

ISEN Yncréa Ouest

Projet FISA — Intelligence Artificielle

**Prédiction de la Présence des Employés
via Deep Learning**

Smart Workplace — Anticipation de l'occupation des bureaux

Auteurs

Ewan BARRE Baptiste LE-BAIL

Année universitaire 2025–2026

Table des matières

1	Introduction	2
1.1	Contexte : le Smart Workplace	2
1.2	Objectifs	2
2	Analyse des données	3
2.1	Description du dataset	3
2.2	Analyse exploratoire	3
2.2.1	Évolution temporelle	3
2.2.2	Présence par jour de la semaine	4
2.2.3	Corrélations	4
2.2.4	Autocorrélation de la série	5
3	Méthodologie	6
3.1	Découpage temporel des données	6
3.2	Stratégie de prédiction autorégressive	6
3.3	Présentation des modèles	6
3.3.1	XGBoost (baseline)	6
3.3.2	Réseaux récurrents (RNN, LSTM, GRU)	6
3.3.3	Architectures Transformer	6
3.4	Métriques d'évaluation	7
4	Résultats	8
4.1	Tableau comparatif des métriques	8
4.2	Résultats par modèle	8
4.3	Analyse détaillée par modèle	9
4.4	Comparaisons globales	10
5	Conclusion	12
5.1	Meilleure approche	12
5.2	Limites	12
5.3	Perspectives	12

1 Introduction

1.1 Contexte : le Smart Workplace

L'essor du travail hybride a profondément transformé l'organisation des espaces de bureau. Dans un contexte de *Smart Workplace*, les entreprises cherchent à optimiser l'utilisation de leurs surfaces : salles de réunion, postes de travail, espaces communs. Anticiper le nombre d'employés présents chaque jour est un enjeu stratégique pour réduire les coûts énergétiques et améliorer le confort des occupants.

Ce projet vise à prédire l'affluence journalière pour la **semaine suivante** (5 jours ouvrés) à partir de 250 jours d'historique, en comparant plusieurs familles de modèles : réseaux récurrents (RNN, LSTM, GRU), gradient boosting (XGBoost), et architectures Transformer (PatchTST, TimeXer, iTransformer, VanillaTransformer).

1.2 Objectifs

- Comparer 8 modèles sur les mêmes métriques (MAE, MAPE, RMSE, R^2) pour une évaluation équitable.
- Identifier la meilleure approche pour une prévision courte-portée ($h = 5$ jours) avec un dataset de taille limitée (250 jours ouvrés).
- Analyser les forces et faiblesses de chaque famille de modèles.

2 Analyse des données

2.1 Description du dataset

Le dataset `df_venues_final.csv` contient **250 jours ouvrés** (septembre 2022 – septembre 2023). Après nettoyage, 15 features sont retenues :

Table 1 – Variables du dataset (après nettoyage)

Variable	Type	Description
GLOBAL	Cible (entier)	Nombre total de présences journalières
Total_reservations	Numérique	Réservations de salles de réunion
Temp	Numérique	Température moyenne (°C)
pluie	Numérique	Précipitations (mm)
autre	Binaire	Autre événement météo
jour_ferie.	Binaire	Jour férié
pont.conge.	Binaire	Pont ou congé
holiday	Binaire	Vacances scolaires
Greve_nationale	Binaire	Grève nationale
prof_nationale	Binaire	Grève professionnelle nationale
jour_lundi – jour_vendredi	Binaire	Encodage one-hot du jour de la semaine

2.2 Analyse exploratoire

2.2.1 Évolution temporelle

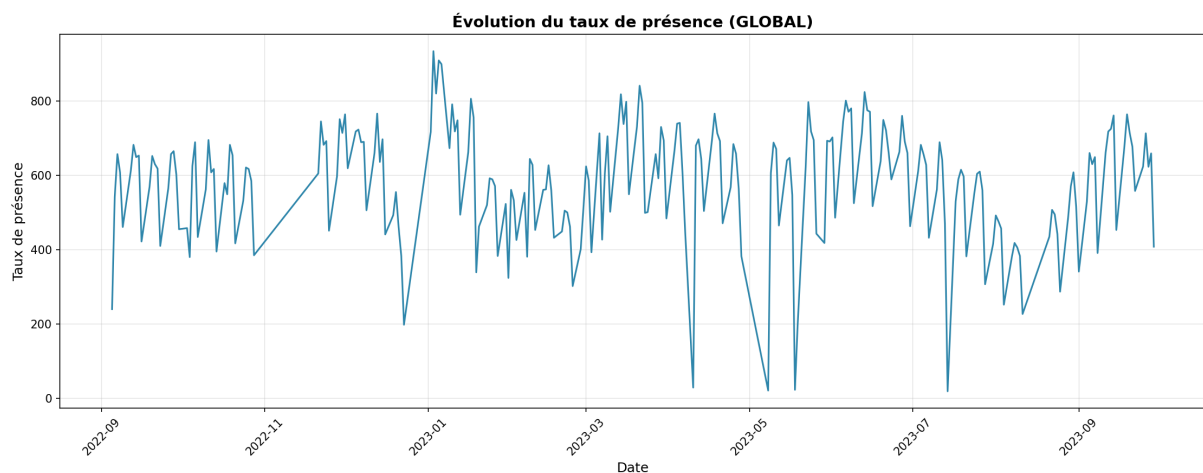


Figure 1 – Évolution journalière de GLOBAL sur 250 jours. On observe une saisonnalité hebdomadaire marquée et des creux importants lors des vacances.

La série présente une **saisonnalité hebdomadaire** nette (les vendredis sont systématiquement plus creux) et des chutes lors des vacances scolaires (été 2023, fin décembre 2022). La moyenne est de 604 venues/jour, avec un écart-type de 127 venues.

2.2.2 Présence par jour de la semaine

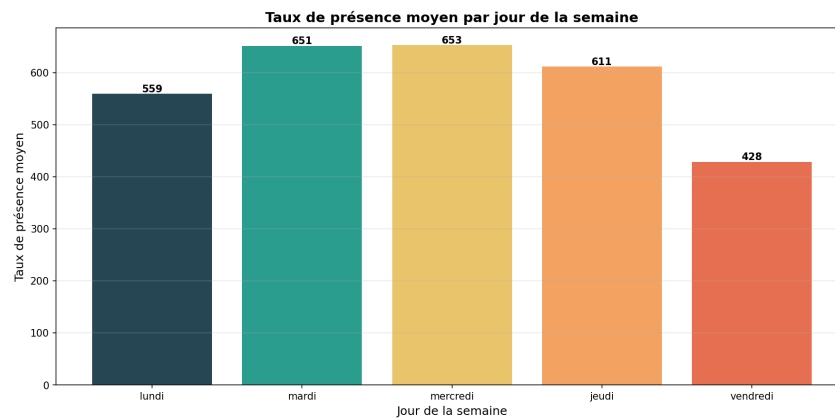


Figure 2 – Présence moyenne par jour. Le vendredi affiche ~30% de présence en moins que le mercredi.

2.2.3 Corrélations

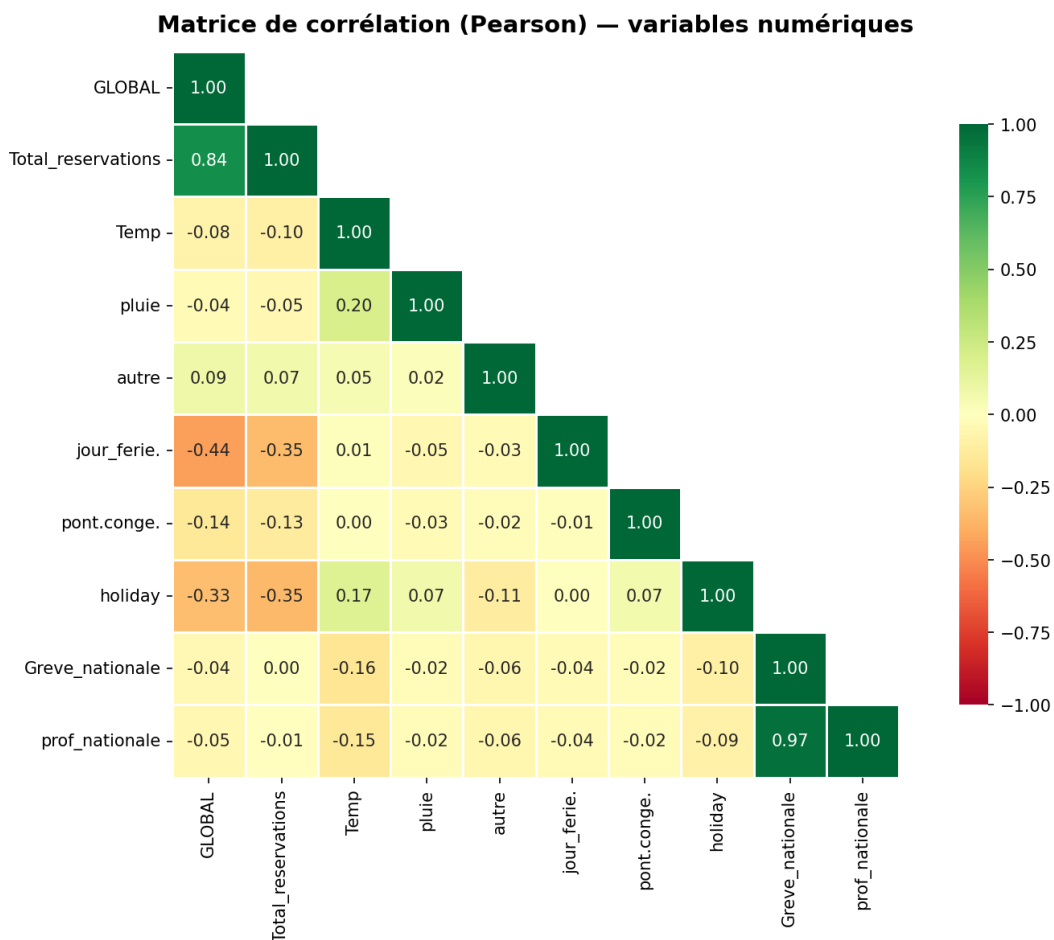


Figure 3 – Matrice de corrélation de Pearson. **Total_reservations** est la variable la plus corrélée avec **GLOBAL** ($r = 0.89$). La température présente une corrélation négative modérée ($r \approx -0.30$).

Total_reservations ($r = 0.89$) est le meilleur prédicteur de l'affluence : plus les salles sont réservées, plus les gens viennent. On peut également noter la corrélation négative modérée de la température ($r \approx -0.30$) : les jours plus chauds sont légèrement moins fréquentés, probablement en raison de la tentation de rester à l'extérieur.

2.2.4 Autocorrélation de la série

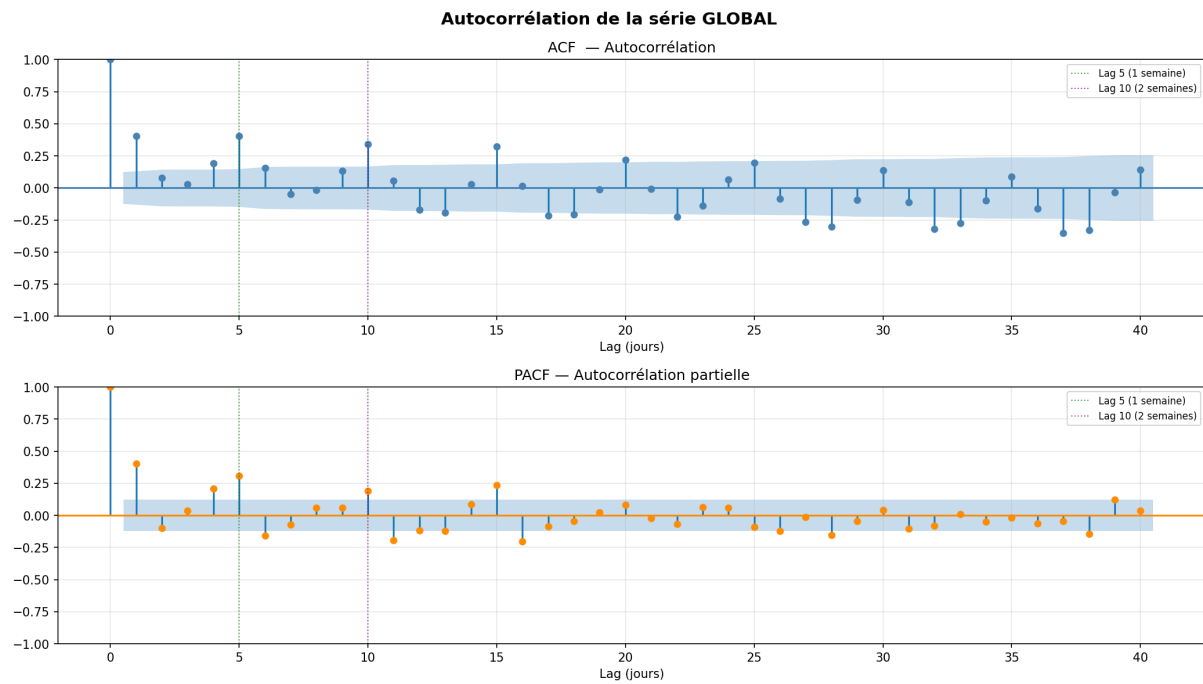


Figure 4 – ACF et PACF de la série GLOBAL. Pics significatifs aux lags 1, 5 et 10, confirmant la dépendance hebdomadaire. La PACF chute dans l’intervalle de confiance au-delà du lag 5, justifiant `WINDOW = 5`.

L’ACF révèle une mémoire hebdomadaire : la présence d’un lundi ressemble à celle du lundi précédent (lag 5). La PACF montre que les dépendances *directes* au-delà de 5 jours sont marginales, ce qui justifie directement le choix de `WINDOW = 5` pour tous les modèles.

3 Méthodologie

3.1 Découpage temporel des données

Un découpage **strictement chronologique** est imposé pour éviter toute fuite d'information (*data leakage*) :

$$\underbrace{\overbrace{\text{S36/2022} - \text{S34/2023}}^{\text{Train (225 j)}} \mid \overbrace{\text{S35-S38/2023}}^{\text{Val (20 j)}} \mid \overbrace{\text{S39/2023}}^{\text{Test (5 j)}}}_{250 \text{ jours ouvrés}} \quad (1)$$

- **Train (225 j)** : apprentissage des poids du modèle.
- **Val (20 j)** : *early stopping* — les poids minimisant la val loss sont conservés. Évaluation directe 1-pas en avant.
- **Test (5 j)** : évaluation finale, **jamais vu** pendant l'entraînement. Prédiction autorégressive sur 5 jours.

Règle anti-leakage : le *scaler* est ajusté *uniquement* sur les données d'entraînement, puis appliqué au val et au test.

3.2 Stratégie de prédiction autorégressive

Pour prédire les 5 jours du test, tous les modèles utilisent une **fenêtre glissante autorégressive** : la prédiction du jour t est injectée dans la fenêtre pour prédire $t + 1$, en utilisant les vraies valeurs des features exogènes (météo, calendrier) connues à l'avance. Les erreurs s'accumulent au fil des jours.

3.3 Présentation des modèles

3.3.1 XGBoost (baseline)

Modèle de *gradient boosting* sur arbres de décision. La fenêtre de 5 jours est **aplatie** en un vecteur de $5 \times 15 = 75$ features. Pas de normalisation requise. Interprétable via l'importance des features. Hyperparamètres : 300 estimateurs, profondeur max = 4, LR = 0.05, régularisation L1+L2.

3.3.2 Réseaux récurrents (RNN, LSTM, GRU)

Les trois architectures traitent la séquence temporelle pas à pas :

- **RNN** : $h_t = \tanh(W_h h_{t-1} + W_x x_t)$. Architecture minimale, souffre théoriquement du *vanishing gradient* mais efficace sur WINDOW=5.
- **LSTM** : cellule de contexte $c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$ avec 3 portes (oubli, entrée, sortie). Plus expressif.
- **GRU** : simplifie le LSTM (2 portes), moins de paramètres.

Architecture commune : 2 couches, taille cachée = 64, dropout = 0.2, 54 081 paramètres. Entraînement : Adam (LR = 10^{-3}), ReduceLROnPlateau, early stopping (patience = 40).

3.3.3 Architectures Transformer

Remplacent la récurrence par un mécanisme d'**attention multi-tête** :

- **VanillaTransformer** : token = un jour entier (endo + exo). Auto-attention sur 5 tokens.
- **PatchTST** : token = un jour de la série endogène seulement.
- **TimeXer** : PatchTST + cross-attention $\text{Attn}(\mathbf{Q}_{\text{endo}}, \mathbf{K}_{\text{exo}}, \mathbf{V}_{\text{exo}})$.

— **iTransformer** : token = une variable entière (attention inter-features).

Config. commune : $d_{\text{model}} = 64$, 4 têtes, 2 couches, dropout = 0.1. Entraînement : AdamW (LR = 3×10^{-4}), CosineAnnealingLR, early stopping (patience = 40).

3.4 Métriques d'évaluation

Table 2 – Métriques utilisées (calculées sur val et test)

Métrique	Formule	Optimum
MAE	$\frac{1}{n} \sum \hat{y}_i - y_i $	↓
MAPE	$\frac{100}{n} \sum \frac{ \hat{y}_i - y_i }{y_i}$	↓
RMSE	$\sqrt{\frac{1}{n} \sum (\hat{y}_i - y_i)^2}$	↓
R ²	$1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$	↑

4 Résultats

4.1 Tableau comparatif des métriques

Table 3 – Métriques — Validation (1-step) et Test (autorégressif 5 jours). **Gras** = meilleure valeur test.

Modèle	Val (1-step)				Test (autorégressif)			
	MAE	MAPE	RMSE	R ²	MAE	MAPE	RMSE	R ²
RNN	33.8	6.0%	45.4	0.851	34.8	6.7%	50.7	0.762
LSTM	34.2	6.2%	41.3	0.877	65.5	11.7%	75.4	0.474
GRU	32.1	5.9%	42.0	0.873	65.4	11.8%	73.9	0.495
XGBoost	58.4	9.7%	70.5	0.642	54.9	9.9%	60.5	0.662
VanillaTransf.	34.6	6.1%	44.2	0.859	62.4	11.5%	67.5	0.579
PatchTST	63.4	11.5%	75.9	0.585	44.0	8.4%	51.4	0.755
iTransformer	45.0	7.9%	54.0	0.790	60.0	10.7%	62.7	0.637
TimeXer	43.7	7.4%	53.3	0.795	39.0	6.9%	43.4	0.826

4.2 Résultats par modèle

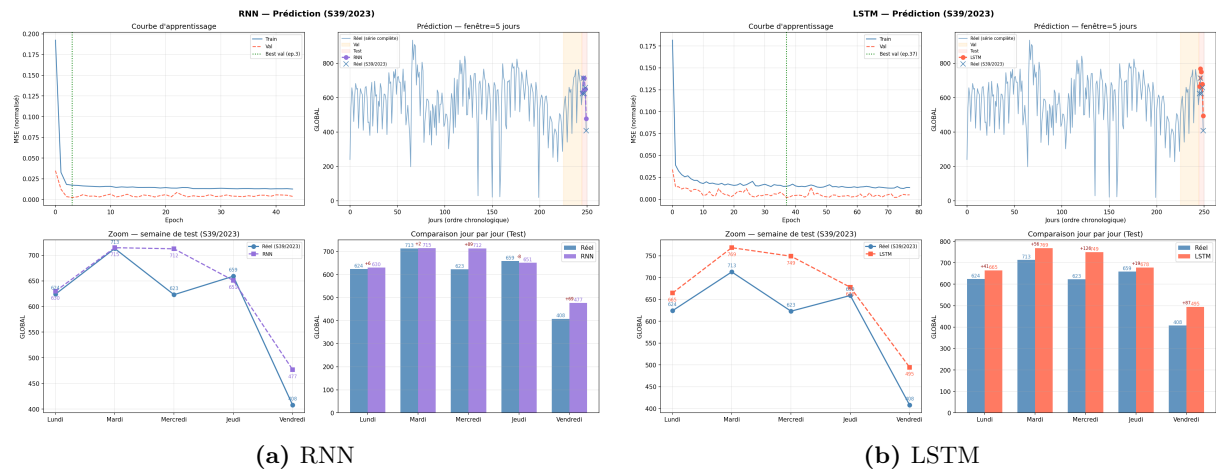


Figure 5 – RNN vs LSTM : courbe d'apprentissage, série complète, zoom et barres.

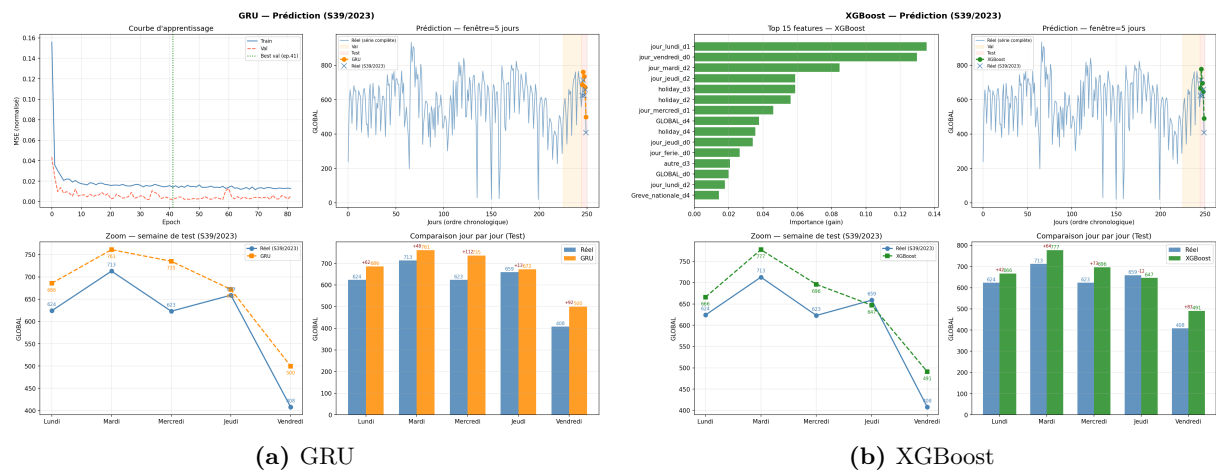


Figure 6 – GRU vs XGBoost : courbe d'apprentissage/importance des features, série complète, zoom et barres.

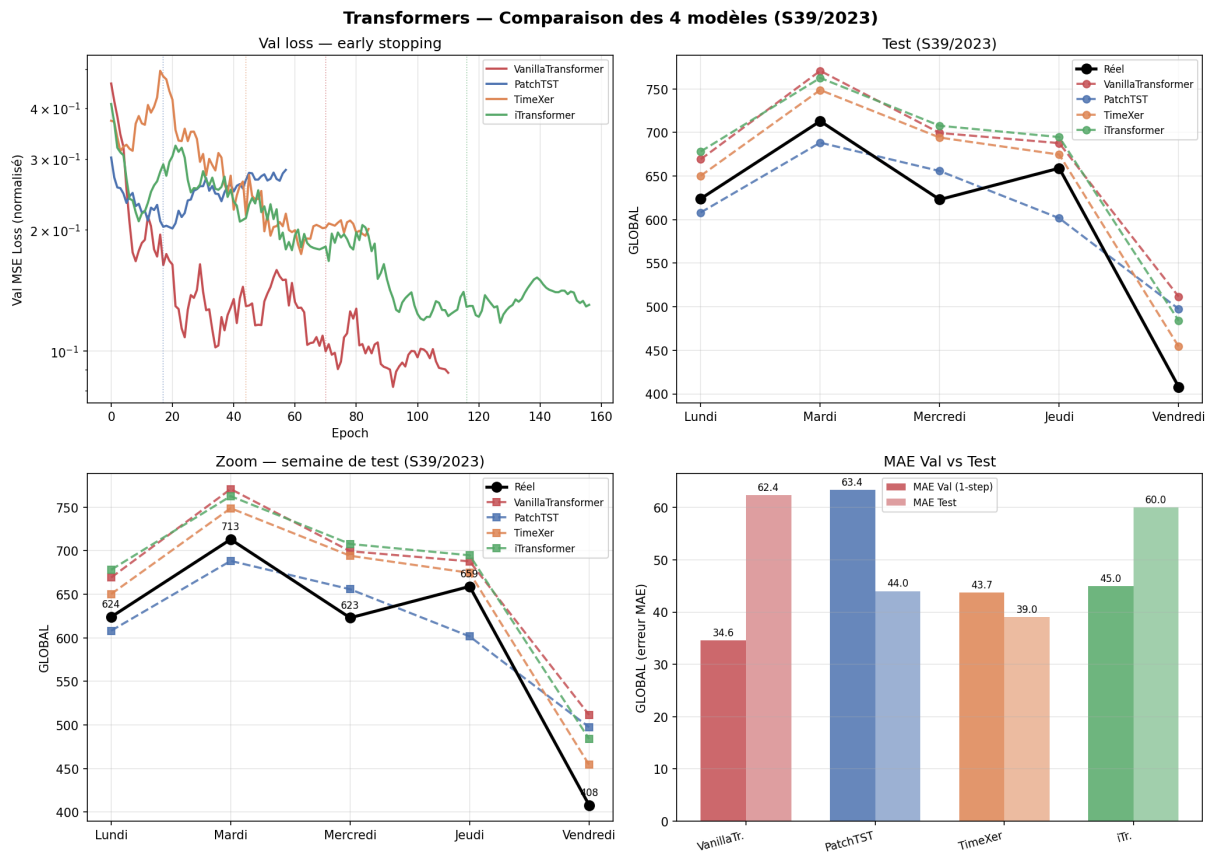


Figure 7 — Comparaison des 4 architectures Transformer sur la semaine de test S39/2023.

4.3 Analyse détaillée par modèle

RNN — 2^e meilleur modèle (test MAE = 34.8, $R^2 = 0.762$). Le RNN est la surprise de cette étude. Sa simplicité est un atout : avec moins de paramètres, il généralise mieux sur 225 jours d'entraînement. Sur une fenêtre de 5 jours seulement, le *vanishing gradient* théorique n'a pas le temps de se manifester. La cohérence val (MAE = 33.8) / test (MAE = 34.8) confirme l'absence d'overfitting.

LSTM — Overfitting marqué (test MAE = 65.5, $R^2 = 0.474$). Malgré d'excellentes métriques de validation (MAE = 34.2, $R^2 = 0.877$), le LSTM s'effondre en test. Ses 3 portes et sa cellule de mémoire lui permettent de mémoriser des patterns très spécifiques du train, mais échouent à généraliser sur S39/2023. L'écart val/test est le plus marqué de tous les modèles — signe classique d'overfitting.

GRU — Même diagnostic que LSTM (test MAE = 65.4, $R^2 = 0.495$). Le GRU présente un profil quasi-identique au LSTM : excellent en val (MAE = 32.1, meilleur de tous), mais effondrement en test. Malgré la fusion des portes et moins de paramètres, sa capacité reste trop grande pour ce dataset. LSTM et GRU sont davantage adaptés à des séries longues (plusieurs années) où leur mémoire à long terme est utile.

XGBoost — Baseline stable (test MAE = 54.9, $R^2 = 0.662$). XGBoost offre une performance intermédiaire mais surtout **stable** : val MAE = 58.4 et test MAE = 54.9 (meilleure cohérence après RNN). En aplatissant la fenêtre temporelle, il perd l'ordre séquentiel, mais l'importance des features révèle que **Total_reservations** et l'encodage du jour de la semaine sont les meilleurs prédicteurs. Recommandé en production pour sa robustesse et son interprétabilité.

VanillaTransformer — Attention insuffisante (test MAE = 62.4, $R^2 = 0.579$). Sur seulement 5 tokens (jours), le mécanisme d’auto-attention ne trouve pas assez de structure temporelle à exploiter. La complexité (103 233 paramètres) n’est pas justifiée par la taille du dataset, résultant en des performances inférieures à XGBoost.

PatchTST — Bon compromis (test MAE = 44.0, $R^2 = 0.755$). PatchTST se classe en 3^e position malgré une validation médiocre (MAE = 63.4). En traitant chaque jour de la série endogène comme un token distinct, il apprend des représentations temporelles plus généralisables que le VanillaTransformer. L’inversion val/test est positive : le modèle généralise mieux qu’il ne valide.

iTransformer — Attention inter-variables inadaptée (test MAE = 60.0, $R^2 = 0.637$). L’iTransformer traite chaque variable comme un token (attention entre features). Cette approche est pertinente sur des datasets multivariés denses, mais avec seulement 15 features et 5 jours, les corrélations inter-variables sont mieux capturées par la matrice de corrélation que par un mécanisme d’attention dédié.

TimeXer — Meilleur modèle (test MAE = 39.0, $R^2 = 0.826$). TimeXer domine grâce à sa **cross-attention** endo/exo : les tokens de la série endogène interrogent directement les features exogènes, apprenant comment la météo, les grèves et les fêtes modulent l’affluence. Là où PatchTST traite les deux séparément, TimeXer crée un dialogue explicite entre les deux flux. La cohérence val/test (43.7 → 39.0) est remarquable et confirme une bonne généralisation.

4.4 Comparaisons globales

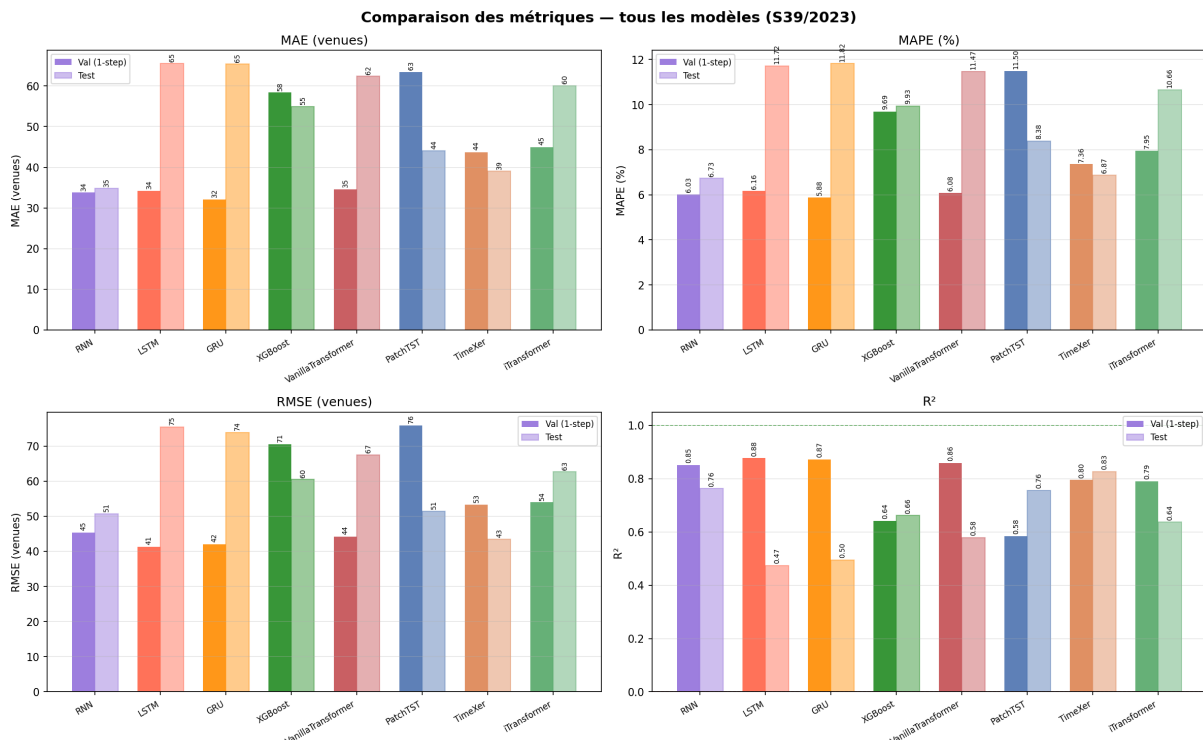


Figure 8 – Barplots MAE, MAPE, RMSE, R^2 pour tous les modèles (barres opaques = val, transparentes = test).

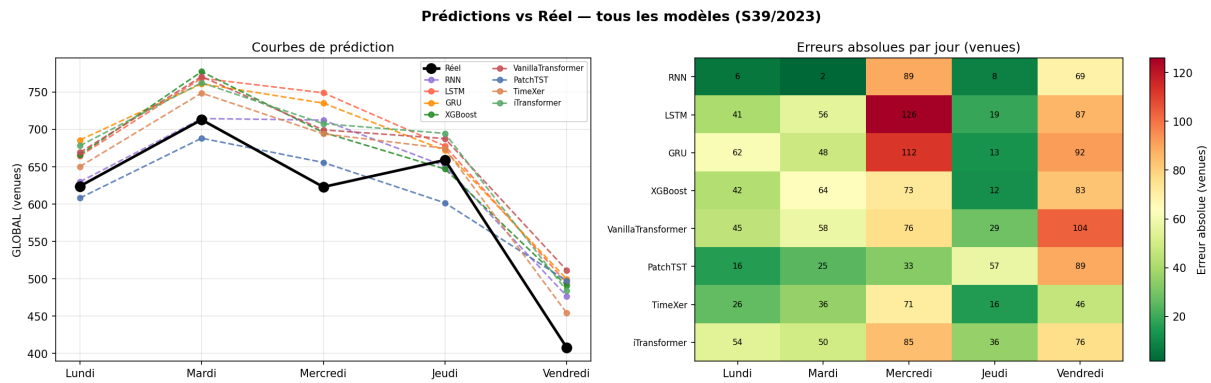


Figure 9 – Prédictions sur la semaine de test S39/2023 (gauche) et heatmap des erreurs absolues par jour et par modèle (droite).

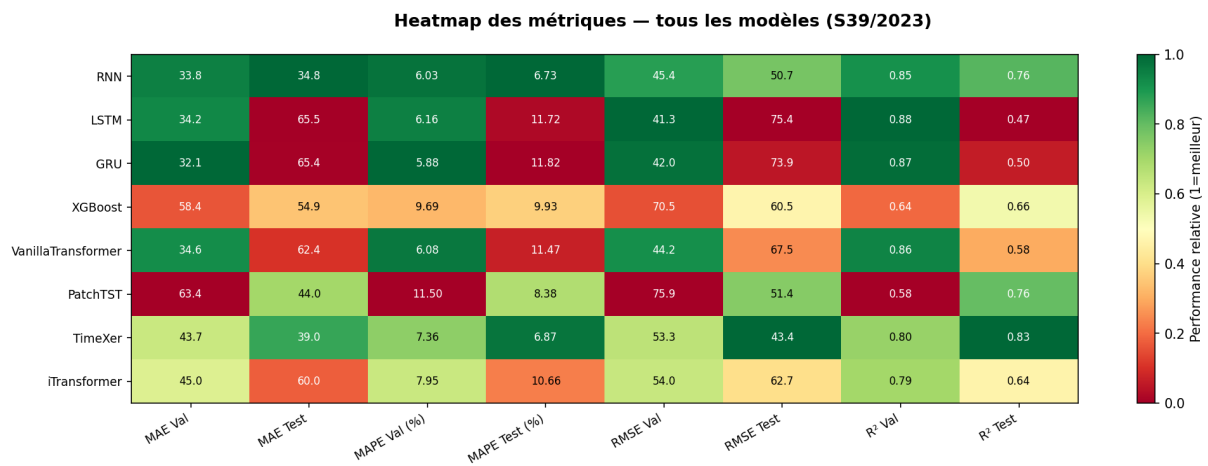


Figure 10 – Heatmap des métriques normalisées : vert = meilleur, rouge = moins bon.

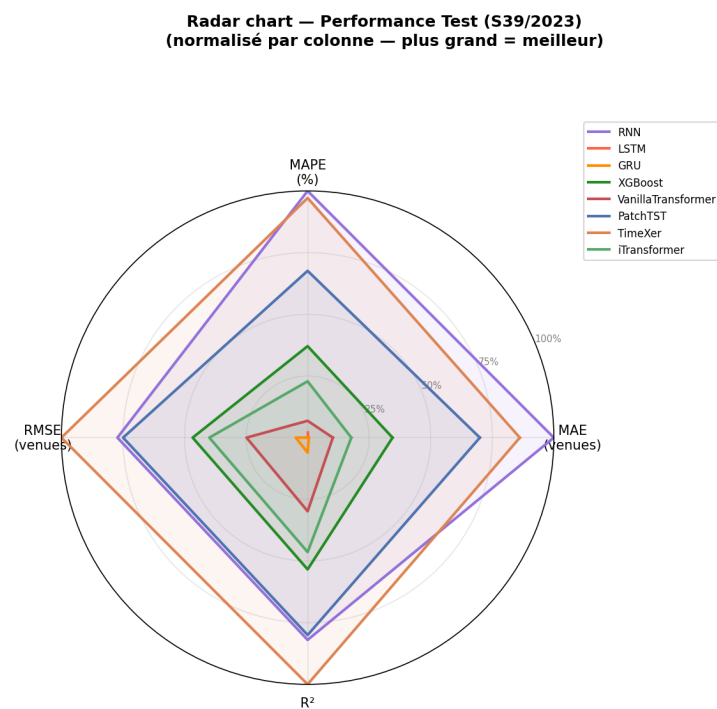


Figure 11 – Radar chart des performances test normalisées. TimeXer domine sur les quatre axes simultanément.

5 Conclusion

5.1 Meilleure approche

TimeXer est le modèle le plus performant ($MAE = 39.0$, $MAPE = 6.9\%$, $R^2 = 0.826$). Sa cross-attention endo/exo capture efficacement l'influence des facteurs contextuels sur la présence, ce que les architectures récurrentes classiques peinent à faire.

RNN, malgré sa simplicité, se classe en deuxième position grâce à sa meilleure généralisation sur un dataset limité. **XGBoost** reste recommandé en production pour sa robustesse, sa rapidité d'inférence et son interprétabilité.

5.2 Limites

- **Dataset limité** : 250 jours est insuffisant pour des architectures profondes. 2–3 ans de données amélioreraient significativement les Transformers.
- **Une seule semaine de test** : S39/2023 est une semaine ordinaire. Les performances sur des semaines atypiques (grèves, vacances) restent inconnues.
- **Accumulation d'erreurs** : en mode autorégressif, les erreurs se propagent sur les 5 jours, rendant le vendredi plus difficile à prédire.
- **Météo parfaite** : les valeurs météo réelles sont utilisées, constituant un avantage artificiel par rapport à un scénario de production réel.
- **Overfitting LSTM/GRU** : malgré l'early stopping et le dropout, la val loss ne représente pas suffisamment la difficulté du test.

5.3 Perspectives

Pour améliorer ce système en production : utilisation de prévisions météo réelles, entraînement sur un historique plus long, intégration de badges d'accès, et ré-entraînement hebdomadaire automatique au fil des nouvelles données.