École Normale Supérieure de Paris-Saclay, centre Borelli

Plateau de Saclay, Paris

Stage de recherche - Licence Mathématiques

Étude et prévision de modèle type trafic routier à l'aide des outils du Deep Learning

Rapport de stage de
Amer Essakine, Quentin Boiret
supervisé par
Argyris Kalogeratos et Xavier Cassagnou

---

29 Mars 2023-29 Juin 2023

Remerciements

Nous tenons à remercier toutes les personnes qui ont contribué au bon déroulement de notre stage.
Nous adressons nos remerciements à Argyris Kalogeratos, qui nous a fourni les documents introductifs et nous a particulièrement guidés pendant les débuts de notre stage, puis nous a mis en relation avec d'autres membres du laboratoire Borelli et nous a finalement aidés pendant le déploiement des algorithmes de deep-learning.
Nous tenons à remercier également Xavier Cassagnou qui nous a aidés sur les premiers codes à écrire, nous a fourni l'article principal sur lequel nous nous sommes appuyé pendant tout le stage et s'est montré présent à chaque fois que ce fut nécessaire.
Nous souhaitons également exprimer nos remerciements envers Eloi Campagne pour nous avoir fourni les données de consommation d'énergie des différentes régions, ainsi que pour ses retours sur les résultats obtenus et ses suggestions concernant la méthode GAM.

Contents

1 Introduction

En 2008, le Texas a inauguré la plus large autoroute du monde, la Katy Freeway, avec 26 voies de largeur ! Le but était bien sûr la fluidification du trafic en dédoublant progressivement les voies. On constate pourtant une augmentation du temps (de l'ordre de 50% supplémentaire entre 2018 et 2021 !) qu'il faut pour sortir de Houston à partir de son centre-ville en passant par ces mêmes voies , et ce, malgré l'augmentation continuelle du nombre de voies ; comment expliquer ce paradoxe, et surtout comment le résoudre ? L'état du trafic routier texan n'est qu'une illustration des nombreux intérêts qu'il existe à étudier le comportement d'un trafic routier.
Le problème de la prévision d'un trafic, à savoir prévoir l'évolution des grandeurs caractéristiques d'un transport de quantité d'un point à un autre sur un réseau, est une problématique classique de la recherche qui a traversé les décennies et s'est confrontée à beaucoup de méthodes et d'approches différentes. Certains cas sont analytiquement solutionnés, car on dispose d'équation fondamentale pour décrire le phénomène, comme le cas d'un écoulement d'un fluide dans un réseau de canalisation. Certains, cependant, sont toujours étudiés car il n'existe pas d'équation pilote du fait de la complexité du modèle (météo, date et heure, accident, événement exceptionnel). La problématique a retrouvé un nouvel essor avec l'avènement du machine learning, méthode adéquate pour des problèmes sans équation pilote traitant d'un grand nombre de paramètres cachés. Nous allons ici nous interroger sur la façon dont le machine learning a amélioré nos connaissances sur la problématique des prévisions de trafic. Dans notre stage, nous nous sommes intéressés en particulier à l'état du réseau électrique français et à la façon de prévoir l'évolution de l'état de son trafic. Comment prévoir les ajustements de la production électrique à l'aide des données de l'état du trafic à des instants antérieurs ?
Dans une première partie, nous allons présenter de façon générique la méthode et les outils principaux du machine learning. Puis nous nous arrêterons sur les méthodes traditionnelles utilisées avant l'avènement du machine learning et qui nous servirons de méthode étalon. Par la suite, nous présenterons en détail l'algorithme de machine learning qui nous a été utile pour étudier le trafic et ses résultats. Finalement, nous conclurons par une approche comparative des différentes méthodes rencontrées.

2 Le machine learning et ses outils

Nous allons ici introduire les outils du machine learning que l'on a utilisés après avoir défini la façon conceptuelle avec laquelle fonctionne le machine learning.

2.1 le concept

Proposons une analogie avec l'apprentissage d'un enfant pour comprendre l'esprit de la méthode [1]. Lorsque l'on souhaite apprendre à lire à un enfant, nous ne faisons pas un cours sur le calligraphie ou la description détaillée des courbes que composent une lettre,

puis un mot. À la place, on lui présente, d'une façon plus ou moins guidée, des exemples de mots, de façon dont ils se prononcent et comment ils se placent dans une phrase. Puis on lui fait faire des exercices d'application que l'enseignant corrige avec l'enfant en lui indiquant ce qu'il a réussi ou pas. Ensuite, On attend de l'enfant qu'à partir de ces exemples, il puisse le généraliser à d'autres cas, en particulier ceux qu'il n'a pas encore rencontrés.
On distingue ici 3 étapes :

- la phase d'apprentissage: l'enfant apprend la leçon au travers d'exemples
- la phase de validation: l'enseignant corrige les erreurs d'apprentissage de l'enfant
- la phase de test: l'enfant utilise ce qu'il a appris en cours et applique ses connaissances dans les déclinaisons infinies de la vie de tous les jours.

Le machine learning procède de la même façon en apprenant à partir des exemples qu'on lui donne, puis en validant ses hypothèses à partir d'autres cas, et enfin en généralisant ses résultats à tous les autres cas qu'il n'a pas rencontrés.

2.2 Le modèle mathématique

Voyons maintenant comment formaliser ça avec des équations. Les trois étapes décrites précédemment peuvent se formaliser au travers la détermination d'une fonction à qui on apprendrait pour chaque entrée x, à lui attribuer la bonne sortie y [2]. Prenons une base de données sur laquelle on souhaite faire les bonnes attributions. Comment y déterminer la fonction ? Voyons les étapes dans les sections suivantes.

2.2.1 L'apprentissage et la validation

Sur la phase d'apprentissage, on va extraire de la base de données une sous-partie qui nous servira de source d'exemples sur laquelle la fonction va s'entraîner. Pour simplifier, supposons qu'on ait fait l'hypothèse que la fonction soit de la forme :

$$f(x) = w * x + b \tag{1}$$

où w et b sont les paramètres à fixer en entraînant le modèle. Pour mesurer si notre fonction est sur la bonne voie et apprend correctement, on lui adjoint une métrique qui mesure l'erreur, on l'appelle fonction de perte, ici notée J telle que :

$$J(w, b) = \sum [f(x_i) - y_i]^2 \tag{2}$$

où x_i doit être associé à y_i. Cette fonction est généralement minimisée par des algorithmes optimisés à partir des méthodes de descente de gradient. Il existe d'autres formes de fonction de perte que celle quadratique qu'on introduira lorsqu'on les utilisera. Une fois l'apprentissage sur une sous-partie de l'ensemble fait, on évalue la pertinence des paramètres choisis sur un autre sous-ensemble sur lequel il n'a pas appris, c'est la validation. il existe plusieurs types de validation, puisqu'il existe plusieurs façons de découper l'ensemble initial en sous-ensembles. Le principe reste le même pour chacune d'entre elles.

2.2.2 La généralisation

Finalement, on teste le machine learning sur un nouveau sous-ensemble qui n'a pas encore été utilisé dans l'apprentissage et la validation. On attend que la fonction générée soit capable de s'adapter à de nouveaux cas qui n'ont pas encore été rencontrés, à la manière de l'enfant qui a appris à lire, on s'attend à ce qu'il soit capable de lire de nouveaux mots qu'il n'a pas rencontrés lors de son apprentissage.

2.3 Les métriques

Afin de quantifier nos résultats et de les comparer entre eux, nous utiliserons les métriques les plus utilisées dans la littérature [3]. Si on note $(y_i)_i$ les valeurs attendues et $(\hat{y}_i)_i$ les valeurs prédites, alors les métriques les plus couramment utilisées sont :

- La métrique MAE (Mean Absolute Error)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3}$$

- la métrique MAPE (Mean Absolute Percentage Error)

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100 \tag{4}$$

- la métrique RMSE (Root Mean Square Error)

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{5}$$

2.4 Réseaux de convolutions sur les graphes

2.4.1 Les réseaux de neurones

Les réseaux de neurones sont un sous-ensemble de l'apprentissage automatique (machine learning). Leur structure est inspirée du cerveau humain, imitant la manière dont les neurones biologiques communiquent entre eux. Les réseaux sont architecturés sous forme de couches de neurones interconnectés qui transmettent les informations les uns aux autres.
L'idée est que l'on peut modéliser des phénomènes assez complexes en les décomposant en des phénomènes plus simples. La fonction f sur laquelle nous effectuons l'apprentissage peut donc être composée de n fonctions plus simples, c'est-à-dire $f = f_1 \circ \ldots \circ f_n$. Ce qui rend cette méthode très efficace est la possibilité d'utiliser la rétropropagation du gradient (back-propagation). Cette technique permet de calculer rapidement et facilement les gradients de

la fonction de perte en utilisant la règle des chaînes.
En général, chaque couche f_i est la combinaison d'une fonction linéaire et d'une fonction non linéaire a dite fonction d'activation : $f_i(x) = a(w * x + b)$. Parmi les fonctions d'activation les plus utilisées dans la littérature

- **ReLU** : $ReLU(x) = \mathbb{1}_{x>0}$ (où $\mathbb{1}$ est la fonction indicatrice)
- **Softmax** : Fonction de normalisation pour obtenir une distribution de probabilités finis.
- **GLU** : $GLU(a, b) = a \odot b$, où $\odot$ représente la multiplication matricielle terme à terme.

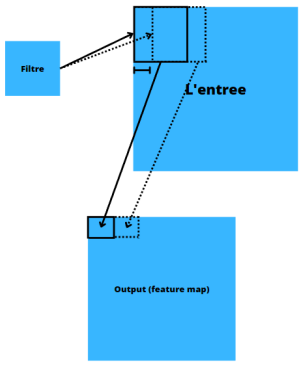
2.4.2 Réseaux de convolution

Les réseaux de neurones à rétropropagation rencontrent des difficultés lorsqu'il s'agit de traiter des images en raison de la complexité computationnelle requise pour ce type de données. En effet, le nombre de paramètres du système augmente rapidement avec la taille de l'image. Par exemple, une petite image de 28x28 pixels nécessite 764 poids pour chaque nœud, tandis qu'une image en couleur de 64x64 pixels nécessite 12 228 paramètres. La reconnaissance d'images nécessite donc des réseaux à rétropropagation suffisamment complexes, ce qui peut entraîner un surajustement (overfitting).
Pour surmonter ces problèmes, les réseaux convolutifs sont utilisés[4]. Les réseaux convolutifs sont essentiellement des réseaux de neurones qui utilisent une opération appelée convolution au lieu de la multiplication matricielle classique. Cette substitution vise à réduire la complexité computationnelle des problèmes de reconnaissance et à exploiter les propriétés intrinsèques des données pouvant être modélisées sous forme de grilles, telles que les images, afin d'améliorer les performances de l'algorithme.

Définition 1 *La convolution du signal d'entrée I avec le noyau K est donnée par :*

$$K * I(i, j) = \sum_m \sum_n I(m, n)K(i - m, j - n)$$

L'idée est d'effectuer une multiplication matricielle locale sur des regroupements de pixels afin de rechercher des motifs dans la grille. Cette approche permet de réduire considérablement le nombre de paramètres par rapport aux réseaux de neurones classiques, car ces paramètres sont partagés par les différents pixels de la grille.



Dans notre stage, nous avons souhaité utiliser des réseaux convolutifs pour prédire des données spatio-temporelles qui ne possèdent pas nécessairement une topologie de grille. Pour cela, il était nécessaire d'adapter le réseau en utilisant la théorie des graphes.

2.4.3 Théorie du graphe

Un graphe $\mathcal{G}$ est un triplet $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$, où $\mathcal{V}$ est un ensemble fini de nœuds, $\mathcal{E}$ est un sous-ensemble de V^2 dont les éléments sont appelés des arêtes, et W est la matrice d'adjacence telle que $W_{i,j}$ représente le poids entre les nœuds i et j.
On définit également des matrices associées au graphe :

- La matrice des degrés associée à $\mathcal{G}$ est une matrice diagonale dont les éléments sont donnés par $D_{i,i} = \sum_j W_{i,j}$.
- La matrice laplacienne normalisée de $\mathcal{G}$ est définie comme $L = I_n - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$.
- La matrice U contient les vecteurs propres de L, et Λ est une matrice diagonale correspondant aux valeurs propres de L. Plus précisément, on a $L = U \Lambda U^T$.

2.4.4 Réseaux de convolutions spectrales

La convolution régulière, utilisée pour les grilles, n'est pas directement applicable aux graphes. Cependant, il existe deux méthodes permettant d'adapter les réseaux de convolutions aux données structurées sous forme de graphe. La première consiste à trouver une façon de représenter le graphe sous forme de grille, tandis que la deuxième consiste à travailler dans le domaine spectral, où les graphes sont représentés par des matrices. Dans le cadre de notre stage, nous avons adopté l'approche spectrale.[5]
On définit l'opération de convolution sur un graphe comme la multiplication matricielle de l'entrée $x \in \mathbb{R}^N$ et du noyau $g_\theta = \text{diag}(\theta)$, paramétrisé par $\theta \in \mathbb{R}^N$, dans le domaine de Fourier.

$$g_\theta \star_\mathcal{G} x = g_\theta(\Lambda) x = U g_\theta U^T x \tag{6}$$

Approximation de premier ordre Cependant, l'evaluation de cette equation est coûteux en termes de calculs, car la multiplication avec la matrice des vecteurs propres U se fait en $\mathcal{O}(N^2)$. Additionellement, on doit calculer les valeurs propres de L. Pour contourner ce problème, Une approche consiste à effectuer une approximation de premier ordre du laplacien L. En plus de cela, nous approchons $\lambda_{\max}$ par environ 2. L'équation de convolution se simplifie alors comme suit :

$$g_\theta \star_\mathcal{G} x \approx \theta_0 x - \theta_1 (L - I_N) x \approx \theta_0 x - \theta_1 (D^{-\frac{1}{2}} W D^{\frac{1}{2}}) x \tag{7}$$

Afin de réduire le nombre de paramètres, nous supposons que $\theta_0 = \theta_1$. Par conséquent,

$$g_\theta \star_\mathcal{G} x \approx \theta (I_N + D^{-\frac{1}{2}} W D^{\frac{1}{2}}) x \tag{8}$$

Ensuite, l'idée est d'empiler K couches de convolutions de premier ordre afin d'obtenir un effet similaire à l'approximation de Chebyshev. La complexité de calcul reste la même.

3 Les méthodes traditionnelles

Nous avons à notre disposition des données sur la consommation énergétique des 12 régions de France, sur une période s'étendant du 1er janvier 2014 au 31 décembre 2018. Ces données ont été collectées toutes les 30 minutes. De plus, nous disposons également des enregistrements de la température moyenne pour chaque région à chaque instant de mesure. Pour chaque méthode, nous utilisons les données des quatre années 2014, 2015, 2016 et 2017 afin de prédire la consommation pour la dernière année.

3.1 La méthode Arima

La méthode ARIMA (AutoRegressive Integrated Moving Average) est une méthode statistique pour étudier l'évolution temporelle d'un noeud au cours du temps [6]. La méthode est plus ancienne, datant du début des années 2000, à l'époque où les méthodes d'utilisation de machine learning était moins efficace. Elle est utilisée dans les problèmes où il n'y a pas de réseau et donc où il n'y a pas de dépendance spatiale du problème. Cette méthode servira de point de départ utile pour comparer les autres méthodes utilisées et voir leur efficacité relative.

3.1.1 Présentation du modèle

L'idée de la méthode est de trouver des schémas récurrents de la série statistique pour ensuite les reproduire. La méthode se décompose comme suit :

- AR (p): Autoregression d'ordre p. Chaque signal peut être déterminé connissant les p points précédents donc
$$\forall t, X_t = \sum_{k=1}^{p} \alpha_k X_k \tag{9}$$
- MA: Prise en compte du bruit des anciens signaux
- I: la régression va prendre en compte un éventuel décalage du signal périodique.

Dans sa conception, la méthode ARIMA est simple. La régression va se faire sur un signal périodique, dont la valeur moyenne se décale avec le temps. La méthode est parfaitement adéquate dans le cas de signaux périodiques comme les températures ou la consommation électrique. On constatera cependant les échecs pour le trafic routier, où les signaux périodiques sont sensiblement affectés par les perturbations.

3.1.2 Pourquoi utiliser ARIMA ?

On souhaite utiliser ARIMA, car il s'agit d'une méthode déterministe connue et utilisée depuis au moins une vingtaine d'années. Elle nous servira de méthode étalon avec laquelle nous pourrons ensuite comparer les résultats des méthodes deep learning.

3.2 La méthode GAM

On utilise également la méthode GAM (Generalized Additive Models). C'est une méthode statistique qui généralise la régression linéaire à des problèmes non linéaires, permettant des relations plus complexes entre les données [7]. L'approche des GAM repose sur l'hypothèse que l'effet de chaque variable peut être modélisée de manière indépendante, en ajoutant simplement les effets individuels de chaque variable. Cela permet ainsi de modéliser des phénomènes non linéaires tout en conservant une structure additive. Cette méthode, en plus de présenter plus de paramètres que la méthode ARIMA, est celle effectivement utilisée par le réseau EDF.

3.2.1 Présentation du modèle

Un problème de régression linéaire peut être formalisé de la manière suivante :

$$Y = \beta_0 x_0 + \beta_1 x_1 + \dots \beta_n x_n$$

Avec Y représentant la variable à prédire, les x_i sont les données d'entrée et les β_i les poids à prédire. Dans le cas des GAM, on abandonne l'hypothèse selon laquelle la variable Y peut être calculée en utilisant une combinaison linéaire des variables d'entrée en affirmant simplement que nous pouvons utiliser une combinaison non-linéaire de variables, représentée par la fonction lisse s :

$$Y = s(x_0) + s(x_1) + \dots + s(x_n)$$

Et nous cherchons à représenter la fonction s sous la forme d'une spline cubique, qui est une fonction polynomiale par morceau définie par l'équation :

$$s(x) = \sum_{k=1}^{n} \beta_i B_i(x)$$

Les polynômes B_i sont de degré 3, et β_i sont des réels. Au cours de l'algorithme, nous cherchons à optimiser à la fois les valeurs des coefficients β_i, les coefficients des B_i, ainsi que n, le nombre de termes.

3.2.2 Pourquoi utiliser la méthode GAM ?

Nous mettrons en parallèle les résultats de la méthode ARIMA avec celle de la méthode GAM, puisqu'il s'agit de deux méthodes déterministes. Nous nous attendons à de meilleurs résultats avec la méthode GAM, puisqu'elle se présente comme une méthode plus robuste (utilisation de fonction régulière au lieu d'un simple modèle linaire). Puis nous mettrons en parallèle ces résultats avec ceux obtenus à l'aide des algorithmes de machine learning.

4 Résultats des approches traditionnelles

Nous présentons dans cette section les résultats obtenus à partir des méthodes déterministes sur le problème de la prédiction de la consommation électrique.

4.1 Résultats des métriques

À l'aide des métriques définies dans la partie I, voici les résultats pour chaque méthode déterministe:

Méthode	ARIMA	GAMs
MAPE	26.452	1.9667
MAE	26.452	1.9667
RMSE	27.04	2.6459

Comme il était attendu, la méthode GAM fournit de meilleurs résultats que la méthode ARIMA. Nous montrons graphiquement ce que ça représente dans la section suivante.

4.2 Analyse des résultats

4.2.1 ARIMA

Bien que la méthode ARIMA produise des résultats plus ou moins satisfaisant à court terme (par exemple, une erreur MAPE d'environ 2% pour la première semaine), cette méthode n'est pas fiable pour les prévisions à long terme (Figure 1).

4.2.2 GAM

Pour la méthode GAM, nous avons pris en compte comme paramètres la température ainsi que les valeurs des 3 instants précédents.

$$Y_t = s(T_t) + s(Y_{t-1}) + (Y_{t-2}) + (Y_{t-3})$$

La méthode GAM représente une première approche combinant les techniques de la statistique et de Machine Learning pour réaliser des prédictions. Elle peut produire des résultats relativement satisfaisants, avec une erreur MAPE d'environ 2%. Cependant, la méthode GAM ne tient pas compte de l'éventuelle dépendance spatiale entre les différentes régions en termes de consommation.

Dans le paragraphe suivant, nous explorerons l'utilisation de la méthode STGCN pour évaluer si l'hypothèse de dépendance spatiale peut permettre d'améliorer la précision des prévisions.

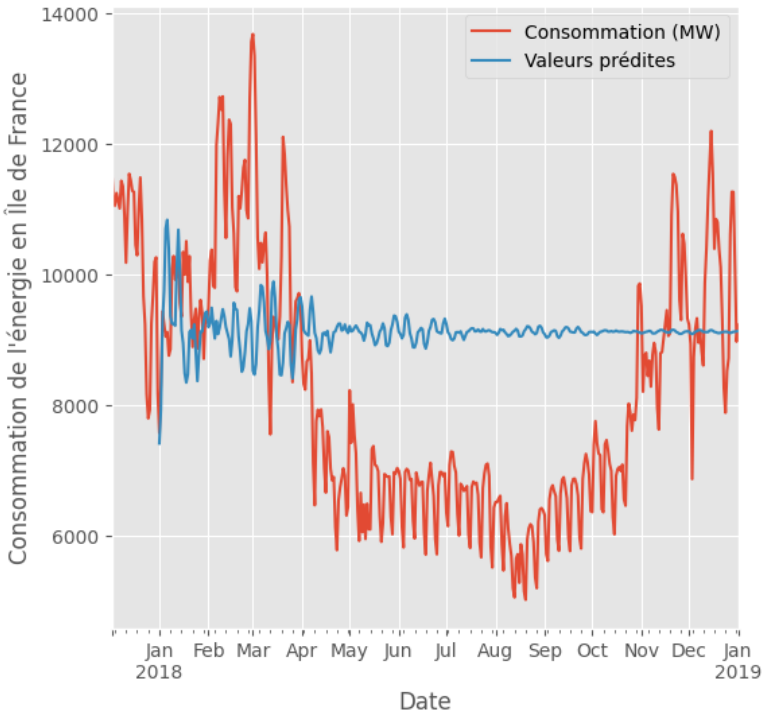


Figure 1: prévision pour la méthode ARIMA

5 La méthode Deep-learning: STGCN

5.1 Description du problème

Nous utilisons des réseaux de convolution graphique spatio-temporelle pour prédire les données de consommation d'énergie [8]. Ce sont des réseaux de neurones qui comportent de nombreux blocs de convolution spatio-temporelle, chacun étant constitué d'une couche de convolution spatiale entre deux couches de convolutions temporelles.
Le problème de la prévision de l'énergie consiste à prédire la consommation d'énergie pour les H prochaines périodes de temps, en se basant sur les M observations précédentes de la consommation et de la température. Mathématiquement, le problème s'exprime comme suit :

$$\tilde{e}_{t+1}, \ldots, \tilde{e}_{t+H} = \underset{e_{t+1},\ldots,e_{t+H}}{\arg\max} \log P(e_{t+1}, \ldots, e_{t+H} | e_{t-M+1}, \ldots, e_t, T_{t-M+1}, \ldots, T_t)$$

Avec $e_t \in \mathbb{R}^{12}$ le vecteur représentant la consommation d'énergie pour les différentes régions à l'instant t, et T_t le vecteur des observations de température à l'instant t.
Pour modéliser les dépendances spatiales entre les données, on suppose que les entrées du vecteur e_t ne sont pas indépendantes, mais liées par des connexions par paires dans un graphe. Ainsi, chaque e_t peut être considéré comme un signal sur un graphe $\mathcal{G}$. À l'étape

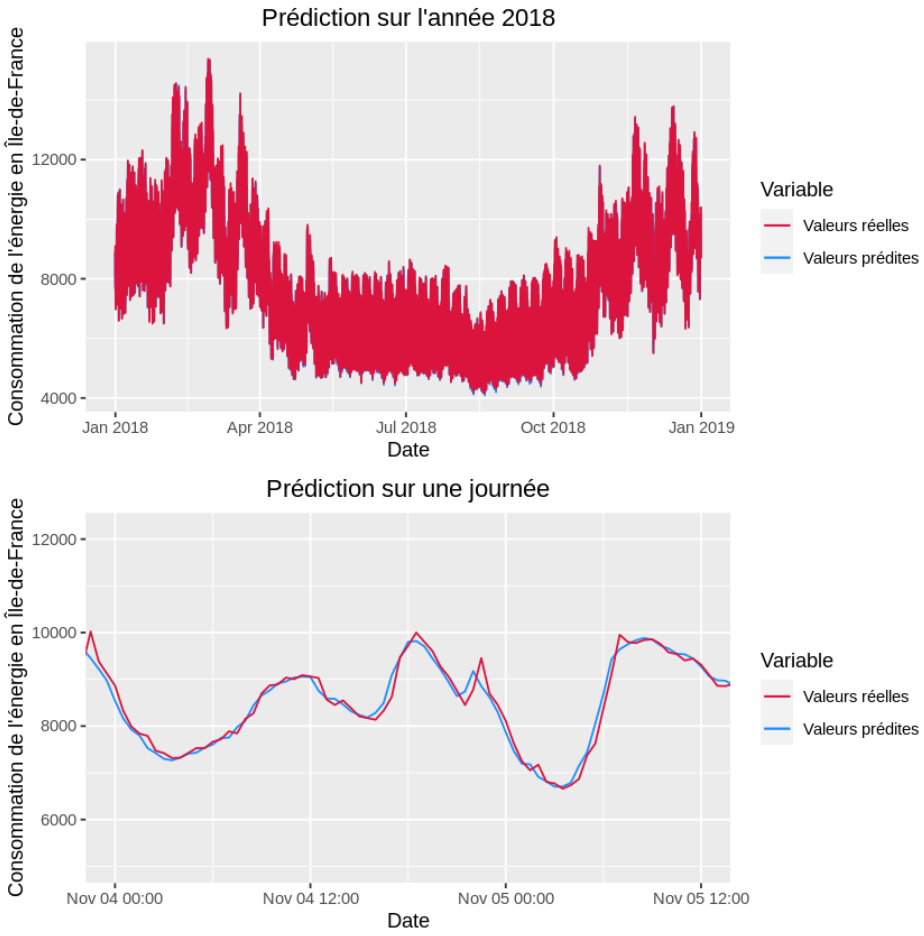


Figure 2: prévision pour la méthode GAM. En haut, la prédiciton sur l'année. En bas, une sous partie de la figure pour visualiser l'allule de la différence entre les deux courbes.

t, les observations sont représentées par un graphe $\mathcal{G} = (\mathcal{V}_t, \mathcal{E}, W)$, où $\mathcal{V}_t$ est l'ensemble des nœuds correspondant aux couples d'observations de la consommation d'énergie et de la température, $\mathcal{E}$ est l'ensemble des nœuds indiquant l'interdépendance spatiale entre les différentes régions, et $W \in \mathbb{R}^{12\times12}$ est la matrice d'adjacence.
Nous notons C_i le nombre d'attributs d'entrée (dans notre cas, les attributs sont la température et la consommation) et C_0 le nombre d'attributs de sortie (dans notre cas, $C_0 = 1$). De plus, nous utilisons M pour représenter le nombre de pas de temps utilisés pour la prédiction (ici, $M = 12$). Dans le cadre de notre problème, nous souhaitons prédire les valeurs de consommation d'énergie pour les 3 prochains pas de temps ($H = 3$).

5.1.1 Convolution spatiale

La dépendance spatiale du problème est modélisée par les nœuds du graphe, qui capturent les relations spatiales entre les différentes régions. On utilise donc l'opération de convolution

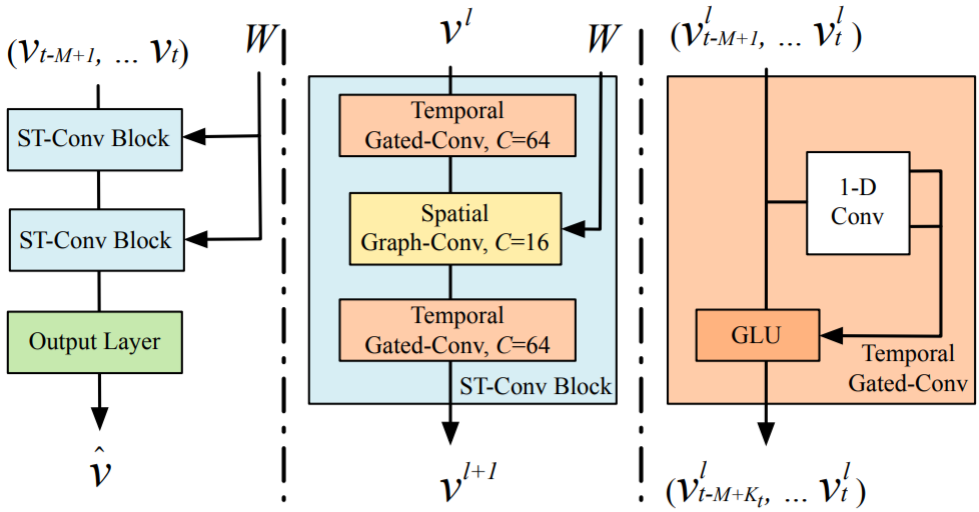


Figure 3: Architecture de STGCN, figure tiré de [8]

sur graphe, permettant de mieux capter les motifs et les caractéristiques spatiales.
La convolution spatiale peut être généralisée à des tenseurs 2D $X \in \mathbb{R}^{n \times C_i}$ par :

$$\Theta \star_{\mathcal{G}} X = Z = (I_N + D^{-\frac{1}{2}} W D^{\frac{1}{2}})\Theta X$$

Avec $\Theta \in \mathbb{R}^{C_i \times C_0}$ le tenseur des paramètres et la sortie $Z \in \mathbb{R}^{n \times C_0}$. La convolution peut être généralisée à des tenseurs $X \in \mathbb{R}^{M \times n \times C_i}$ en appliquant la convolution spectrale 2D à chaque pas de temps.

5.1.2 Convolution temporelle

Pour capturer les motifs et les caractéristiques temporelles, on utilise l'opération de convolution sur des grilles avec un noyau de longueur K_t, suivi d'une fonction d'activation appelée unité linéaire à portes (GLU) pour incorporer la non-linéarité au bloc.
Étant donné un tenseur $X \in \mathbb{R}^{M \times C_i}$ à l'entrée du bloc temporel, on effectue une convolution temporelle avec un noyau $\Gamma_t \in \mathbb{R}^{K_t \times C_i \times 2C_0}$ pour obtenir une sortie $[PQ] \in \mathbb{R}^{M - K_t + 1 \times C_0}$. Ensuite, nous appliquons la fonction GLU à P et Q. Par conséquent, la convolution temporelle avec portes peut être définie comme suit :

$$\Gamma \star_{\mathcal{T}} X = P \odot \sigma(Q) \in \mathbb{R}^{(M - K_t + 1) \times C_0}$$

De manière similaire, la convolution temporelle peut également être généralisée aux variables 3D en utilisant le même noyau de convolution Γ pour chaque noeud $X_i \in \mathbb{R}^{M \times C_i}$ (à chaque région). On le note également $\Gamma \star_{\mathcal{T}} X$ avec $X \in \mathbb{R}^{M \times n \times C_i}$

5.1.3 Convolution spatio-temporelle

Comme indiqué sur la figure 3, le STGCN est l'empilement de plusieurs blocs spatio-temporels (figure de gauche), chaque bloc étant constitué d'un bloc de convolution spatiale entre deux blocs de convolution temporelle. Pour l'entrée $V^l \in \mathbb{R}^M \times n \times C_l$ du l-ème bloc spatio-temporel, la sortie $V^{l+1} \in \mathbb{R}^{(M-2(K_t-1))\times n \times C_{l+1}}$ est calculée selon la formule suivante :

$$V^{l+1} = \Gamma_1^l \star_{\mathcal{T}} ReLU(\Theta^l \star_{\mathcal{G}} (\Gamma_0^l \star_{\mathcal{T}} V^l)$$

Où Γ_1^l et Γ_0^l sont les noyaux des blocs de convolution temporels supérieurs et inférieurs respectivement, et Θ^l est le noyau du bloc de convolution spatiale.
Dans la pratique, on utilise deux blocs spatio-temporels et on ajoute une couche de convolution temporelle supplémentaire avec une couche entièrement connectée comme couche de sortie à la fin, comme montré sur la figure. On utilise la norme L2 comme fonction de perte, que l'on peut écrire :

$$L(\tilde{e}, w_\theta) = \sum_t \|\tilde{e}(e_{i-M+1}, ..., e_i) - e_{t+1}\|^2$$

Avec w_θ représentant tous les paramètres entraînables du modèle, $\tilde{e}$ désignant les valeurs prédites par le modèle et e représentant les valeurs réelles.

5.2 Dépendence spatiale

Une des difficultés qui empêche l'application de la méthode STGCN est la construction d'un graphe pondéré permettant de modéliser correctement la dépendance spatiale du problème. Dans le cas du trafic routier, la structure des routes offre une définition naturelle des graphes, et la distance entre les capteurs fournit une mesure de la connectivité entre deux nœuds. En revanche, pour le problème de la consommation d'énergie, la distance ne joue pas un rôle aussi crucial, ce qui rend la tâche de prendre en compte l'aspect spatiale plus difficile.
On propose dans cette paragraphe de construire une matrice d'adjacence permettant de bien modéliser la dépendance spatiale entre les différentes régions.

5.2.1 Matrice identité

Une première approche consiste à traiter le problème sans prendre en compte la dimension spatiale, ce qui équivaut à utiliser la matrice identité comme matrice d'adjacence. Cette approche est équivalente à appliquer un réseau de convolution à chaque réseau indépendamment des autres.

5.2.2 Matrice de proximité

Une première approche naturelle qu'on a prise pour rendre compte de la dimension spatiale, est de considérer la matrice de proximité spatiale. Plus précisément :

$$\begin{cases} W_{i,j} = 1, & \text{Si les regions } i \text{ et } j \text{ sont voisins} \\ W_{i,j} = 0, & \text{Sinon} \end{cases}$$

La matrice de proximité permet de rendre en compte des phénomenes propagatifs d'une région a l'autre et influençant la consommation d'une region, par exemple une vague de chaleur ou les variations du climat en general. On cherche ensuite notre matrice d'adjacence comme combinaison linéaire de la matrice identité et la matrice de proximité.

$$W = \lambda W_{proximite} + (1 - \lambda) I_n$$

Avec $\lambda \ll 1$, l'idée est que la majeure partie du motif appris sur la consommation d'une région provient de l'historique de cette région, mais cette prédiction peut être améliorée en supposant que la consommation est légèrement influencée par les régions voisines. Le poids w_1 doit être suffisamment faible pour que l'influence spatiale n'interfère pas avec l'information extraite du signal.
Pour déterminer une valeur appropriée de λ, nous considérons λ comme un hyperparamètre de l'algorithme. En utilisant une recherche par grille sur une plage de valeurs et en comparant les performances de la matrice, nous avons sélectionné la valeur $\lambda = 0.9754$ comme la plus optimale.

5.2.3 Matrice de corrélation

La correlation de deux signaux X et Y se définit par :

$$\rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y}$$

En mesurant la covariance ainsi que la variance des observations de X et Y, on peut déterminer la corrélation, qui quantifie dans quelle mesure les variations d'un signal sont associées aux variations de l'autre signal. La matrice d'adjacence est alors définie comme suit :

$$W_{i,j} = \rho_{T_i, T_j}$$

Avec T_i représentant le signal des observations de la température de la région i, il est également possible de définir une matrice de corrélation en se basant sur la corrélation des observations de la consommation d'énergie. Cependant, la méthode utilisant la corrélation sur la consommation d'énergie donne de meilleurs résultats en raison de la relation de causalité entre la température et la consommation d'énergie.
De plus, afin de simplifier la matrice et d'éliminer les nœuds ayant de petites valeurs susceptibles d'interférer avec la précision des prédictions, nous ne conservons que les termes les plus significatifs. Nous définissons donc le nombre de nœuds à conserver comme un hyperparamètre du système, et après une phase de validation, nous choisissons de ne conserver que le nœud ayant le poids le plus élevé.

5.2.4 Algorithme FGL-3SR

L'algorithme FGL-3SR, développé au sein du laboratoire Centre Borelli, constitue une approche novatrice pour l'inférence des nœuds d'un graphe à partir des données disponibles. En se basant sur les données observées, cet algorithme permet d'apprendre la matrice lapalcien decrivant la dependance entre les differents noeuds.[9]
La matrice résultante de l'algorithme, qui décrit les dépendances entre les différentes régions, est obtenue à partir des données de température des 12 régions. Cette matrice est ensuite combinée avec la matrice identité en utilisant une combinaison convexe.
En testant différentes valeurs, nous avons choisi d'attribuer à λ la valeur 0.9759 et de conserver tous les nœuds, à l'exception des deux ayant les valeurs les plus petites. Nous conservons plus de valeurs que dans le cas de la matrice de corrélation, car les valeurs obtenues par FGL-3SR sont plus pertinentes pour le problème. De plus, la matrice résultante de FGL-3SR est déjà suffisamment sparse.

6 Résultats des approches de Deep Learning

Nous présentons ici les résultats obtenus à partir des méthodes présentées dans la section précédente. Nous les mettrons en parralèle avec ceux obtenus par les méthodes traditionnelles.

6.1 Présentation des modèles

Tous les modèles s'appuyant sur les méthodes de deep-learning utilisent les mêmes principes (opérations de convolution, etc). Seuls diffèrent le choix des matrices d'adjacence. Nous allons présenter les résultats à partir de quatre d'entre elles : l'identité, la proximité, la corrélation et l'algorithme FGL-3SR

6.2 Résultats

6.2.1 Les métriques

Dans ce tableau sont regroupés les résultats en fonction des différents métriques. Nous rappelons également ceux des méthodes traditionnelles pour une meilleure visibilité:

Méthode	ARIMA	GAMs	identité	proximité	corrélation	FGL-3SR
MAPE	26.452	1.9667	1.3295	1.2571	1.2526	1.1797
MAE	26.452	1.9667	1.1777	1.1400	1.1341	1.0796
RMSE	27.04	2.6459	1.5947	1.5550	1.5641	1.4859

Nous observons une amélioration nette des résultats notamment entre l'approche sans dépendance spatiale (l'idendité) et la matrice da'adjacence la plus optimisée (FGL-3SR) de l'ordre de 11%.

Une première approche superficielle entre la méthode ARIMA et la méthode deep-learning avec la matrice idendité, illustrée par les graphes (figure 4), pourrait nous faire penser que les résultats sont satisfaisants avec cette dernière. Mais en réalité la dépendance spatiale est essentielle car un gain de cette sorte n'est pas négligeable lorsque l'on fait des productions d'ajustement d'électricité.
Finalement, on constate la nécessite de procéder par des méthodes d'optimisation pour obtenir la meilleure matrice d'adjacence. L'approche du "bon-sens" avec la matrice de proximité ne donne pas d'amélioration des résultats par rapport à celle sans dépendance spatiale.

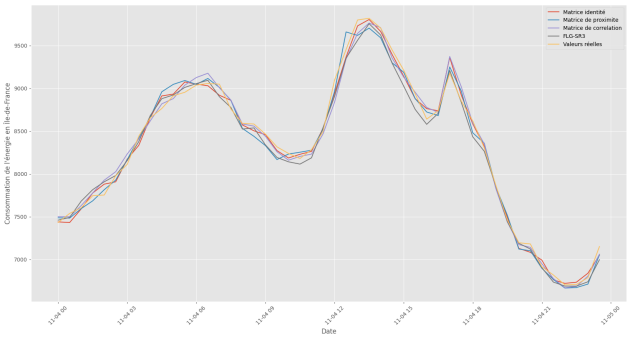


Figure 4: Graphique des résultats des méthodes de deep learning sur la journée du 4 novembre

6.2.2 Les régions

Pour visualiser les résultats plus en détails, nous avons également tracé les résultats des prévisions pour chaque région (Figure 5). L'objectif est d'une part de s'assurer qu'i n'y ait pas d'anomalie d'une région à l'autres, d'autre part que le gain en prévision ne soit pas concentré que sur une unique région. Au contraire, Les gains sont distribués de façon assez uniforme.
On constate une nette amélioration entre la méthode GAM (vert) et les méthodes deep-learning. Et au sein des méthodes deep-learning, la méthode FGL-3SR se distingue assez nettement.

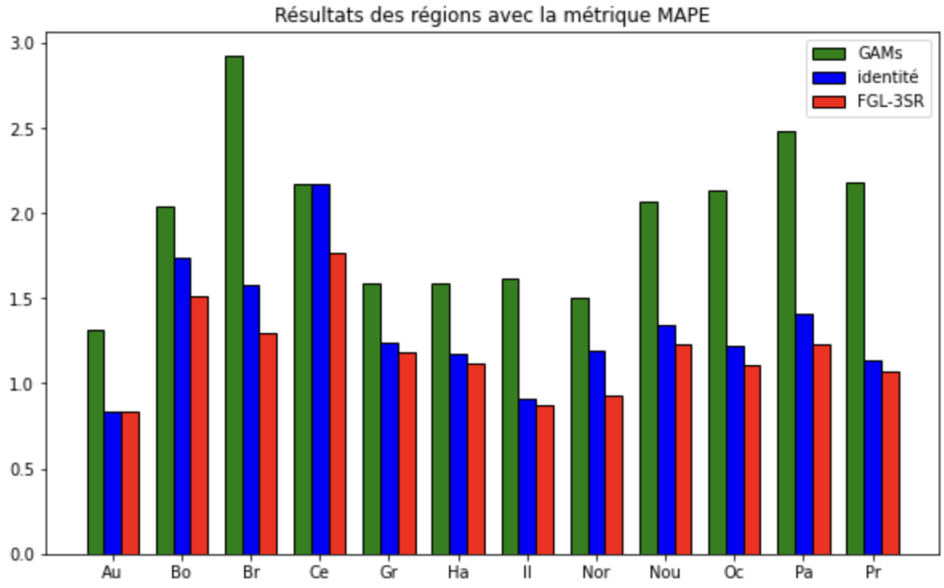


Figure 5: Graphique des résultats par région (classé par ordre alphabétique et abrégé) en fonction de la métrique MAPE

7 Conclusion

À partir des résultats obtenus pendant ce stage, nous avons mis en évidence l'intérêt de l'utilisation des méthodes d'apprentissage automatique par rapport aux approches déterministes. Après avoir optimisé le paramètre de la dépendance spatiale à l'aide de l'algorithme FGL-3SR, la méthode STGCN s'est avérée fournir les meilleurs résultats parmi toutes les approches présentées. Ainsi, les prédictions de la consommation électrique sont plus fiables avec cette méthode. Cependant, il est difficile de choisir un graphe permettant de justifier l'amélioration apportée par la STGCN. Une modélisation naïve de la dépendance spatiale du problème peut conduire à une conclusion hâtive quant à l'invalidité de l'approche STGCN. Pour surmonter cette difficulté, nous avons adopté une approche consistant à optimiser les hyperparamètres afin de construire une matrice d'adjacence. Plus récemment, des tentatives ont été faites pour incorporer l'apprentissage du graphe dans l'architecture de STGCN[10].

Bibliographie

[1] G Dreyfus et al. *Apprentissage statistique*. Springer, 2008. ISBN: 978-2-212-11464-5.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning, chapitre 5*. `http://www.deeplearningbook.org`. MIT Press, 2016.

[3] Yaguang Li et al. "Recurrent neural network Data-Driven traffic forecasting". In: (2018).

[4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning, chapitre 9*. `http://www.deeplearningbook.org`. MIT Press, 2016.

[5] Max Welling Thomas N. Kipf. "Semi-Supervised Classification with Graph Convolutional Networks". In: (2017).

[6] Didier Delignières. "Séries temporelles – Modèles ARIMA". In: (Mars 2020).

[7] Mitchell Lyons. *generalised additive models (GAMs)*. URL: `https://environmentalcomputing.net/statistics/gams/`.

[8] Zhanxing Zhu Bing Yu Haoteng Yin. "Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting". In: (juillet 2018).

[9] Argyris Kalogeratos et al. "Learning Laplacian Matrix from Graph Signals with Sparse Spectral Representation". In: (2021).

[10] Yuanbo Xu et al. "Dynamic traffic correlations based spatio-temporal graph convolutional network for urban traffic prediction". In: (2022).