De ARIMA aux Transformers

Analyse de Séries Temporelles et NLP Avancé

Réalisée par :

- **KHATTABI Mohammed**
- **OUAHAB Abderrahim**
- **ZOURDAN Youssef**

2025/03/07

Encadré par :

- **M. Abdelwahab Naji**

Sommaire

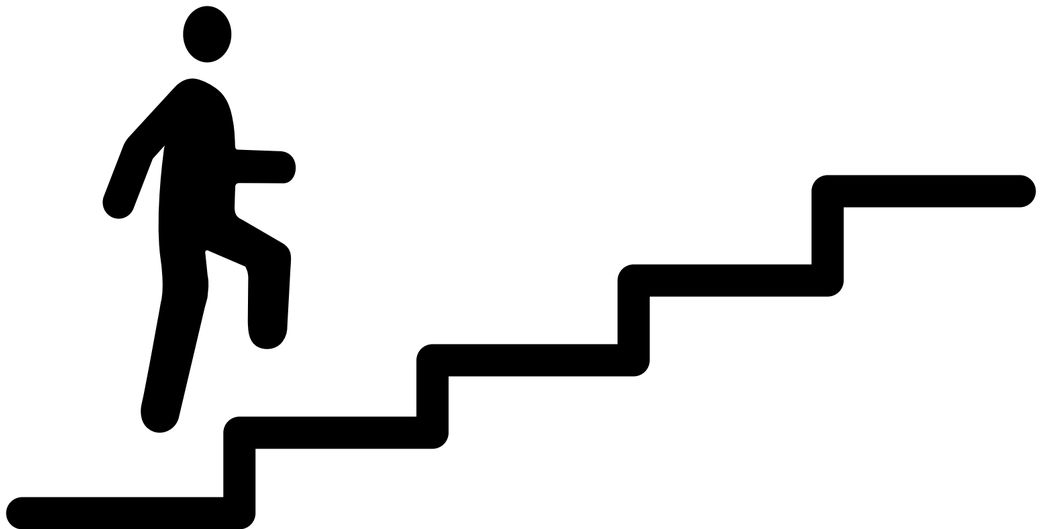
1. Time Series : Objectifs et Défis
2. Premiers Pas en Time Series
3. Découverte des Jeux de Données
4. Deep Learning : ANN, RNN, et LSTM
5. Analyse de Séries Temporelles et Modèles Avancés
6. NLP et Techniques Avancées
7. Perspectives et Conclusions

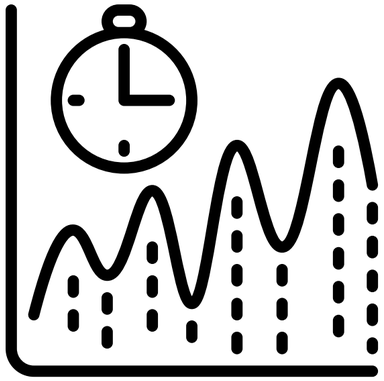
1. Time Series : Objectifs et Défis

- **Compréhension des séries temporelles**
- **Préparation des données**
- **Visualisation et analyse**
- **Modélisation et prévision**
- **Optimisation et évaluation**

- **Dépendance temporelle et identification des tendances et saisonnalités.**
- **Stationnarité et transformations pour stabiliser la série.**
- **Choix du bon modèle**
- **Gestion des erreurs résiduelles et surajustement.**
- **Problèmes liés aux RNN/LSTM : vanishing gradient, overfitting, convergence.**

2. Premiers Pas en Time Series

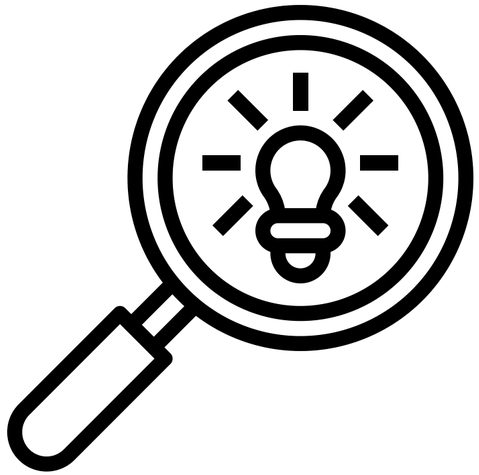


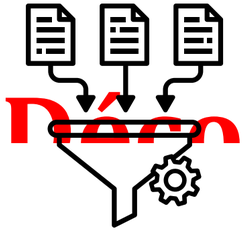


Données organisées chronologiquement.

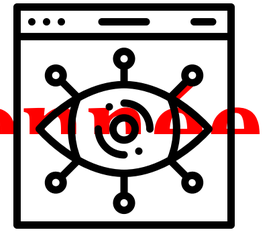
- **Collecte et visualisation des données.**
- **Identification des tendances et saisonnalités.**
- **Modélisation et prévision des données.**

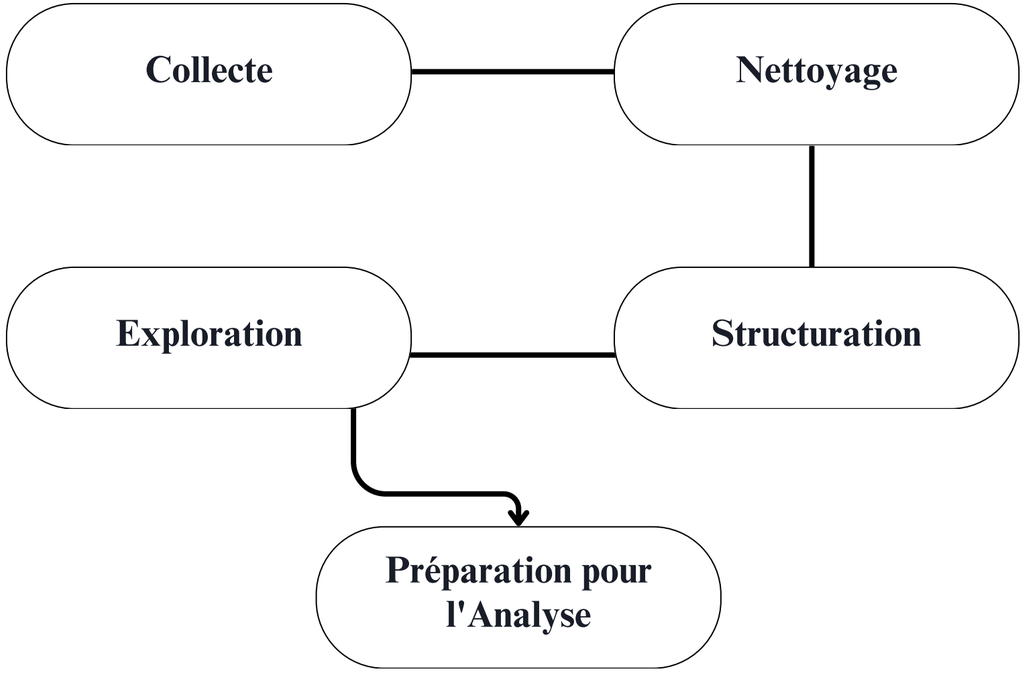
3. Découverte des Jeux de Données

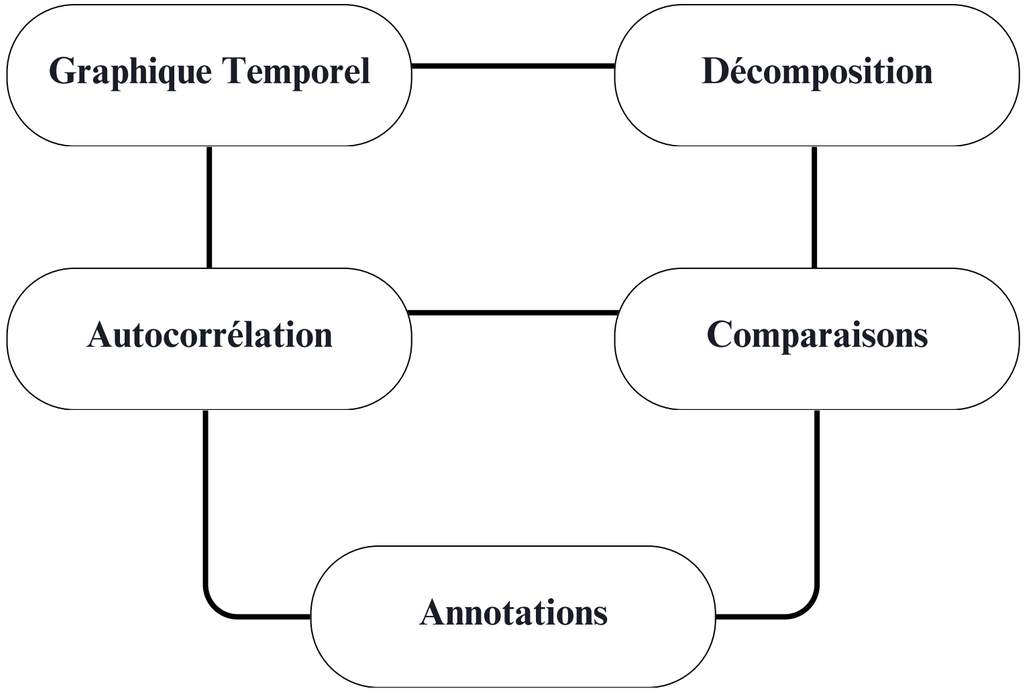


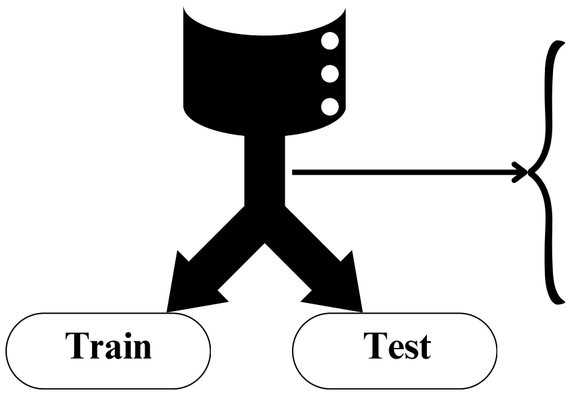


3. Découverte des Jeux de Données





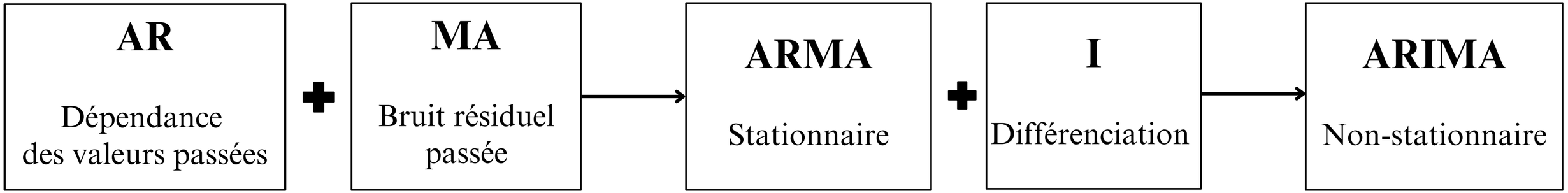


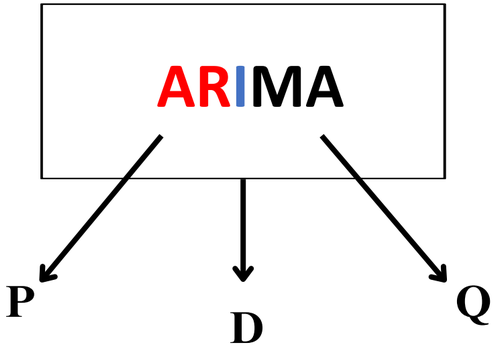


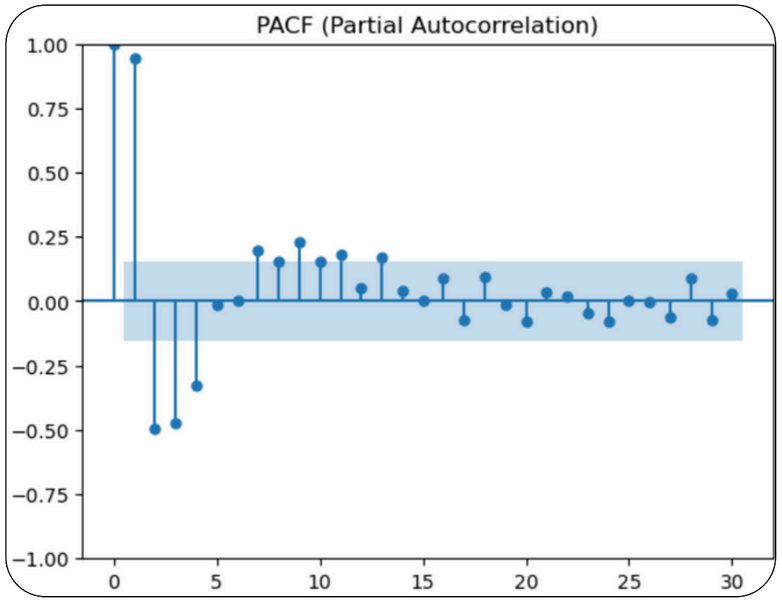
- **Séparation temporelle**
- **Ratio classique**

- **Eviter les fuites de données**
- **Taille du test**

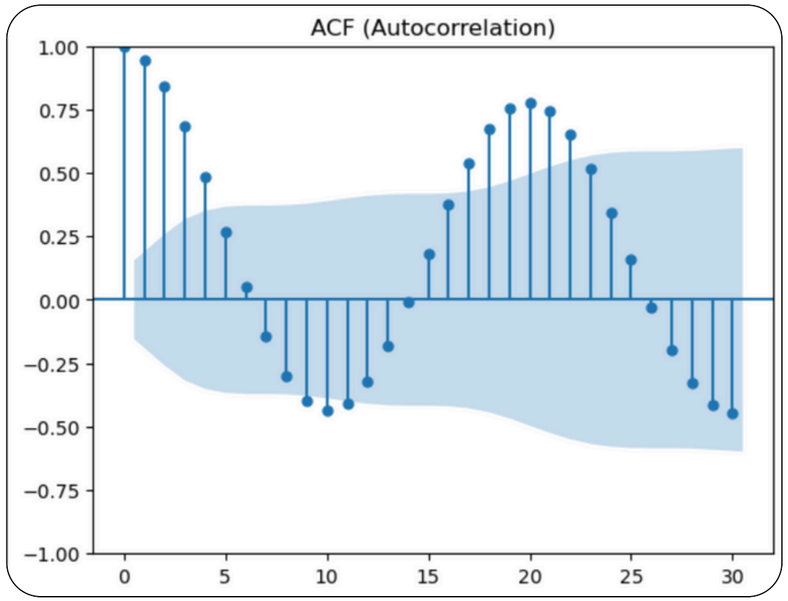
```python
# Charger les données
data = pd.read_csv('data.csv', parse_dates=['date'], index_col='date')

# Diviser les données
train_size = int(len(data) * 0.8)  # 80% pour l'entraînement
train, test = data[:train_size], data[train_size:]
```


AR
Dépendance des valeurs passées
MA
Bruit résiduel passée
ARMA
Stationnaire
I
Différenciation
ARIMA
Non-stationnaire





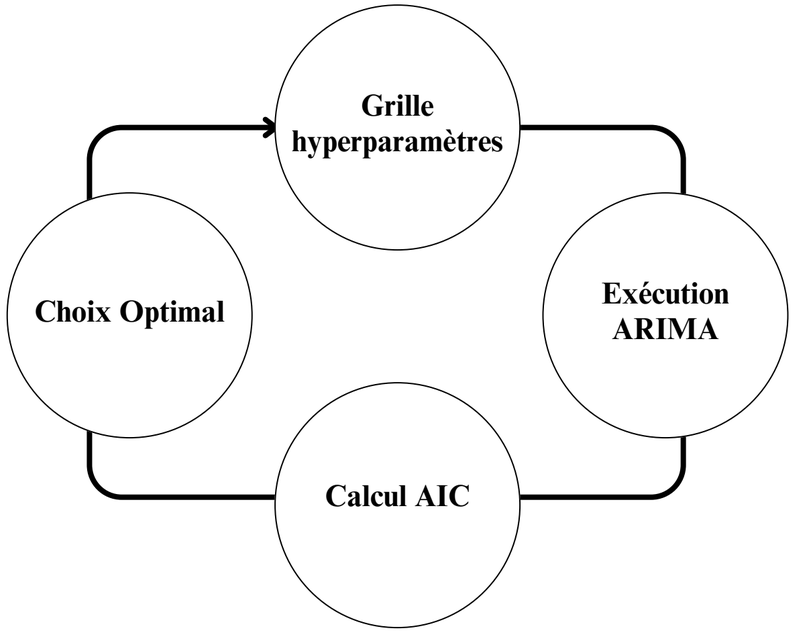
corrélation directe entre une série et un lag spécifique



corrélation entre une série temporelle et ses lags

Grid Search

Trouver les meilleures valeurs des hyperparamètres



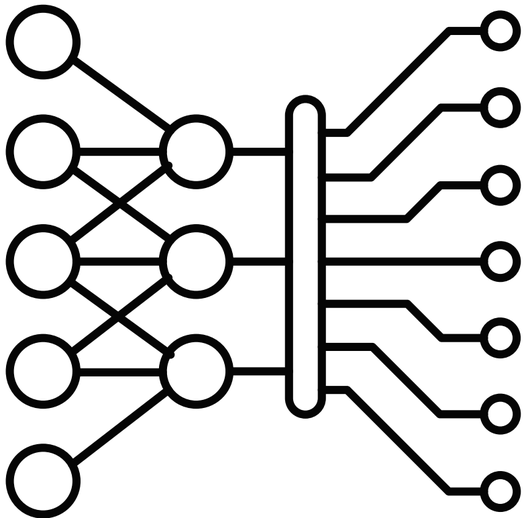

```
p=2, d=0, q=0 -> AIC=937.2276302268851
...
p=3, d=2, q=1 -> AIC=912.3296883733982
p=3, d=2, q=2 -> AIC=915.533357524179
p=3, d=2, q=3 -> AIC=911.0013625037545
Meilleurs paramètres ARIMA: (0, 2, 3) avec AIC: 907.1687054410694
```

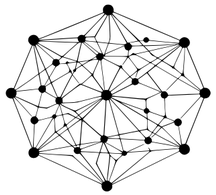
Évaluation des modèles

mesurer leur performance et d'identifier les améliorations nécessaires

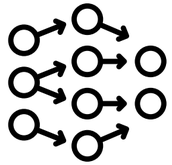
```
Residual Forecast Error: -1.3208072068007628
Forecast Bias: -1.3208072068007628
Mean Absolute Error: 47.09943294486856
Mean Squared Error: 3674.176001779176
Root Mean Squared Error: 60.614981661130415
```

4. Deep Learning : ANN, RNN et LSTM

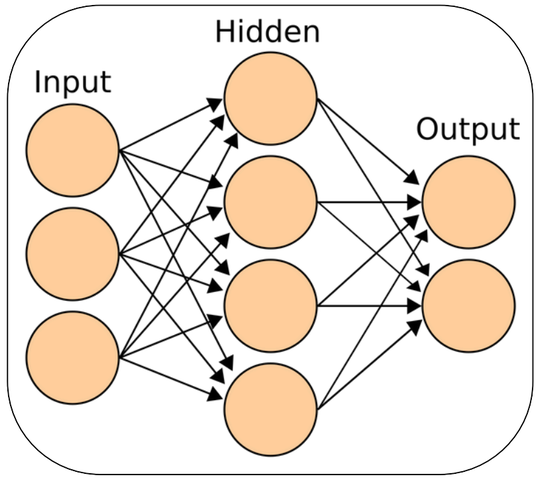




Deep Learning

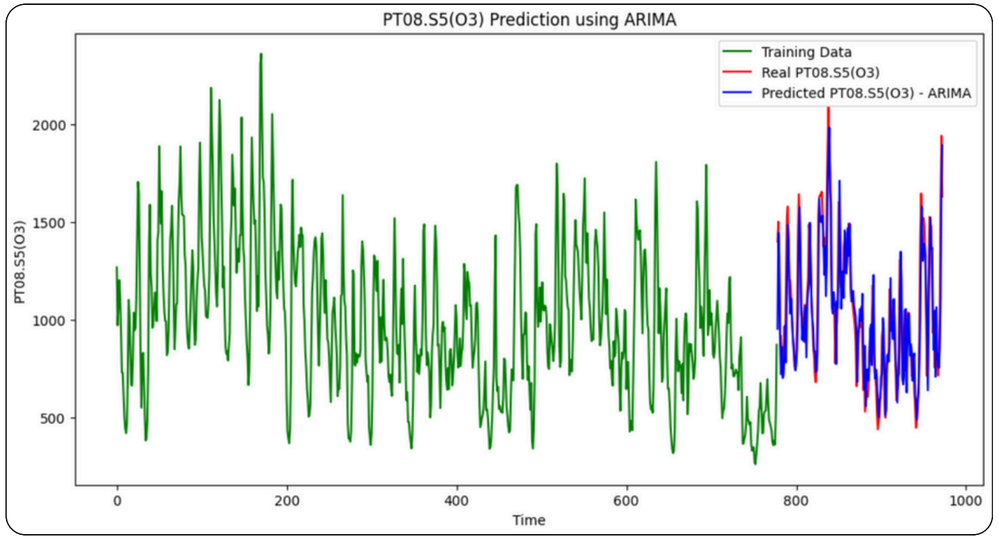


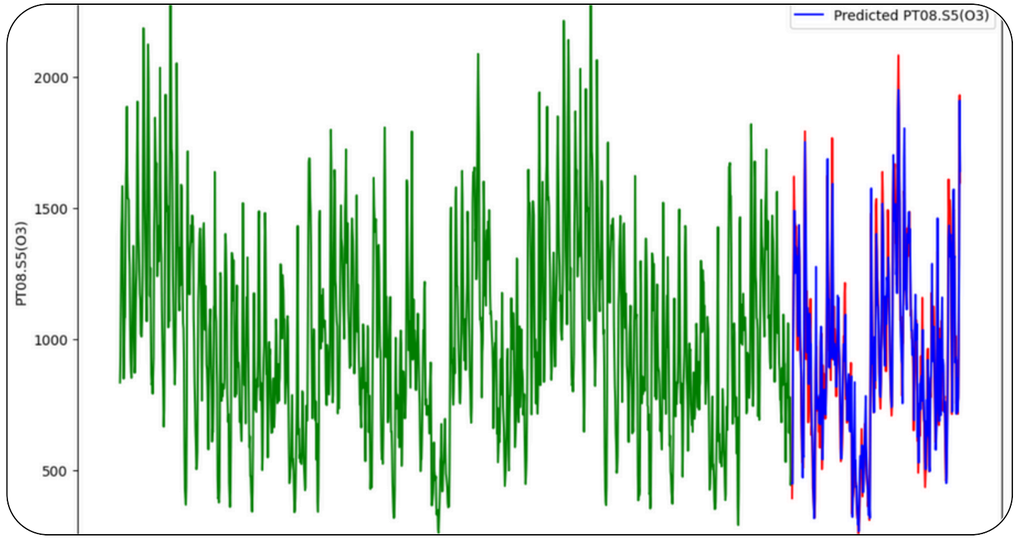
- Le Deep Learning est une approche basée sur les réseaux de neurones pour capturer des relations complexes dans les données.
- Contrairement a ARIMA : il permet de modéliser des tendances non linéaires.



Avantages

- Capture des dépendances à long terme.
- Gestion des tendances et saisonnalités.

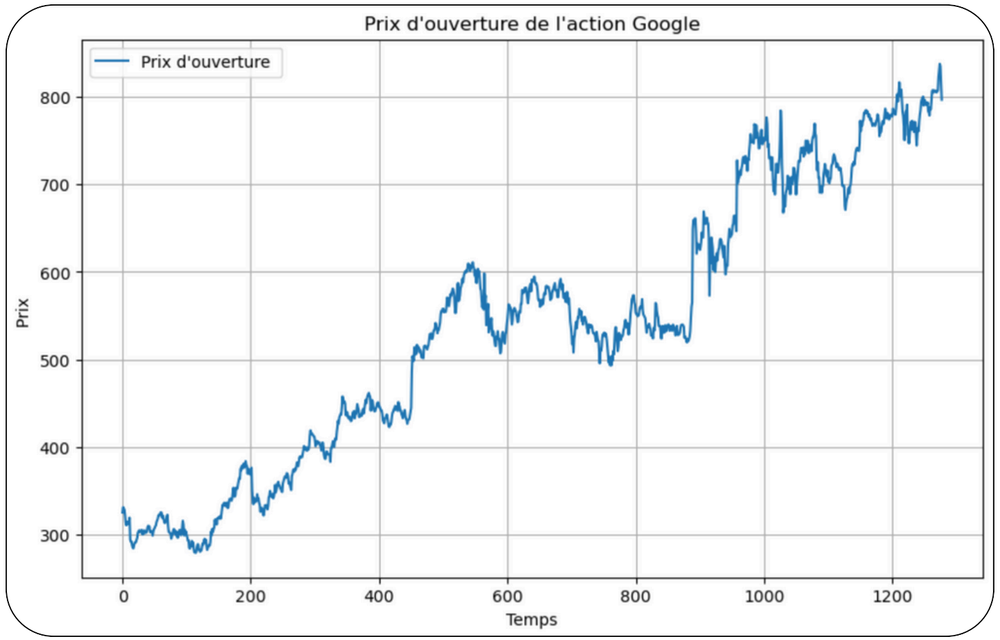


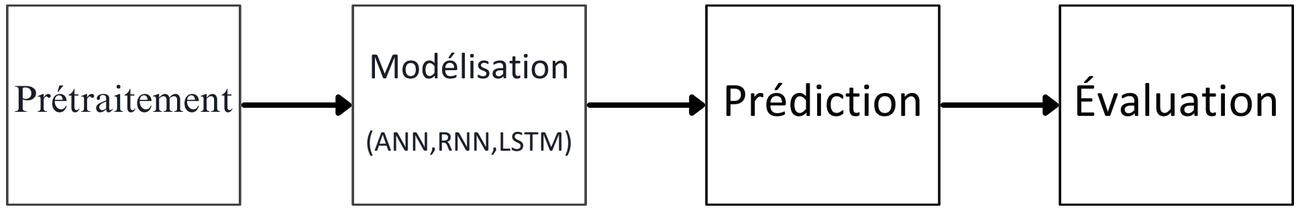


Model	MAE	MSE	RMSE	R2
ARIMA	123.409645	24965.311545	158.004150	0.781957
LSTM	103.505379	17977.183253	134.079019	0.860822

Google Stock Analysis

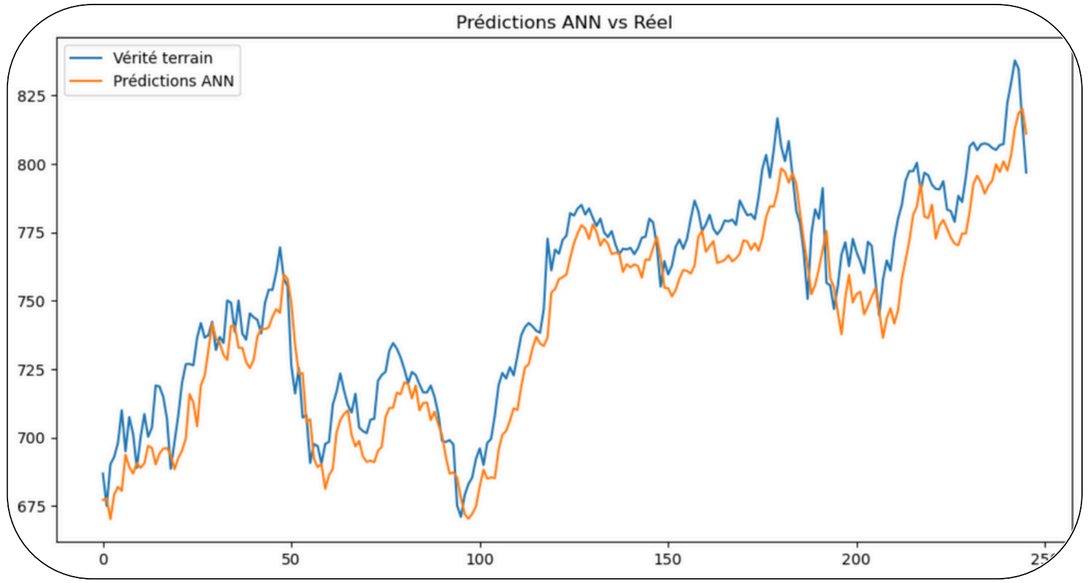
- **Objectif** : Prédire l’évolution du cours des actions Google en utilisant ANN, RNN et LSTM.
- **Données utilisées** : Historique des prix des actions Google “Google Stock Price”.



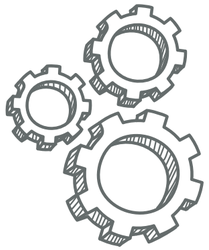


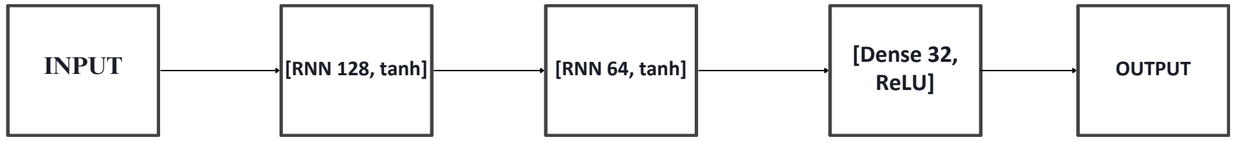
Modélisation ANN

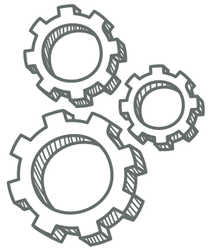
- Optimisation avec Adam et fonction de perte MSE.
- 100 epochs, batch size=32, validation split=20%.



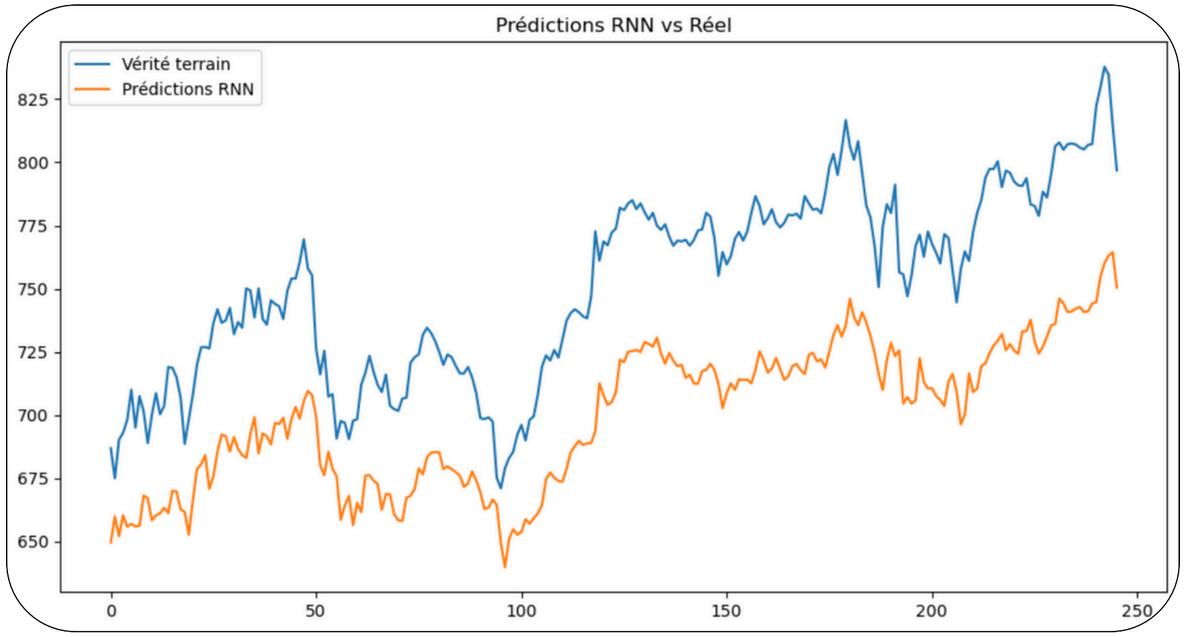
Modélisation RNN



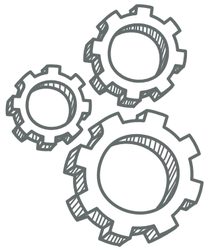


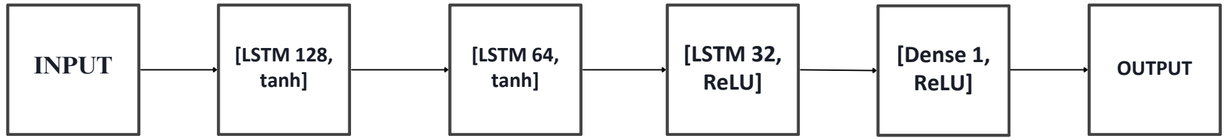


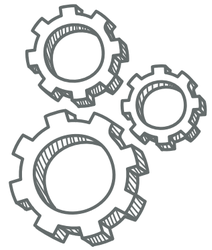
- Optimisation avec Adam et fonction de perte MSE.
- 100 epochs, batch size=32, validation split=20%.



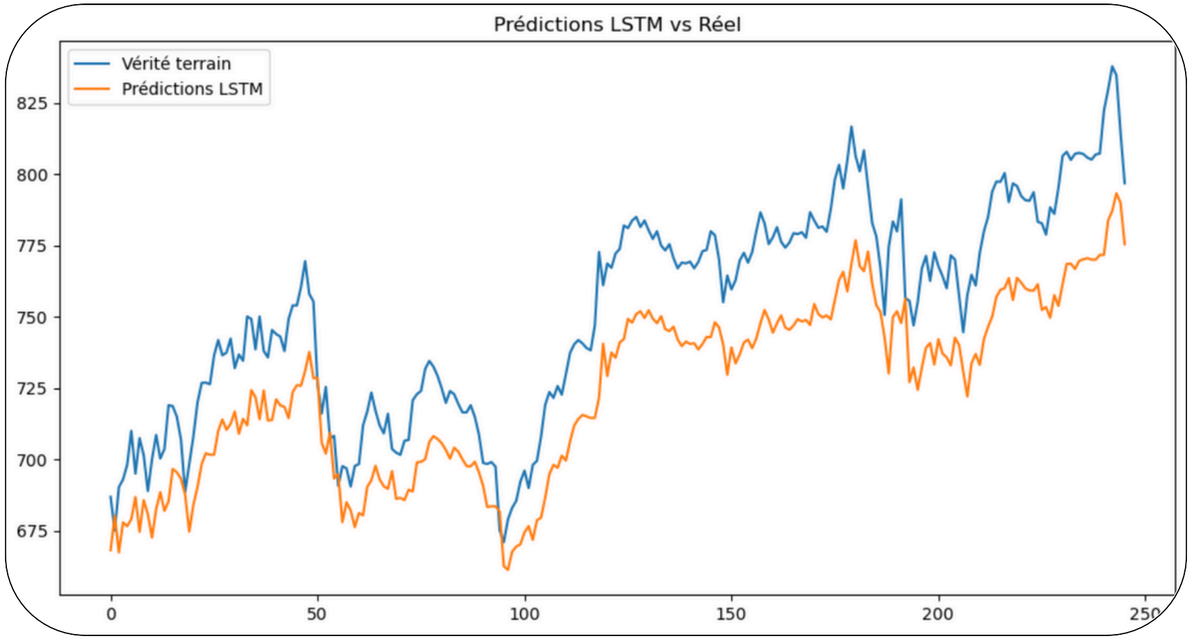
Modélisation LSTM







- Optimisation avec Adam et fonction de perte MSE.
- 100 epochs, batch size=32, validation split=20%.



5.Analyse de Séries Temporelles et Modèles Avancés

Présentation du Dataset

Variables clés : Date, PT08.S5(O3)

Décomposition

- Tendance
- Saisonnalité
- Résidus

Date	PT08.S5(O3)
2004-03-10	1267.50
2004-03-10	972.25
2004-03-10	1074.00
2004-03-10	1203.25
2004-03-10	1110.00
...	...
2005-04-04	1728.50
2005-04-04	1269.00
2005-04-04	1092.00
2005-04-04	769.75
2005-04-04	816.00

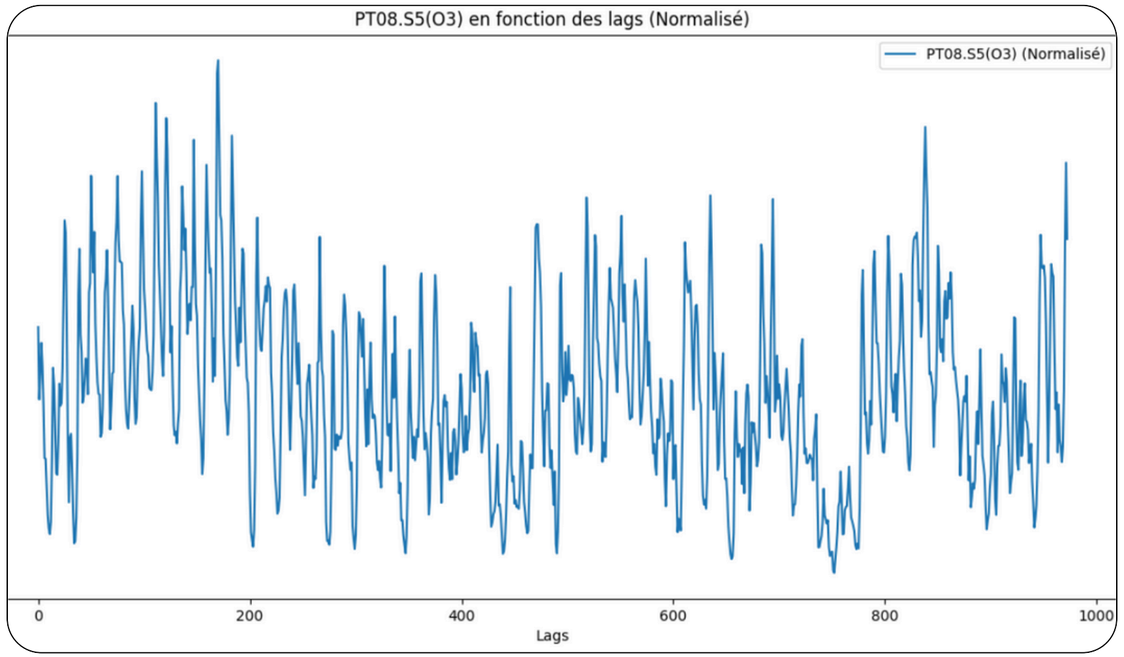
9357 rows × 1 columns

Nettoyage des Données

- Anomalies
- Normalisation

PT08.S5(O3) en fonction des Lags

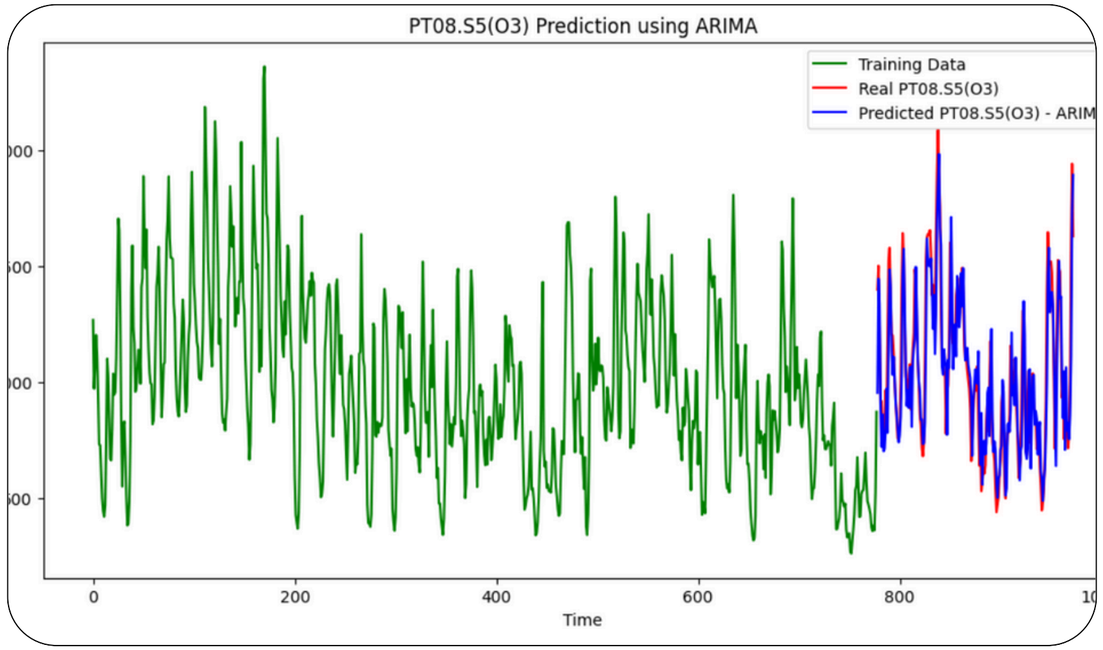
- Pics et creux réguliers
- Autocorrélation détectée



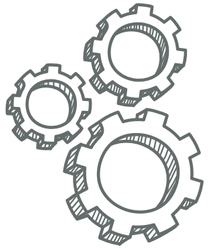
Modélisation ARIMA

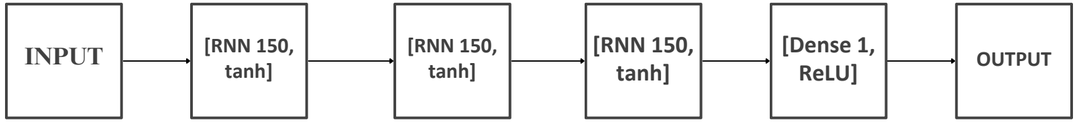
- **Architecture du modèle ARIMA :**
  - Modèle ARIMA (2,1,1) sélectionné via Grid Search en minimisant l'AIC .
- **Entraînement :**
  - Validation par glissement (Walk Forward Validation) .

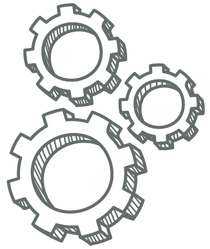
```
Residual Forecast Error: 16.099592752419454
Forecast Bias: 16.099592752419454
Mean Absolute Error: 123.4096448698548
Mean Squared Error: 24965.311545484074
Root Mean Squared Error: 158.00415040588038
R² Score: 0.7819570341404055
```



Modélisation RNN

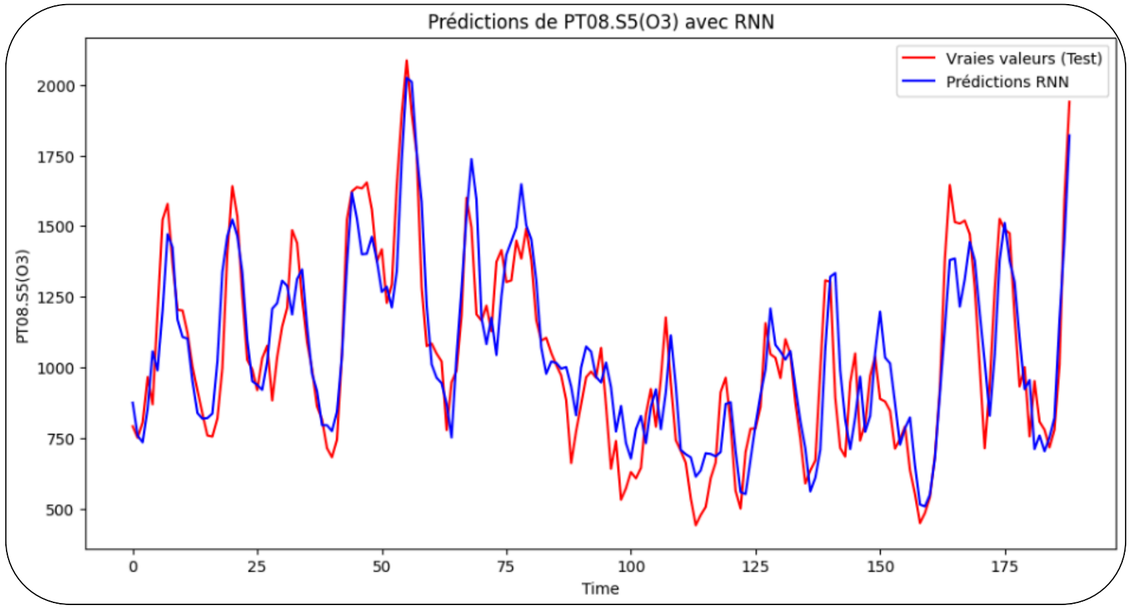




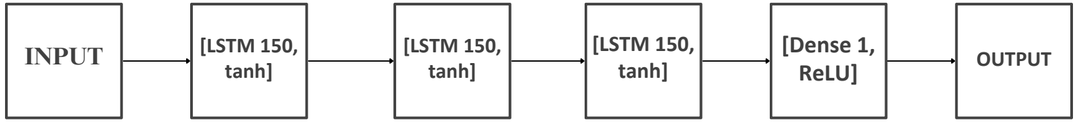


- Couches de Dropout (30%) pour éviter le surapprentissage.
- Optimisation avec Adam et fonction de perte MSE.
- 150 epochs, batch size=64, validation split=20%

```
Evaluation du modèle:
Mean Squared Error (MSE): 19888.14932535974
Mean Absolute Error (MAE): 111.59860953326151
R² Score: 0.8460504789003471
```

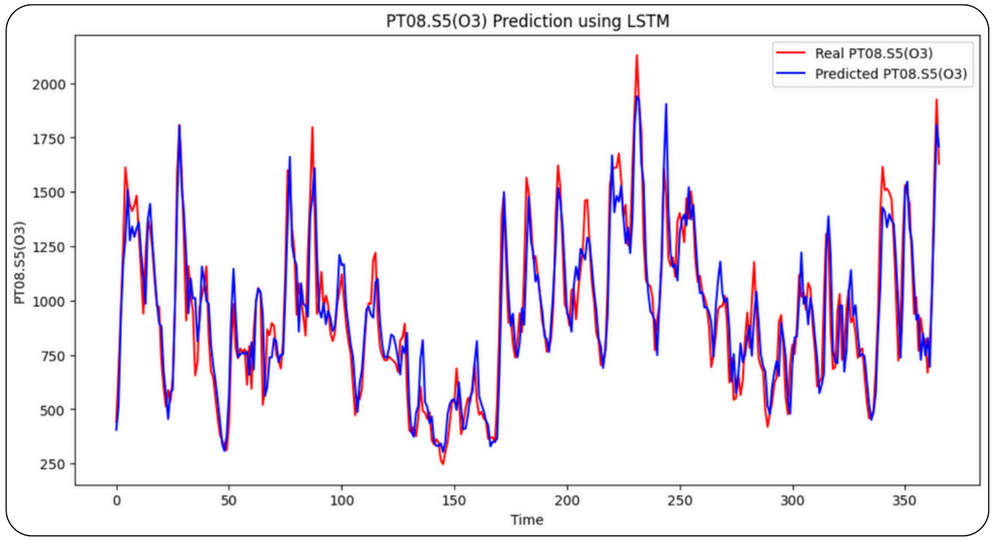


Modélisation LSTM

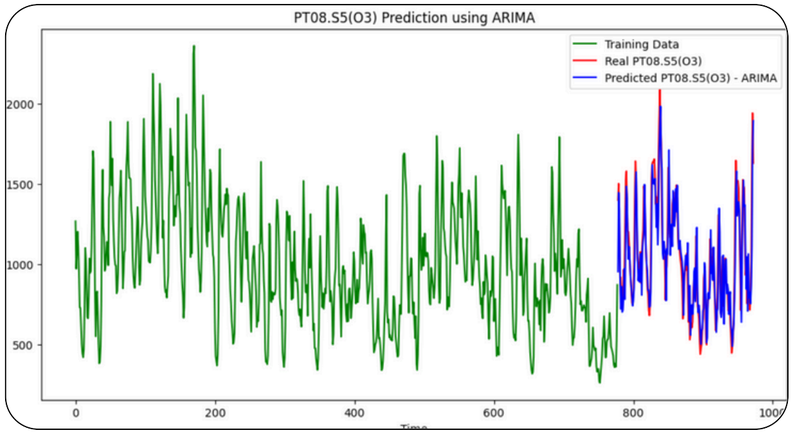


- Couches de Dropout (30%) pour éviter le surapprentissage.
- Optimisation avec Adam et fonction de perte MSE.
- 150 epochs, batch size=64, validation split=20%

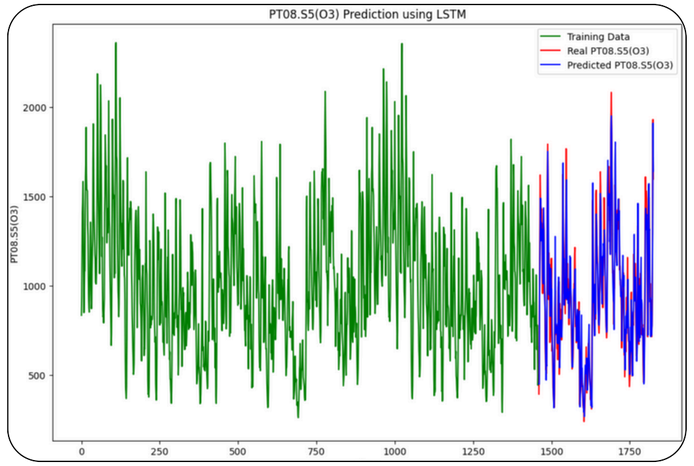
```
Evaluation du modèle:
Mean Squared Error (MSE): 17572.524961459403
Mean Absolute Error (MAE): 102.97603910010308
R² Score: 0.8639751865258364
```



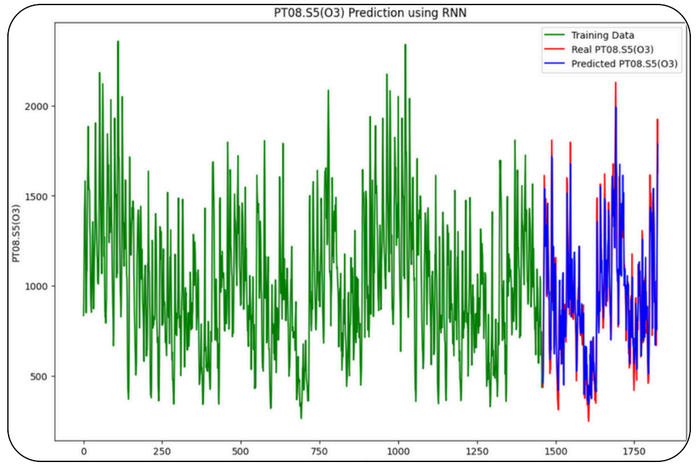
Comparaison finale des performances



ARIMA



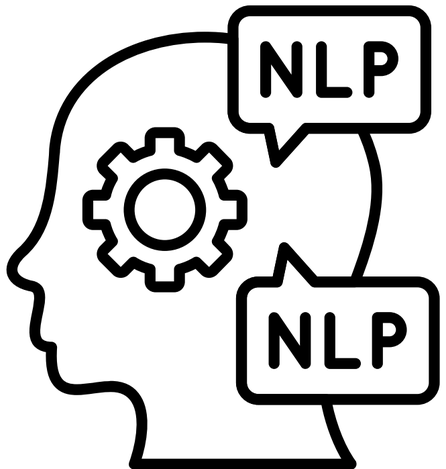
LSTM



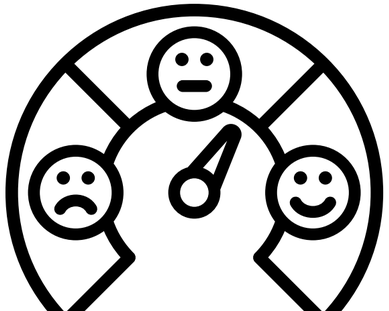
RNN

Model	MAE	MSE	RMSE	R2
ARIMA	123.409645	24965.311545	158.004150	0.781957
RNN	115.913238	21282.121348	145.883931	0.835376
LSTM	102.976039	17572.524961	132.561401	0.863975

6.NLP et Techniques Avancées



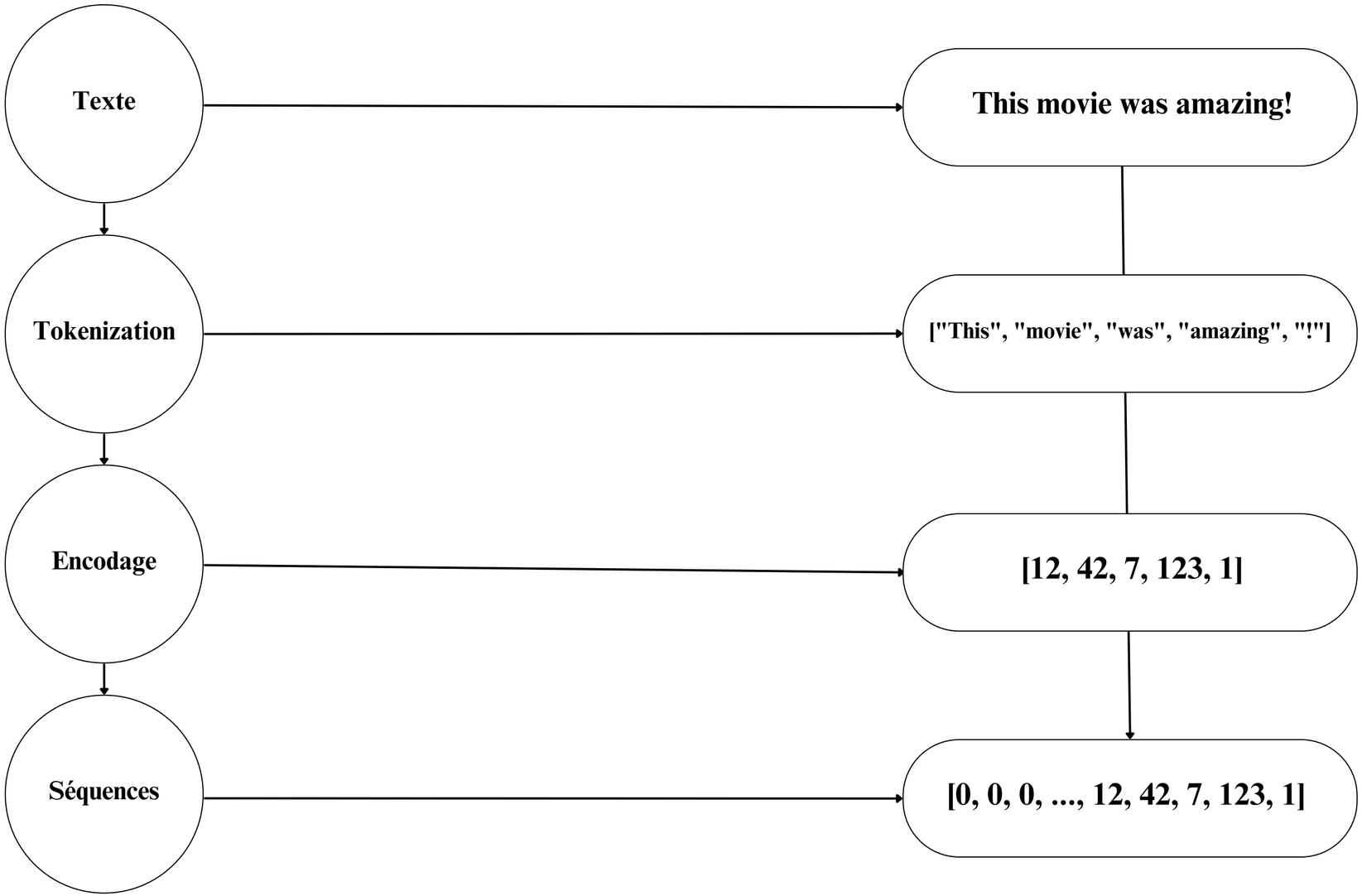
Sentiment Analysis



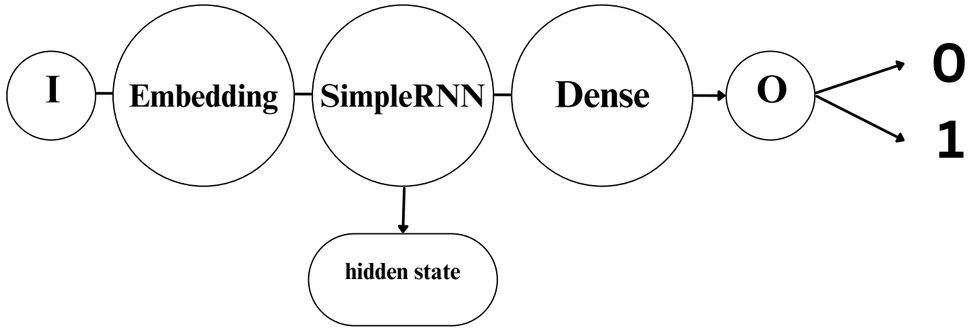
Objectif

- **Analyser les sentiments des critiques de films (positif/négatif).**
- **Comparer les performances des modèles RNN, LSTM et GRU.**
- **Intégrer un mécanisme d'attention pour améliorer les résultats.**
- **Utilisation de tronsformers .**

Préparation et Transformation



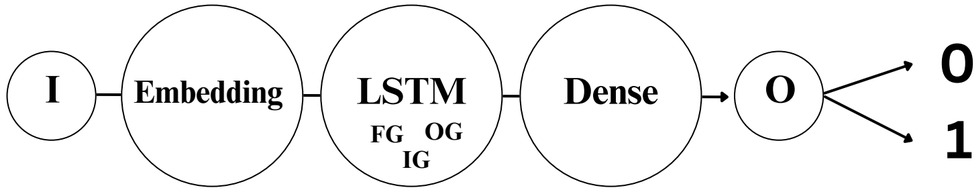
Modélisation avec RNN et LSTM




```
positive_review = "it was amazingly very great"
predict(positive_review)

negative_review = "that movie really sucked. I hated it and wouldn't watch it again."
predict(negative_review)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step
[0.9949482]
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
[0.06237121]
```

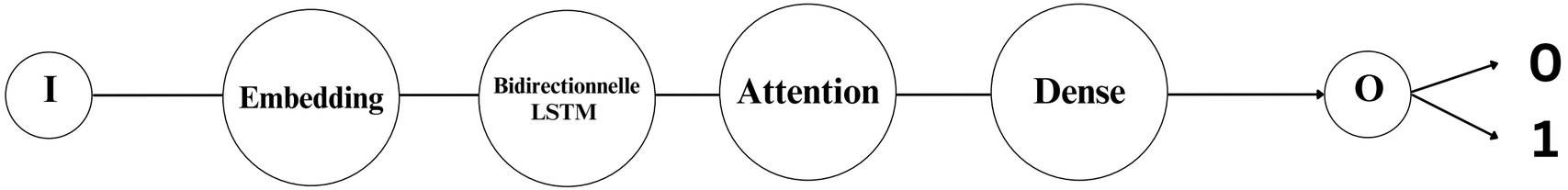



```
positive_review = "That movie was! really loved it and would great watch it again because it was ama
predict(positive_review)

negative_review = "that movie really sucked. I hated it and wouldn't watch it again. Was one of the v
predict(negative_review)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 44ms/step
[0.9041555]
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 44ms/step
[0.03734026]
```

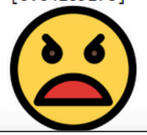
Mécanisme d'Attention



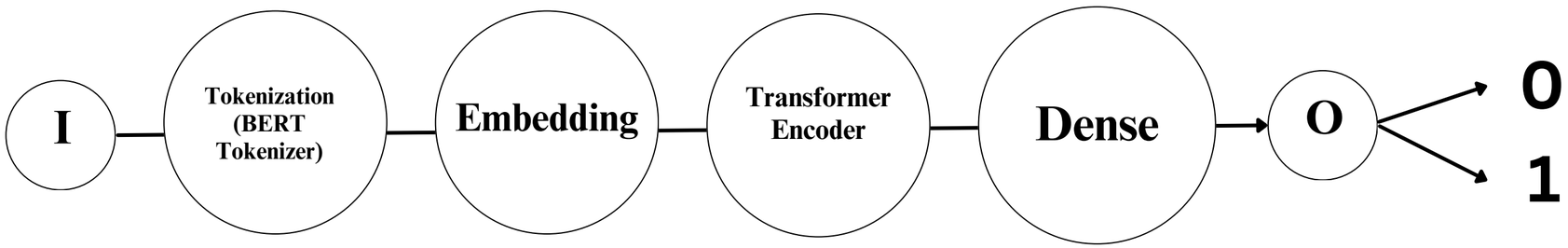

```
positive_review = "That movie was! really loved it and would great watch it agai
predict(model_lstm_attention,positive_review)

negative_review = "that movie really sucked. I hated it and wouldn't watch it aga
predict(model_lstm_attention,negative_review)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 64ms/step
[0.9814324]
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 62ms/step
[0.04269176]
```

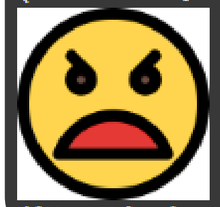


Modélisation avec Transformers



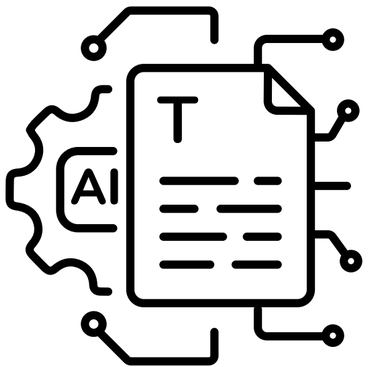

```
# Exemple de prédiction
text = "the thing that really ? people and this film proves that as the direc
print(predict_sentiment(text))
text = "This movie was absolutely bad! I hated every minute of it."
print(predict_sentiment(text))
```

```
('Positive', <tf.Tensor: shape=(), dtype=float32, numpy=0.801487922668457>)
```



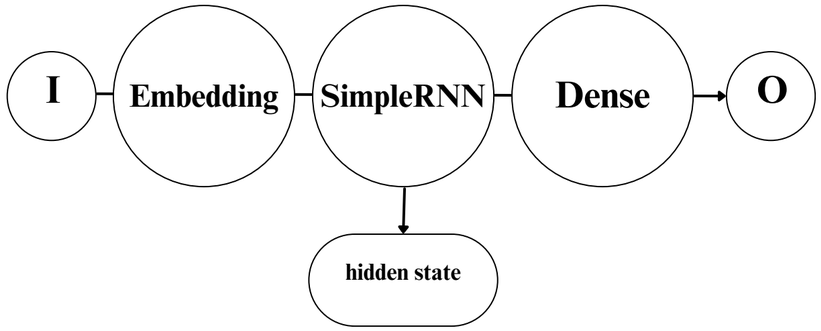
```
('Negative', <tf.Tensor: shape=(), dtype=float32, numpy=0.23638194799423218>)
```

Text Generation



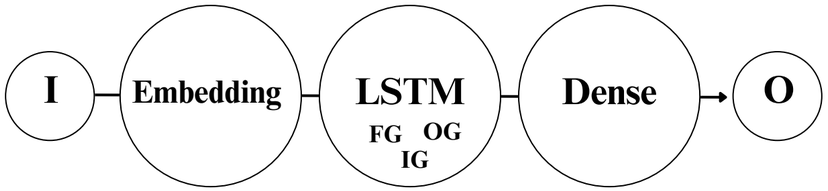
Préparation et Transformation

Modélisation avec RNN, LSTM et GRU



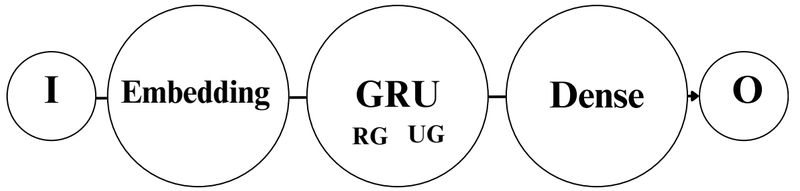

```
# Exemple de génération de texte
print(generate_text(model_rnn, start_string="ensao", temperature=0.1))
```

```
ensao offre des formations en ingénierie
```




```
# Exemple de génération de texte
print(generate_text(model_lstm, start_string="DSCC", temperature=0.3))
```

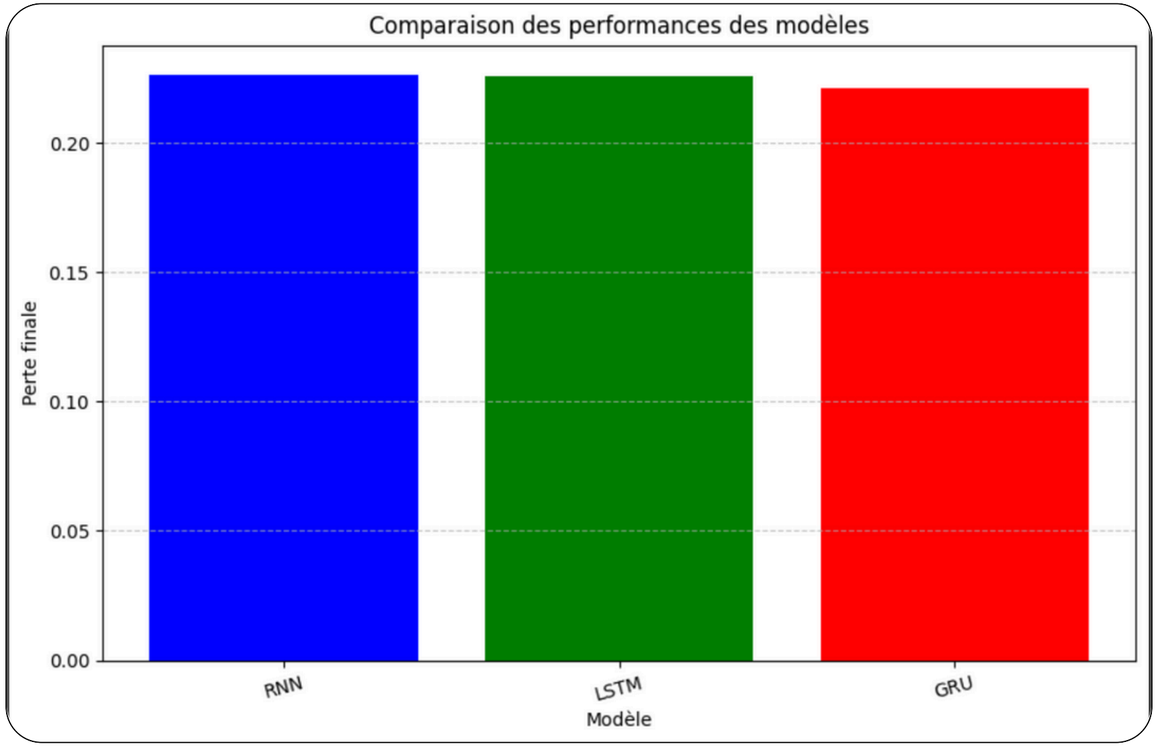
```
DSCC est dédiée aux sciences des donnée
```




```
# Exemple de génération de texte
print(generate_text(model_gru, start_string="ensao", temperature=0.5))
```

```
ensao offre des formations en ingénierie
```

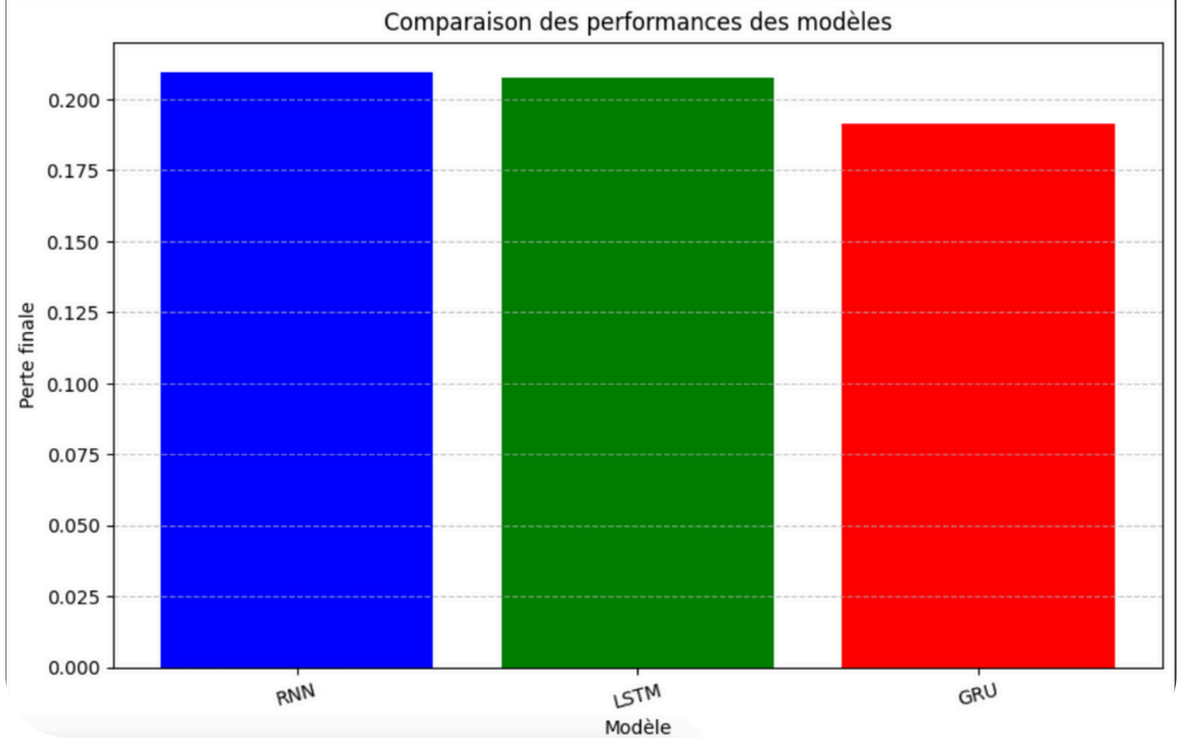
Comparaison des résultats



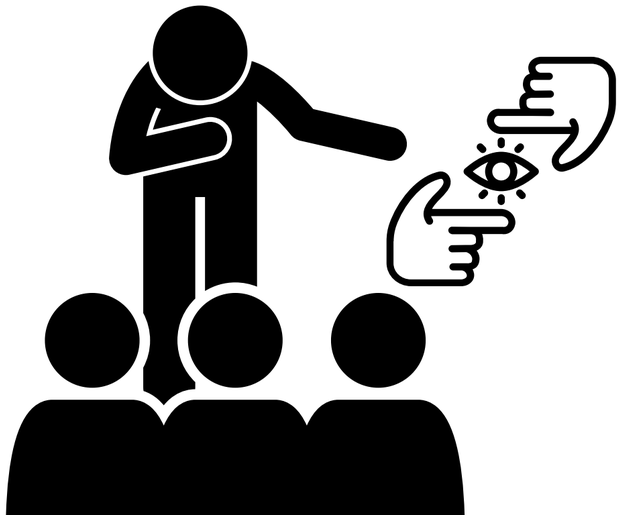

```
Résultats de la comparaison des modèles :
RNN : Perte finale = 0.2261
LSTM : Perte finale = 0.2226
GRU : Perte finale = 0.2215
```

Utilisation de GloVe

```
sultats de la comparaison des modèles :
NN : Perte finale = 0.2097
LSTM : Perte finale = 0.2078
GRU : Perte finale = 0.1914
```



7.Perspectives et Conclusions



Limites

Vanishing Gradient

Complexité

Interprétabilité

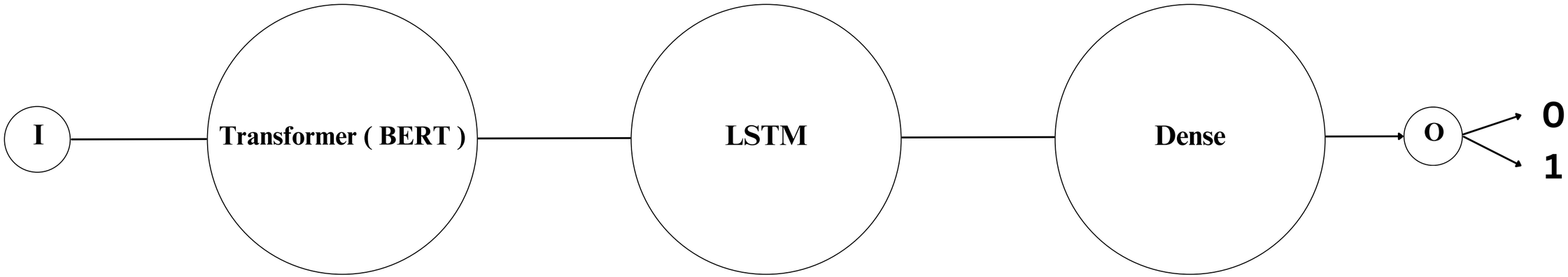
Sensibilté

Modèles Préentraînés

Transfer Learning

Contexte Global

Transformers Hybrides



MERCI POUR VOTRE ATTENTION