Temperature Forecasting with TinyML

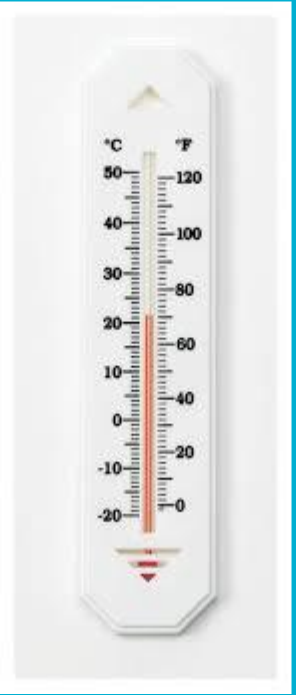
Group Group:

Allison Lampe, Sonia Aung, Claire Chiang, Zourong Jiang

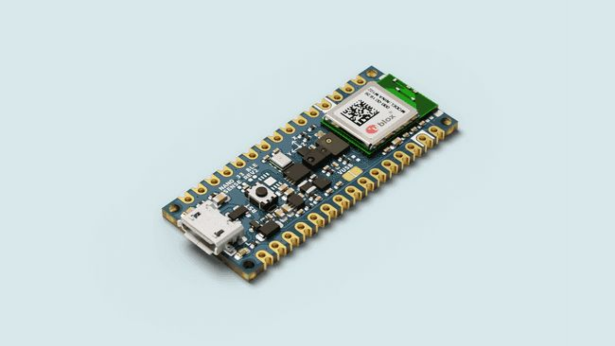
Team members

- Claire Chiang - Model training
- Zourong Jiang - Data preprocessing, Initial model architecture building
- Allison Lampe - Microcontroller configuration and data collection
- Sonia Aung - Testing
- Creating final deliverables - Everyone

Motivation



- Temperature variations directly affect human comfort and daily decisions (e.g.: Sudden temperature drops may require proactive actions)
- Cloud-based solutions increase latency, energy consumption, and privacy risks
- TinyML allows real-time temperature forecasting directly on low-power devices
- This project explores the feasibility of deploying ML models on resource-constrained hardware



Arduino Nano 33 BLE Sense

Data Processing

Collection

- Data source: Historic weather data from meteostat.com
- Time range: Jan 1, 2024 – Dec 1, 2024
- Raw features: Temperature (°C), Relative humidity (%), Air pressure (hPa)

Goal

- Input different training figures
- Determine which features lead to optimal output
- Tradeoffs between higher data and higher memory usage

Data Processing

Pipeline

- Raw data consists of a continuous hourly temperature time series
- Remove missing values to ensure clean sequences
- Time series is converted into supervised learning samples using a sliding window
- Input: temperature readings from the past 24 hours
- Output: temperature of the next hour

Model Architecture

Input(24) -> Dense(32,ReLU) -> Dense(16,ReLU) -> Output(1)

Dense(32): extracts short-term temperature patterns

Dense(16): compresses features to reduce model size

Output: next-hour temperature prediction

Total parameters: 1,345(~5.25KB)

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 32)	800
dense_1 (Dense)	(None, 16)	528
dense_2 (Dense)	(None, 1)	17

Total params: 1,345 (5.25 KB)
Trainable params: 1,345 (5.25 KB)
Non-trainable params: 0 (0.00 B)

Model training

Strategy

- Pretraining on large-scale historical weather data (which from Meteostat.com)
- Fine-tuning on small-scale Arduino sensor data (which is collected by our team)
- Model export with INT8 quantization for microcontroller deployment

Web Data Pretraining

- One year of historical weather data used for pretraining
- Large dataset helps the model learn general temperature dynamics

Model Training

Device Fine-tuning and Normalization

- Limited Arduino sensor data collected
- Normalization parameters computed from web training data
- Ensure stable scaling and prevents overfitting to small device dataset

Towards Microcontroller Deployment

- Converted the trained model to TFLite INT8 format
- Prepared normalization parameters for on-device preprocessing
- Designed an end-to-end pipeline targeting Arduino Nano 33 BLE Sense
- On-device evaluation not fully successful due to limited device data
- More continuous sensor data is required for reliable deployment testing

Results

- Increase or decrease in 30 minutes - only 53% success rate
- Average distance between predicted and real temperature - 1.30 C
- More data would be preferred, but it may just not be enough parameters to properly predict temperature

Temp(C)	Hum(%)	Pres(hPa)	Pred_Temp	Real_Temp
6.88	37.28	993.85	6.79	5.33
6.79	36.95	993.88	6.52	5.27
6.68	36.83	993.86	6.76	5.23
6.56	38.26	993.86	6.83	5.22
6.44	38.78	993.89	6.16	5.19
6.33	41.17	993.93	6.27	5.19
6.29	40.36	993.93	6.24	5.23
6.24	40.62	993.94	5.48	5.26
6.24	40.38	993.96	5.81	5.3
6.24	41.09	993.96	5.61	5.33
6.23	40.99	993.98	5.63	5.37
6.2	40.48	993.98	5.66	5.4
6.16	40.68	994	5.66	5.44
6.19	39.69	994	5.68	5.44
6.19	38.97	993.98	5.72	5.45
6.16	41.94	994.03	5.74	5.46
6.15	41.89	994	5.74	5.49

Future Improvements

- Multi season training and testing
- More data
- Focus more on long term predictions