**Laporan Praktikum IoT : Penerapan Machine Learning dengan TensorFlow Lite di ESP32**

*Difa Aqilah*

*Fakultas Vokasi, Universitas Brawijaya*

*Email :* [*difaaqilah04@student.ub.ac.id*](mailto:difaaqilah04@student.ub.ac.id)

**Abstrak**

Dalam praktikum ini, dilakukan implementasi model machine learning pada perangkat mikrokontroler ESP32 menggunakan TensorFlow Lite. Model klasifikasi yang digunakan adalah model untuk mengenali jenis bunga Iris. Proses dimulai dari pelatihan model di Google Colab, lalu model dikonversi dari format Keras menjadi .tflite, dan selanjutnya diubah menjadi array C yang disimpan dalam file iris\_model.h. File ini kemudian diintegrasikan ke dalam proyek PlatformIO agar dapat dijalankan langsung di ESP32.

Klasifikasi dilakukan secara lokal di ESP32 dan hasilnya ditampilkan melalui Serial Monitor. Sistem ini terbukti mampu memberikan prediksi dengan akurasi tinggi dan waktu proses yang cepat, sekitar 1086 mikrodetik. Dengan praktikum ini, peserta dapat memahami secara langsung bagaimana cara menggabungkan model machine learning dengan perangkat IoT berbasis mikrokontroler.

Kata Kunci — *ESP32, Wokwi, TinyML*

**PENDAHULUAN**

Perkembangan Internet of Things (IoT) telah membawa perubahan besar dalam cara kita memproses dan menganalisis data. Salah satu tren baru yang muncul adalah **TinyML**, yaitu integrasi machine learning pada perangkat kecil seperti ESP32. TinyML memungkinkan pemrosesan data dilakukan langsung di perangkat, tanpa perlu koneksi ke server eksternal.

Dalam praktikum ini, dilakukan percobaan untuk mengimplementasikan model klasifikasi bunga Iris yang sebelumnya telah dilatih di Google Colab. Model ini diubah ke format .tflite lalu dikonversi menjadi header file iris\_model.h, agar dapat dijalankan di dalam mikrokontroler ESP32 menggunakan PlatformIO. Hasil prediksi ditampilkan melalui Serial Monitor sehingga proses klasifikasi bisa langsung diamati secara real-time.

Pada era digital, perangkat yang mampu memproses data secara lokal menjadi sangat penting, terutama untuk sistem IoT yang sering bekerja di lingkungan terbatas. Konsep TinyML membuat ESP32 tidak hanya berfungsi sebagai pengumpul data, tetapi juga sebagai pemroses data yang cerdas.

Dataset Iris dipilih karena sederhana namun cukup representatif untuk pengenalan pola. Dataset ini terdiri dari data panjang dan lebar kelopak serta sepal tiga jenis bunga. Model yang dilatih dengan data ini dapat digunakan untuk menguji kemampuan ESP32 dalam menjalankan machine learning secara langsung. Dengan mengubah model ke dalam format .tflite dan kemudian ke iris\_model.h, mahasiswa dapat langsung melihat kinerja klasifikasi real-time di perangkat embedded.

**TUJUAN**

Adapun tujuan utama dari praktikum ini adalah:

1. Mempelajari cara menghubungkan antara PlatformIO/Wokwi dengan model CNN atau Google Colab.
2. Mengintegrasikan model yang dilatih di Google Colab ke dalam proyek PlatformIO.
3. Mengamati dan menampilkan hasil klasifikasi model secara langsung di ESP32, termasuk mengecek apakah prediksi yang dilakukan sesuai atau tidak.

**METODELOGI**

**Alat dan Bahan:**

* Mikrokontroler ESP32
* Platform Wokwi (untuk simulasi)
* Google Colab / CNN (untuk pelatihan model)
* PlatformIO / Arduino IDE via Visual Studio Code

**Langkah-langkah Implementasi:**

1. Persiapan Proyek

* Membuat proyek baru di PlatformIO melalui VSCode dan memilih board ESP32 sebagai target.

2. Pembuatan dan Konversi Model

* Melatih model klasifikasi bunga Iris di Google Colab dengan library Keras.
* Mengonversi model ke format .tflite dengan TFLiteConverter.
* Mengubah file .tflite menjadi array C menggunakan hex\_to\_c\_array() dan menyimpannya dalam iris\_model.h.

3. Integrasi Model ke PlatformIO

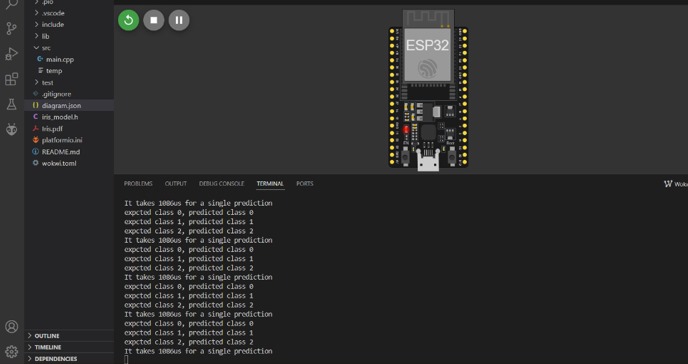
* Memasukkan iris\_model.h ke dalam proyek PlatformIO.
* Menulis program utama di main.cpp menggunakan pustaka eloquent\_tinyml dan tflm\_esp32.

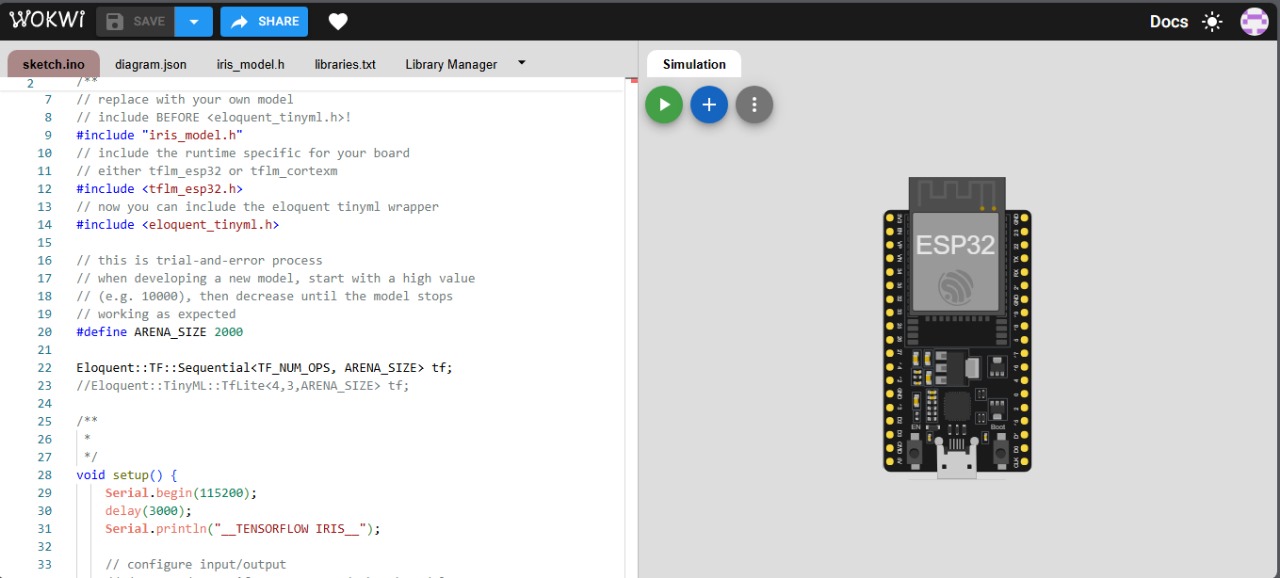
4. Penulisan Program Prediksi

* Menentukan jumlah input dan output model.
* Melakukan prediksi untuk tiga sampel data (x0, x1, x2).
* Menampilkan hasil prediksi dan waktu proses ke Serial Monitor.

5. Simulasi di Wokwi

* Membuat diagram koneksi ESP32 di Wokwi.
* Menghubungkan pin TX/RX ke Serial Monitor virtual.
* Menjalankan simulasi dan mengamati hasil klasifikasi secara langsung.





A screenshot of a computer

AI-generated content may be incorrect.

**LAMPIRAN**

**Kode Program Pada VSCode**

#include <Arduino.h>

#include <iris\_model.h>

#include <tflm\_esp32.h>

#include <eloquent\_tinyml.h>

#define ARENA\_SIZE 2000

Eloquent::TF::Sequential<TF\_NUM\_OPS, ARENA\_SIZE> tf;

void setup() {

    Serial.begin(115200);

    delay(3000);

    Serial.println("\_\_TENSORFLOW IRIS\_\_");

    tf.setNumInputs(4);

    tf.setNumOutputs(3);

    tf.resolver.AddFullyConnected();

    tf.resolver.AddSoftmax();

    while (!tf.begin(irisModel).isOk())

        Serial.println(tf.exception.toString());

}

void loop() {

    if (!tf.predict(x0).isOk()) {

        Serial.println(tf.exception.toString());

        return;

    }

    Serial.print("expcted class 0, predicted class ");

    Serial.println(tf.classification);

    if (!tf.predict(x1).isOk()) {

        Serial.println(tf.exception.toString());

        return;

    }

    Serial.print("expcted class 1, predicted class ");

    Serial.println(tf.classification);

    if (!tf.predict(x2).isOk()) {

        Serial.println(tf.exception.toString());

        return;

    }

    Serial.print("expcted class 2, predicted class ");

    Serial.println(tf.classification);

    Serial.print("It takes ");

    Serial.print(tf.benchmark.microseconds());

    Serial.println("us for a single prediction");

    delay(1000);

}

**Kode Diagram Pada Wokwi**

{

  "version": 1,

  "author": "subairi",

  "editor": "wokwi",

  "parts": [ { "type": "board-esp32-devkit-c-v4", "id": "esp", "top": 0, "left": 0, "attrs": {} } ],

  "connections": [ [ "esp:TX", "$serialMonitor:RX", "", [] ], [ "esp:RX", "$serialMonitor:TX", "", [] ] ],

  "dependencies": {}

}