**Implementasi Model Klasifikasi Iris Flower Menggunakan TensorFlow Lite Micro pada ESP32**

*Ibnu Jaisyurrahman Faiz*

*Program Studi Teknologi Informasi, Fakultas Vokasi, Universitas Brawijaya*

*Email: (*[*ibnufaiz72ub@student.ub.ac.id*](mailto:ibnufaiz72ub@student.ub.ac.id)*)*

**Abstrak:** Penelitian ini bertujuan untuk mengimplementasikan model klasifikasi bunga Iris menggunakan TensorFlow Lite Micro pada mikrokontroler ESP32. Model yang digunakan merupakan hasil pelatihan dengan dataset Iris Flower, yang mengklasifikasikan bunga ke dalam tiga jenis berdasarkan panjang dan lebar sepal serta petal. Program dijalankan pada ESP32 dengan menggunakan pustaka EloquentTinyML untuk memproses model TensorFlow Lite dalam lingkungan terbatas. Hasil pengujian menunjukkan bahwa ESP32 mampu menjalankan prediksi secara real-time dengan akurasi yang baik dan waktu inferensi yang singkat. Implementasi ini membuktikan bahwa mikrokontroler low-power seperti ESP32 dapat digunakan untuk menjalankan aplikasi machine learning ringan, khususnya dalam klasifikasi data sederhana.

Kata Kunci : *ESP32, TensorFlow Lite Micro, EloquentTinyML, Machine Learning, Klasifikasi, Bunga Iris, Embedded AI, TinyML*

**Abstract:** This study aims to implement an Iris flower classification model using TensorFlow Lite Micro on an ESP32 microcontroller. The model is trained on the Iris Flower dataset, which classifies flowers into three species based on sepal and petal length and width. The program runs on the ESP32 using the EloquentTinyML library to process the TensorFlow Lite model within a constrained environment. Test results show that the ESP32 is capable of performing real-time predictions with good accuracy and low inference time. This implementation demonstrates that low-power microcontrollers like the ESP32 can effectively run lightweight machine learning applications, especially for simple classification tasks.

Key Word: *ESP32, TensorFlow Lite Micro, EloquentTinyML, Machine Learning, Classification, Iris Flower, Embedded AI, TinyML*

1. **Pendahuluan**
2. **Latar belakang**

Perkembangan teknologi Machine Learning (ML) tidak hanya terbatas pada perangkat komputer berspesifikasi tinggi, tetapi juga mulai merambah ke perangkat mikrokontroler melalui konsep TinyML (Tiny Machine Learning). TinyML memungkinkan implementasi model pembelajaran mesin pada perangkat dengan sumber daya terbatas, seperti mikrokontroler ESP32. Salah satu tantangan utama adalah bagaimana model yang kompleks dapat dijalankan secara efisien dalam lingkungan dengan memori dan daya komputasi terbatas.

Dalam eksperimen ini, digunakan model klasifikasi Iris Flower, yang merupakan salah satu dataset paling terkenal dalam dunia Machine Learning. Model tersebut dilatih untuk mengklasifikasikan tiga jenis bunga iris berdasarkan empat parameter morfologis: panjang dan lebar sepal serta petal. Dengan mengimplementasikan model ini pada ESP32 menggunakan pustaka TensorFlow Lite Micro dan EloquentTinyML, eksperimen ini menjadi representasi nyata bagaimana sistem kecerdasan buatan dapat diintegrasikan ke dalam perangkat IoT dan sistem tertanam.

1. **Tujuan eksperimen**

* Menerapkan model klasifikasi bunga Iris hasil pelatihan ke dalam mikrokontroler ESP32.
* Menguji kemampuan ESP32 dalam menjalankan prediksi model Machine Learning secara real-time menggunakan TensorFlow Lite Micro.
* Menganalisis performa prediksi (kecepatan dan akurasi) dari model pada lingkungan dengan sumber daya terbatas.
* Menunjukkan potensi penerapan TinyML untuk aplikasi klasifikasi sederhana pada sistem tertanam.

1. **Metodologi**
2. **Alat dan Bahan**

Untuk membuat Implementasi Model Klasifikasi Iris Flower Menggunakan TensorFlow Lite Micro pada ESP32. Berikut adalah daftar alat dan bahan yang digunakan:

* Visual Studio Code
* PlatformIO (plugin untuk VS Code)
* Python + TensorFlow (untuk pelatihan dan konversi model Iris)
* Library:

eloquent\_tinyml

tflm\_esp32

iris\_model.h

* File/Model yang Dibutuhkan:

Dataset Iris Flower

File model .tflite hasil training

File iris\_model.h hasil konversi dari .tflite menggunakan xxd

1. **Langkah Implementasi**

Langkah untuk Implementasi Model Klasifikasi Iris Flower Menggunakan TensorFlow Lite Micro pada ESP32 sebagai berikut:

* Install Visual Studio Code.
* Install ekstensi PlatformIO IDE.
* Buat proyek baru menggunakan PlatformIO dengan board ESP32 Dev Module dan framework Arduino.
* Gunakan Python + TensorFlow untuk melatih model klasifikasi bunga Iris.
* Simpan model ke format .tflite menggunakan:

converter = tf.lite.TFLiteConverter.from\_keras\_model(model)

tflite\_model = converter.convert()

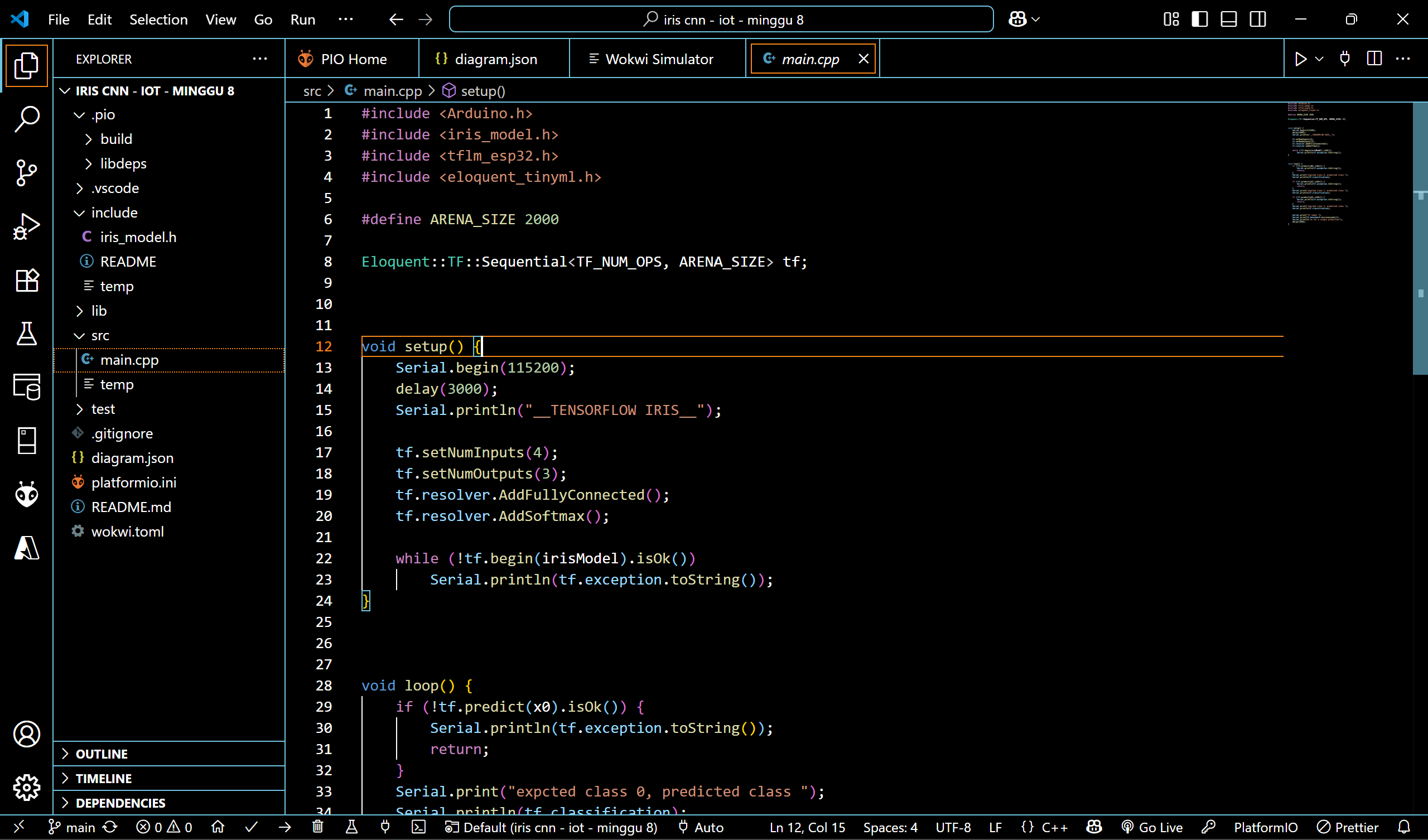
* Konversi model .tflite ke .h menggunakan perintah:

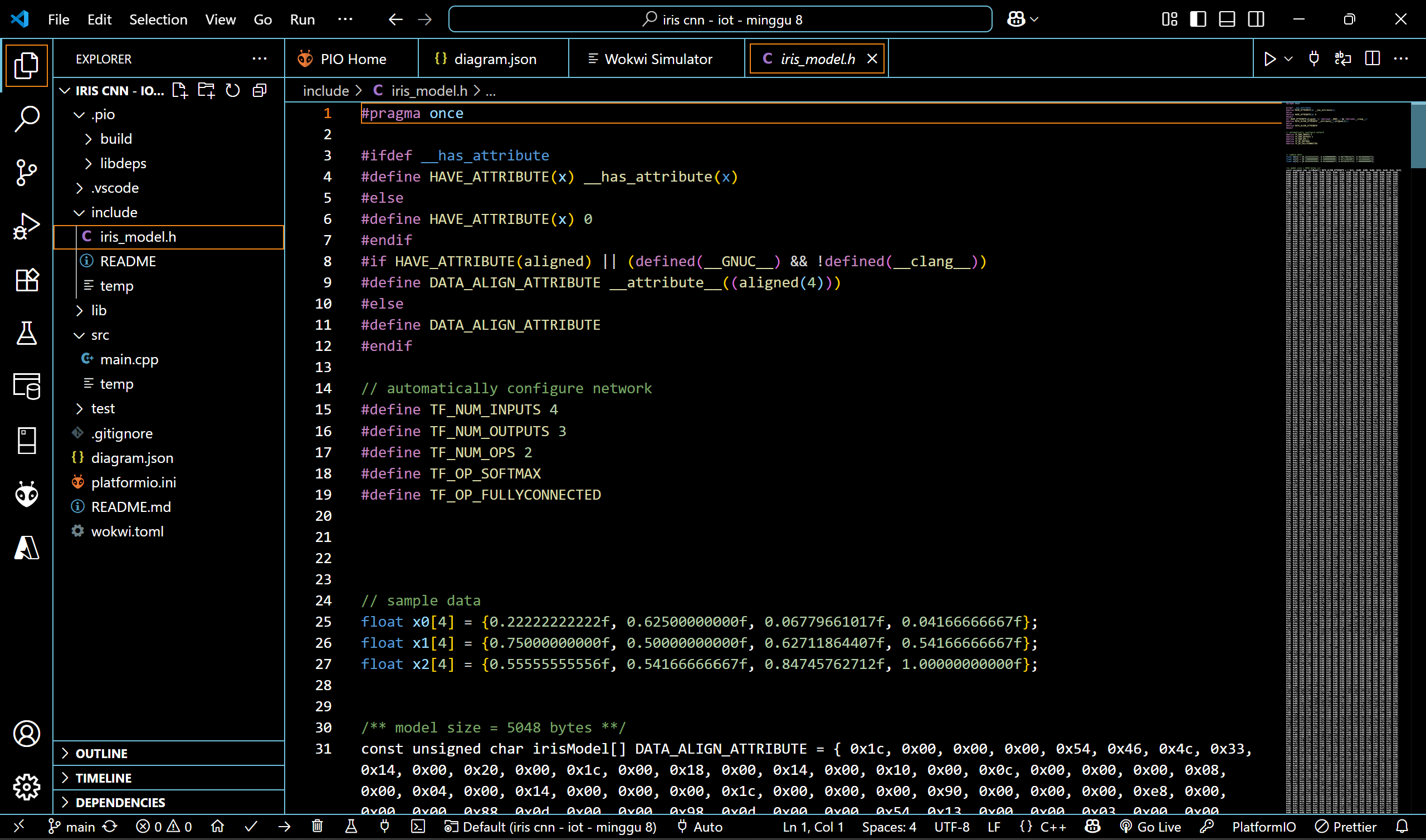
xxd -i iris\_model.tflite > iris\_model.h

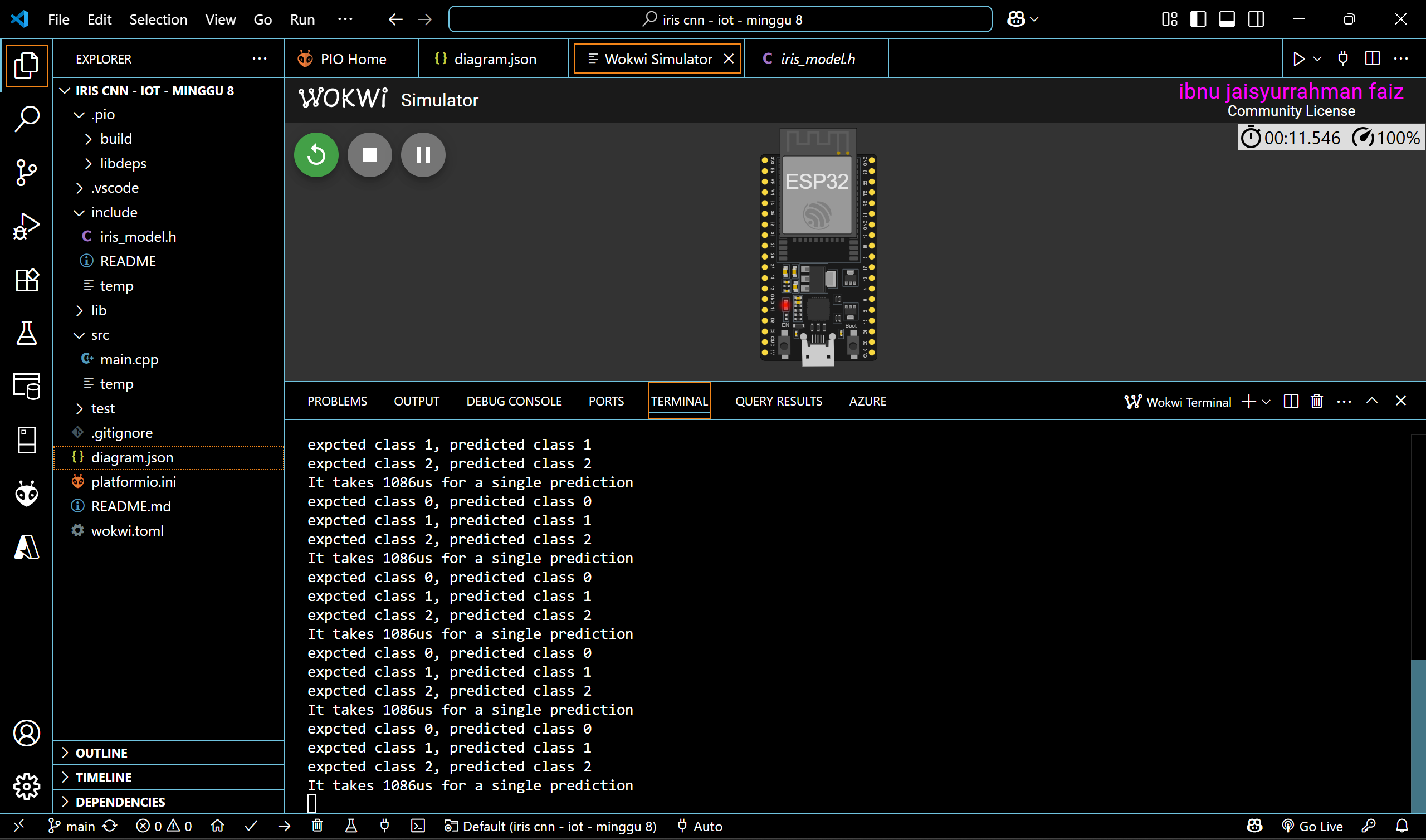
* Salin file iris\_model.h ke dalam folder include/ atau src/.
* Jalankan proses build di PlatformIO.
* Cek apakah kompilasi berhasil dan tidak ada error.

1. **Hasil dan Pembahasan**
2. **Hasil Eksperimen**

Hasil eksperimen menunjukkan bahwa model klasifikasi Iris Flower berhasil dijalankan dengan baik menggunakan TensorFlow Lite Micro dalam lingkungan simulasi berbasis PlatformIO di Visual Studio Code. Tiga sampel data uji yang mewakili masing-masing kelas (class 0, 1, dan 2) diproses berulang kali, dan hasil prediksinya selalu sesuai dengan kelas yang diharapkan. Prediksi untuk input x0, x1, dan x2 masing-masing secara konsisten menghasilkan class 0, 1, dan 2. Hal ini menunjukkan bahwa model mampu mengenali pola data secara akurat. Selain itu, waktu yang dibutuhkan untuk satu kali inferensi adalah sebesar 1086 mikrodetik, yang mencerminkan efisiensi waktu proses prediksi dalam konteks perangkat dengan sumber daya terbatas. Meskipun muncul peringatan terkait duplikasi operasi saat inisialisasi model (“Calling AddBuiltin with the same op more than once is not supported”), proses inferensi tetap berjalan lancar tanpa error yang mengganggu, dan hasil tetap akurat. Secara keseluruhan, eksperimen membuktikan bahwa model klasifikasi sederhana dapat diimplementasikan dan dijalankan secara efektif di lingkungan mikrokontroler menggunakan TinyML, bahkan tanpa perangkat keras fisik.







1. **Lampiran**

* Kode program main.cpp

#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 program diagram.json

{

  "version": 1,

  "author": "ibnu jaisyurrahman faiz",

  "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": {}

}

}