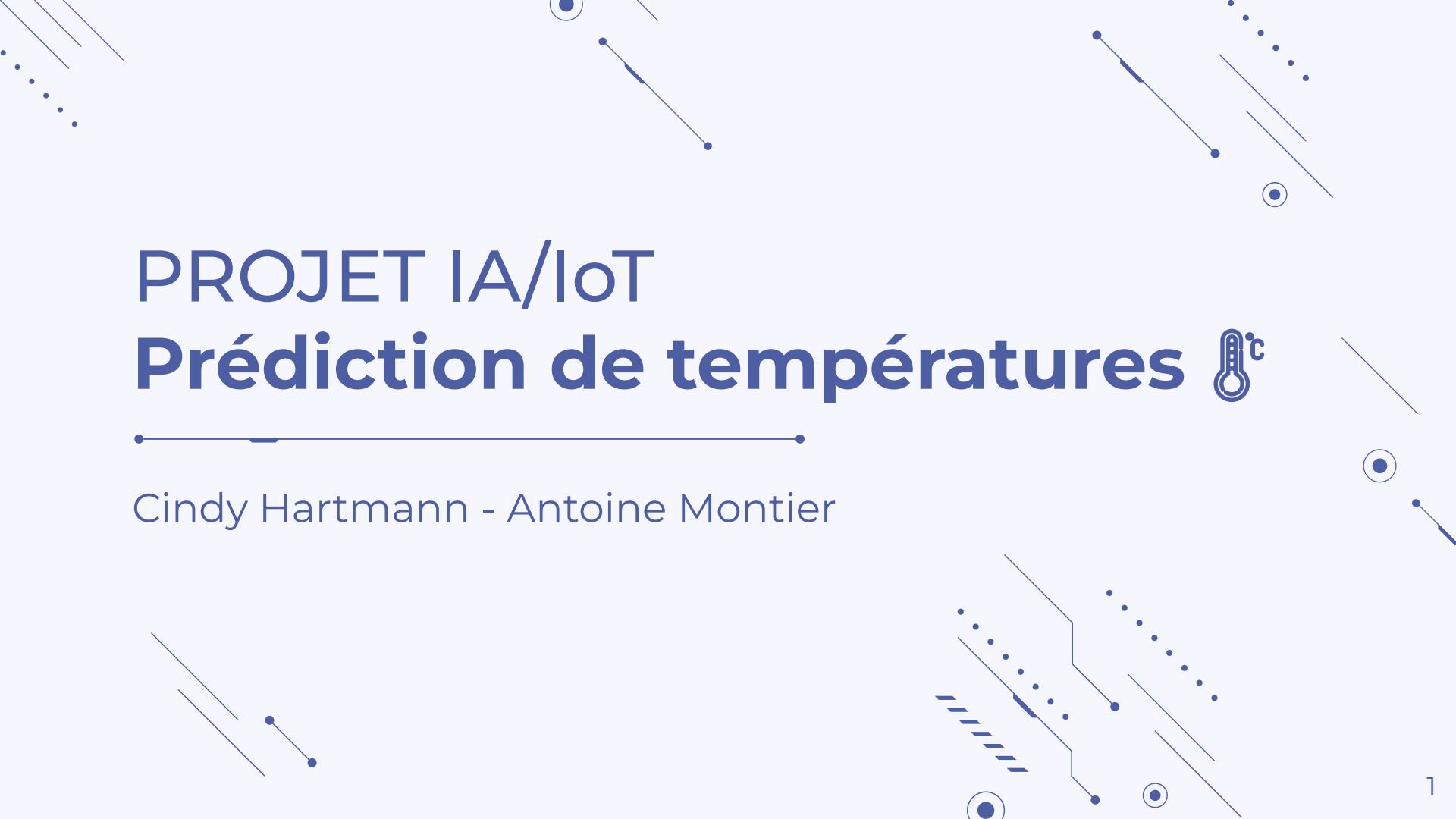


PROJET IA/IoT

Prédiction de températures



Cindy Hartmann - Antoine Montier

PLAN

01

Récupération des données

Prototype
Connexion BLE



Capteur d'humidité et de température Arduino DHT11



Carte Arduino Nano 33 BLE Rev2
256KB SRAM 1MB Flash

02

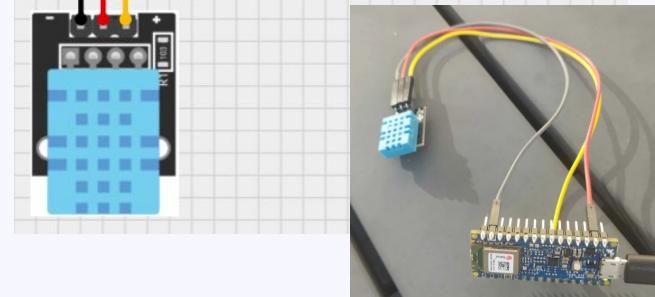
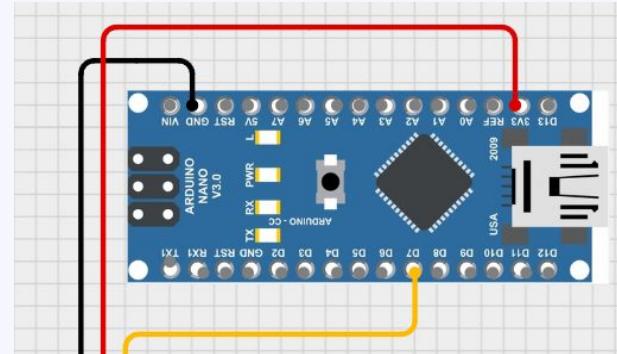
Implémentation du modèle d'IA

Différents modèles testés

1/ Récupération des données

DHT sensor library (1.4.6) d'Adafruit

```
test_thermometer.ino
1  #include "DHT.h"
2
3  #define DHTPIN 7
4  #define DHTTYPE DHT11
5
6  DHT dht(DHTPIN, DHTTYPE);
-- 
13 void loop() {
14   float h = dht.readHumidity();
15   float t = dht.readTemperature();
16 }
```



1/ Récupération des données



Dispositif de captation
(à placer à l'extérieur)

BLE - Bluetooth
Low Energy

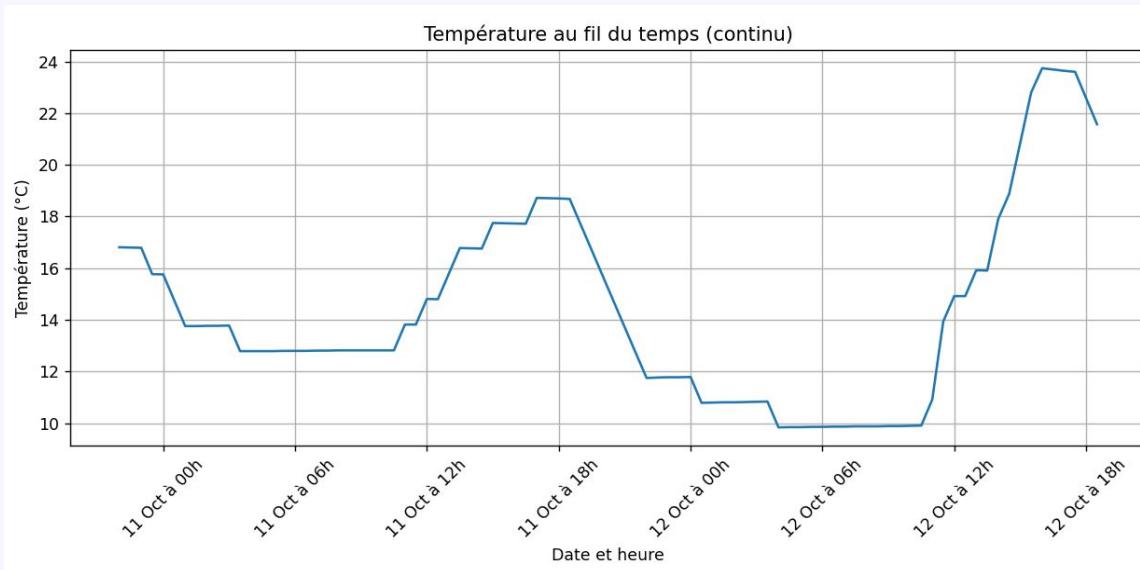
Mode direct



Mode Dongle
(2e carte Arduino)



1/ Récupération des données



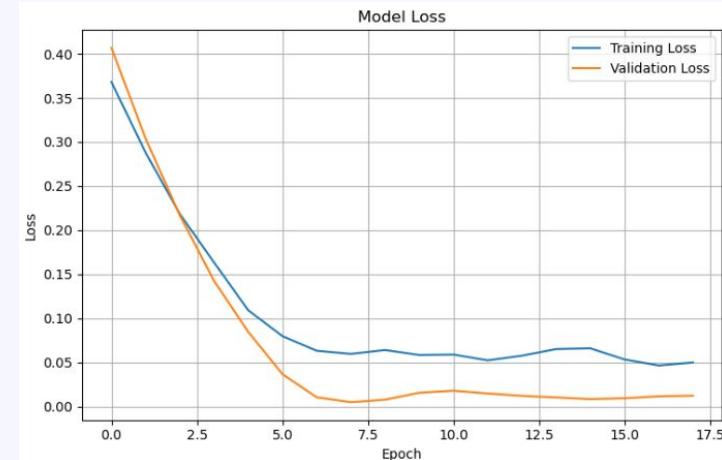
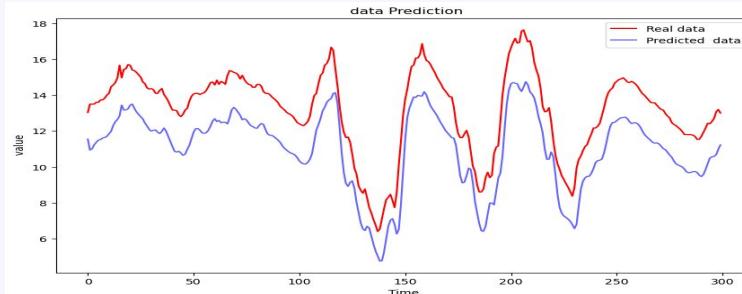
- Récupération de l'historique des dernières mesures pour plus de robustesse
- Complétées par les données de l'api historical weather (open-meteo.com)

2/ Implémentation du modèle

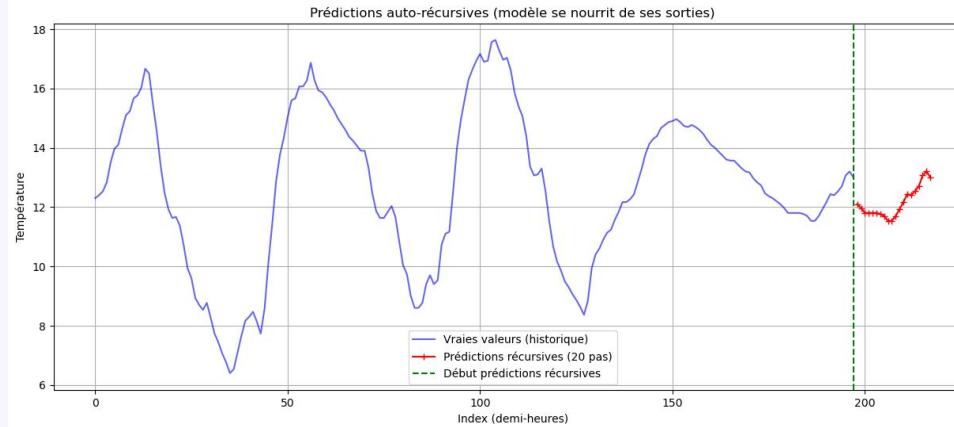
LSTM (Long Short-Term Memory)

- Mémoire des données temporelles
- Dépendances entre mesures successives
- Prédiction de séquences continues

3 couches (~5kb de paramètres)
 R^2 score : 0.1249

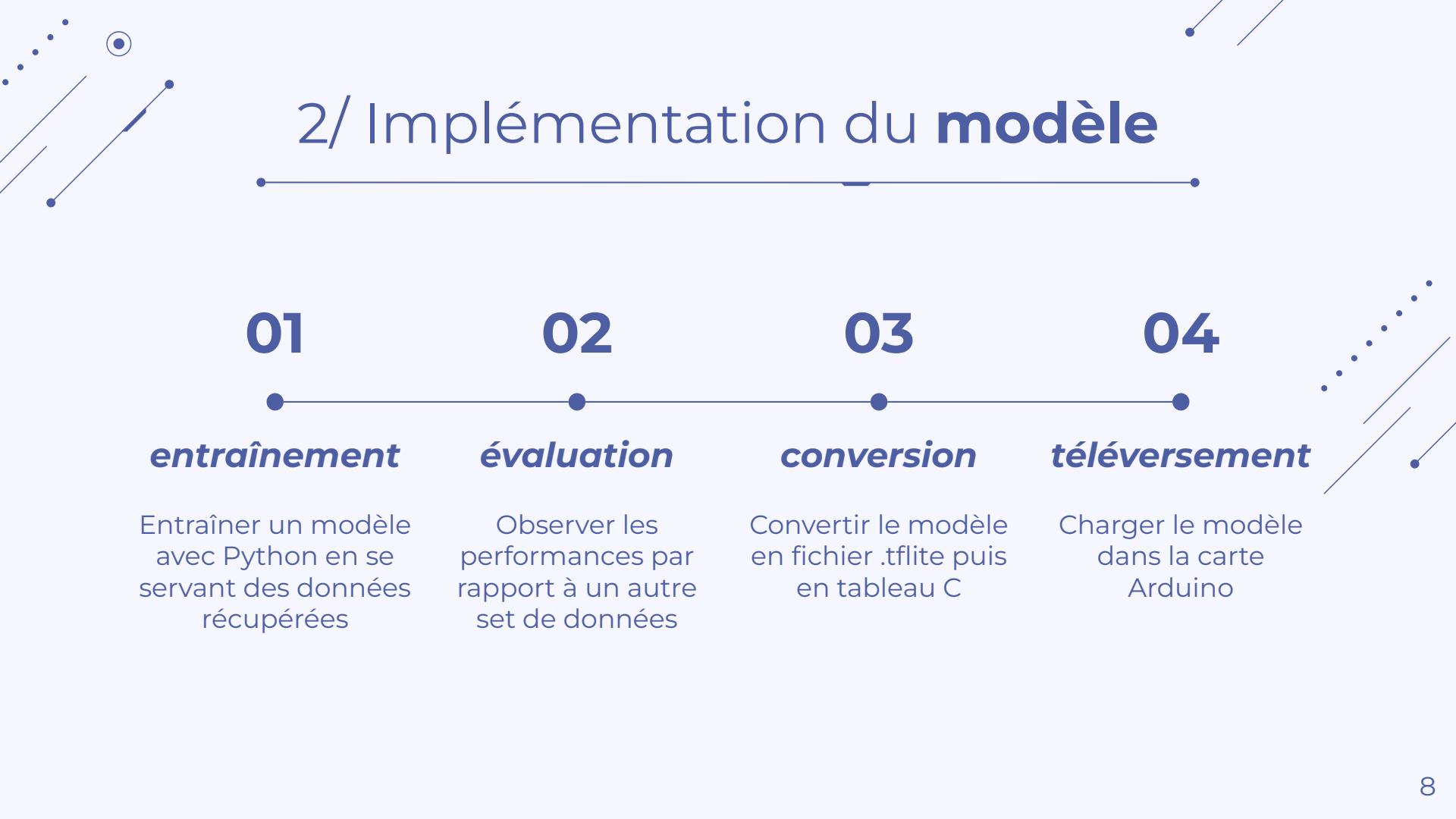


2/ Implémentation du modèle



Problèmes :

- Trop lourd pour la carte Arduino
- Opérations non supportées



2/ Implémentation du **modèle**

01

entraînement

Entraîner un modèle avec Python en se servant des données récupérées

02

évaluation

Observer les performances par rapport à un autre set de données

03

conversion

Convertir le modèle en fichier .tflite puis en tableau C

04

téléversement

Charger le modèle dans la carte Arduino

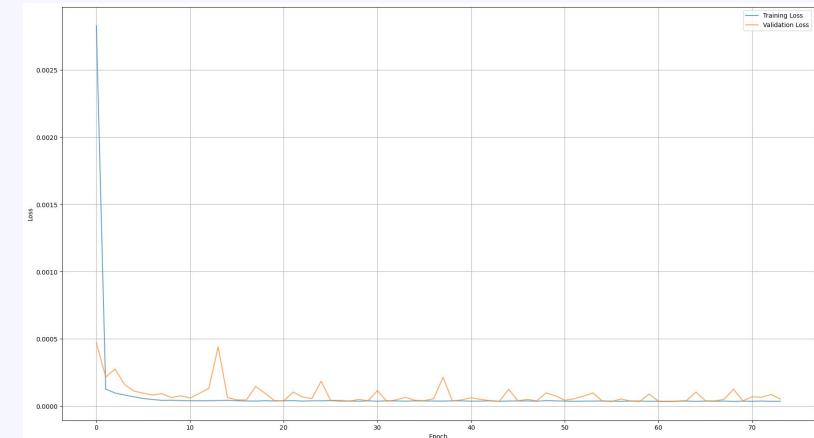
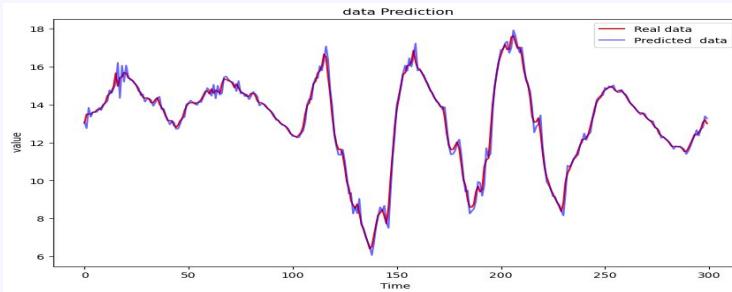
2/ Implémentation du **modèle**

GRU (Gated Recurrent Unit)

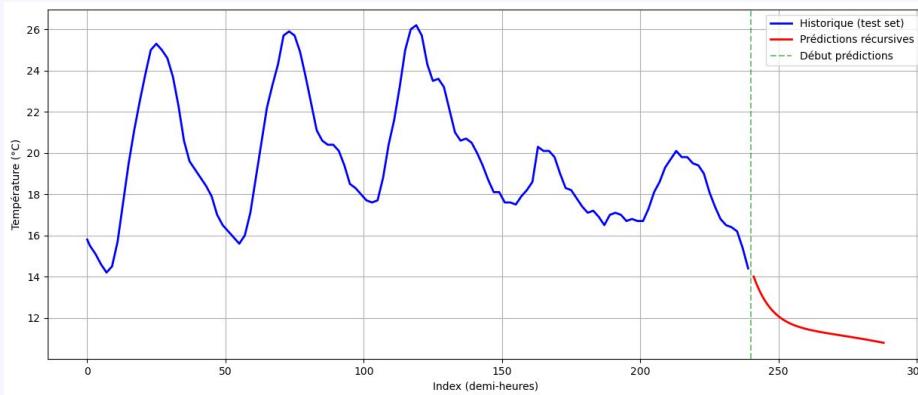
→ Similaire à LSTM en plus simple et léger

3 couches (~500b de paramètres)

R² score : 0.9840



2/ Implémentation du modèle



Problèmes

- écarts importants entre le modèle Python et le modèle Arduino
- modèle encore trop lourd si on veut avoir des résultats satisfaisants

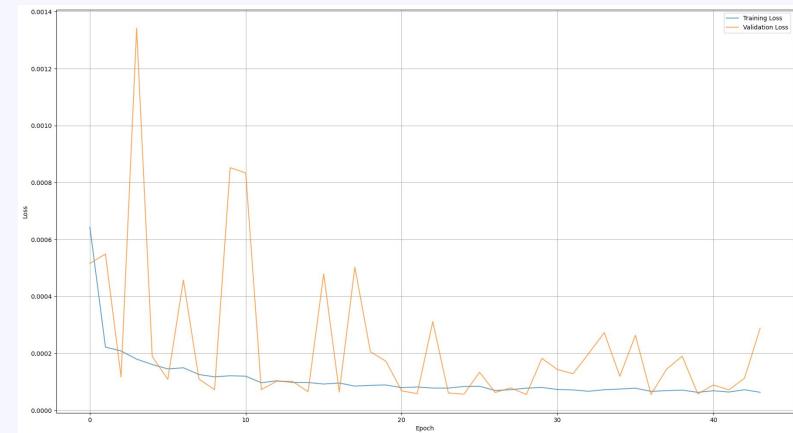
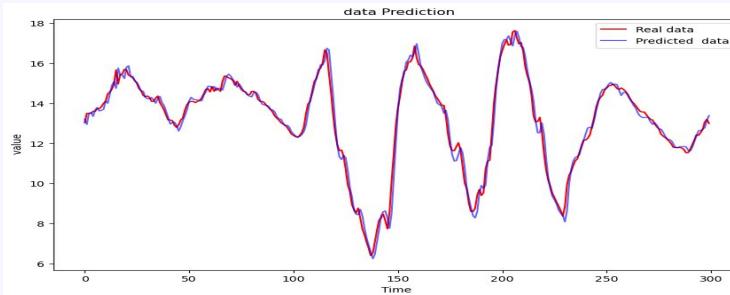
2/ Implémentation du **modèle**

MLP (Multi-Layer Perceptron)

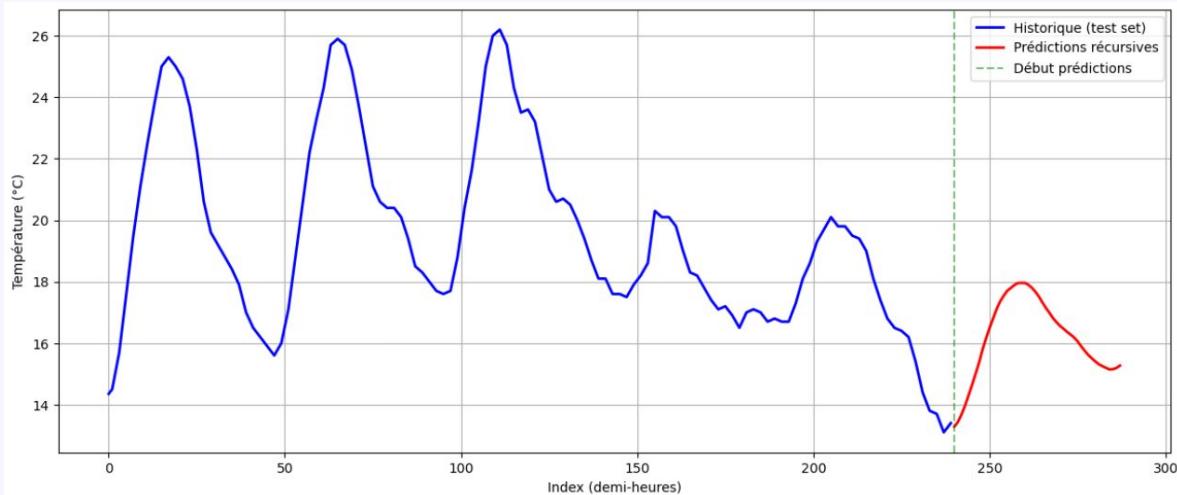
→ Légèrement moins bon, mais beaucoup plus léger et simple à implémenter

4 couches (~9kb de paramètres)

R² score : 0.9774



2/ Implémentation du **modèle**



Comparaison des modèles

	LSTM	GRU	MLP
R ²	0.1249	0.9840	0.9774
Poids (KB)	48.4	30.6	6.4
Compatibilité	Non	En partie	Oui

CONCLUSION

Axes à poursuivre

Automatisation

de la collecte des données

Modèle GRU

à coder en C pour comparer avec MLP

Visualisation

des résultats (envoi BLE, serveur web)

MERCI !

Ressources :

- Pili-Zhangqiu, Wireless PC Communication with the Arduino Nano 33 Series, GitHub.
<https://github.com/pili-zhangqiu/Wireless-PC-Communication-with-the-Arduino-Nano-3-Series?tab=readme-ov-file>
- Mistry, S., Pajak, D. and Siebeneicher, H. Get Started With Machine Learning, Docs.arduino.cc.
<https://docs.arduino.cc/tutorials/nano-33-ble-rev2/get-started-with-machine-learning/>
- Videvelop, TensorFlow Lite Micro Library for Arduino, GitHub.
<https://github.com/tensorflow/tflite-micro-arduino-examples>

CRÉDITS: Ce modèle de présentation a été créé par **Slidesgo**, comprenant des icônes de **Flaticon**, des infographies et des images de **Freepik**