THE UNIVERSITY OF KANSAS

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

EECS 700: Embedded Machine Learning

Fall 2024

Project Report

Student Name: Sai Rithvik Gundla — Student ID: 3109847
Student Name: Dimple Galla — Student ID: 3149112
Student Name: Manoj Prakash Reddy — Student ID: 3149217

EECS 700: Embedded Machine Learning– Academic Project

Abstract— **This project implements a temperature prediction system using a Long Short-Term Memory (LSTM) model deployed on a resource-constrained microcontroller, the XIAO ESP32S3. By leveraging historical temperature data collected through sensors, the system predicts the temperature one hour ahead with minimal latency. The LSTM model was trained on normalized hourly data, and its architecture was optimized for edge computing devices. The trained model was converted into TensorFlow Lite (TFLite) format to enable deployment on the ESP32S3 using Edge Impulse SDK.**

The system's efficient performance, demonstrated by a Root Mean Squared Error (RMSE) of 0.025 and an inference latency of 0.584 seconds, makes it ideal for real-world applications in agriculture, environmental monitoring, and smart cities. The project highlights the potential of TinyML in enabling AI-powered solutions for edge devices, showcasing how machine learning models can deliver accurate predictions with minimal resource usage. By bridging the gap between data-driven forecasts and embedded systems, this solution paves the way for innovative IoT applications in weather prediction.

1. INTRODUCTION

In our increasingly interconnected world, where climate data plays a vital role in decision-making and planning, real-time weather forecasting has become essential. This project focuses on predicting future temperatures (one hour ahead) using historical weather data by training a Long Short-Term Memory (LSTM) model. The trained model is converted to TensorFlow Lite (TFLite) and deployed on an XIAO ESP32S3 microcontroller for predictions. This implementation of TinyML (Tiny Machine Learning) demonstrates how advanced AI models can be applied in embedded systems to provide efficient, accurate solutions.

Why LSTM?
LSTM networks excel in capturing temporal dependencies and complex nonlinear relationships in time-series data. Unlike traditional methods, LSTMs:
- Retain and utilize past observations over long durations.
- Handle vanishing gradient issues effectively, ensuring stable learning of long-term dependencies.
- Learn intricate patterns, making them ideal for weather forecasting tasks.

By leveraging these capabilities, this project addresses the challenge of short-term temperature prediction with high accuracy and efficiency.

2. APPLICATIONS:

This system is designed for a variety of real-world applications:

Agriculture: Assists in determining irrigation schedules and protecting crops from extreme weather.
Smart Cities: Integrates with urban planning systems for energy management and disaster preparedness.

The system’s portability and adaptability make it suitable for diverse environments and use cases, further demonstrating the power of embedded AI.

3. PROBLEM STATEMENT:

Weather forecasting systems often rely on high-powered infrastructure, which is not feasible for edge devices or IoT applications. This project aims to fill this gap by providing an LSTM-based prediction system tailored for resource-constrained environments like the ESP32S3.

4. DATASET:

The dataset comprises hourly weather data collected from sensors, specifically focusing on parameters such as:

-Temperature (°C)

-Humidity (%)

The data spans 5 weeks, ensuring robustness and reliability for training and testing. Data was retrieved from Lawrence Weather Data using a DHT22 Sensor and processed with Python.

Preprocessing Steps:

-Cleaning: Removal of noise and handling of missing values.

-Normalization: Scaling features to a [0, 1] range using Min-Max Scaling.

-Transformation: Generating input-output pairs with 24-hour input windows to predict the next hour's temperature.

-The dataset was split into training (80%), validation (10%), and testing (10%).

The past 24 hours were used as input to predict the next hour, ensuring relevance and temporal consistency. Normalization and scaling were applied to maintain uniform and consistent input features.

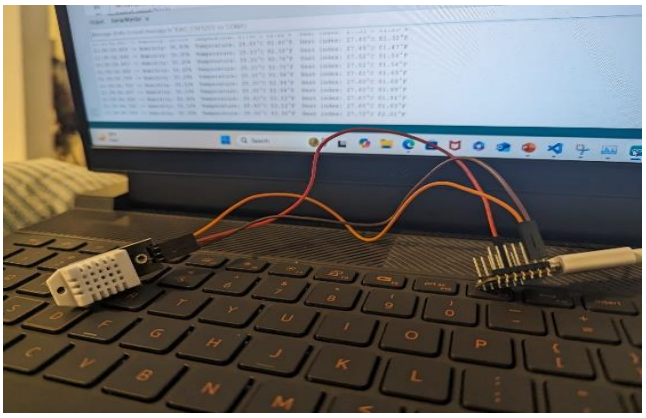

Fig1. Temperature and Humidity collected through DHT 22 Sensor

```
21:35:36.074 -> Humidity: 37.80%  Temperature: 26.90°C
21:35:38.082 -> Humidity: 37.40%  Temperature: 26.80°C
21:35:40.099 -> Humidity: 37.70%  Temperature: 26.80°C
21:35:42.076 -> Humidity: 37.50%  Temperature: 26.80°C
21:35:44.095 -> Humidity: 37.80%  Temperature: 26.80°C
21:35:46.088 -> Humidity: 37.80%  Temperature: 26.80°C
21:35:48.094 -> Humidity: 38.10%  Temperature: 26.70°C
21:35:50.112 -> Humidity: 37.70%  Temperature: 26.70°C
21:35:52.145 -> Humidity: 38.00%  Temperature: 26.70°C
```

Fig2. Readings captured from IDE of temperature and humidity

5. Model Design:

LSTM Algorithm

The model architecture is designed to handle sequential data efficiently:

LSTM Layer: 128 units, learning temporal dependencies from sequential input data.

Dense Layer: A single neuron with linear activation, producing continuous regression output.

Training Parameters

Optimizer: Adam, leveraging adaptive learning rates for efficient weight updates.

Loss Function: Mean Squared Error (MSE), minimizing prediction errors.

Batch Size: 32, optimized for stability and hardware constraints.

Epochs: 20 with early stopping to prevent overfitting.

6. LSTM MODEL FOR TEMPERATURE PREDICTION:

The project transitioned to utilizing a **Long Short-Term Memory (LSTM)** model for temperature prediction on embedded systems, particularly with a focus on real-time and edge-based applications. This decision was motivated by LSTM's effectiveness in handling time-series data, ensuring accurate and efficient predictions, even on constrained devices.

Training the Model:

The project began with the collection of historical temperature data, which was preprocessed into sequences suitable for time-series analysis. The data underwent normalization to ensure consistent input values for the model. The LSTM model, implemented using TensorFlow, was trained to predict future temperatures based on previous time-step sequences.

The training process included:

- Dividing the dataset into training and testing sets.
- Configuring an LSTM model with appropriate layers for time-series forecasting.
- Optimizing the model using the Adam optimizer and Mean Squared Error (MSE) as the loss function.

During testing, the model demonstrated a high degree of accuracy in predicting temperature trends, with minimal error margins across the validation dataset.

7. REAL-TIME SCENARIO:

Consider a remote agricultural field where temperature variations directly affect crop yield. A compact TinyML-enabled sensor equipped with the trained LSTM model is deployed to monitor and predict temperature changes in real time. The system processes live temperature data, continuously analyzing trends and providing accurate predictions. These forecasts enable farmers to make proactive decisions such as irrigation scheduling or deploying protective measures to safeguard crops during extreme conditions. This operates seamlessly on an embedded device, ensuring low power consumption and portability.

8. RESULTS:

The **LSTM-based TinyML solution** successfully predicts temperature trends with remarkable accuracy on edge devices. Key outcomes include:

- Efficient deployment on embedded systems with limited computational resources.
- Predictions with minimal latency.
- A robust model capable of adapting to varying environmental conditions. The project highlights the

transformative potential of TinyML for edge-based temperature forecasting, improving decision-making in sectors like agriculture, environmental monitoring, and IoT applications.

The following graph compares the predicted vs actual temperature values, highlighting the model's accuracy and ability to capture trends.

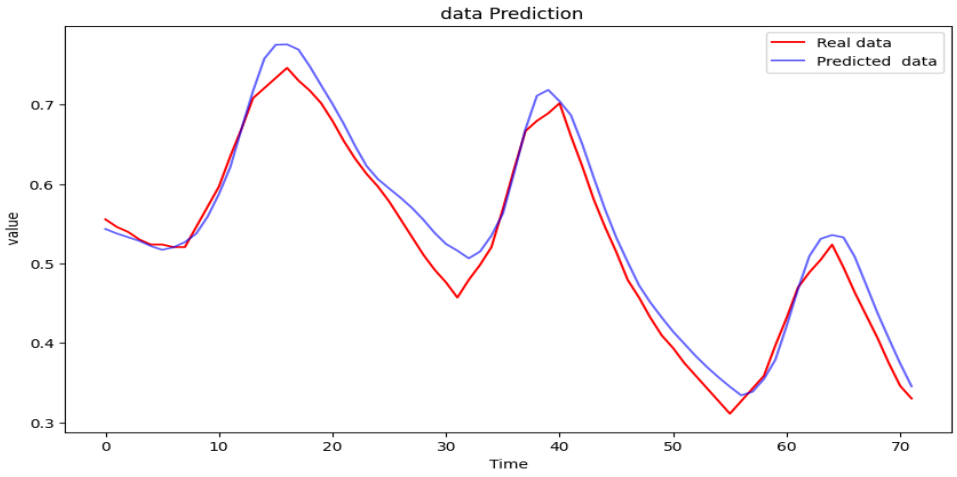

Fig3. Real Data Vs Predicted Data as depicted in above graph.

Model Performance:

- **Real Value (1 Hour Ahead):** 0.555
- **Predicted Value (1 Hour Ahead):** 0.543
- **Error:** 0.012

The model's predicted values closely match the actual temperature trends, demonstrating its accuracy and efficiency in forecasting future temperatures.

Normalization and Inference:

- **Normalization:** The data was preprocessed using **Min-Max Scaling** to rescale the temperature values to a specific range (typically between 0 and 1) before making predictions.
- **Reversing Normalization:** After generating the predictions, the inference results were **rescaled back to Celsius** using the **MinMaxScaler**, ensuring the output was in the correct unit of measurement.

Quantitative Evaluation:

Layer	MACs	RAM	Input Buffer Size	Output Buffer Size	Parameters
LSTM_1 (LSTM)	22,536,192	124,884 bytes	336	21,504	67,072
Dense_1 (Dense)	128	No Additional RAM	128	1	129

Table1. Quantitative evaluation

9. CHALLENGES FACED:

During deployment, several challenges were encountered:

Temperature Variability

- Capturing accurate predictions across different environmental conditions (e.g., extreme weather changes) required additional fine-tuning of the model

Real-Time Inference

- Achieving low-latency predictions (0.584 seconds per inference) while maintaining accuracy required extensive profiling and optimization of the TensorFlow Lite model.

By addressing these challenges through efficient coding practices, the system achieved reliable performance on resource-constrained devices.

10. REFERENCES

- TensorFlow Lite Micro Documentation.
- Edge Impulse SDK User Guide.