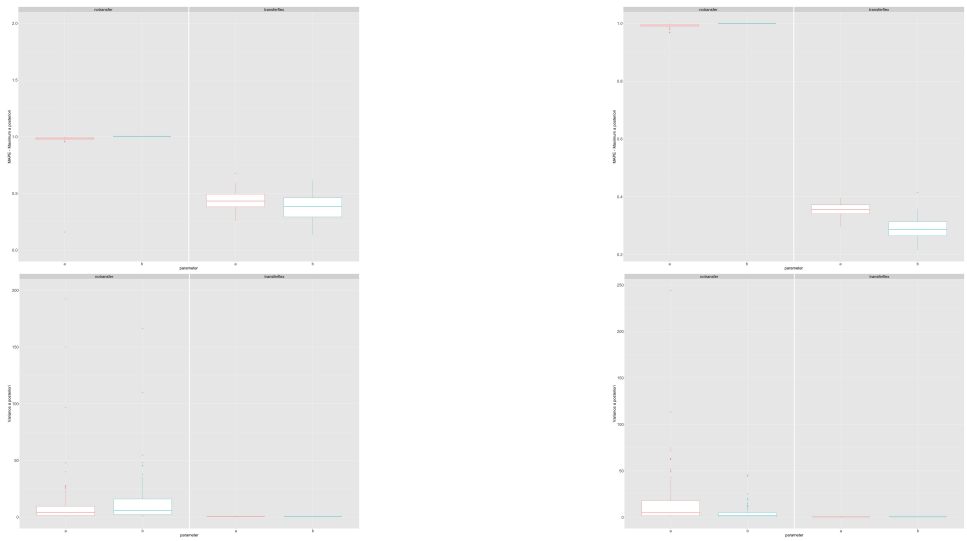
(a) $\beta^* = \beta^h$ and $\sigma^* = 1$ (b) $\beta^* = \beta^h$ and $\sigma^* = 4$

Figure 4.20 – **Autoregressive model** - For both panels, [top] Average length of prediction intervals for $T = 100$ points of historical data, [bottom] Percentage of values contained within the prediction intervals over the 100 replications. The results are displayed for the two models MWT [left] and MTF [right], for each panel.

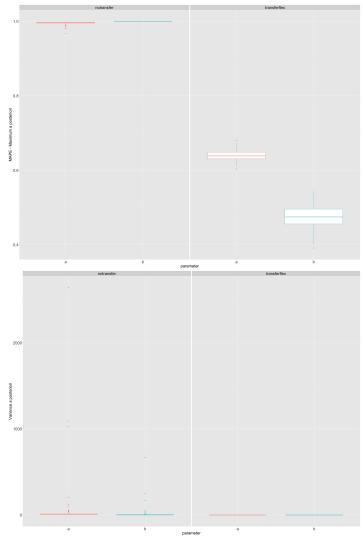


(a) $\beta^* = \rho\beta^h$ and $\sigma^* = 1$ (b) $\beta^* = \rho\beta^h$ and $\sigma^* = 4$

Figure 4.21 – **Autoregressive model** - For both panels, [top] Average length of prediction intervals for $T = 100$ points of historical data, [bottom] Percentage of values contained within the prediction intervals over the 100 replications. The results are displayed for the two models MWT [left] and MTF [right], for each panel.



(a) (b)



(c)

Figure 4.22 – **Hierarchical Poisson model** - [top] $\mathcal{E}(\beta^h, \hat{\beta}^h)$ is obtained for all the parameters. [bottom] The posterior variance boxplots for all the parameters. The results are displayed for the two models MWT [left] and MTF [right], for all three panels. For each model, the results are shown for parameters a and b side by side.

Chapter 5

Application of Bayesian transfer for panel data: end of month electrical consumption forecasting for residential customers

5.1 Introduction

For companies such as Électricité de France (EDF), panel data modeling and its application are a topic of interest, closely related to consumer behavior and consumption forecasting. This is the case, for instance, for forecasting of individual residential consumption, based on household features and consumers' behavior.

Recent approaches have been developed for the issue of residential load forecasting.

Iyengar et al. (2018) propose a Bayesian-based approach to model and detect energy inefficient residential buildings. In Sun et al. (2020), a Bayesian deep learning framework is developed for forecasting of the aggregated residential load. Brusaferri et al. (2019) also use a Bayesian deep learning approach for short-term residential load forecasting. Bessani et al. (2020) apply a Bayesian network for short-term residential load forecasting at half hourly period.

A common issue, arising in industrial applications such as forecasting of individual residential consumption, is the lack of historical data to forecast the behavior of new customers. This lack of historical data prevents us from using deep learning-based approaches, as they rely on large volumes of data to be trained properly.

One way to overcome this issue is to use information and knowledge on existing cus-

tomers and apply it to new customers. This can be seen as a transfer learning situation where the existing customers constitute the source domain and the new customers of interest the target domain. Transfer learning can be an efficient way to predict behavior of new customers. It has the advantage of reducing training time of models for new observations, and requires a smaller amount of training data when transferring on the source domain to learn the source task.

In this Chapter, we propose an application of the general methodology presented in Chapter 4 to the forecasting of individual residential customers. Specifically, we focus on end-of month forecasting.

In Section 5.2, we present the dataset we use for this industrial use-case. In Section 5.3 we introduce the notations used throughout the Chapter. Section 5.4 is dedicated to the presentation of the models as well as the results obtained on real world data. A discussion and conclusion of this chapter is provided in 5.5.

5.2 Description of the dataset

The dataset considered is the dataset from the CER Smart Metering Project (see Commission for Energy Regulation (2012)). We focus on a subgroup of the dataset. The construction of this subgroup, by means of clustering methods and deep learning, is presented in Chapter 6.6.

For each individual household in the dataset, we have load curves, initially at half-hourly period, that are aggregated to a daily period. Similarly, the temperature curves are available and aggregated at a daily period. The dataset provides questionnaires indicating various information relating to each of the households. For the purpose of this study we consider only a few of those information. The details on the selected variables are provided in Section 5.3.

The models are trained on a training subset of 725 individual households. For our experiments, we evaluate the transfer learning method on 93 individuals from the same subgroup as the training subset.

5.3 Notations

Recalling the notations introduced in Chapter 4, we denote:

— $\mathcal{D}^1 = (X_t^{(i)}, V_t^{(i)})_{\substack{i=1,\dots,n \\ t=1,\dots,T}}$, the panel of residential customers with long historical data.

- $X_t^{(i)}$ represents the consumption of the household i at time t
- The number of individuals in the panel is $n = 725$, and the length of each time series is $T = 365$.
- the variables $V_t^{(i)}$ of a given customer i are:
  - the outside temperature $T_t^{(i)}$ recorded for the day t, $t = 1, \ldots, T$.
  - $\mathbb{1}_h^{(i)}$ the binary variable indicating the presence of electric heating in the household i.
  - $\mathbb{1}_{wh}^{(i)}$ the binary variable indicating the presence of an electric sanitary hot water system in the household i.
- $\mathcal{D}^* = (X_t^*, V_t^*)$, the data of a household with shorter historical data.
- where $t \in \{t_1, \ldots, t_2\}$, and $t_2 - t_1 < T$.
- the variables V_t^* of the customer i are:
  - the outside temperature T_t^* recorded for the day t, $t \in \{t_1, \ldots, t_2\}$.
  - $\mathbb{1}_h^*$ the binary variable indicating the presence of electric heating in the household i.
  - $\mathbb{1}_{wh}^*$ the binary variable indicating the presence of an electric sanitary hot water system in the household i.

5.4 Application to end-of-month electrical consumption forecasting

We wish to forecast the individual end-of-month electrical consumption of residential customers. The issue being that when the historical data available for an individual is short, forecasting methods provide unreliable forecasts.

We adapt the methodology presented in Chapter 4 of Bayesian transfer learning for panel data analysis. The goal is to use information from customers with longer historical consumption, after training Bayesian hierarchical models on them and applying the information to the prior of the parameters of an individual model of a new customer, with shorter historical data. The model on the new customer is then trained with the short historical available and the informative prior. After the training, we focus on forecasting, using the posterior predictive distribution. The forecast of the end-of-month consumption is done in two steps:

— We forecast, each point of the horizon left of the month for the new customer, given the length of its historical data at the beginning of the month.

— The forecasts obtained at each timestep considered are summed to have a forecast of the end-of-month consumption of the new customer.

5.4.1 Weakly informative approach

We apply and compare two models described in the following section, for estimating the individual consumption of a panel of i customers, with historical time series of length T . We denote the models as follows:

— $(M1^{\mathcal{D}^1})$: the consumption is modeled with a Bayesian hierarchical regression using the temperature available at time t, the possession of an electrical heating system, and the possession of an electrical water heating system.

— $(M2^{\mathcal{D}^1})$: this model is an adaptation of $(M1^{\mathcal{D}^1})$ where an additional term is added, the lagged consumption of the previous day.

Remark 20

- *The lagged consumption of the previous day is considered because we want to see whether using historical data available is relevant to the task at hand. We restrict ourselves to the previous day only because the ultimate goal is to forecast the consumption using short historical data. In real world applications, the consumption of previous days may not always be available, thus we restrict ourselves in this study. Future applications may involve going further in the past, and comparing whether there is an interest in doing so.*
- *We choose to use few variables here, because those are the most frequently available for residential customers. It could be relevant for future applications to add more variables to the models, for instance, the possession of an air conditioning, a heated pool or an electric vehicle as they may have a great influence on individual consumption.*

For each individual household, we have the following model named $(M1^{\mathcal{D}^1})$:

$$X_t^{(i)} \sim \mathcal{N}(\beta_{intpt}^{(i)} + \beta_h^{(i)}(T_t^i - u_h)\mathbb{1}_{T_t^{(i)} < u_h}\mathbb{1}_h + \beta_{wh}^{(i)}\mathbb{1}_{wh}, \sigma_X^2),$$

for $t \in 1, \ldots, T$

The parameters u_h indicates the minimum temperature level for which the household start their heating system. The parameters $\beta^{(i)} = \begin{pmatrix} \beta^{(i)}_{intpt} \\ \beta^{(i)}_{h} \\ \beta^{(i)}_{wh} \end{pmatrix}$ follow a multivariate prior normal distribution as such:

$$\beta^{(i)}|\beta^{M1}, \Sigma^{M1} \sim \mathcal{N}(\beta^{M1}, \Sigma^{M1}), \text{for } i = 1, \ldots, n$$

The mean vector of the normal distribution is:

$$\beta^{M1} = \begin{pmatrix} \beta_{intpt} \\ \beta_{h} \\ \beta_{wh} \end{pmatrix}$$

Each hyperparameter $\beta_{.}$ follows a weakly informative hyperprior (a normal distribution with large variance). The covariance matrix Σ^{M1} is:

$$\Sigma^{M1} = \begin{pmatrix} \sigma^2_{\beta_{intpt}} & 0 & 0 \\ 0 & \sigma^2_{\beta_h} & 0 \\ 0 & 0 & \sigma^2_{\beta_{wh}} \end{pmatrix}$$

The diagonal elements of Σ^{M1} each follow a Half-Cauchy prior distribution (see Gelman (2006)).

For the purpose of comparison, we consider another model named ($M2^{\mathcal{D}^1}$). We consider a supplementary variable to our model which is the lag of consumption of the previous day:

$$X^{(i)}_t \sim \mathcal{N}(\beta^{(i)}_{intpt} + \beta^{(i)}_h (T^i_t - u_h)\mathbb{1}_{T^{(i)}_t < u_h}\mathbb{1}_h + \beta^{(i)}_{wh}\mathbb{1}_{wh} + \beta^{(i)}_{lag} X^{(i)}_{t-1}, \sigma^2_X), \text{for } t = 1, \ldots, T$$

The parameters $\beta^{(i)} = \begin{pmatrix} \beta^{(i)}_{intpt} \\ \beta^{(i)}_{h} \\ \beta^{(i)}_{wh} \\ \beta^{(i)}_{lag} \end{pmatrix}$ follow a multivariate prior normal distribution as such:

$$\beta^{(i)}|\beta^{M2}, \Sigma^{M2} \sim \mathcal{N}(\beta^{M2}, \Sigma^{M2}), \text{for } i = 1, \ldots, n$$

The mean vector of the normal distribution is:

$$\beta^{M2} = \begin{pmatrix} \beta_{intpt} \\ \beta_h \\ \beta_{wh} \\ \beta_{lag} \end{pmatrix}$$

Each hyperparameter $\beta_{.}$ follows a weakly informative hyperprior as in $(M1)$. The covariance matrix Σ^{M2} is:

$$\Sigma^{M2} = \begin{pmatrix} \sigma^2_{\beta_{intpt}} & 0 & 0 & 0 \\ 0 & \sigma^2_{\beta_h} & 0 & 0 \\ 0 & 0 & \sigma^2_{\beta_{wh}} & 0 \\ 0 & 0 & 0 & \sigma^2_{\beta_{lag}} \end{pmatrix}$$

The training subset is used to learn $(M1^{\mathcal{D}^1})$ and $(M2^{\mathcal{D}^1})$. Once training is done, we sample in the posterior distribution. The samples are used to obtain the mean and covariance of the posterior distribution of β^{M1} and β^{M2} denoted $\hat{\mu}^{M1}_\beta$, $\hat{\mu}^{M2}_\beta$, $\hat{\Sigma}^{M1}_\beta$ and $\hat{\Sigma}^{M2}_\beta$. The mean and covariance are the information that we are going to transfer to the model of the new individual with shorter historical data.

5.4.2 Informative approach

From the estimation of the model $(M1^{\mathcal{D}^1})$, we get the mean and covariance of the posterior distribution of the parameters $\hat{\mu}^{M1}_\beta$ and $\hat{\Sigma}^{M1}_\beta$. We consider the hierarchical model $(M1)$ for the dataset $\mathcal{D}^*$ with shorter historical data:

$$X^*_t \sim \mathcal{N}(\tilde{\beta}^*_{intpt} + \tilde{\beta}^*_h(T^i_t - \tilde{u}_h)\mathbb{1}_{T^*_t < \tilde{u}_h}\mathbb{1}_h + \tilde{\beta}^*_{wh}\mathbb{1}_{wh}, \sigma^2_{X^*})$$

The parameters $\tilde{\beta}^* = \begin{pmatrix} \tilde{\beta}^*_{intpt} \\ \tilde{\beta}^*_h \\ \tilde{\beta}^*_{wh} \end{pmatrix}$ follow a multivariate normal distribution.

We have:

$$\tilde{\beta}^* \sim \mathcal{N}(K_{M1} \cdot \hat{\mu}^{M1}_\beta, l^{-1}_{M1} \cdot \hat{\Sigma}^{M1}_\beta),$$

where $K_{M1} = diag(k_{M1})$. $\hat{\mu}_{\beta_{M1}}$ and $\hat{\sigma}_{\beta_{M1}}$ are the mean and covariance of the posterior re-

sulting from model $(M1)$. As in 4.2.2, the parameters k_{M1} and l_{M1} to model the similarity between the new individual and the dataset $\mathcal{D}^1$.

Similarly, we can adapt a model $(M2)$ for the individual, using the posterior mean and covariance denoted $\hat{\mu}_\beta^{M2}$ and $\hat{\Sigma}_\beta^{M2}$ of model $(M2^{\mathcal{D}^1})$. The model $(M2)$ is the following:

$$X_t^* \sim \mathcal{N}(\tilde{\beta}_{intpt}^* + \tilde{\beta}_h^*(T_t^i - \tilde{u}_h)\mathbb{1}_{T_t^* < \tilde{u}_h}\mathbb{1}_h + \tilde{\beta}_{wh}^*\mathbb{1}_{wh} + \tilde{\beta}_{lag}^* X_{t-1}^{(i)}, \sigma_{X^*}^2)$$

The parameters $\tilde{\beta}^* = \begin{pmatrix} \tilde{\beta}_{intpt}^* \\ \tilde{\beta}_h^* \\ \tilde{\beta}_{wh}^* \\ \tilde{\beta}_{lag} \end{pmatrix}$ follow a multivariate normal distribution:

$$\tilde{\beta}^* \sim \mathcal{N}(K_{M2} \cdot \hat{\mu}_\beta^{M2}, l_{M2}^{-1} \cdot \hat{\Sigma}_\beta^{M2}),$$

where $K_{M2} = diag(k_{M2})$. As for the model $(M1)$, the parameters k_{M2} and l_{M2} are introduced to model the similarity between the new individual and the dataset $\mathcal{D}^1$.

The models $(M1)$ and $(M2)$ are trained on the testing subset, with short historical data.

5.4.3 Forecasting the end-of-month electrical consumption of individual customers

After inference of the models $(M1)$ and $(M2)$, we can sample from the posterior predictive distribution (see Equation (4.13)) for a forecasting horizon h.

For a given individual, we denote t_{first} and t_{last} the first and last days of the month considered. The value of an individual end-of-month consumption is:

$$C^* = \sum_{t=t_{first}}^{t_{last}} X_t^* \tag{5.1}$$

When some historical data is available at the beginning of the month (e.g. the first week of data), we adapt the calculation above. Let $t_{first} \leq t_p \leq t_{last}$ the last day of historical data available for the individual, for the month considered. We forecast the values of X_t^*

for $t_p < t \leq t_{last}$. We denote $\hat{X}_t^*$ the forecast. The end-of-month consumption is:

$$\hat{C}^* = \sum_{t=t_{first}}^{t_p} X_t^* + \sum_{t=t_p+1}^{t_{last}} \hat{X}_t^* \tag{5.2}$$

In a Bayesian setting, we rely on the posterior predictive distribution to obtained the point estimates of the forecast as well as prediction intervals.

From (4.13), we sample $J = 1000$ realizations to forecast $j = 1, \ldots, J$ values of X_t^* for $t_p < t \leq t_{last}$. Let $\hat{X}_t^{*(j)}$ denote the j-eth forecast of X_t^*, for $t_p < t \leq t_{last}$. We then have a collection of end-of-month consumption realizations denoted $\hat{C}^* = \hat{C}^{*(j)}$, from which we can deduce point estimates (here, we consider the maximum a posteriori) and intervals.

Error evaluation for point estimates

To evaluate the error between the real value of individual end-of-month and a point estimate, we define:

$$\mathcal{E}(C^*, \hat{C}^*) = \left| \frac{C^* - \hat{C}^*}{C^*} \right| \tag{5.3}$$

5.4.4 Results

In this Section, we show the results obtained from the approach. We wish to forecast the consumption at the end of a given month, for customers with shorter historical data, for different forecasting horizons.

The numerical applications are conducted with R (R Core Team (2019)), the Bayesian models and estimation are carried out with the Stan software (see Carpenter et al. (2017) and Stan Development Team (2018)) and used in the R interface to Stan (Stan Development Team (2020)).

We focus on forecasting the consumption at the end of february. Specifically, we consider the following length of historical data:

— $t_2 = 37$ days, we predict the next 22 days

— $t_2 = 45$ days, we predict the next 14 days

— $t_2 = 53$ days, we predict the next 7 days

We predict each day, using a sample from the posterior predictive distribution and we sum the values of each realization of the sample to obtain the predictions. Point estimates

are given using the Maximum a Posteriori, and we compute the MAPE between the point estimates and the real value.

Hereinafter, we display the performances of the models on the forecasting of the end-of-month consumption, for varying horizons of forecasting. Results are summarized for the 93 individuals of the testing subset on which the methodology has been applied.

We compare the performances of $(M1)$ and $(M2)$ with the informative approach and without transferring information, as is done in 4.3.

The forecasting errors obtained when the historical data available is the shortest ($t_2 =$ 37 days) are shown on Figure 5.1. We can see that the best model here in terms of performances is the $(M2)$ model with transfer of information learned from the panel with longer historical data. The $(M1)$ model display comparable median errors with and without transfer (left and right of Figure 5.1 respectively). The error of $(M1)$ with transfer is more dispersed.

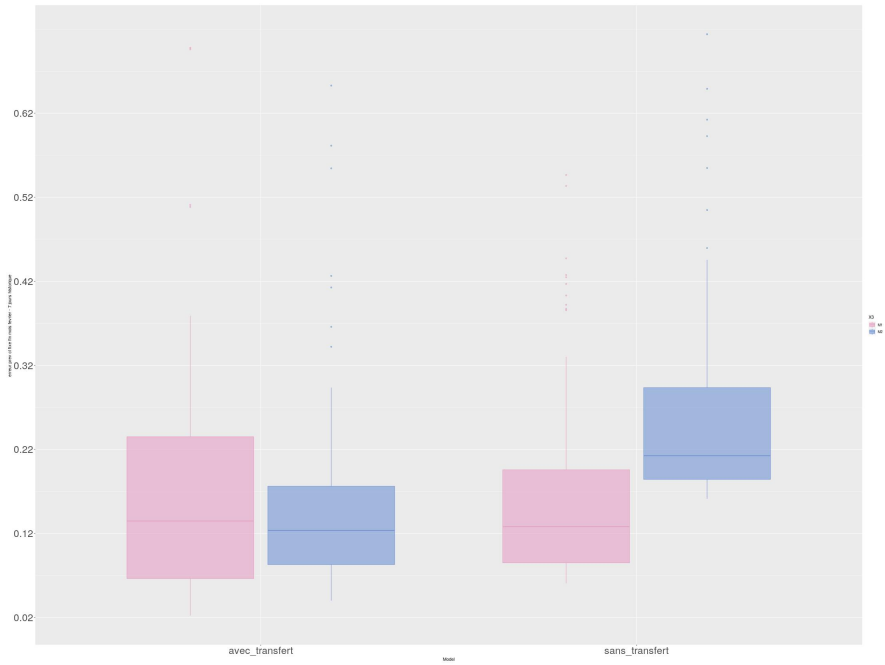


Figure 5.1 – Boxplots of $\mathcal{E}(C^*, \hat{C}^*)$ obtained with the point estimates for each individual in the training set, given a horizon of 22 days of data. [Left] The error is displayed for the $M1$ [Pink] and $M2$ models [Blue]

When we wish to forecast the consumption of the new individuals using $t_2 = 45$ days of data, the results obtained with both models are displayed in Figure 5.2. The

errors have decreased compared to Figure 5.1, which is to be expected, as more training data is available. The performances are similar as previously, as the $(M2)$ model with transfer seems to give out the lowest errors. Again, the $(M1)$ model with transfer has more dispersed errors.

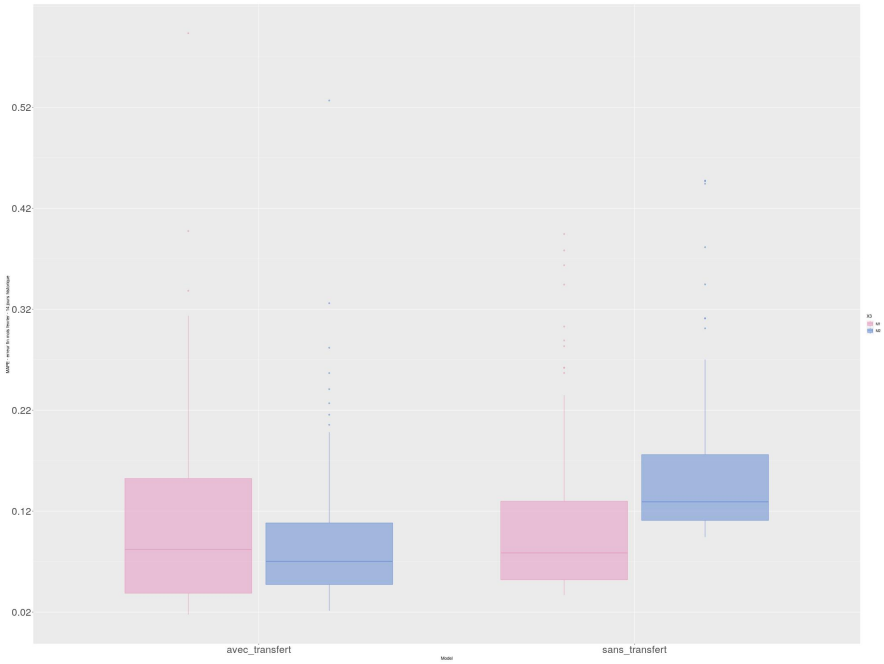


Figure 5.2 – Boxplots of $\mathcal{E}(C^*, \hat{C}^*)$ obtained with the point estimates for each individual in the training set, given a horizon of 14 days of data. [Left] The error is displayed for the $M1$ [Pink] and $M2$ models [Blue]

Finally, the errors obtained with the longest historical data of the testing set considered here are displayed in Figure 5.3. The errors have decreased compared to the two other cases, as is to be expected. The best model, out of the four considered, is again the $(M2)$ model with transfer. All the boxplots of errors are smaller, because the larger the historical data is available, the closer the performance between informative and noninformative approaches will get. The $(M1)$ model with transfer gives out the second best performances here, and the errors are less dispersed than in the previous cases.

We also look are the length of 80% prediction intervals for the end-of-month consumption of each of the individuals and the real value of their consumption at the different horizons considered. The results are displayed in 5.4 for the varying horizons considered. When the horizon is the shortest (i.e. $t_2 = 37$ days), the coverage probability of the

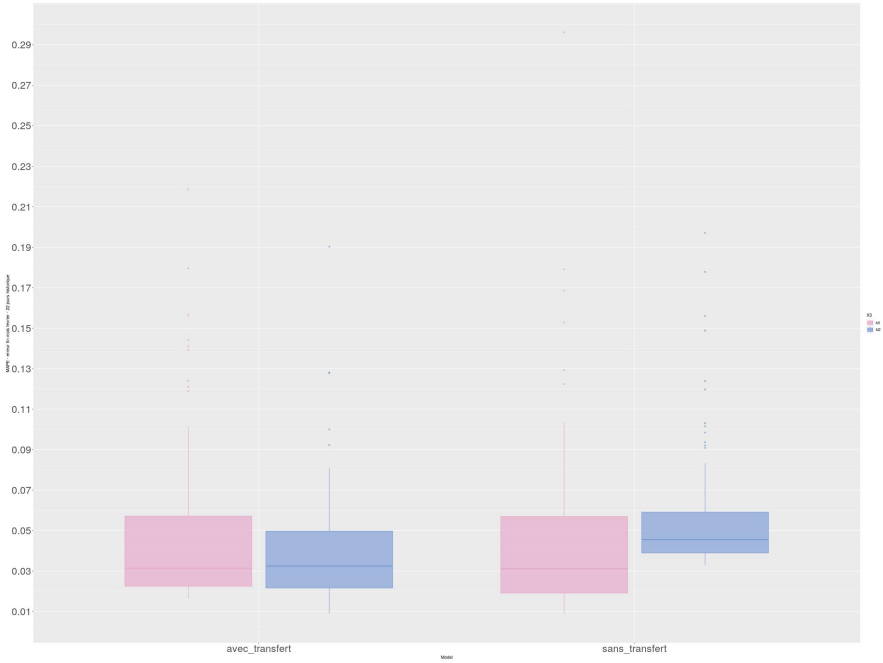


Figure 5.3 – Boxplots of $\mathcal{E}(C^*, \hat{C}^*)$ obtained with the point estimates for each individual in the training set, given a horizon of 7 days of data. [Left] The error is displayed for the $M1$ [Pink] and $M2$ models [Blue]

prediction intervals for all the models is very small. The largest coverage probability is obtained with the $(M2)$ model without transfer, however, this is also the model with the largest intervals (see Figure 5.4a). When the horizon grows ($t_2 = 45$ days), the coverage probabilities for all models increases (see Figure 5.4b). The largest coverage probability is again obtained with the $(M2)$ model without transfer, but the length of the prediction intervals is much larger than with the others models. The $(M2)$ model with transfer has the second largest coverage probability in this case. The length of prediction intervals is also much smaller than the $(M2)$ model without transfer. For the last case ($t_2 = 53$ days), the observations are similar to the two other cases. The coverage probability of the $(M1)$ model without transfer improves slightly, but the length of intervals is the smallest of the four. This model fails to capture correctly the true value of the end-of-month consumption. Overall, the $(M2)$ model with transfer seems to have reasonable sized prediction intervals. The coverage probability is around 70% of the testing subset in this case.

Given all the observations, the $(M2)$ model with transfer is the best one out of the four for the different horizons considered.

Overall, considering the results obtained on real world data, we see a benefit to using the Bayesian transfer learning methodology as described in Chapter 4. Even as the horizon of forecasting grows, the transferring approach remains relevant with both models.

5.5 Conclusion

Individual load forecasting is a challenging task due to the great variety of residential customers. We have showed here an application of the Bayesian approach proposed in Chapter 4 to real world data. The forecasting is improved with the use of Bayesian transfer learning. This is promising for potential industrial applications. The main drawback of Bayesian approaches, is the duration of training of the models. Once the weakly informative approach is done, however, the transfer learning on individual customers is quite fast. In real-world applications, some information on household are complicated to obtain. For this reason, and to simplify the individual models considered here, we focus on integrating only certain information available to our models.

However, it may be interesting to add variables to both models $M1$ and $M2$ such as the housing surface, calendar information like holidays, and week-ends, possession of an electric vehicle, the possession of an electric air conditioning system etc. One can easily imagine adding a similar threshold parameter for the case of air conditioning. Future applications may consider adapting the models with other variables and evaluate their relevancy.

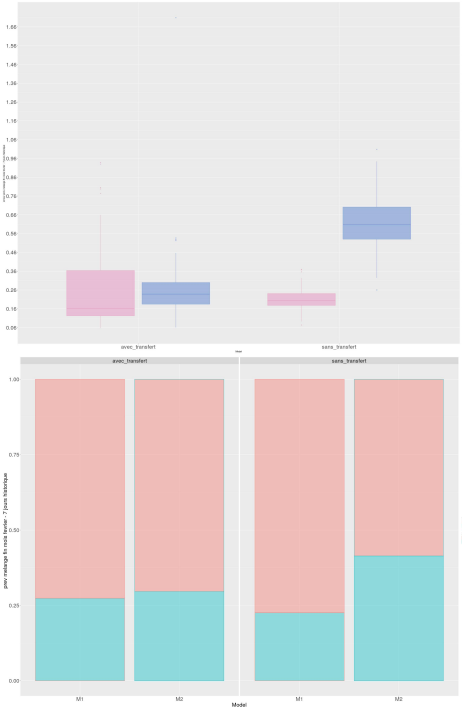
Bessani et al. (2020) use only certain lagged consumption to their model, based on the mutual information between the consumption at time t and the consumption at time $t-k$. The lagged consumption integrated to the model are those that present the highest mutual information index. This may be an interesting approach for future applications, to integrate temporal dependency to the Bayesian models.

The results shown on the testing subset of the CER dataset shows the benefit of the transfer learning through the prior for the individual model of new customers. The ($M2$) model with transfer gives out the best results in terms of error out of the four models considered. The observations are consistent with what was seen in Chapter 4. The performances of the transfer, for forecasting, seems to be degraded when looking at the coverage probabilities. This matches the observations made in 4.3.2, where the model without transfer seems to outperform the model with transfer. However, usually, the prediction intervals obtained with the models without transfer are much larger, making them more imprecise as well.

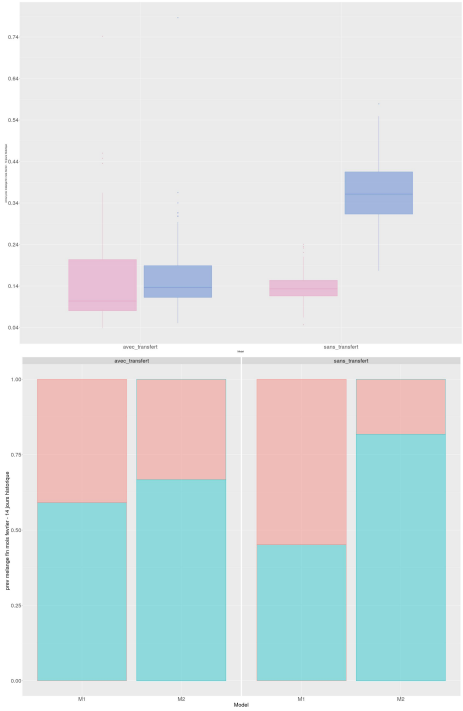
In this Chapter, the dataset used for $\mathcal{D}^1$ the panel with long historical data and $\mathcal{D}^*$ the individuals with shorter historical data is a subset of the CER dataset (see Commission for Energy Regulation (2012)). The construction of this subset is detailed in Chapter

6. The goal is to have $\mathcal{D}^1$ as close as possible to the new customer $\mathcal{D}^*$. We detail the methodology for the building the subset, using deep learning methods and clustering as well as affecting the individuals to the clusters. The results presented in this Chapter focus on a fixed cluster.

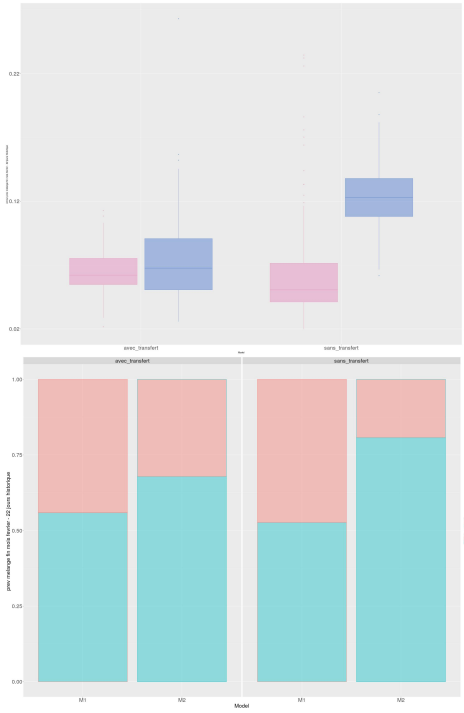
We discuss the affectation of the new individuals to the clusters using the majority rule. The issue being that miss-classified individuals will have a subsequent forecast based on the model of the wrong class. To alleviate this issue, we show in Chapter 6 a method to obtain mixture predictions for the new individual instead of affecting them to a unique class.



(a) Horizon: 22 days



(b) Horizon: 14 days



(c) Horizon: 7 days

Figure 5.4 – For each panel: [top] length of prediction intervals, [bottom] coverage probability. The results are displayed for $M1$ [Pink] and $M2$ models [Blue]. For each case, [Left] boxplots are the models with transfer and [Right] are the models without transfer.

Chapter 6

Classifying new customers into homogeneous groups with deep learning

6.1 Introduction

In Chapter 5, we describe an application of Bayesian transfer learning to forecasting the end-of-month consumption of residential customers. The methodology is applied to a subgroup of the database from the CER Commission for Energy Regulation (2012). In this Chapter, we describe the methodology designed to build the subgroup used in Chapter 5.

When modeling the individual electrical consumption, many factors may intervene such as: the outside temperature, the possession of an electric heating system, an electric sanitary water heating, the surface of the household, the number of appliances, the number of individuals in the household etc. Considering the diversity of customers whether in terms of behavior, household characteristics or appliances, it appears relevant to regroup them into homogeneous subgroups before adopting a Bayesian approach. In real world applications, household characteristics may not always be available directly when the customer subscribes to a utility contract. This poses an issue when we rely only on those characteristics to regroup the individuals. The widespread deployment of smart meters allows utility companies to have precise information on the household consumption, using the load curve (at different time periods). For those reasons, we focus on building homogeneous groups of customers based on their individual load curve rather than only the household characteristics.

The availability of smart meter data, while beneficial for load analysis and demand response, poses new challenges due to the high dimensional nature of the data. Many statistical analysis task fail in high dimension, thus requiring a prior dimensionality re-

duction step. Individual load curves are nonlinear time series, thus necessitating appropriate dimensionality reduction approaches. Wavelet-based approaches are a classical tool for time series representation and dimensionality reduction. Vlachos et al. (2003) adopt a wavelet-based approach for time series clustering.Antoniadis et al. (2013) present clustering strategy based on the wavelet transform to reduce dimension of time series. Auder et al. (2018) adopt the strategies proposed by Antoniadis et al. (2013) to the clustering of electrical load curves, including the wavelet-based dimensionality reduction step. Other statistical methods for dimensionality reduction techniques applied to time series include Piecewise Aggregate approximation, The Discrete Fourier Transform model-based approaches such as Generalized Additive Models (see Laurinec and Lucká (2018) for an application) etc. For more information on those approaches for dimensionality reduction on time series, refer to Laurinec and Lucká (2016).

Deep learning methods have also been considered for the dimensionality reduction of time series. Madiraju et al. (2018) propose a deep learning based approach for dimensionality reduction for temporal clustering. The architecture of the encoder relies on 1D convolutional layers as well as Bidirectionnal Long Short Term Memory blocks. Ma et al. (2019) use a Recurrent neural network encoder-decoder architecture combined with clustering for time series. Richard et al. (2020) propose a 1D convolutionnal autoencoder based approach to dimensionality reduction for clustering electrical load curves. Kong et al. (2020b) use a stacked autoencoder architecture for dimensionality reduction and feature extraction of electrical load curve.

A review of time series clustering has been proposed by Aghabozorgi et al. (2015).

Neural networks are massively used nowadays for multiclass-classification. Several applications exist in image and speech recognition (see for instance He et al. (2016), Simonyan and Zisserman (2015) and Graves et al. (2013)). A review of deep learning methods applied to time series classification is provided in Fawaz et al. (2019).

In this Chapter, we focus on building homogeneous groups of customers based on their individual electrical load curves, using the CER database (Commission for Energy Regulation (2012)). Hereinafter, the following aspects are tackled:

- Dimensionality reduction based on deep neural networks. We focus on two architectures of autoencoders designed to reduce the dimension of individual load curves.
- Clustering the individual into groups. We use clustering methods, on the load curves in reduced dimension, in order to build homogeneous groups of individuals.
- Multi-class classification of new individuals, with short historical data, into the

groups. We design a deep neural network to classify the individuals to the labels obtained from the previous clustering step.

We adopt a similar approach to Richard et al. (2020), as we adapt a 1D convolutional autoencoder to first reduce the dimension of the individual load curves and perform clustering on the latent space.

Section 6.2 briefly describes the notations used throughout the Chapter. In Section 6.3, we specify the two architectures of neural networks used to perform dimensionality reduction on the time series. In Section 6.4, we present the clustering methods applied to the reduced load curves and the methods used to select the optimal number of clusters. Section 6.4 details the method and structure of the neural network used to classify new individuals into the clusters. In Section 6.6, we describe the dataset used and the results obtained for the dimensionality reduction methods, the clustering and the classification. In Section 6.7, we present an adaptation of the end-of-month forecasting developed in Chapter 5, where we make use of softmax scores to obtain mixture predictions. Section 6.8 is dedicated to concluding this Chapter and perspectives for future work.

6.2 Notations

The notations used in this Chapter are the following: Hereinafter we refer to $n = 365$ as the length of individual load curves, $m = 3159$ the number of individuals. We denote $X = (X_t^{(i)})_{\substack{1\leq i\leq m,\\ 1\leq t\leq n}}$, the matrix of dimension (m, n), containing the consumption time series of m individual customers, and of length n, where $X_t^{(i)}$ indicates the consumption at time $t \in \{1 \dots n\}$ of each individual customer $i \in \{1, \dots, m\}$.

- $n = 365$ denotes the length of each individual load curve for the training set
- $m = 3159$ denotes the number of individual customers
- $X = (X_t^{(i)})_{\substack{1\leq i\leq m,\\ 1\leq t\leq n}}$, is the matrix of dimension (m, n) containing the consumption time series of m individual customers, and of length n.
- $I = (I_p^{(i)})_{\substack{1\leq i\leq m,\\ 1\leq p\leq d}}$, is the matrix of dimension (m, n) containing the load curve of m individual customers in an encoded space of dimension d. I is the reduced representation of X, obtained with a dimensionality reduction technique. We refer to I as the latent space or encoded space. d is the dimension of encoding.
- C denotes the number of desired clusters built using I, given any clustering method. C_k denotes the k-eth cluster of size n_k, for $1 \leq k \leq C$.

6.3 Dimensionality reduction

Each of the individual load curves are in dimension 365 (over a year, at a daily period). Since we wish to cluster individuals using their load curves, clustering techniques may fail in high dimension. Thus, before clustering the customers, we wish to reduce the dimension of their load curves.

Deep neural networks have been gaining traction in various domains (Krizhevsky et al. (2012), He et al. (2016)). A specific architecture of neural networks has been applied to perform dimensionality reduction: autoencoders Hinton and Salakhutdinov (2006). They take the load curves X as input and output as such:

$$\begin{aligned} I &= e(X), \\ X &= d(I), \end{aligned}$$

where e and d are respectively the encoding and decoding function, constituted of several nonlinear transformations, also referred to as layers of the neural network. I denotes the load curves in reduced dimension, we refer to I as the latent or encoded space.

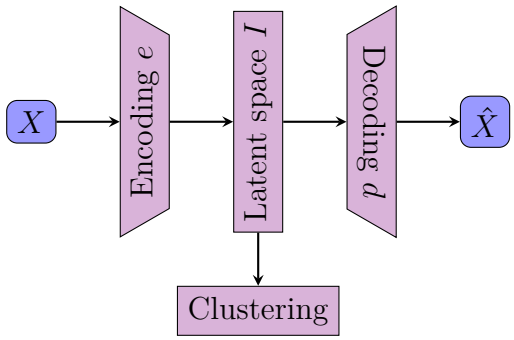


Figure 6.1 – Outline of an autoencoder (Section 6.3). Encoding and Decoding contain several layers designed to first reduce the dimension of the inputs X and then reconstruct them. Clustering is operated on the latent space I

Here we consider two types of layers for the encoding and decoding functions: Fully Connected layers (or Dense layers) and one-dimensional convolutional layers (1D-convolutions).

6.3.1 Dense layers

An autoencoder constituted of $i+1$ Dense layers is defined by:

$$\begin{aligned}
\mathbf{h}_1 &= \sigma(\mathbf{W}_1^T \cdot X + \mathbf{b}_1), && (6.1)\\
&\vdots \\
I &= \sigma(\mathbf{W}_d^T \cdot \mathbf{h}_{d-1} + \mathbf{b}_d), \\
\mathbf{h}_{d+1} &= \sigma(\mathbf{W}_{d+1}^T \cdot I + \mathbf{b}_{d+1}), \\
&\vdots \\
\mathbf{h}_i &= \sigma(\mathbf{W}_i^T \cdot \mathbf{h}_{i-1} + \mathbf{b}_i), \\
X &= \sigma(\mathbf{W}_{i+1}^T \cdot \mathbf{h}_i + \mathbf{b}_{i+1}), && (6.2)
\end{aligned}$$

Here, X are both the inputs and output of the neural network, $h_j,\ 1 \leq j \leq i$ are the outputs of the hidden layers, W_p and b_p are the weights and biaises of the networks, $1 \leq p \leq i$. I is the latent space, a reduced representation of the inputs X obtained through several successive linear and nonlinear transformations.

The function σ is the activation function, it can be a linear of non linear transformation, depending on the task the neural network performs. A common choice for σ is the Rectified Linear Unit (ReLU):

$$\sigma(x) = \max(0, x), \text{for all } x \in \mathbb{R} \tag{6.3}$$

6.3.2 1D convolutions layers

Usually convolutional neural networks are applied to image data and refer to 2D convolutions. We focus on a specific case of convolutions, namely 1D convolutions designed specifically for signals. For an extended review on 1D convolutions refer to Kiranyaz et al. (2021). An example of neuron of a 1D-convolutional layer is as such:

$$h_{i,j}^l \quad = \quad \sigma(b_j + \sum_{p=1}^{P} w_{p,j}^l x_{i+p-1,j}^{l-1}) \tag{6.4}$$

where $h_{i,j}^l$ is the i-eth output of the layer $l,\ 1 \leq l \leq L$. b_j are the biases and $w_{p,j}$ the kernels (also referred to as weights) for the j-eth feature map and x^{l-1} is the signal (or output of the previous layer).

An autoencoder designed with 1D-convolutional layer is built similarly to an autoencoder with Dense layers (as described in (6.1)-(6.2)), by stacking convolutional layers (varying in dimension to encode the input data) as defined in (6.4).

6.4 Clustering

We wish to build homogeneous subgroups of individuals, using clustering methods. However clustering is often a complicated task to perform in high dimension. Considering that we wish to use build customers' individual load curves to build the clusters, we are going to use a representation of the load curves in a reduced dimension to run the clustering algorithms.

Similarly to using Principal component analysis as a pre-processing step prior to clustering, we use the encoded space of the autoencoder and proceed to clustering on this data representation.

6.4.1 Hierarchical agglomerative clustering (HAC)

Hierarchical agglomerative clustering is a method designed to group similar observations into clusters.

Let us consider a set of m individual to cluster. Initially, each individual constitutes its own class. The HAC algorithm seeks to construct C groups, where $C < m$, based on the similarity of each individual.

Iteratively, the clusters are built by merging previous clusters, based on their similarity. Specifically, the algorithm computes a dissimilarity matrix between each of the clusters and seeks the minimum as a criteria of merging. The process is repeated until $C = 1$.

The main advantage of HAC is the fact that the number of clusters is not specified by the user beforehand. The relevant number of groups to identify the partition afterwards remains a challenge as for the other clustering algorithms. It is also computationally expensive, compared to other methods and does not scale to large datasets. The main method to select an appropriate number of clusters is through the use of a dendrogram which shows the hierarchy between all the individuals. This method of selection, like most methods is highly subjective and is usually efficient when the size of the dataset is small.

More details on Hierarchical agglomerative clustering and choices of dissimilarity are provided in Ward (1963), Candillier (2006) and Murtagh and Legendre (2014).

6.4.2 K-Means and K-Medoids clustering

In this section, we describe two partition based-methods of clustering applied to the data in reduced dimension.

K-Means

K-Means, first introduced by is a classical clustering method. Given a set of m observations, the algorithm aims at regrouping each of them into C groups. Several implementations of the algorithm exist (see Lloyd (1982),MacQueen et al. (1967) and Forgy (1965)).

Here, we describe the implementation of Hartigan and Wong (1979). The K-Means algorithm is applied to the latent space $I = (I_{i,j})_{\substack{1\leq i\leq m,\\ 1\leq j\leq d}}$, obtained from the autoencoder. d refers to the dimension of the latent space after encoding. In K-Means, the number of desired clusters C is defined by the user.

The algorithm is the following:

- **Initialize** C centroids randomly from the m data points
- Assign each data point to the cluster of which they are closest to the centroid.
- For each cluster k, $1 \leq k \leq C$, update its centroid (mean of the points of the cluster k)
- For each cluster k, $1 \leq k \leq C$, if the centroid of k was updated:
  - ▷ Assign each data point I_i, $1 \leq i \leq m$ to the cluster, where the updated centroid is the closest
  - ▷ Update the cluster centroid by calculating the mean of data points.
- **Repeat until** no data point changes its assignment.

K-Means relies on iteratively assigning data points to clusters. Given a data point I_i and a centroid denoted M_k, assignment is done by minimizing the cluster Within-sum-of-square:

$$SSE_k = \sum_{I_i \in k} (I_i - M_k)^2 \tag{6.5}$$

K-Medoids

The K-Medoids clustering algorithm (see Kaufman and Rousseeuw (1990)) is a partitioning methods similar to k-means. As with k-means, the number of clusters is specified before running the algorithm.

As for the K-Means algorithm, we apply the K-Medoids algorithm on the latent space obtained from the autoencoder denoted $I = (I_{i,j})_{\substack{1 \leq i \leq m, \\ 1 \leq j \leq d}}$, where d is the dimension of the encoding.

Let C be the number of clusters specified. The K-Medoids algorithm applied to the latent space I is:

- **Initialize** M_k, $1 \leq k \leq C$ medoids from the m data points
- Assign each data point to the cluster of which they are closest to the medoid.
- For each cluster k, $1 \leq k \leq C$:
  - ▷ For each non medoid data point I_i where $1 \leq i \leq m_k$, m_k the number of points of the cluster k.
  - ▷ Swap the roles of I_i and M_k:
  - ▷ I_i becomes the medoid of the cluster k, compute the dissimilarity of the cluster:

$$D_k = \sum_{j=1}^{m_k} d(I_i, I_j) \tag{6.6}$$

  - ▷ Replace the medoid M_k with the point I_i such that D_k is minimal
  - ▷ Reallocate each non medoid data point to the closest medoid
- **Repeat until** a dissimilarity decreases and swaps occur.

Remark 21 *The K-Medoids algorithm is more robust to outliers and atypical observations than the K-Means. This is due to the use of medoids as centers of the clusters as well as the use of distances less sensitive to outliers.*

As for scalability is high dimension, both methods suffer from it. Thus we apply them on a reduced representation of the curve rather than the initial data.

6.4.3 Selection of the number of cluster

A challenging aspect of clustering is selecting a number of cluster that seems suitable to the data at hand. Many methods exists and can be separated into two categories : internal and external evaluation methods.

External methods of evaluation consists in evaluating the clustering based on a testing dataset containing known class labels. In our case, known labels are unavailable for the considered data, thus we focus on internal evaluation methods.

Internal evaluation methods focus on measuring the within class and between class inertia. The Silhouette coefficient Rousseeuw (1987), the Davies-Bouldin index Davies and Bouldin (1979) and Dunn's index Dunn (1974) are among the usual methods of internal evaluation.

Using the notations from Hassani and Seidl (2017), we define Dunn's index as follows. Let C_i denote the i-eth cluster and $d(x, y)$ the distance between two data points x and y, Dunns's index is:

$$\mathcal{D} = \frac{\min_i \min_j (\min_{x \in C_i, y \in C_j} d(x, y))}{\max_k (\max_{x, y \in C_k} d(x, y))}. \tag{6.7}$$

The numerator of Equation (6.7) minimizes the intercluster distance between clusters while the denominator maximizes the intracluster distance The optimal number of cluster is obtained when $\mathcal{D}$ is maximal.

Let C denote the number of clusters, The Davies-Bouldin index is defined by:

$$\mathcal{DB} = \frac{1}{C} \sum_{i=1}^{C} \max_{i \neq j} \frac{\frac{1}{n_i} \sum_{x \in C_i} d(x, c_i)) + \frac{1}{n_j} \sum_{x \in C_j} d(x, c_j)}{d(c_i, c_j)}, \tag{6.8}$$

where n_i, $1 \leq i \leq C$ is the size of the cluster C_i. The optimal number of cluster, based on the Davies-Bouldin index, is obtained when $\mathcal{DB}$ is minimal.

The Silhouette score, is also a popular choice of method for selecting a suitable number of cluster. For $x \in C_i$, we define:

$$a(x) = \frac{1}{n_i - 1} \sum_{y \in C_i, y \neq x} d(x, y), \tag{6.9}$$

and

$$b(x) = \min_{j \neq i} \frac{1}{n_j} \sum_{y \in C_j} d(x, y). \tag{6.10}$$

The Silhouette coefficient is:

$$S = \frac{1}{C} \sum_{i=1}^{C} (\frac{1}{n_i} \sum_{x \in C_i} \frac{b(x) - a(x)}{max(a(x), b(x))}). \tag{6.11}$$

The optimal number of clusters, given a clustering method, is the one that maximizes the Silhouette coefficient.

In section 6.6, we compare the three evaluation methods presented here, to select an appropriate number of clusters, obtained with the three methods of clustering considered.

Selection of the optimal number of clusters, despite the use of existing methods remains a highly subjective task.

6.5 Affecting a new customer to the clusters

Once the clustering labels have been set, we wish to assign those labels to new customers. We build a neural network to assign customers' load curves to the given labels. Considering that customers' load curves in reality can be of varying length, as they may subscribe during any time of the year to electrical contracts. We set the minimal length of load curve for a new customer is 28 days of any given month. We build the training data set as such:

- X are the original load curves of length n of m individuals.

  We transform the data, by transposing the matrix so that the dimension is $(m \times n, 2)$. Thus the data contains $m \times n$ rows, and 2 columns (the load value at time t for individual i, and the individual labels of the customers'.

  We select the first p days of each month and create a column labeling the month of the year for data point.

  Finally, we re-transpose the data by months' and customers' labels. We obtain two matrices:

  1. $\tilde{X}_{load}$ of dimension $(12 \times m, p)$, contains, for each row p points of the load curve of a given customer for a given month.
  2. $\tilde{X}_{month} \in \mathbb{R}^{(12 \times m) \times 12}$ contains the label of each month of the year, corresponding to each row of $\tilde{X}_{load}$.

  For a given customer $i \in \{1, \ldots, m\}$, initially, its load curve is a row of the matrix X:

$$X^{(i)} = (X_1^{(i)}, \ldots, X_n^{(i)}) \tag{6.12}$$

  Let t_1^l be the first day of the l-eth month, $1 \leq l \leq 12$. Following the transformation

described above, for the customer i, we have:

$$\tilde{X}^{(i)}_{load} = \begin{pmatrix} X^{(i)}_{t^1_1} & \dots & X^{(i)}_{t^1_1+p} \\ \vdots & \ddots & \vdots \\ X^{(i)}_{t^{12}_1} & \dots & X^{(i)}_{t^{12}_1+p} \end{pmatrix}, \tag{6.13}$$

and

$$\tilde{X}^{(i)}_{month} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \vdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}. \tag{6.14}$$

Thus, we have:

$$\tilde{X}_{load} = \begin{pmatrix} \tilde{X}^{(1)}_{load} \\ \vdots \\ \tilde{X}^{(m)}_{load} \end{pmatrix}, \tag{6.15}$$

and

$$\tilde{X}_{month} = \begin{pmatrix} \tilde{X}^{(1)}_{month} \\ \vdots \\ \tilde{X}^{(m)}_{month} \end{pmatrix}. \tag{6.16}$$

- We denote $Y \in \mathbb{R}^{(12\times m)\times C}$, the label matrix corresponding to each row of $\tilde{X}_{load}$ and $\tilde{X}_{month}$, C being the number of clusters ($C > 2$).

  Let $(y_{j,i}),\ 1 \leq j \leq (12 \times m),\ 1 \leq i \leq C$ be the elements of Y, we have:

$$y_{j,i} = \begin{cases} 1, & \text{if the } j\text{-eth individual belongs to label } i, \\ 0, & \text{otherwise.} \end{cases} \tag{6.17}$$

The task to solve is a multiclass classification. Neural networks are interesting tools in this context. Considering the nature of the data at hand, we develop a neural network approach that takes as input both the monthly time series and the months' labels associated, and aims at classifying them into the provided labels.

A neural network adapted to a multiclass classification task is trained using the categorical cross-entropy loss function:

$$\mathcal{L}(y, \hat{y}) \quad = \quad - \sum_{j=1}^{12\times m} \sum_{i=1}^{C} y_{j,i} \log \mathbb{P}(y_{j,i} = 1) \tag{6.18}$$

Specifically, the neural network is made of two separate blocks (see Figure 6.2):

- The convolutional block takes the time series as inputs of 1D-convolutional layers. The output of a convolutional block, made of L_{Conv} layers, can be written as such:

$$\tilde{I}_{load} \quad = \quad f_{L_{Conv}}(f_{L_{Conv}-1}(\dots f_1(\tilde{X}_{load}))), \tag{6.19}$$

  where each function $f_{L_{Conv}}$, $1 \leq l \leq L_{Conv}$, is as described in Equation (6.4).

— The Dense block takes the months' labels as inputs. Similarily, the output of a dense block, made of L_{Dense} layers, can be written as:

$$\tilde{I}_{month} \quad = \quad f_{L_{Dense}}(f_{L_{Dense}-1}(\dots f_1(\tilde{X}_{month}))), \tag{6.20}$$

  where each function $f_{L_{Dense}}$, $1 \leq l \leq L_{Dense}$, is as described in Equation (6.2).

- Both blocks output's are flattened and concatenated before passing through a classifying layer with a softmax activation function. Flattening the output tensors consists in converting them into one dimensional tensors.

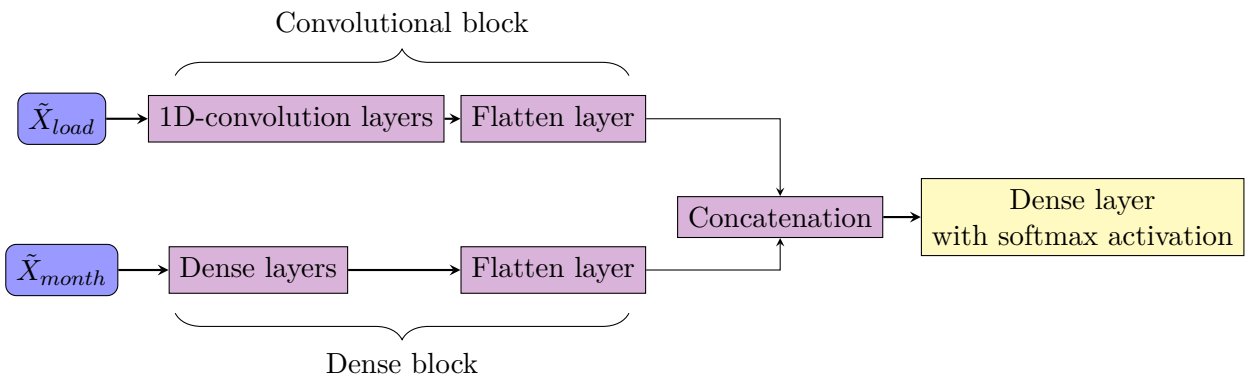


Figure 6.2 – Outline of the neural network (Section 6.5) used for classification of the customer's to the label obtained from clustering

Remark 22 *The activation function used for multiclass classification is the softmax function defined by:*

$$softmax(x)_i = \frac{\exp(x_i)}{\sum_{j=1,\dots,C} \exp(x_j)}, \tag{6.21}$$

where $x = (x_i)_{i=1,\dots,C}$, C *the number of classes.*

The softmax used as the activation function of the last layer of the neural network assign a score to each possible label. The predicted label is found by finding the label associated to the maximal score. We denote $\hat{y}$ the predicted label, defined by:

$$\hat{y} = \underset{i=1,\dots,C}{\operatorname{argmax}}(\operatorname{softmax}(x)_i) \tag{6.22}$$

6.6 Application to real data

For the implementation of the neural networks aforementioned, we use Abadi et al. (2015) and the keras package in R (see Chollet et al. (2015) and Chollet and Allaire (2018)). The K-Medoids algorithm is implemented with the *pam* function from the cluster package in R Maechler et al. (2021). We use the *hclust* and *kmeans* functions to apply respectively Hierarchical clustering and the K-Means algorithms to our data. The functions are directly available in R (see R Core Team (2019)).

Richard et al. (2020)

6.6.1 Description of the dataset

As in Chapter 5, our approach is applied to the dataset from the CER Smart Metering Project (see Commission for Energy Regulation (2012)). This customer behavior trial has been conducted between 2009 and 2010 and contains over 5000 residential and small businesses' data collected from smart meters.

We focus on the data containing individual electrical load curves of residential Irish customers and their corresponding survey data. The load curves are initially at half hourly period, we aggregate them by day and use the first 365 days of data.

Recalling our earlier notations, the length of each individual curve is $n = 365$ and the number of individuals considered here is $m = 3159$.

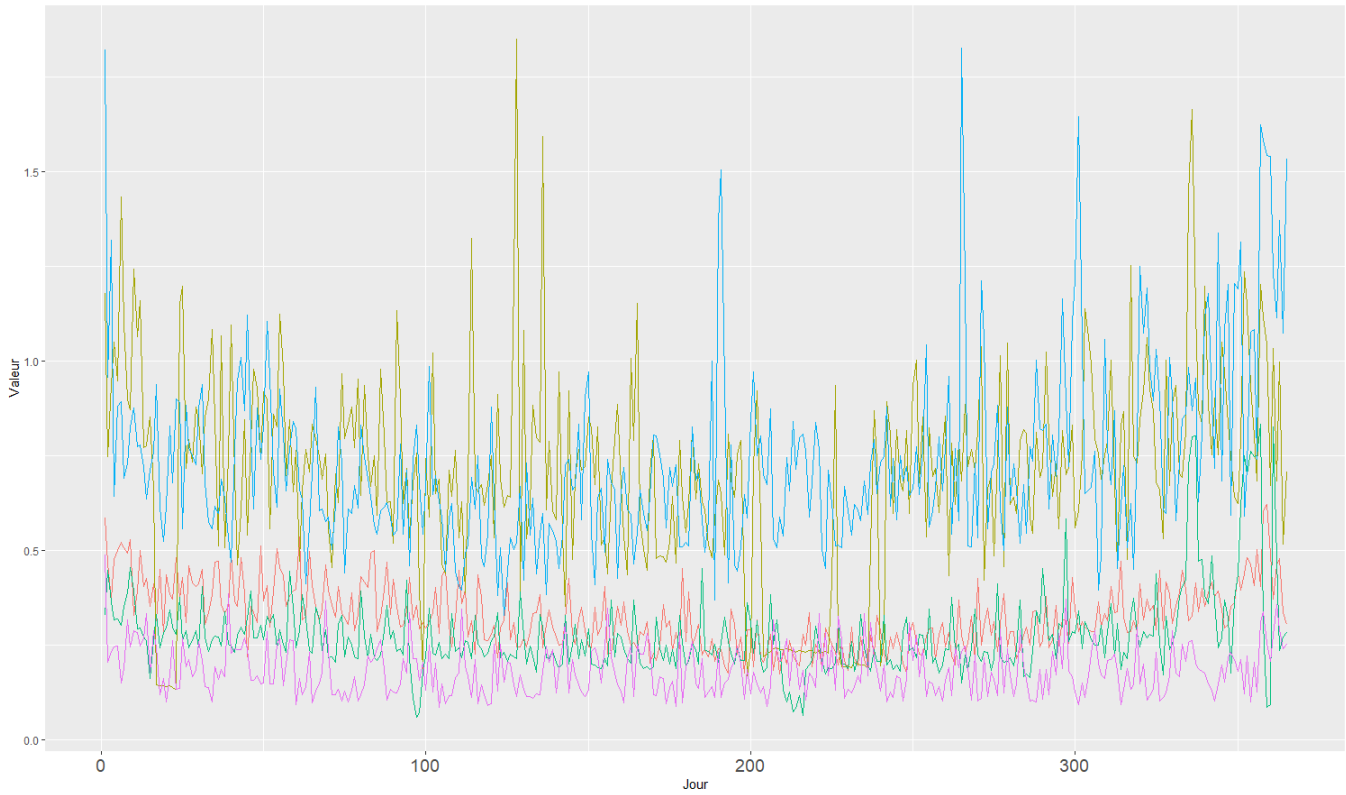


Figure 6.3 – Example of 5 individual load curves randomly selected from the CER dataset (Commission for Energy Regulation (2012)).

In Figure 6.3, we show a sample of 5 customer's individual load curves taken randomly from the CER dataset. The curves are each highly noisy and show very diverse patterns. The yellow curve for instance shows a few consumption gaps, where the value drops drastically (probably due to a period of vacation or absence). Consumption behavior are very heterogeneous amongst the customers, thus complicating any clustering effort in high dimension.

The example of curves given in Figure 6.3, while illustrating the heterogeneity of the customers based on their consumption, also show some similarities. Here, we can see that the yellow and blue curve seem closer on terms of level of consumption than the others. This gives us an idea that some customers are similar to others and justifies the use of clustering methods to create homogeneous groups.

To train and evaluate the models, the database is split accordingly:

- training subset: 80% of the database

- validation subset : 10% of the database
- testing subset : 10% of the database

The number of individuals in the testing subset is: $m_{test} = 308$.

6.6.2 Evaluation error for dimensionality reduction

Considering the load curve of any given customer i, $X^{(i)} = (X_t^{(i)})_{1 \leq t \leq n}$ and its estimation $\hat{X}^{(i)} = (\hat{X}_t^{(i)})_{1 \leq t \leq n}$, we evaluate the error between $X^{(i)}$ and $\hat{X}^{(i)}$ using the Mean Absolute Error (MAE).

We define the MAE, named $\mathcal{E}_{MAE}$ as such:

$$\mathcal{E}_{MAE}(X^{(i)}, \hat{X}^{(i)}) \quad = \quad \frac{1}{n} \sum_{t=1}^{n} |X_t^{(i)} - \hat{X}_t^{(i)}|, \ 1 \leq i \leq m. \tag{6.23}$$

Subsequently, we use $\mathcal{E}_{MAE}$ to evaluate the error obtained between $X^{(i)}$ and $\hat{X}^{(i)}$ its estimation obtained with the autoencoders.

We compare errors obtained on autoencoders made either of Dense layers (in brown on Figure 6.4) or 1D-convolutional layers (in red on Figure 6.4). Boxplots displayed side by side of both methods on Figure 6.4 for different dimension of the bottleneck of the autoencoder. The number of hidden units in the bottleneck (encoding dimension d) has an important influence on the quality of the reconstruction of the autoencoder.

We show results obtained with both types of layers when $d = 20, 2, 32, 8$. When $d = 8, 20, 32$, 1D-convolutional autoencoders outperform their Dense counterpart. However, when the encoding dimension is small ($d = 2$), both autoencoder show similar performances, and the highest median reconstruction errors are obtained in this case. Overall, the best results are obtained with the 1D-convolutional autoencoder, when the encoding dimension is $d = 32$.

Hereinafter, we use the latent space extracted from that autoencoder for the clustering of the customers.

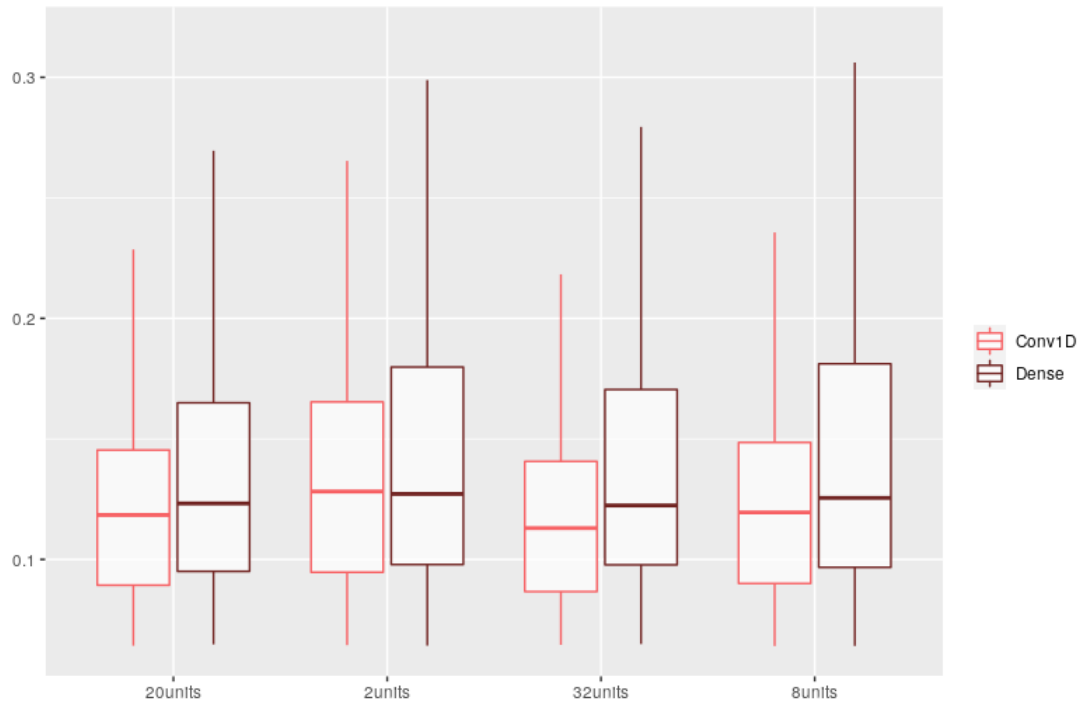


Figure 6.4 – Boxplots of $\mathcal{E}_{MAE}(X^{(i)}, \hat{X}^{(i)})$ obtained with autoencoders with 1D-convolutional layers [Red] and autoencoders with Dense layers when the encoding dimension is $d = 20, 2, 32, 8$

6.6.3 Selection of the number of clusters and vizualisation

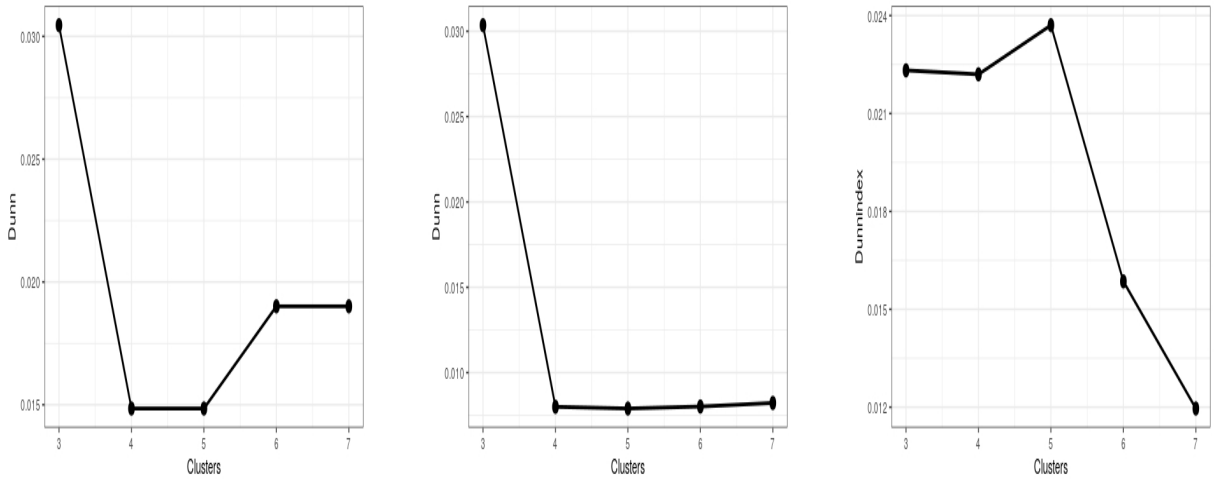
Selection of the number of clusters

Looking at the maximal Dunn index, for the three clustering methods, in Figure 6.5, we can see that the optimal number of cluster for HAC and K-means is 3, whereas it is 5 for the K-medoids methods.

As for the Davies-Bouldin index, the minimal value for the three methods are attained with 3 clusters, pointing towards 3 as the optimal number of clusters.

The Silhouette is maximal for all three methods, when the number of clusters is equal to 3.

Overall, the analysis of the three indexes points towards 3 as the preferable number of clusters. Hereinafter, we consider the three methods of clusterinf HAC, K-means and K-medoids, when 3 is the number of clusters.



(a) HAC (b) K-Means (c) K-Medoids

Figure 6.5 – The Dunn index is displayed for the three clustering methods, when the number of cluster varies from 3 to 7

Visualization of the clusters when $C = 3$

Originally, we perform the clustering algorithms on the latent space of the autoencoder of dimension 32. Visualizing data in high dimension is challenging, thus we do not display the 32 dimensions of the latent space simultaneously.

We apply the t-SNE algorithm (see Van der Maaten and Hinton (2008)) to reduce the dimension of the latent space for visualization purposes. The t-SNE algorithm is a nonlinear dimensionality reduction method that maps high dimensional data into a lower dimensional space.

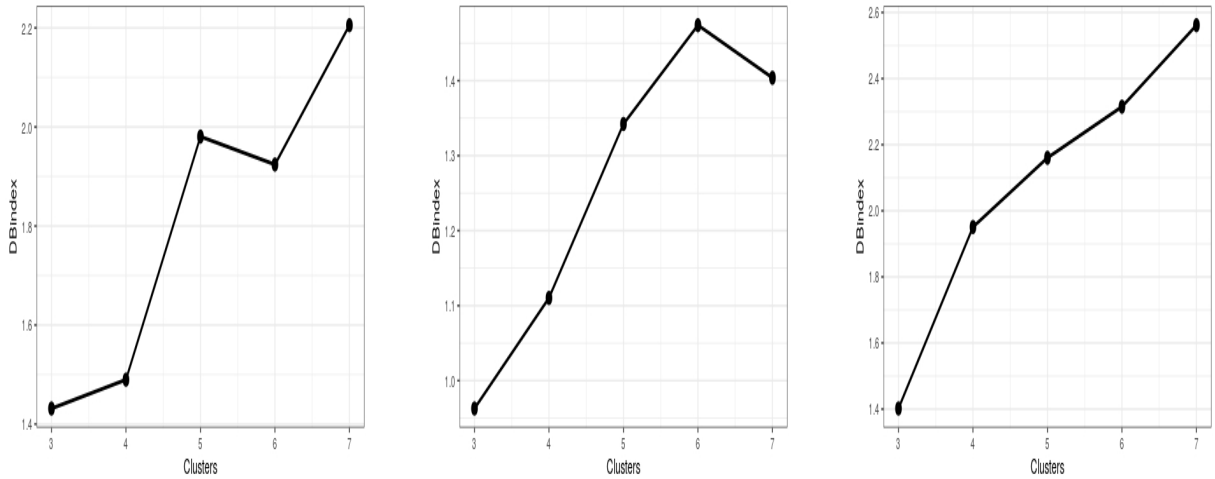
Recalling that, we have, for each observation $1 \leq i \leq m$ the latent space denoted I_i. The t-SNE method is the following:

1. First the algorithm calculates the similarity between two observations I_i and I_j, $i \neq j$ in high dimension. Following the notations from Van der Maaten and Hinton (2008), let $p_{j|i}$:

$$p_{j|i} = \frac{\exp(||I_i - I_j||^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(||I_i - I_k||^2/2\sigma_i^2)},$$

be the similarity between I_i and I_j. Let p_{ij}:

$$p_{ij} = \frac{p_{i|j} + p_{j|i}}{2m},$$



(a) HAC (b) K-Means (c) K-Medoids

Figure 6.6 – The Davies-Bouldin index is displayed for the three clustering methods, when the number of cluster varies from 3 to 7

where $p_{ij} = p_{ij}$, $p_{ij} = 0$ and $\sum_{i,j} p_{ij} = 1$.

2. Likewise, we calculate the similarity between a map of the two counterparts of I_i and I_j, denoted y_i and y_j in the lower dimensional space.

$$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\sum_k \sum_{k \neq l} (1 + ||y_k - y_l||^2)^{-1}},$$

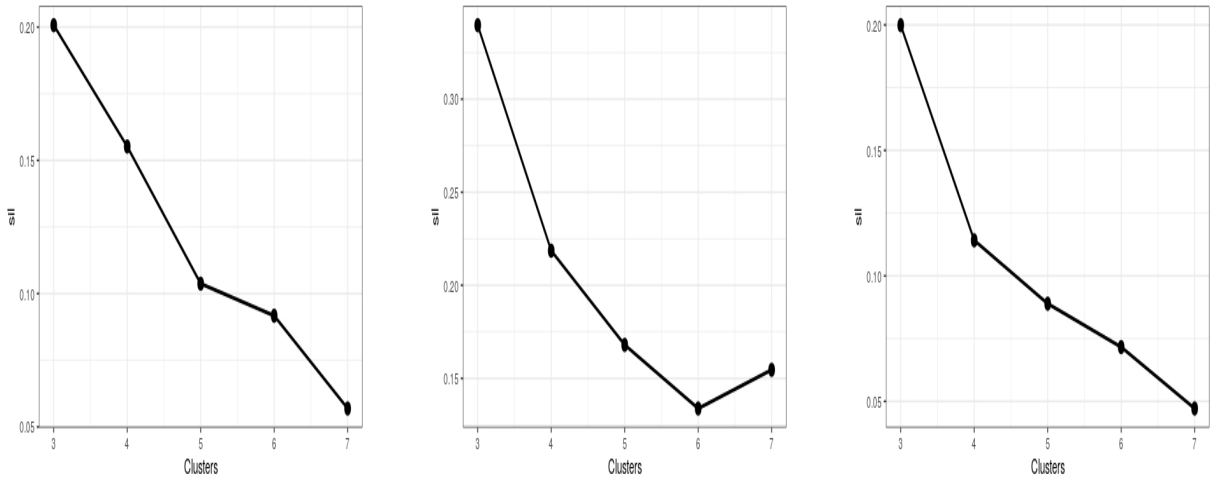
3. The correct mapping of y_i, $1 \leq i \leq m$ in the lower dimensional space is obtained by minimizing the Kullback-Leibler divergence between the similarities in high and low dimension. It is given by:

$$KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}.$$

Here, we use t-SNE to map the latent space of dimension 32 into a reduced space of dimension 2 for visualization purposes.

In Figure 6.10, we display the three clusters obtained from the K-Medoids algorithm applied to the initial latent space, on the t-SNE axis.

In Figure 6.9, we display graphical results of the clustering obtained with the K-Means algorithm, when the number of clusters is $C = 3$. On top of Figure 6.9, we can see that the clusters' size are highly unbalanced. The third cluster (in Blue) is much larger in



(a) HAC (b) K-Means (c) K-Medoids

Figure 6.7 – The silhouette index is displayed for the three clustering methods, when the number of cluster varies from 3 to 7

size than the two others. The daily mean curves (bottom of Figure 6.11) are obtained by aggregating the individual curves of each customer in the clusters. The mean curves of the first and third cluster (in blue and red) are rather close, showing the difficulty of the algorithm to distinguish the individuals.

For the Bayesian weakly informative approach described in Chapter 3, we wish to use the posterior information of the hyperparameters (the global effects of the subpopulation) to transpose it to new individuals. When using weakly informative priors, the model need enough data to estimate correctly the posterior distribution of the parameters, so that it can be usable afterwards. Thus, if the cluster or subgroup size is too small, the model fails to estimate the posterior distribution of the hyperparameters. For instance, if we wish to estimate the electrical consumption from a regression model containing a heating part, and the subgroup lacks observation with electric heating, this complicates the estimation of the posterior.

K-Medoids seems more robust to atypical individuals compared to HAC and K-Means, and gives more balanced classes. HAC does not scale well for larger dataset, making it difficult to adapt to real customers database. The mean load curve by clusters obtained for the HAC and K-Means are difficult to separate, whereas with the K-Medoids methods, each curve is distinct.

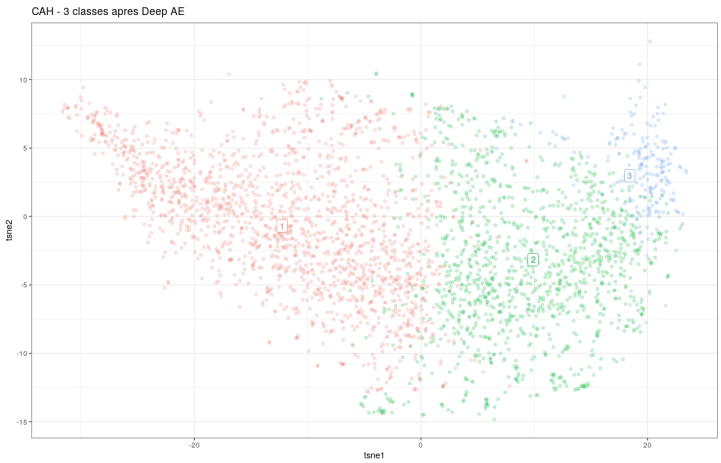


Figure 6.8 – [Top] The individuals, colored by clusters obtained with HAC, are displayed on the two axis from the t-SNE method.

6.6.4 Quality of the affectation to the clusters

Global affectation to the label

To evaluate the quality of a multiclass classification on a testing subset, one may consider looking at scores such as the accuracy, the precision and recall or the kappa coefficient (Cohen (1960),Landis and Koch (1977)).

Given a binary confusion matrix with two classes:

	class 1	class 2
class 1	True positive (TP)	False Negative (FN)
class 2	False positive (FP)	True Negative (TN)

In the case of binary classification, the precision is defined by:

$$Precision = \frac{TP}{TP + FP},$$

and the recall is as follows:

$$Precision = \frac{TP}{TP + FN}.$$

In the context of multiclass classification, the precision and recall are calculated for each class.

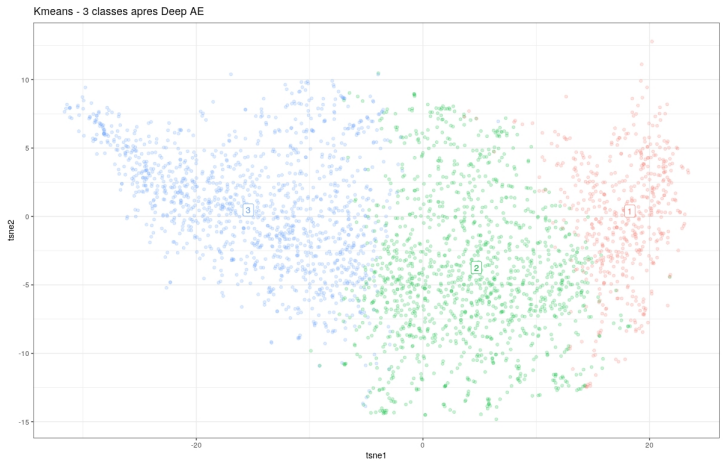


Figure 6.9 – [Top] The individuals, colored by clusters obtained with K-Means, are displayed on the two axis from the t-SNE method.

The accuracy score is defined as such:

$$Accuracy \quad = \quad \frac{TP + TN}{TP + TN + FP + FN}$$

Given a confusion matrix, the accuracy is retrieved by summing the diagonal elements and dividing it over the number of individuals to classify. It gives an overall measure of the prediction performance of the model on new observations. The accuracy score does not however indicate if the model correctly predicts each class individually. To have more information on this aspect, it can be interesting to consider for each class the precision and recall scores.

A review of metrics in the case of multiclass classification is given in Grandini et al. (2020).

For the purpose of this application, we set the length of historical data of a given customer as $p = 28$. We split the load curves of each individual by month, thus having a matrix of dimension $(L \times m_{test}) \times p$. The number of customers in our testing subset is $m_{test} = 308$, the matrix is of dimension 3696×28. Each individual is spread over $L = 12$ rows of the matrix and for each row, a label is predicted by the model as in Equation (6.22). Thus, each individual customer has 12 possible labels. First, We evaluate the quality of the prediction on each row, as if each row was a potential customer. The prediction rendered by the neural network is summarized in the confusion matrix in Table 6.1. The accuracy of the model in this case is 84.71% which is quite high. Yet considering the accuracy score

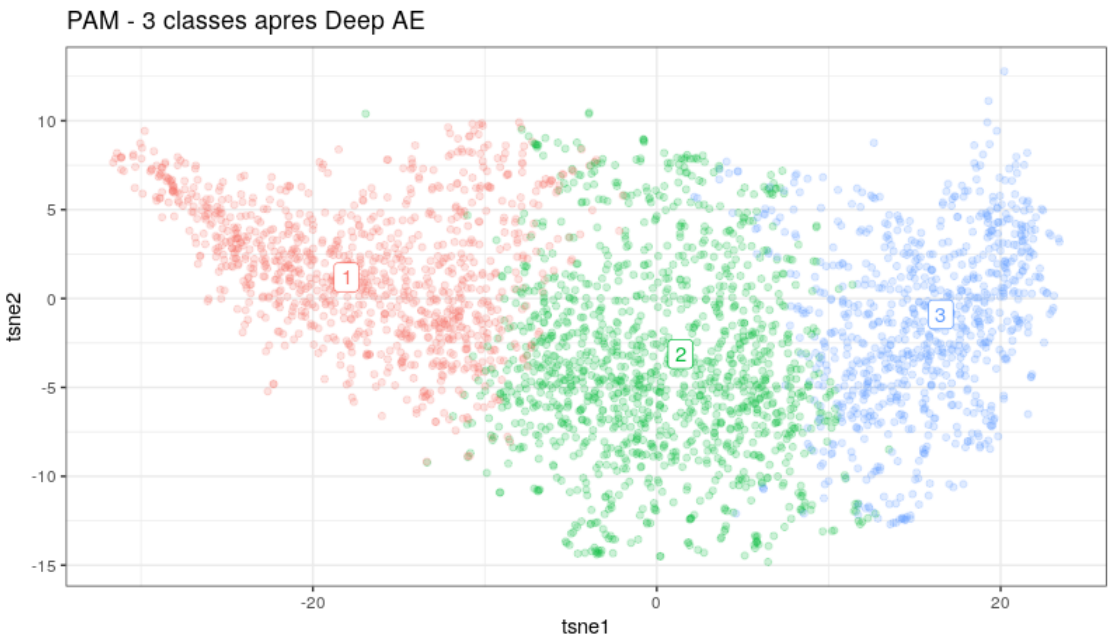


Figure 6.10 – The individuals, colored by clusters obtained with the K-Medoids algorithm, are displayed on the two axis from the t-SNE method. Cluster number 1, 2 and 3 are displayed in [Red], [Green] and [Blue] respectively.

alone is not enough to evaluate the quality of the model on the testing data.

Table 6.1 – Confusion matrix and obtained for the individuals split across $L \times m_{test}$ rows (3696 observations)

		Predicted label		
		1	2	3
Actual label	1	1140	83	1
	2	202	1193	93
	3	7	179	798
Accuracy (%)		84.71		

Table 6.2 – Precision and recall scores obtained for the individuals split across $L \times m_{test}$ rows (3696 observations)

	Class			Overall score
	1	2	3	
Precision	93.14	80.17	81.60	84.97
Recall	84.51	81.99	89.46	85.32

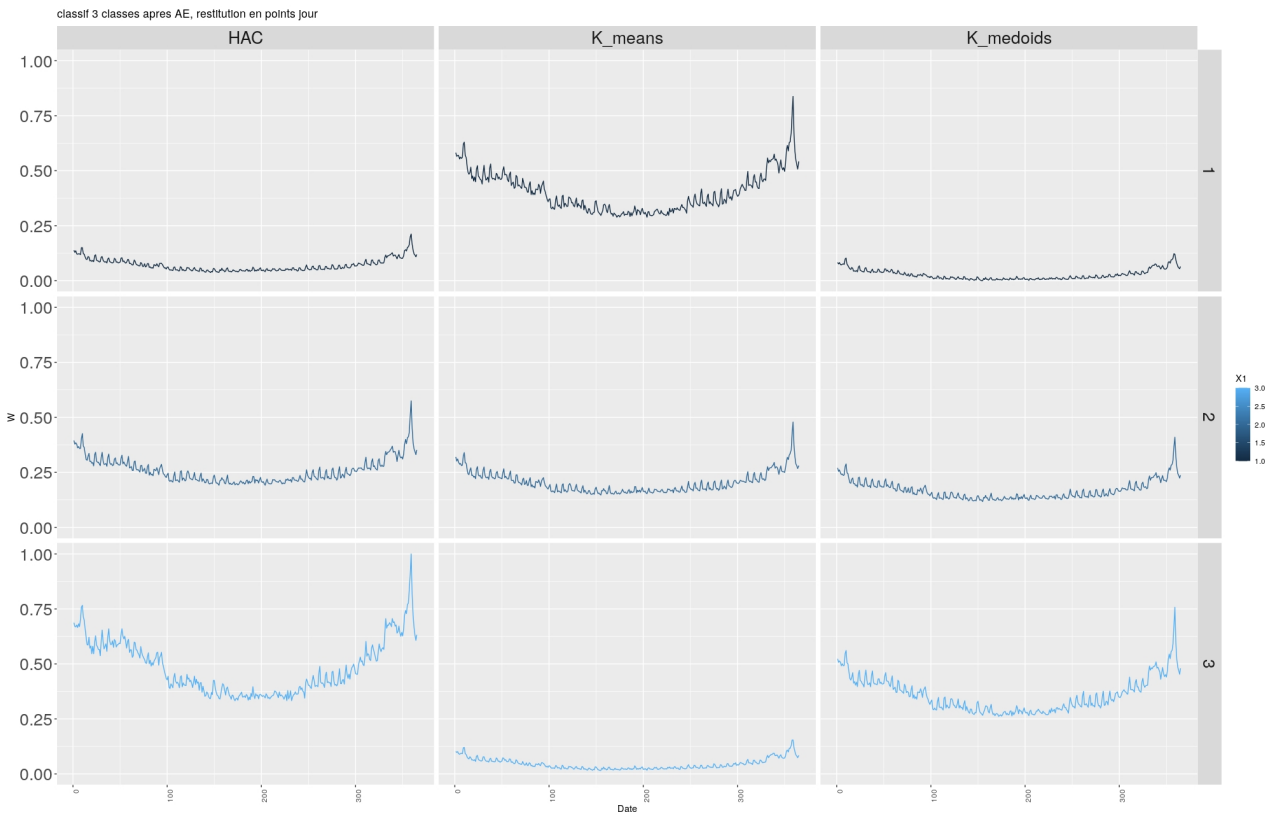


Figure 6.11 – [Top] This graphic shows the daily mean curve, colored by cluster. Cluster number 1, 2 and 3 are displayed in respectively. For visualization and comparison purposes, the mean curves are scaled here.

The precision and recall results for each class are displayed in Table 6.2. The precision of class 1 is the highest, as the model is correct 93.14% of the time when predicting that an observation belongs to this class. The overall precision and recall are both above 80% which is quite satisfactory.

One way to predict for the individual its label is to take the label that is the most frequent over the 12. Then, we have:

$$\hat{Y} \quad = \quad \underset{i=1,\ldots,C}{\operatorname{argmax}}(\mathbb{P}(\hat{y}_l = i)), \;\; 1 \leq l \leq 12 \tag{6.24}$$

The confusion matrix and accuracy are displayed in Table 6.3. We see that accuracy reaches 89.94%. Looking at the precision and recall summarized in Table 6.4, we see that the scores have increased for all the clusters.

Another possibility involves averaging the softmax scores obtained for each individual,

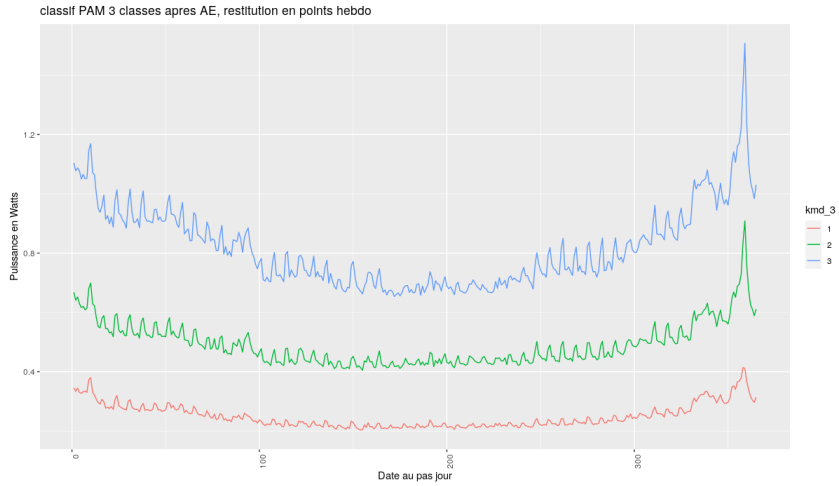


Figure 6.12 – This graphic shows the daily mean curve, colored by the clusters obtained with the K-Medoids algorithm. Cluster number 1, 2 and 3 are displayed in [Red], [Green] and [Blue] respectively.

Table 6.3 – Confusion matrix and obtained for the individuals when predicting the most frequent label over each L

		Predicted label		
		1	2	3
Actual label	1	100	2	0
	2	15	106	3
	3	0	11	71
Accuracy (%)		89.94		

Table 6.4 – Precision and recall scores obtained for the individuals split across $L \times m_{test}$ rows ($ntrain$ observations)

	Class			Overall score
	1	2	3	
Precision	98.04	85.48	86.59	90.04
Recall	86.96	89.08	95.95	90.66

by class, over the number of months available. Precisely, this is done as such:

$$\hat{Y} = \underset{i=1,\ldots,C}{\text{argmax}}\left(\frac{\sum_{k=1}^{12}(\text{softmax}(x)_i)_k}{12}\right), \tag{6.25}$$

where $\text{softmax}(x)_i)_k$ is the value obtained at the end of the neural network, for the label i and corresponding to the month k of the data of the customer.

We can generalize this method, given the historical data available for the customers. Let K be the number of month for which we have 28 values. Then, the predicted class of the customer is:

$$\hat{Y} = \underset{i=1,\ldots,C}{\text{argmax}}\left(\frac{\sum_{k=1}^{K}(\text{softmax}(x)_i)_k}{K}\right) \tag{6.26}$$

The confusion matrix and accuracy are displayed in Table 6.5. We see that accuracy reaches 89.94%, the same as in Table 6.3. There is a gain of one individual for cluster 1 and two for cluster 2. Looking at the precision and recall summarized in Table 6.6, we see that the scores have increased for all the clusters. The precision of cluster 1 reaches 99.02%. The lowest precision and highest recall are obtained for cluster 3.

Table 6.5 – Confusion matrix and obtained for the individuals when averaging the score, and predicting the label associated to the highest averaged score

		Predicted label		
		1	2	3
Actual label	1	101	1	0
	2	14	108	2
	3	0	11	71
Accuracy (%)		89.94		

Table 6.6 – Precision and recall scores obtained for the individuals split across $L \times m_{test}$ rows ($ntrain$ observations)

	Class			Overall score
	1	2	3	
Precision	99.02	87.10	86.59	90.90
Recall	87.83	90	97.26	91.45

The results obtained are overall quite high. However, when the historical data available is lower, there is a risk of misclassifiying individuals. This poses problems as after classification, a Bayesian forecasting model is applied to the individuals. If the individual is affected to the wrong class, the information transferred to the individual model corresponds to the one obtained from training on the wrong class. In the next section, we show an alternative to alleviate this issue for forecasting.

6.7 Weighting predictions using the softmax scores

In Chapter 5, we show an application of Bayesian transfer learning to the case of end-of-month forecasting. The results are shown for a fixed cluster. In this section, we present a method to obtain forecasts using the softmax scores of the clusters. Instead of affecting an individual to a specific cluster with the method in Section 6.6.4, we focus on using the scores to obtain a mixture of predictions. This approach allows to avoid the issue of missclassified individuals.

Recalling our previous notations, $\mathcal{D}^*$ denotes the observation a new individual with a short dataset. Let p_i, $i = 1, \ldots, C$ denote the probability that an individual belongs to class i. Let $\nu_i(\theta)$ denote the prior distribution of the parameter θ, of a Bayesian model, for cluster i. Let $\pi(\theta)$ denote the mixture prior:

$$\pi(\theta) = \sum_{i=1}^{C} \nu_i(\theta) p_i \tag{6.27}$$

The posterior distribution is then:

$$\pi(\theta|\mathcal{D}^*) \propto \sum_{i=1}^{C} p_i \nu_i(\theta) L_\theta(\mathcal{D}^*) \tag{6.28}$$

$$\pi(\theta|\mathcal{D}^*) \propto \sum_{i=1}^{C} p_i m_i \pi_i(\theta|\mathcal{D}^*), \tag{6.29}$$

$L_\theta(\mathcal{D}^*)$ denotes the likelihood of the model. $\pi_i(\theta|\mathcal{D}^*)$ are the posterior distributions obtained from training the model on the $\mathcal{D}^*$ with the information transferred from the longer dataset in cluster i.

$m_i(\mathcal{D}^*)$ are the marginal likelihoods relating to cluster i: $m_i(\mathcal{D}^*) = \int_\Theta L_\theta(\mathcal{D}^*)\pi_i(\theta)d\theta$.

Therefore the posterior $\pi(\theta|\mathcal{D}^*)$ is a mixture of the posterior distributions $\pi_i(\theta|\mathcal{D}^*)$, with weights $\tilde{p}_i$:

$$\tilde{p}_i = \frac{p_i m_i(\mathcal{D}^*)}{\sum_{j=1}^{C} p_j m_j(\mathcal{D}^*)}, \; i = 1, \ldots, C. \tag{6.30}$$

Remark 23 *The weights $(\tilde{p}_i)_{i=1,\ldots,C}$ can be interpreted as the updated probabilities that the individual belongs to cluster i, $i = 1, \ldots, C$.*

The prediction of a new data point X^*, for a fixed class i, is obtained using the posterior predictive distribution:

$$p_i(X^*|\mathcal{D}^*) = \int_\Theta f(X^*|\theta, \mathcal{D}^*)\pi_i(\theta|\mathcal{D}^*)\, d\theta dk. \tag{6.31}$$

The mixture prediction of the new data point X^* is:

$$\begin{aligned} p(X^*|\mathcal{D}^*) &= \int_\Theta f(X^*|\theta, \mathcal{D}^*)\pi(\theta|\mathcal{D}^*)\, d\theta \\ &= \int_\Theta f(X^*|\theta, \mathcal{D}^*) \sum_{i=1}^{C} \tilde{p}_i \pi_i(\theta|\mathcal{D}^*)\, d\theta \\ &= \sum_{i=1}^{C} \tilde{p}_i p_i(X^*|\mathcal{D}^*). \end{aligned} \tag{6.32}$$

We apply this methodology to the case of end-of-month forecasting.

The short historical data available for an individual is:

$$\mathcal{D}^* = \{X_t^*, V_t^*, t_1 \leq t \leq t_{p-1}\}$$

. X_t^* denotes the daily consumption of the individual at time t, and V_t^* the variables included to the models $(M1)$ and $(M2)$ as detailed in 5.3. t_1 is the day at which the first data point is available (we may have $t_1 > t_{first}$, when t_{first} is the first day of the month considered).

We sample $J = 1000$ realizations from $p(X^*|\mathcal{D}^*)$ in (6.32) to forecast $j = 1, \ldots, J$ values of X_t^* for $t_p < t \leq t_{last}$.

Let $\hat{X}_t^{*(j)}$ denote the j-eth forecast of X_t^*, for $t_p < t \leq t_{last}$. From the Equation (5.2), we deduce the J realizations of the mixture end-of-month forecast denoted $\hat{C}^* = \hat{C}^{*(j)}$. We have:

$$\hat{C}^{*(j)} = \sum_{t=t_{first}}^{t_p} X_t^* + \sum_{t=t_p+1}^{t_{last}} \hat{X}_t^{*(j)} \tag{6.33}$$

As in Section 5.4.4, we evaluate the errors obtained, for the testing subset used in Chapter 5, with the following length of historical data:

(a) first week of the month: we have 37 days of historical available, the horizon of forecasting is 22 days.

(b) second week of the month: we have 45 days the horizon of forecasting is 14 days.

(c) third week of the month: we have 52 days, the horizon of forecasting is 7 days.

We look at the same indicators as in Section 5.4.4:

— MAPE between the real value of the end-of-month consumption and the MAP estimate of C^*.

— Length of prediction intervals.

— Coverage probability of the intervals i.e. the probability that the real value of C^* falls into the prediction intervals.

Figure 6.13 displays the boxplots of errors for the three cases. Overall the best performances, for all the cases, are obtained with the mixture of predictions from the $M2$ model with transfer (the model with the auto-regressive part). The mixture predictions from the $M1$ model with transfer has the second best performances, except when the length of historical data is larger (see Figure 6.13c). In case (c), the mixture predictions from the $M2$ model with and without transfer give out the lowest error. However, the error with the $M2$ model without transfer is more dispersed.

When we compare the results of the mixture predictions with the ones obtained for a fixed cluster in Chapter 5, we see an improvement of all the performances of the four models. Looking at Figure 5.1, the median error of $(M2)$ with transfer is around 0.12, for the fixed cluster situation, whereas, the median error with the mixture prediction is around 0.8 (on Figure 6.13a). When the forecasting horizon is smaller, we see that the median error for $(M2)$ with transfer is slightly above 0.03 for the fixed cluster and slightly below 0.02 for the mixture prediction. We see that, for the boxplots of error, when the forecasting horizon is larger, the better the performances are for the mixture predictions. When the forecasting horizon is smaller, there is still an improvement, but the gap between the two situations is smaller.

The length of prediction intervals and coverage probabilities for all the models, and all the situations are displayed in Figure 6.14, for the mixture predictions. When compared with the fixed cluster situation in Figure 5.4, we see that for all the forecasting horizons, the coverage probabilities are higher in the mixture situation. The observations are similar, because the highest coverage probabilites are given by the mixture of predictions from the model $(M2)$ without transfer. This is to be put in perspective with the length of prediction intervals, which are also much higher compared to the other models. The coverage probabilities for the model $(M2)$ with transfer are much higher when the forecasting horizon is large (see Figure 6.14a, with the mixture prediction case). The median length

of intervals in this case is around 0.24, and is comparable to the one for the fixed situation in Figure 5.4a. When the forecasting horizon is smaller (see Figure 6.14c), the coverage probabilities of $(M2)$ with and without transfer are close, but the length of prediction intervals is smaller with transfer. Compared with the fixed cluster situation (see Figure 5.4c), the coverage probabilities of $(M2)$ with transfer are slightly higher for the mixture situation. The length of intervals are also a little higher, the median is slightly above 0.06 (compared to 0.07 for the fixed cluster situation) but the values are more dispersed.

Overall, the $(M2)$ model with transfer gives out the best results for the mixture predictions, when looking at all the indicators above. We see an improvement of the results compared to the fixed cluster situation presented in Chapter 5. Using mixture predictions seems to be a good idea, especially when there is not a large amount of historical data available. The more data we have, the closer the results will be for both situations.

6.8 Conclusion and perspectives

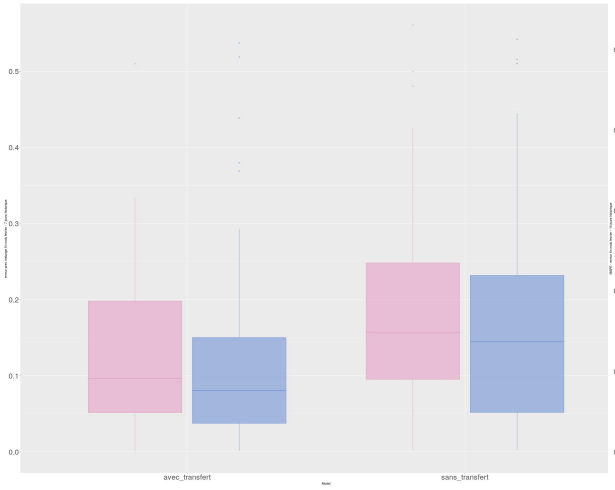
Our approach of clustering is a two step approach : first we perform dimensionality reduction on the curves with a deep autoencoder, second we use a clustering method on the latent space resulting from the autoencoder. Future research may include applying directly a neural network combining dimensionality reduction and clustering directly as proposed notably by Madiraju et al. (2018) and Guo et al. (2017).

The clustering approach retained is the K-medoids, as it is robust to outliers and atypical individuals comapred to the HAC and K-means.

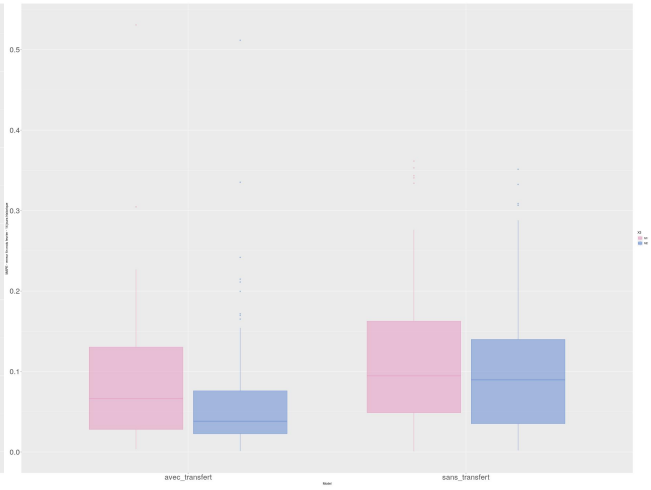
We also develop a neural network, designed to affect a new individual, with short historical data, to the classes obtained from the clustering. In Chapter 5, we have presented an application of the Bayesian transfer learning methodology of Chapter 5, to the case of end-of-month consumption forecasting. The results were shown for a fixed cluster, where the individuals are affected to one cluster alone with the majority rule of Section 6.6.4. In Section 6.7, we present a method to obtain mixture predictions from the softmax scores outputs of the affectation neural network. We compare the results obtained with the fixed cluster situation in Chapter 5. Overall, the mixture predictions of the model $(M2)$ with transfer give out the lowest error. We see an improvement with the mixture predictions, for all the forecasting horizons considered compared to the fixed cluster case. The improvement of performances is larger especially when the forecasting horizon is short. Using mixture predictions, instead of hard affectation to the classes, is relevant in

situations where the amount of data available for a new customer is small.

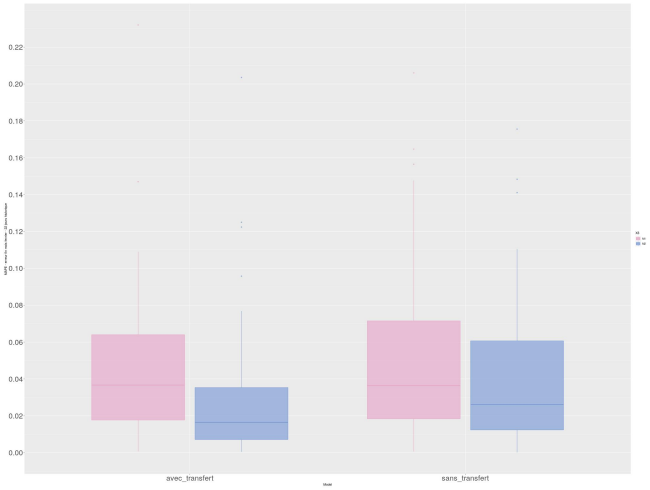
As discussed in Section 5.5, the models considered could be improved by the addition of variables (air conditioning, possession of an electrical vehicle etc.). For future research, it can be relevant to see whether the possible addition of variables, improves the forecasts for both situations: fixed cluster and mixture.



(a) Horizon: 22 days

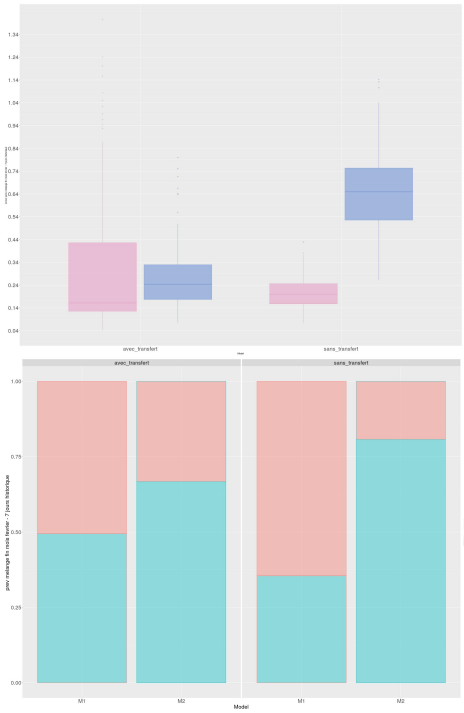


(b) Horizon: 14 days

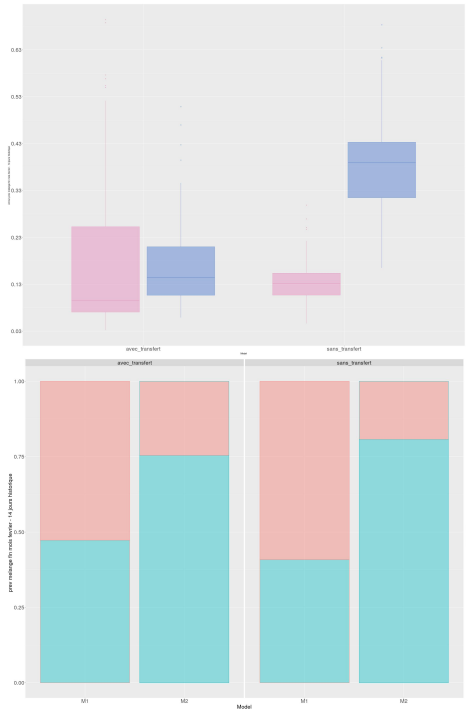


(c) Horizon: 7 days

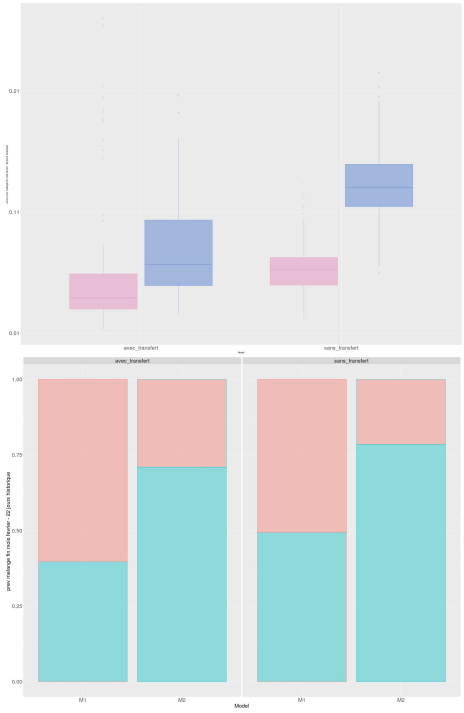
Figure 6.13 – Boxplots of the MAPE obtained with the point estimates for each individual in the testing set. The error is displayed for the $M1$ [Pink] and $M2$ models [Blue]. For each case, [Left] boxplots are the models with transfer and [Right] are the models without transfer.



(a) Horizon: 22 days



(b) Horizon: 14 days



(c) Horizon: 7 days

Figure 6.14 – For each case: [top] length of HPD prediction intervals, [bottom] coverage probability. The results are displayed for $M1$ [Pink] and $M2$ models [Blue]. For each case, [Left] boxplots are the models with transfer and [Right] are the models without transfer.

Conclusion

In this thesis, we proposed methods based on the combination of Bayesian modelling and deep learning to the estimation and prediction of time series. We dedicate this Chapter to sum up the various methodologies and results tackled in the manuscript as well as the possible perspective that may emerge from them.

The methodologies developed are closely related to transfer learning, from several aspects: through the use of fine tuning for neural networks or the Bayesian approach to transfer learning. Our methodologies focus mainly on regression, as the industrial applications on which they are developed are regression based.

The work presented here is largely motivated by the necessity for EDF to develop new supply offers and consumption services tailored to the needs of new customers, when the historical data available is short.

The methodologies developed rely on deep learning and Bayesian methods in two ways:

- Through the use of Bayesian neural networks.
- Using first deep learning models to build homogeneous subsets in the data and applying Bayesian (non deep) models to the subgroups built.

Both types of methodologies are applied to real world datasets provided by EDF for industrial applications relating the individual consumption of either non residential or residential customers.

Methodology of estimation of a multi-target regression model in high dimension

We have developed a methodology of estimating a multi-target regression model in high dimension in Chapter 2. The results obtained on real world data are presented in Chapter 3. The three strategies of modeling rely on deep learning methods and are adapted to account for two industrial constraints.

Fine tuning is used on all the models, and shows improvement on the autoencoder and when the curve is reconstructed from the latent space. However, the best results are

obtained with the Bayesian neural network **BayesRN1** without fine tuning, for the ENR strategy. Future research may involve understating the effects of fine tuning on the various strategies.

The autoencoders considered and the neural networks for the direct estimation with MNR are built with Dense layers. Thus the time dependency of the inputs is not taken into account by those neural networks. It would be interesting to see whether using layers adapted to deal with time series such as 1D-convolutional layers or LSTM layers outperform the Dense layers.

The industrial application is related to the issue of solar panel sizing for non residential customers. Thus, we focus on emphasizing the prediction of consumption during hours of sunlight, using a set of weights built from solar power production data from EDF. However, the explanatory variables used for modeling do not contain any information relating to this application. Specifically, we lack information on the industrial sites, their dimension, their geographical location, the type of electrical usage etc. Those information could enrich the models, as for instance, knowing the geographical location, we could collect data relating to radiance. If available, it would be relevant to add such variables to the models and see whether there is an improvement.

The direct estimation of the curves through the MNR strategy is conducted using regular neural networks. Adaptating the strategy to Bayesian neural networks could be relevant as we would directly be able to build prediction intervals from their posterior distribution. For instance, adapting a Bayesian convolutional neural network may be interesting, the main drawback being the potential training time that is longer with Bayesian neural networks.

Bayesian neural networks' potential are promising as the samples drawn from the posterior predictive distribution could be used for other industrial applications relating to load profiling.

We have not explored the possibility of constructing informative priors for Bayesian neural networks, from expert knowledge. Cui et al. (2021) provide a methodology to incorporate domain knowledge to the priors of Bayesian neural networks. Future research may involve adapting a Bayesian transfer learning approach as presented in Chapter 4 to the case of neural networks and comparing it with the approach from Cui et al. (2021).

Bayesian transfer learning for panel data

In Chapter 4, we present a general methodology of Bayesian transfer learning for panel data.

The methodology, adapted from Launay et al. (2015), consists in two steps:

- The weakly informative approach: a Bayesian hierarchical model is trained on panel data with long historical data.
- The informative approach: the posterior mean and covariance from the first step are transferred to the prior of the parameters of the model of a new individual with shorter historical data.

We evaluate the Bayesian transfer on three simulated situations: the polynomial regression, the autoregressive model and the hierarchical Poisson model. Globally, we see a benefit to using this transfer strategy. The performances of the transfer strategy are mitigated for the hierarchical Poisson model when the new individual differs from the original panel. While the point estimates of the parameters (see Figure 4.22) obtained for the model with transfer are better than without transfer for all scenarios, the coverage probabilities are smaller for the scenarios where the new individual differs. This is possibly due to the fact that, contrary to the other simulation situations, we cannot transfer the covariance matrix, the correlation between parameters is not accounted for in the priors of the informative approach. Future research may involve extending the Bayesian transfer learning for the univariate case.

The Bayesian transfer learning methodology is applied to real world data in Chapter 5. The industrial application is the forecasting of the individual consumption at the end of the month of residential customers. The results are presented for a fixed subset of the data. Future applications may involve enriching the models with more information on the customers behavior. We restrict ourselves to adding the lagged consumption of the previous day in the model ($M2$), however, if available it would be relevant to go further in the past. For instance, adding information relating to calendar days (weekend versus weekdays), the average consumption of similar day types, or retaining the lagged consumption based on the mutual information index as in Bessani et al. (2020), are interesting leads to explore.

In Chapter 6, we present the methodology to construct the subset used in Chapter 5.

We use deep learning based methods for dimensionality reduction on daily individual load curves and perform clustering on the latent space to build homogeneous groups of

customers. We also develop a double input deep neural network for multi class classification of new individuals to the clusters we built.

From the affectation neural networks, we are able to construct probabilities that the individual belongs to either of the classes. We then obtain mixture predictions, using the models described in Chapter 6, by weighing the forecasts with the probabilities scores. The results are presented on the same testing subset as in Chapter 6, for the same forecasting horizon. There is a clear benefit to mixing the forecast instead of the fixed cluster situation as there are improvements for forecasting errors as well as coverage probabilities. Again, the addition of variables to the models and the mixture prediction resulting would be interesting to consider.

For the clustering of load curves, future research could involve estimating the regularity of the individual time series, following a methodology proposed by Echelard et al. (2015). The individuals would be clustered not on their curves, but on those estimators of regularity.

Finally, it would be interesting to extend the methodology developed for Bayesian transfer learning to multiclass classification problems, for instance for targeting adapter supply offers and innovative services to new customers with shorter historical data.

Bibliography

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `http://tensorflow.org/`. Software available from tensorflow.org.

S. Aghabozorgi, A. S. Shirkhorshidi, and T. Y. Wah. Time-series clustering–a decade review. Information Systems, 53:16–38, 2015.

E. Aldrich. wavelets: Functions for Computing Wavelet Filters, Wavelet Transforms and Multiresolution Analyses, 2020. URL `https://CRAN.R-project.org/package=wavelets`. R package version 0.3-0.2.

E. Aljalbout, V. Golkov, Y. Siddiqui, M. Strobel, and D. Cremers. Clustering with deep learning: Taxonomy and new methods. arXiv preprint arXiv:1801.07648, 2018.

K. Amarasinghe, D. L. Marino, and M. Manic. Deep neural networks for energy load forecasting. In 2017 IEEE 26th International Symposium on Industrial Electronics (ISIE), pages 1483–1488, 2017. doi: 10.1109/ISIE.2017.8001465.

A. Antoniadis, X. Brossat, J. Cugliari, and J.-M. Poggi. Clustering functional data using wavelets. International Journal of Wavelets, Multiresolution and Information Processing, 11(01):1350003, 2013.

H. Aras and N. Aras. Forecasting residential natural gas demand. Energy Sources, 26(5): 463–472, 2004.

D. M. S. Arsa, G. Jati, A. J. Mantau, and I. Wasito. Dimensionality reduction using deep belief network in big data case study: Hyperspectral image classification. In 2016

International Workshop on Big Data and Information Security (IWBIS), pages 71–76, 2016. doi: 10.1109/IWBIS.2016.7872892.

B. Auder, J. Cugliari, Y. Goude, and J.-M. Poggi. Scalable clustering of individual electrical curves for profiling and bottom-up forecasting. Energies, 11(7):1893, 2018.

B. H. Baltagi et al. Econometric analysis of panel data, volume 4. Springer, 2008.

S. Barbon Junior, S. M. Mastelini, A. P. A. Barbon, D. F. Barbin, R. Calvini, J. F. Lopes, and A. Ulrici. Multi-target prediction of wheat flour quality parameters with near infrared spectroscopy. Information Processing in Agriculture, 7(2):342–354, 2020. ISSN 2214-3173. doi: https://doi.org/10.1016/j.inpa.2019.07.001. URL https://www.sciencedirect.com/science/article/pii/S2214317318304554.

M. Bessani, J. A. Massignan, T. M. Santos, J. B. London, and C. D. Maciel. Multiple households very short-term load forecasting using bayesian networks. Electric Power Systems Research, 189:106733, 2020. ISSN 0378-7796. doi: https://doi.org/10.1016/j.epsr.2020.106733. URL https://www.sciencedirect.com/science/article/pii/S0378779620305368.

D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. Journal of the American statistical Association, 112(518):859–877, 2017.

C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural network. In F. Bach and D. Blei, editors, Proceedings of the 32nd International Conference on Machine Learning, volume 37 of Proceedings of Machine Learning Research, pages 1613–1622, Lille, France, 07–09 Jul 2015. PMLR. URL http://proceedings.mlr.press/v37/blundell15.html.

H. Borchani, G. Varando, C. Bielza, and P. Larrañaga. A survey on multi-output regression. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 5(5): 216–233, 2015.

A. Brusaferri, M. Matteucci, P. Portolani, and A. Vitali. Bayesian deep learning based method for probabilistic forecast of day-ahead electricity prices. Applied Energy, 250:1158–1175, 2019. ISSN 0306-2619. doi: https://doi.org/10.1016/j.apenergy.2019.05.068. URL https://www.sciencedirect.com/science/article/pii/S0306261919309237.

A. J. Burnham, J. F. MacGregor, and R. Viveros. Latent variable multivariate regression modeling. Chemometrics and Intelligent Laboratory Systems, 48(2):167–180, 1999.

H. Cai, X. Jia, J. Feng, W. Li, Y.-M. Hsu, and J. Lee. Gaussian process regression for numerical wind speed prediction enhancement. Renewable Energy, 146:2112–2123, 2020.

L. Candillier. Contextualisation, Visualisation et Évaluation en Apprentissage Non Supervisé. PhD thesis, Université Charles de Gaulle de Lille 3, september 2006. URL `http://lcandillier.free.fr/publis/these.pdf`.

B. Carpenter, A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell. Stan: A probabilistic programming language. Journal of statistical software, 76(1), 2017.

K. Chen, K. Chen, Q. Wang, Z. He, J. Hu, and J. He. Short-term load forecasting with deep residual networks. IEEE Transactions on Smart Grid, 10(4):3943–3952, 2019. doi: 10.1109/TSG.2018.2844307.

H. Cho, Y. Goude, X. Brossat, and Q. Yao. Modelling and forecasting daily electricity load via curve linear regression. In A. Antoniadis, J.-M. Poggi, and X. Brossat, editors, Modeling and Stochastic Learning for Forecasting in High Dimensions, volume 217 of Lectute Notes in Statistics, pages 35–54. Springer, June 2015. ISBN 9783319187310. doi: 10.1007/978-3-319-18732-7_3. 2nd Workshop on Industry and Practices for Forecasting, WIPFOR 2013 ; Conference date: 05-06-2013 Through 07-06-2013.

K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078, 2014.

F. Chollet and J. J. Allaire. Deep Learning with R. Manning Publications Co., USA, 1st edition, 2018. ISBN 161729554X.

F. Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

J. Cohen. A coefficient of agreement for nominal scales. Educational and psychological measurement, 20(1):37–46, 1960.

Commission for Energy Regulation. Cer smart metering project - electricity customer behaviour trial, 2009-2010 [dataset]. Irish Social Science Data Archive, 1st Edition, 2012. ISSN SN: 0012-00. URL https://www.ucd.ie/issda/data/commissionforenergyregulationcer/.

P. D. Congdon. Applied Bayesian hierarchical methods. CRC Press, 2010.

J. Cugliari, Y. Goude, and J. Poggi. Disaggregated electricity forecasting using wavelet-based clustering of individual consumers. In 2016 IEEE International Energy Conference (ENERGYCON), pages 1–6, 2016. doi: 10.1109/ENERGYCON.2016.7514087.

T. Cui, A. Havulinna, P. Marttinen, and S. Kaski. Informative bayesian neural network priors for weak signals. Bayesian Analysis, 1(1):1–31, 2021.

A. Damianou and N. Lawrence. Deep gaussian processes. In Artificial Intelligence and Statistics, pages 207–215, 2013.

D. L. Davies and D. W. Bouldin. A cluster separation measure. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-1(2):224–227, 1979. doi: 10.1109/TPAMI.1979.4766909.

A. Deihimi and H. Showkati. Application of echo state networks in short-term electric load forecasting. Energy, 39(1):327–340, 2012.

J. V. Dillon, I. Langmore, D. Tran, E. Brevdo, S. Vasudevan, D. Moore, B. Patton, A. Alemi, M. Hoffman, and R. A. Saurous. Tensorflow distributions, 2017.

J. F. Driver and F. Alemi. Forecasting without historical data: Bayesian probability models utilizing expert opinions. Journal of medical systems, 19(4):359–374, 1995.

J. C. Dunn. Well-separated clusters and optimal fuzzy partitions. Journal of Cybernetics, 4(1):95–104, 1974. doi: 10.1080/01969727408546059. URL https://doi.org/10.1080/01969727408546059.

A. V. Dyrrdal, A. Lenkoski, T. L. Thorarinsdottir, and F. Stordal. Bayesian hierarchical modeling of extreme hourly precipitation in norway. Environmetrics, 26(2):89–106, 2015. doi: https://doi.org/10.1002/env.2301. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/env.2301.

A. Echelard, J. L. Véhel, and A. Philippe. Statistical estimation for a class of self-regulating processes. Scandinavian Journal of Statistics, 42(2):485–503, 2015.

European Commission. Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation), 2016.

H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller. Deep learning for time series classification: a review. Data mining and knowledge discovery, 33(4): 917–963, 2019.

A. Feller and A. Gelman. Hierarchical models for causal effects. Emerging Trends in the Social and Behavioral Sciences: An interdisciplinary, searchable, and linkable resource, pages 1–16, 2015.

C. Finn, K. Xu, and S. Levine. Probabilistic model-agnostic meta-learning. arXiv preprint arXiv:1806.02817, 2018.

J. Fitzpatrick, P. Carroll, and D. Ajwani. Creating and characterising electricity load profiles of residential buildings. In V. Lemaire, S. Malinowski, A. Bagnall, T. Guyet, R. Tavenard, and G. Ifrim, editors, Advanced Analytics and Learning on Temporal Data, pages 182–203, Cham, 2020. Springer International Publishing. ISBN 978-3-030-65742-0.

E. W. Forgy. Cluster analysis of multivariate data : efficiency versus interpretability of classifications. Biometrics, 21:768–769, 1965.

V. Fortuin, M. Hüser, F. Locatello, H. Strathmann, and G. Rätsch. Deep self-organization: Interpretable discrete representation learning on time series. In International Conference on Learning Representations, 2019. URL `https://openreview.net/forum?id=rygjcsR9Y7`.

C. Fu, T. Sayed, and L. Zheng. Multivariate bayesian hierarchical modeling of the non-stationary traffic conflict extremes for crash estimation. Analytic Methods in Accident Research, 28:100135, 2020. ISSN 2213-6657. doi: https://doi.org/10.1016/j.amar.2020.100135. URL `https://www.sciencedirect.com/science/article/pii/S2213665720300257`.

Y. Gal. Uncertainty in deep learning. PhD thesis, University of Cambridge, 2016.

Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In international conference on machine learning, pages 1050–1059, 2016.

Y. Gal, J. Hron, and A. Kendall. Concrete dropout. In Advances in neural information processing systems, pages 3581–3590, 2017.

J.-M. Galharret and A. Philippe. Priors comparison in Bayesian Mediation Framework with binary outcome. working paper or preprint, Mar. 2019. URL https://hal.archives-ouvertes.fr/hal-02070053.

P. Gallinari, Y. Lecun, S. Thiria, and F. Fogelman Soulie. Memoires associatives distribuees: Une comparaison. In Proceedings of COGNITIVA 87, Paris, La Villette, May 1987. Cesta-Afcet, 1987.

A. Gelman. Prior distributions for variance parameters in hierarchical models. Bayesian Analysis, 1(3):515–533, 2006.

H. Golmohammadi. Prediction of octanol–water partition coefficients of organic compounds by multiple linear regression, partial least squares, and artificial neural network. Journal of Computational Chemistry, 30(15):2455–2465, 2009. doi: https://doi.org/10.1002/jcc.21243. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/jcc.21243.

I. Goodfellow, Y. Bengio, and A. Courville. Deep learning. MIT press, 2016.

M. Grandini, E. Bagli, and G. Visani. Metrics for multi-class classification: an overview. arXiv preprint arXiv:2008.05756, 2020.

A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In 2013 IEEE international conference on acoustics, speech and signal processing, pages 6645–6649. IEEE, 2013.

C. Guan, P. B. Luh, L. D. Michel, Y. Wang, and P. B. Friedland. Very short-term load forecasting: wavelet neural networks with data pre-filtering. IEEE Transactions on Power Systems, 28(1):30–41, 2012.

S. Guang. Network traffic prediction based on the wavelet analysis and hopfield neural network. International Journal of Future Computer and Communication, 2(2):101, 2013.

X. Guo, X. Liu, E. Zhu, and J. Yin. Deep clustering with convolutional autoencoders. In International conference on neural information processing, pages 373–382. Springer, 2017.

J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics), 28(1):100–108, 1979. ISSN 00359254, 14679876. URL http://www.jstor.org/stable/2346830.

M. Hassani and T. Seidl. Using internal evaluation measures to validate the quality of diverse stream clustering algorithms. Vietnam Journal of Computer Science, 4(3): 171–183, 2017.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770–778, 2016.

W. He. Load forecasting via deep neural networks. Procedia Computer Science, 122:308 – 314, 2017. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2017.11.374. URL http://www.sciencedirect.com/science/article/pii/S1877050917326170. 5th International Conference on Information Technology and Quantitative Management, ITQM 2017.

H. Helmiriawan and Z. Al-Ars. Multi-target regression approach for predictive maintenance in oil refineries using deep learning. International Journal of Neural Networks and Advanced Applications, 6:18–24, 2019.

J. Hensman, N. Fusi, and N. D. Lawrence. Gaussian processes for big data. In Uncertainty in Artificial Intelligence, page 282, 2013.

J. M. Hernàndez-Lobato and R. Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In F. Bach and D. Blei, editors, Proceedings of the 32nd International Conference on Machine Learning, volume 37 of Proceedings of Machine Learning Research, pages 1861–1869, Lille, France, 07–09 Jul 2015. PMLR. URL http://proceedings.mlr.press/v37/hernandez-lobatoc15.html.

G. Hinton. Deep belief networks. Scholarpedia, 4(5):5947, Jan. 2009. doi: 10.4249/scholarpedia.5947.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. science, 313(5786):504–507, 2006.

G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. Neural computation, 18(7):1527–1554, 2006.

H. S. Hippert, C. E. Pedreira, and R. C. Souza. Neural networks for short-term load forecasting: A review and evaluation. IEEE Transactions on power systems, 16(1): 44–55, 2001.

S. Hochreiter and J. Schmidhuber. Long short-term memory. Neural computation, 9(8): 1735–1780, 1997.

T. Hong and S. Fan. Probabilistic electric load forecasting: A tutorial review. International Journal of Forecasting, 32(3):914 – 938, 2016. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2015.11.011. URL `http://www.sciencedirect.com/science/article/pii/S0169207015001508`.

J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. Proceedings of the national academy of sciences, 79(8):2554–2558, 1982.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. Neural networks, 2(5):359–366, 1989.

H. Hu, L. Wang, and S.-X. Lv. Forecasting energy consumption and wind power generation using deep echo state network. Renewable Energy, 154:598–613, 2020. ISSN 0960-1481. doi: https://doi.org/10.1016/j.renene.2020.03.042. URL `https://www.sciencedirect.com/science/article/pii/S0960148120303645`.

S. Iyengar, S. Lee, D. Irwin, P. Shenoy, and B. Weil. Watthome: A data-driven approach for energy efficiency analytics at city-scale. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 396–405, 2018.

H. Jaeger and H. Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. science, 304(5667):78–80, 2004.

L. Kaufman and P. Rousseeuw. Partitioning Around Medoids (Program PAM), chapter 2, pages 68–125. John Wiley & Sons, Ltd, 1990. ISBN 9780470316801. doi: https://doi.org/10.1002/9780470316801.ch2. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470316801.ch2.

A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? In Advances in neural information processing systems, pages 5574–5584, 2017.

A. Kendall, V. Badrinarayanan, , and R. Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. In Proceedings of the British Machine Vision Conference (BMVC), 2017.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In 2nd International Conference on Learning Representations, 2014.

D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. In Advances in neural information processing systems, pages 2575–2583, 2015.

S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman. 1d convolutional neural networks and applications: A survey. Mechanical Systems and Signal Processing, 151:107398, 2021. ISSN 0888-3270. doi: https://doi.org/10.1016/j.ymssp.2020.107398. URL https://www.sciencedirect.com/science/article/pii/S0888327020307846.

T. Kohonen. Self-Organizing Maps, volume 30 of Springer Series in Information Sciences. Springer, 1995.

S. Kong, J. Bai, J. H. Lee, D. Chen, A. Allyn, M. Stuart, M. Pinsky, K. Mills, and C. P. Gomes. Deep hurdle networks for zero-inflated multi-target regression: Application to multiple species abundance estimation. In C. Bessiere, editor, Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, pages 4375–4381. ijcai.org, 2020a. doi: 10.24963/ijcai.2020/603. URL https://doi.org/10.24963/ijcai.2020/603.

W. Kong, Z. Y. Dong, D. J. Hill, F. Luo, and Y. Xu. Short-term residential load forecasting based on resident behaviour learning. IEEE Transactions on Power Systems, 33(1): 1087–1088, 2017.

W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang. Short-term residential load forecasting based on lstm recurrent neural network. IEEE Transactions on Smart Grid, 10(1):841–851, 2019. doi: 10.1109/TSG.2017.2753802.

X. Kong, R. Lin, and H. Zou. Feature extraction of load curve based on autoencoder network. In 2020 IEEE 20th International Conference on Communication Technology (ICCT), pages 1452–1456, 2020b. doi: 10.1109/ICCT50939.2020.9295958.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, NIPS'12, page 1097–1105, Red Hook, NY, USA, 2012. Curran Associates Inc.

J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. biometrics, pages 159–174, 1977.

C. Lang, F. Steinborn, O. Steffens, and E. W. Lang. Applying a 1d-cnn network to electricity load forecasting. In O. Valenzuela, F. Rojas, L. J. Herrera, H. Pomares, and I. Rojas, editors, Theory and Applications of Time Series Analysis, pages 205–218, Cham, 2020. Springer International Publishing. ISBN 978-3-030-56219-9.

T. Launay, A. Philippe, and S. Lamarche. Construction of an informative hierarchical prior for a small sample with the help of historical data and application to electricity load forecasting. TEST: An Official Journal of the Spanish Society of Statistics and Operations Research, 24(2):361–385, June 2015. doi: 10.1007/s11749-014-0416-0. URL `https://ideas.repec.org/a/spr/testjl/v24y2015i2p361-385.html`.

P. Laurinec and M. Lucká. Comparison of representations of time series for clustering smart meter data. In Proceedings of the world congress on engineering and computer science, volume 1, 2016.

P. Laurinec and M. Lucká. Clustering-based forecasting method for individual consumers electricity load using time series representations. Open Computer Science, 8(1):38–50, 2018.

Y. LeCun. A learning scheme for asymmetric threshold networks. Proceedings of COGNITIVA, 85(537):599–604, 1985.

Y. LeCun. PhD thesis: Modeles connexionnistes de l'apprentissage. Universite P. et M. Curie (Paris 6), juin 1987.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. Neural computation, 1(4):541–551, 1989.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.

Y. Lee and A. Bateman. The competitiveness of fair trade and organic versus conventional coffee based on consumer panel data. Ecological Economics, 184:106986, 2021. ISSN 0921-8009. doi: https://doi.org/10.1016/j.ecolecon.2021.106986. URL `https://www.sciencedirect.com/science/article/pii/S0921800921000446`.

H. Li, W. Zhang, Y. Chen, Y. Guo, G.-Z. Li, and X. Zhu. A novel multi-target regression framework for time-series prediction of drug efficacy. Scientific reports, 7(1):1–9, 2017.

K. Lindberg, S. Bakker, and I. Sartori. Modelling electric and heat load profiles of non-residential buildings for use in long-term aggregate load forecasts. Utilities Policy, 58: 63 – 88, 2019. ISSN 0957-1787. doi: https://doi.org/10.1016/j.jup.2019.03.004. URL `http://www.sciencedirect.com/science/article/pii/S0957178719300128`.

F. Lionetti, M. Pastore, U. Moscardino, A. Nocentini, K. Pluess, and M. Pluess. Sensory processing sensitivity and its association with personality traits and affect: A meta-analysis. Journal of Research in Personality, 81:138–152, 2019. ISSN 0092-6566. doi: https://doi.org/10.1016/j.jrp.2019.05.013. URL `https://www.sciencedirect.com/science/article/pii/S0092656619300583`.

H. Liu, J. Cai, and Y.-S. Ong. Remarks on multi-output gaussian process regression. Knowledge-Based Systems, 144:102–121, 2018.

S. Lloyd. Least squares quantization in pcm. IEEE Transactions on Information Theory, 28(2):129–137, 1982. doi: 10.1109/TIT.1982.1056489.

Q. Ma, J. Zheng, S. Li, and G. W. Cottrell. Learning representations for time series clustering. Advances in neural information processing systems, 32:3781–3791, 2019.

D. J. MacKay. A practical bayesian framework for backpropagation networks. Neural computation, 4(3):448–472, 1992.

D. J. MacKay et al. Bayesian nonlinear modeling for the prediction competition. ASHRAE transactions, 100(2):1053–1062, 1994.

J. MacQueen et al. Some methods for classification and analysis of multivariate observations. In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, volume 1, pages 281–297. Oakland, CA, USA, 1967.

N. S. Madiraju, S. M. Sadat, D. Fisher, and H. Karimabadi. Deep temporal clustering : Fully unsupervised learning of time-domain features, 2018.

M. Maechler, P. Rousseeuw, A. Struyf, M. Hubert, and K. Hornik. cluster: Cluster Analysis Basics and Extensions, 2021. URL `https://CRAN.R-project.org/package=cluster`. R package version 2.1.2 — For new features, see the 'Changelog' file (in the package source).

S. Makridakis, E. Spiliotis, and V. Assimakopoulos. The m4 competition: 100,000 time series and 61 forecasting methods. International Journal of Forecasting, 36(1):54–74, 2020. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2019.04.014. URL `https://www.sciencedirect.com/science/article/pii/S0169207019301128`. M4 Competition.

S. Makridakis, E. Spiliotis, and V. Assimakopoulos. M5 accuracy competition: Results, findings, and conclusions. International Journal of Forecasting, 38(4):1346–1364, 2022. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2021.11.013. URL `https://www.sciencedirect.com/science/article/pii/S0169207021001874`. Special Issue: M5 competition.

Z. Marx, M. T. Rosenstein, L. P. Kaelbling, and T. G. Dietterich. Transfer learning with an ensemble of background tasks. In In NIPS Workshop on Inductive Transfer, 2005.

A. G. d. G. Matthews, J. Hron, M. Rowland, and Z. Turner, Richard E.and Ghahramani. Gaussian process behaviour in wide deep neural networks. In 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings, 2018.

W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4):115–133, 1943.

R. Meager. Understanding the average impact of microcredit expansions: A bayesian hierarchical analysis of seven randomized experiments. American Economic Journal: Applied Economics, 11(1):57–91, 2019.

G. D. Merkel, R. J. Povinelli, and R. H. Brown. Short-term load forecasting of natural gas with deep neural network regression. Energies, 11(8):2008, 2018.

E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long. A survey of clustering with deep learning: From the perspective of network architecture. IEEE Access, 6:39501–39514, 2018.

M. Minsky and S. Papert. Perceptrons: An Introduction to Computational Geometry. MIT Press, Cambridge, MA, USA, 1969.

O. Motlagh, A. Berry, and L. O'Neil. Clustering of residential electricity customers using load time series. Applied Energy, 237:11–24, 2019. ISSN 0306-2619. doi: https://doi.org/10.1016/j.apenergy.2018.12.063. URL `https://www.sciencedirect.com/science/article/pii/S0306261918318816`.

F. Murtagh and P. Legendre. Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion? Journal of classification, 31(3):274–295, 2014.

E. Nalisnick, J. M. Hernández-Lobato, and P. Smyth. Dropout as a structured shrinkage prior. arXiv preprint arXiv:1810.04045, 2018.

E. T. Nalisnick. On priors for bayesian neural networks. PhD thesis, UC Irvine, (2018).

B. Narayanan, M. Saadeldin, P. Albert, K. McGuinness, and B. M. Namee. Extracting pasture phenotype and biomass percentages using weakly supervised multi-target deep learning on a small dataset, 2021.

R. M. Neal. Bayesian Learning for Neural Networks. PhD thesis, University of Toronto, CAN, 1995. AAINN02676.

R. M. Neal. Bayesian learning for neural networks. PhD thesis, University of Toronto, 1996.

R. Novak, L. Xiao, Y. Bahri, J. Lee, G. Yang, J. Hron, D. A. Abolafia, J. Pennington, and J. Sohl-Dickstein. Bayesian deep convolutional networks with many channels are gaussian processes. In 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019, 2019.

S. J. Pan and Q. Yang. A survey on transfer learning. IEEE Transactions on knowledge and data engineering, 22(10):1345–1359, 2009.

P. Papoutsis, B. Michel, A. Philippe, and T. Duong. Bayesian hierarchical models for the prediction of the driver flow and passenger waiting times in a stochastic carpooling service. working paper or preprint, July 2020. URL `https://hal.archives-ouvertes.fr/hal-02901476`.

A. Pinkus. Approximation theory of the mlp model in neural networks. Acta Numerica, 8:143–195, 1999. doi: 10.1017/S0962492900002919.

N. G. Polson, V. Sokolov, et al. Deep learning: a bayesian perspective. Bayesian Analysis, 12(4):1275–1304, 2017.

M. Qin, Z. Li, and Z. Du. Red tide time series forecasting by combining arima and deep belief network. Knowledge-Based Systems, 125:39–52, 2017.

R Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, 2019. URL `https://www.R-project.org/`.

R. Raina, A. Y. Ng, and D. Koller. Constructing informative priors using transfer learning. In Proceedings of the 23rd international conference on Machine learning, pages 713–720, 2006.

C. E. Rasmussen and C. K. I. Williams. Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press, 2006.

M. Q. Raza, N. Mithulananthan, and A. Summerfield. Solar output power forecast using an ensemble framework with neural predictors and bayesian adaptive combination. Solar Energy, 166:226–241, 2018. ISSN 0038-092X. doi: https://doi.org/10.1016/j.solener.2018.03.066. URL `https://www.sciencedirect.com/science/article/pii/S0038092X18302998`.

S. A. Renganathan, R. Maulik, S. Letizia, and G. V. Iungo. Data-driven wind turbine wake modeling via probabilistic machine learning. arXiv preprint arXiv:2109.02411, 2021.

G. Richard, G. Hébrail, M. Mougeot, and N. Vayatis. Densenets for time series classification: towards automation of time series pre-processing with cnns. In Workshop on Mining and Learning from Time Series, 2019.

G. Richard, B. Grossin, G. Germaine, G. Hébrail, and A. de Moliner. Autoencoder-based time series clustering with energy applications. arXiv preprint arXiv:2002.03624, 2020.

C. P. Robert and G. Casella. Monte Carlo statistical methods; 2nd ed. Springer texts in statistics. Springer, Berlin, 2005. URL `https://cds.cern.ch/record/1187871`.

C. P. Robert et al. The Bayesian choice: from decision-theoretic foundations to computational implementation, volume 2. Springer, 2007.

F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, 65(6):386, 1958.

P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 20:53–65, 1987. ISSN 0377-0427. doi: https://doi.org/10.1016/0377-0427(87)90125-7. URL `https://www.sciencedirect.com/science/article/pii/0377042787901257`.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. nature, 323(6088):533–536, 1986.

F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 815–823, 2015.

M. Segal and Y. Xiao. Multivariate random forests. WIREs Data Mining and Knowledge Discovery, 1(1):80–87, 2011. doi: https://doi.org/10.1002/widm.12. URL `https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.12`.

M. Shepero, D. van der Meer, J. Munkhammar, and J. Widén. Residential probabilistic load forecasting: A method using gaussian process designed for electric load data. Applied Energy, 218:159 – 172, 2018. ISSN 0306-2619. doi: https://doi.org/10.1016/

j.apenergy.2018.02.165. URL http://www.sciencedirect.com/science/article/pii/S030626191830299X.

H. Shi, M. Xu, and R. Li. Deep learning for household load forecasting—a novel pooling deep rnn. IEEE Transactions on Smart Grid, 9(5):5271–5280, 2018.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In Y. Bengio and Y. LeCun, editors, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015.

P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. Technical report, Colorado Univ at Boulder Dept of Computer Science, 1986.

Stan Development Team. The Stan Core Library, 2018. URL http://mc-stan.org/5. Version 2.18.0.

Stan Development Team. RStan: the R interface to Stan, 2020. URL http://mc-stan.org/. R package version 2.21.2.

L. Sun, J. Du, L.-R. Dai, and C.-H. Lee. Multiple-target deep learning for lstm-rnn based speech enhancement. In 2017 Hands-free Speech Communications and Microphone Arrays (HSCMA), pages 136–140. IEEE, 2017.

M. Sun, T. Zhang, Y. Wang, G. Strbac, and C. Kang. Using bayesian deep learning to capture uncertainty for residential net load forecasting. IEEE Transactions on Power Systems, 35(1):188–201, 2020. doi: 10.1109/TPWRS.2019.2924294.

M. Sun, T. Zhang, Y. Wang, G. Strbac, and C. Kang. Using bayesian deep learning to capture uncertainty for residential net load forecasting. IEEE Transactions on Power Systems, 35(1):188–201, 2020. doi: 10.1109/TPWRS.2019.2924294.

Y. Suo, T. Liu, X. Jia, and F. Yu. Application of clustering analysis in brain gene data based on deep learning. IEEE Access, 7:2947–2956, 2018.

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In Advances in neural information processing systems, pages 3104–3112, 2014.

Q. Tao, F. Liu, Y. Li, and D. Sidorov. Air pollution forecasting using a deep learning model based on 1d convnets and bidirectional gru. IEEE access, 7:76690–76698, 2019.

T. Teeraratkul, D. O'Neill, and S. Lall. Shape-based approach to household electric load curve clustering and prediction. IEEE Transactions on Smart Grid, 9(5):5196–5206, 2018. doi: 10.1109/TSG.2017.2683461.

S. Thomassey and M. Happiette. A neural clustering and classification system for sales forecasting of new apparel items. Applied Soft Computing, 7(4):1177–1187, 2007. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2006.01.005. URL https://www.sciencedirect.com/science/article/pii/S156849460600010X. Soft Computing for Time Series Prediction.

M. Titsias. Variational learning of inducing variables in sparse gaussian processes. In Artificial Intelligence and Statistics, pages 567–574, 2009.

L. Torrey and J. Shavlik. Transfer learning. In Handbook of research on machine learning applications and trends: algorithms, methods, and techniques, pages 242–264. IGI Global, 2010.

D. Tran, M. Dusenberry, M. van der Wilk, and D. Hafner. Bayesian layers: A module for neural network uncertainty. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems 32 (NeurIPS), volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/154ff8944e6eac05d0675c95b5b8889d-Paper.pdf.

O. Triebe, N. Laptev, and R. Rajagopal. Ar-net: A simple auto-regressive neural network for time-series. arXiv preprint arXiv:1911.12436, 2019.

A. Vaghefi, F. Farzan, and M. A. Jafari. Modeling industrial loads in non-residential buildings. Applied Energy, 158:378–389, 2015. ISSN 0306-2619. doi: https://doi.org/10.1016/j.apenergy.2015.08.077. URL https://www.sciencedirect.com/science/article/pii/S0306261915010132.

L. Van der Maaten and G. Hinton. Visualizing data using t-sne. Journal of machine learning research, 9(11), 2008.

D. van der Meer, J. Widén, and J. Munkhammar. Review on probabilistic forecasting of photovoltaic power production and electricity consumption. Renewable and Sustainable Energy Reviews, 81:1484 – 1512, 2018. ISSN 1364-0321. doi: https://

doi.org/10.1016/j.rser.2017.05.212. URL http://www.sciencedirect.com/science/article/pii/S1364032117308523.

R. van Steenbergen and M. Mes. Forecasting demand profiles of new products. Decision Support Systems, 139:113401, 2020. ISSN 0167-9236. doi: https://doi.org/10.1016/j.dss.2020.113401. URL https://www.sciencedirect.com/science/article/pii/S0167923620301561.

E. D. Varga, S. F. Beretka, C. Noce, and G. Sapienza. Robust real-time load profile encoding and classification framework for efficient power systems operation. IEEE Transactions on Power Systems, 30(4):1897–1904, 2015. doi: 10.1109/TPWRS.2014.2354552.

E. Vazquez and E. Walter. Multi-output suppport vector regression. IFAC Proceedings Volumes, 36(16):1783–1788, 2003. ISSN 1474-6670. doi: https://doi.org/10.1016/S1474-6670(17)35018-8. URL https://www.sciencedirect.com/science/article/pii/S1474667017350188. 13th IFAC Symposium on System Identification (SYSID 2003), Rotterdam, The Netherlands, 27-29 August, 2003.

M. Vlachos, J. Lin, E. Keogh, and D. Gunopulos. A wavelet-based anytime algorithm for k-means clustering of time series. In In proc. workshop on clustering high dimensionality data and its applications. Citeseer, 2003.

M. Vladimirova, J. Verbeek, P. Mesejo, and J. Arbel. Understanding priors in Bayesian neural networks at the unit level. In K. Chaudhuri and R. Salakhutdinov, editors, Proceedings of the 36th International Conference on Machine Learning, volume 97 of Proceedings of Machine Learning Research, pages 6458–6467, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/vladimirova19a.html.

X. Wang, K. A. Smith, R. Hyndman, and D. Alahakoon. A scalable method for time series clustering, 2004.

Y. Wang, W. Liao, and Y. Chang. Gated recurrent unit network-based short-term photovoltaic forecasting. Energies, 11(8):2163, 2018.

Y. Wang, Q. Chen, T. Hong, and C. Kang. Review of smart meter data analytics: Applications, methodologies, and challenges. IEEE Transactions on Smart Grid, 10(3):3125–3148, 2019. doi: 10.1109/TSG.2018.2818167.

J. H. Ward. Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association, 58(301):236–244, 1963. ISSN 01621459. URL http://www.jstor.org/stable/2282967.

Y. Wen, P. Vicol, J. Ba, D. Tran, and R. B. Grosse. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. In 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJNpifWAb.

P. J. Werbos. Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. PhD thesis, Harvard University, 1974.

A. Wilson, A. Fern, and P. Tadepalli. Transfer learning in sequential decision problems: A hierarchical bayesian approach. In Proceedings of ICML Workshop on Unsupervised and Transfer Learning, pages 217–227. JMLR Workshop and Conference Proceedings, 2012.

D. Xu, Y. Shi, I. W. Tsang, Y.-S. Ong, C. Gong, and X. Shen. Survey on multi-output learning. IEEE transactions on neural networks and learning systems, 31(7):2409–2429, 2019.

Y. Yang, W. Li, T. A. Gulliver, and S. Li. Bayesian deep learning-based probabilistic load forecasting in smart grids. IEEE Transactions on Industrial Informatics, 16(7): 4703–4713, 2020. doi: 10.1109/TII.2019.2942353.

B. Yildiz, J. Bilbao, and A. Sproul. A review and analysis of regression and machine learning models on commercial building electricity load forecasting. Renewable and Sustainable Energy Reviews, 73:1104 – 1122, 2017. ISSN 1364-0321. doi: https://doi.org/10.1016/j.rser.2017.02.023. URL http://www.sciencedirect.com/science/article/pii/S1364032117302265.

H. Yu, F. Khan, and V. Garaniya. Risk-based fault detection using self-organizing map. Reliability Engineering & System Safety, 139:82–96, 2015. ISSN 0951-8320. doi: https://doi.org/10.1016/j.ress.2015.02.011. URL https://www.sciencedirect.com/science/article/pii/S0951832015000563.

K. Yu, V. Tresp, and A. Schwaighofer. Learning gaussian processes from multiple tasks. In Proceedings of the 22nd international conference on Machine learning, pages 1012–1019, 2005.

W. Zhang, X. Liu, Y. Ding, and D. Shi. Multi-output ls-svr machine in extended feature space. In 2012 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA) Proceedings, pages 130–134, 2012. doi: 10.1109/CIMSA.2012.6269600.

B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu. Convolutional neural networks for time series classification. Journal of Systems Engineering and Electronics, 28(1):162–169, 2017.

R. Zhao, W. Xu, H. Su, and Q. Ji. Bayesian hierarchical dynamic model for human action recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7733–7742, 2019.

A. Y. Ziat. Apprentissage de représentation pour la prédiction et la classification de séries temporelles. PhD thesis, Université Pierre et Marie Curie-Paris VI, 2017.

DOCTORAT BRETAGNE LOIRE / MATHSTIC

Nantes Université

Titre : Méthodes d'apprentissage statistique mêlant approche bayésienne et deep learning

Mot clés : Apprentissage profond, apprentissage par transfert, méthodes bayésiennes, séries temporelles, consommation électrique individuelle

Résumé : Nous proposons des méthodes statistiques mêlant approche bayésienne et deep learning pour la prévision de consommation électrique individuelle. Les travaux sont réalisés en partenariat avec EDF. Deux types de méthodologies sont developpées : l'une faisant usage de réseaux de neurones bayésiens et l'autre utilisant du deep learning pour de la réduction de dimension avant clustering en vue d'appliquer des modèles bayésiens plus classiques sur les clusters. Dans un premier temps, nous présentons une méthodologie d'estimation d'un modèle de régression à plusieurs sorties en grande dimension avec des réseaux de neurones. Celle-ci est appliquée à la prédiction de courbes de charges individuelles de clients non résidentiels. Dans un second temps, nous présentons une méthodologie de transfer learning bayésien adaptée à des données de panel. Nous l'appliquons à la problématique de prévision de consommation à la fin du mois de clients résidentiels en situation d'historique court, pour des clusters de clients. Ces clusters de clients sont obtenus avec des réseaux de neurones.

Title: Statistical learning methods combining the Bayesian approach and deep learning

Keywords: Bayesian methods, deep learning, individual electrical consumption, time series, transfer learning

Abstract: We propose statistical methods combining the Bayesian approach and deep learning for forecasting individual electrical consumption. This work is done in partnership with EDF. Two types of methodologies are developed: one relying on Bayesian neural networks, the other using deep learning for dimensionality reduction prior to clustering. Bayesian (non deep) models are then applied to the clusters. Firstly, we present a methodology to estimate a multi target regression model in high dimension with neural networks. It is applied to the prediction of individual load curves of non residential customers. Secondly, we present a Bayesian transfer learning approach adapted to panel data. The methodology is applied to forecasting the individual end-of-month consumption of residential customers, with short historical data, for specific clusters of customers. Those clusters are built using neural networks.