Jurnal Elkolind Volume 13, Nomor 1, Mei 2026
DOI: http://dx.doi.org/10.33795/elkolind.v13i1.9587

Implementasi *Edge AI* Menggunakan *Neural Network* pada Mesin Sortir Berbasis Arduino

Wahyu Ramadhani Gusti[1*)], Nurfauziah[2]

[1,2]Jurusan Pendidikan Teknik Elektronika, Universitas Negeri Makassar, Jalan Mallengkeri Raya, Makassar, Indonesia

*****Penulis Korespondensi, e-mail: **wahyu.ramadhani.gusti@unm.ac.id**

Received: 08/02/2026 *Revised: 13/04/2026* *Accepted:23/04/2026*

ABSTRAK

Penelitian ini membahas penerapan *Edge Artificial Intelligence* berbasis *neural network* pada sistem klasifikasi warna menggunakan sensor TCS3200 dan mikrokontroler Arduino. Penelitian didasari oleh kebutuhan sistem yang mampu bekerja secara mandiri, akurat, dan *real-time* pada perangkat dengan keterbatasan sumber daya komputasi. Tujuannya adalah merancang dan mengimplementasikan model *Artificial Neural Network* yang ringan namun andal untuk melakukan klasifikasi warna secara langsung. Metode yang digunakan meliputi pengambilan dataset warna RGB, normalisasi data agar sesuai dengan fungsi aktivasi sigmoid, serta pelatihan model *Multilayer Perceptron* (MLP) menggunakan algoritma *backpropagation*. Arsitektur jaringan terdiri dari tiga neuron input (R, G, B), dua *hidden layer* dengan masing-masing 12 dan 8 neuron, serta lima neuron output yang merepresentasikan kelas warna. Proses pelatihan dilakukan menggunakan 100 data latih dengan *learning rate* sebesar 0,1 dan menghasilkan nilai *Mean Squared Error* (MSE) sebesar 0,0099. Evaluasi sistem menggunakan 50 data uji menunjukkan tingkat akurasi klasifikasi mencapai 100% pada seluruh kelas warna. Hasil penelitian ini menunjukkan bahwa pendekatan *Edge AI* berbasis neural network efektif diterapkan pada sistem klasifikasi warna berbasis *embedded system* dengan performa yang stabil dan andal.

Kata Kunci: Edge Artificial Intelligence, Klasifikasi Warna, Artificial Neural Network, Embedded System, Sensor TCS3200

ABSTRACT

This study presents the implementation of Edge Artificial Intelligence based on a neural network for a color classification system using a TCS3200 sensor and an Arduino microcontroller. The research is motivated by the need for a system capable of operating independently, accurately, and in real time on devices with limited computational resources. The objective of this study is to design and implement a lightweight yet reliable Artificial Neural Network model for direct color classification. The research methodology includes RGB color data acquisition, data normalization to match the sigmoid activation function, and training of a Multilayer Perceptron (MLP) model using the backpropagation algorithm. The network architecture consists of three input neurons (R, G, B), two hidden layers with 12 and 8 neurons respectively, and five output neurons representing color classes. The training process was conducted using 100 training samples with a learning rate of 0.1, resulting in a Mean Squared Error (MSE) of 0.0099. System evaluation using 50 testing samples demonstrated a classification accuracy of 100% across all color classes. These results indicate that the proposed Edge AI-based neural network approach is effective for real-time color classification on embedded systems, providing stable and reliable performance.

Keywords: *Edge Artificial Intelligence, Color Classification, Artificial Neural Network, Embedded System, TCS3200 Sensor*

1. PENDAHULUAN

Perkembangan pesat *Artificial Intelligence* (AI) telah mendorong pergeseran paradigma komputasi dari pemrosesan terpusat berbasis *cloud* menuju pemrosesan terdistribusi pada perangkat *edge computing*. Pendekatan ini dikenal sebagai *Edge AI*, proses inferensi model kecerdasan buatan dilakukan langsung pada perangkat dengan komponen terbatas pada *microcontroller* [1]. Proses tersebut mampu menurunkan latensi, meningkatkan efisiensi energi, dan menjaga privasi data [2]. Dalam beberapa tahun terakhir, kemunculan

konsep *Tiny Machine Learning (TinyML)* memungkinkan implementasi model *neural network* ringan pada perangkat *embedded*, termasuk platform Arduino, yang sebelumnya dianggap tidak memadai untuk menjalankan algoritma pembelajaran mesin [3], [4]. Kondisi ini membuka peluang luas bagi pengembangan sistem cerdas berskala kecil, baik untuk kebutuhan industri ringan maupun sebagai media pembelajaran berbasis praktik.

Salah satu aplikasi yang relevan dan banyak digunakan dalam otomasi sederhana adalah mesin sortir berbasis warna, yang umum diterapkan pada proses pemilahan objek di bidang manufaktur, pertanian, serta sistem pelatihan dan praktikum teknik [5], [6]. Sebagian besar implementasi mesin sortir warna berbasis Arduino masih mengandalkan pendekatan konvensional, seperti ambang batas (*treshold*) nilai RGB [7], metode statistik sederhana, atau algoritma klasifikasi klasik dengan kompleksitas rendah [8]. Pendekatan tersebut relatif mudah diimplementasikan, namun memiliki keterbatasan dalam menghadapi variasi intensitas cahaya, *noise* sensor, serta kompleksitas pola warna yang lebih beragam. Oleh karena itu, dibutuhkan pendekatan yang lebih adaptif dan *robust*, salah satunya melalui penerapan *neural network* yang mampu memodelkan hubungan non linier pada data warna secara lebih efektif.

Penelitian ini bertujuan untuk mengimplementasikan *Edge AI* menggunakan *neural network* pada mesin sortir berbasis Arduino untuk klasifikasi warna. Fokus penerapan pada optimalisasi model agar dapat dijalankan secara *on-device* pada perangkat dengan keterbatasan memori dan komputasi [9]. Pendekatan yang digunakan meliputi pengumpulan data warna sebagai dataset pelatihan, perancangan arsitektur *neural network*, *training* model pada komputasi eksternal, serta optimasi model sebelum diimplementasikan pada *microcontroller* Arduino. Seluruh proses inferensi dilakukan secara lokal tanpa ketergantungan pada koneksi internet sehingga sistem dapat beroperasi secara *real-time* dan mandiri.

Beberapa penelitian sebelumnya telah membahas penerapan perangkat Arduino serta *neural network* untuk keperluan pengenalan atau klasifikasi warna. Salah satu penelitian yaitu pengembangan sistem klasifikasi warna cabai berbasis Arduino menggunakan sistem *treshold* [7], [10], namun belum memanfaatkan *neural network* yang dioptimalkan untuk *edge device*. Di sisi lain, kajian mengenai *TinyML* dan *Edge AI* menunjukkan bahwa *neural network* dengan jumlah parameter terbatas dapat dijalankan secara efisien pada *microcontroller* melalui optimasi model dan pemanfaatan *framework* ringan seperti *TensorFlow Lite for Microcontrollers* [11]. Meski demikian, penelitian yang secara spesifik mengintegrasikan neural network berbasis *Edge AI* pada mesin sortir warna berbasis Arduino masih relatif terbatas dalam literatur ilmiah. Hasil penelitian diharapkan dapat memberikan kontribusi nyata baik dari sisi pengembangan teknologi *Edge AI* pada perangkat terbatas maupun sebagai referensi penelitian lanjutan di bidang *embedded intelligence*.

2. METODE PENELITIAN

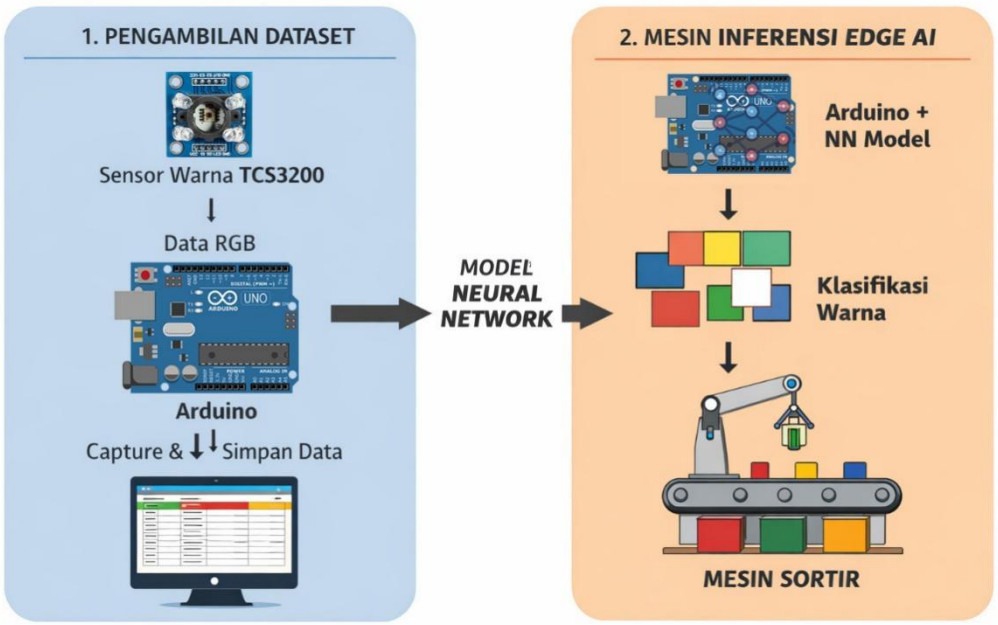
Penelitian yang dilaksanakan merupakan penelitian eksperimental dan rekayasa (*engineering research*). Secara garis besar, prosesnya meliputi perancangan sistem, implementasi, dan evaluasi. Sistem yang dirancang merupakan *prototype* mesin sortir warna berbasis *Edge AI* menggunakan *neural network* pada platform Arduino [5], [12]. Pendekatan yang digunakan adalah pendekatan kuantitatif eksperimental yang dikembangkan secara bertahap. Proses ini terdiri dari pengambilan data warna, pelatihan model *artificial neural network (ANN)*, implementasi model ke dalam perangkat *edge*, hingga evaluasi kinerja sistem [13]. Metode penelitian disusun secara kronologis untuk memastikan setiap tahapan pemecahan masalah berjalan sistematis dan terukur.

2.1 Blok Diagram Penelitian

Sistem terdiri atas dua bagian utama, yaitu tahap pengambilan dataset dan tahap mesin inferensi berbasis *Edge AI* [4]. Pada tahap pengambilan dataset, sensor warna TCS3200 digunakan untuk membaca intensitas warna objek dalam bentuk gabungan komponen *Red, Green, dan Blue (RGB)*. Beberapa sampel yang dibaca pada penelitian seperti warna merah, kuning, hijau, biru, dan ungu. Sensor TCS3200 terhubung

langsung ke Arduino yang berfungsi sebagai *controller* utama sekaligus perangkat akuisisi data [5], [14]. Data RGB yang diperoleh dikirim dan disimpan sebagai dataset *training* melalui program *capture* berbasis Arduino.

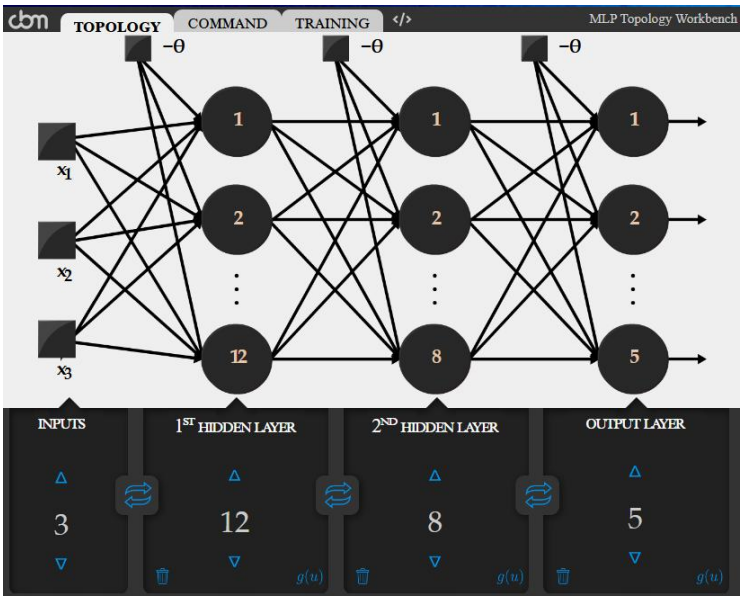
Tahap berikutnya adalah pembuatan mesin inferensi (*inference engine*), model *neural network* yang telah dilatih dan optimalisasi diimplementasikan langsung pada Arduino [15]. Model ini menerima input berupa nilai RGB hasil pembacaan sensor dan menghasilkan keluaran berupa kelas warna hasil prediksi. Sistem dapat diterapkan langsung pada mesin sortir agar hasil klasifikasi dan prediksi warna lebih akurat. Dengan demikian, keseluruhan sistem bekerja secara *standalone* tanpa ketergantungan pada pemrosesan eksternal [16]. Lebih jelas, diagram proses penelitian ditunjukkan pada Gambar1.



Gambar 1: Blok Diagram Penelitian

2.2 Pemilihan Metode *Neural Network*

Metode klasifikasi yang digunakan dalam penelitian ini adalah *Artificial Neural Network (ANN)* dengan algoritma *backpropagation*. *Nodes ANN* dihubungkan dalam bentuk *Multi Layer Perceptron (MLP)*. Pemilihan ANN didasarkan pada kemampuannya dalam memodelkan hubungan non linier antara *input* dan *output* data warna yang dipengaruhi oleh variasi intensitas cahaya dan *noise* sensor [10]. Arsitektur jaringan terdiri dari tiga neuron pada *input layer*, yang merepresentasikan tiga warna primer *RGB (Red, Green, Blue)*. *Output layer* dirancang untuk menghasilkan lima kelas warna, sesuai dengan jumlah warna yang akan diprediksi pada sistem sortir. Penggunaan lima kelas warna bertujuan untuk menyimulasikan kondisi klasifikasi multi kelas yang umum ditemui pada sistem sortir nyata – arsitektur ANN ditunjukkan pada Gambar 2 [17]. Metode *backpropagation* dipilih karena relatif ringan secara komputasi, stabil untuk dataset berukuran kecil hingga menengah, serta telah banyak digunakan pada implementasi *neural network* di *embedded system* [18]–[20].



Gambar 2: Arsitektur *Artificial Neural Network*

2.3 Proses Pengambilan Dataset

Pengambilan dataset dilakukan menggunakan program *capture* berbasis Arduino ditunjukkan pada Gambar 3. Program yang dirancang khusus untuk membaca dan merekam nilai RGB dari sensor warna TCS3200. Setiap warna objek diambil dalam beberapa kali pengukuran untuk memperoleh variasi data yang representatif terhadap kondisi pencahayaan dan posisi objek. Nilai RGB hasil pembacaan kemudian disimpan sebagai dataset mentah. Selanjutnya, dataset tersebut melalui tahap normalisasi, agar nilai input berada dalam rentang yang sesuai dengan fungsi aktivasi sigmoid. Normalisasi ini penting untuk mempercepat proses konvergensi jaringan dan mencegah terjadinya saturasi pada neuron [21].

rgbCapture

```
16  pinMode(tombol, INPUT_PULLUP);
17
18  // Setting frequency-scaling to 20%
19  digitalWrite(S0,HIGH);
20  digitalWrite(S1,LOW);
21  Serial.println("Membuat Datasheet Sensor"); Serial.println("------------------------");
22  Serial.println("Tekan tombol jika sudah siap"); Serial.println("Tekan lagi jika sudah selesai");
23  Serial.println();
24  delay(500);
25 }
26
27 void loop(void) {
28   if (digitalRead(tombol)==LOW){
29     delay(500);
30     while(1){
31
32       float r, g, b, c;
33       digitalWrite(S2,LOW); digitalWrite(S3,LOW);
34       r = pulseIn(Out, LOW);
35       delay(100);
36       digitalWrite(S2,HIGH); digitalWrite(S3,HIGH);
37       g = pulseIn(Out, LOW);
38       delay(100);
39       digitalWrite(S2,LOW); digitalWrite(S3,HIGH);
40       b = pulseIn(Out, LOW);
41       delay(100);
42
43       float red = r / 90.0 ;
44       float green = g / 90.0;
```

Done compiling.

Gambar 3: Program *Capture* Dataset

Program *rgbCapture* digunakan untuk melakukan pengambilan dataset warna berbasis sensor TCS3200 yang terhubung dengan Arduino. Proses pengambilan data dipicu menggunakan tombol, sensor membaca intensitas warna merah (R), hijau (G), dan biru (B) secara bergantian melalui pengaturan pin seleksi

warna. Nilai RGB diperoleh dari durasi pulsa keluaran sensor menggunakan fungsi pulseIn. Setiap variabel pembacaan warna dibagi 90 agar sesuai dengan rentang input fungsi aktivasi sigmoid. Dataset yang telah dinormalisasi selanjutnya digunakan sebagai data latih pada tahap *training* model.

2.4 Multi Layer Perceptron dengan Backpropagation

Model neural network yang digunakan adalah *Multi Layer Perceptron (MLP)* dengan satu atau lebih *hidden layer*, dilatih menggunakan algoritma *backpropagation*. Proses pelatihan dilakukan secara iteratif yang bertujuan untuk meminimalkan nilai *error* antara keluaran jaringan dan target yang diharapkan. Parameter kinerja utama yang digunakan dalam pelatihan adalah *Mean Squared Error (MSE)*, dihitung pada setiap iterasi pelatihan. Nilai MSE digunakan sebagai indikator tingkat kesalahan jaringan, sedangkan jumlah iterasi menunjukkan seberapa cepat jaringan mencapai kondisi konvergen. Proses pelatihan otomatis berhenti ketika nilai MSE telah berada di bawah ambang batas tertentu. Model dengan nilai MSE terkecil dipilih sebagai model terbaik untuk diimplementasikan pada tahap inferensi [22].

2.4.1 Keluaran Neuron pada MLP

Untuk neuron ke-j, keluaran jaringan dirumuskan sebagai:

$$y_j = f\left(\sum_{i=1}^{n} w_{ij}x_i + b_j\right) \quad (1)$$

dengan:

- y_j : keluaran neuron ke-j
- x_j : input atau hasil neuron layer sebelumnya
- b_j : bias

2.4.2 Fungsi aktivasi sigmoid

Nilai keluaran neuron selanjutnya melewati fungsi aktivasi guna membatasi keluaran neuron agar stabil pada pembelajaran *backpropagation* menggunakan persamaan:

$$f(x) = \frac{1}{1+e^{-x}} \quad (2)$$

dengan:

- $f(x)$: fungsi aktivasi
- x : jumlah terbobot input neuron
- e : bilangan *euler*

2.4.3 *Mean Squared Error (MSE)*

Parameter kinerja utama pada proses pelatihan dinyatakan dengan MSE:

$$MSE = \frac{1}{N}\sum_{k=1}^{N}(t_k - y_k)^2 \quad (3)$$

dengan:

- MSE : *Mean Squared Error*
- N : jumlah data pelatihan
- t_k : nilai target ke-k
- y_k : keluaran jaringan ke-k

2.4.4 Pembaruan bobot dengan algoritma backpropagation

Bobot jaringan diperbarui secara iteratif menggunakan:

$$w_{ij}(t+1) = w_{ij}(t) - \eta\frac{\partial \text{MSE}}{\partial w_{ij}} \quad (4)$$

dengan:

- $w_{ij}(t+1)$: bobot baru
- $w_{ij}(t)$: bobot sekarang
- η : *learning rate*

- $\frac{\partial MSE}{\partial w_{ij}}$: gradien *error* terhadap bobot

2.5 Mesin Inferensi (*Edge AI*)

Tahap mesin inferensi merupakan implementasi konsep Edge AI, di mana model *neural network* hasil pelatihan dijalankan langsung pada Arduino. Bobot dan bias hasil pelatihan ditanamkan ke dalam program Arduino sehingga proses inferensi dapat dilakukan secara *real-time* tanpa memerlukan koneksi ke komputer atau *cloud server*. Arduino menerima input nilai RGB dari sensor TCS3200, memprosesnya, dan menghasilkan keluaran berupa kelas warna hasil prediksi. Implementasi inferensi secara *on-device* ini bertujuan untuk membuktikan bahwa *neural network* dapat dijalankan secara efisien pada perangkat dengan keterbatasan memori dan komputasi, sekaligus memenuhi kebutuhan sistem sortir warna berbasis Edge AI.

rgbClassifier | model.h

```
#include <Neurona.h>
#include "model.h"
#define S0 2
#define S1 3
#define S2 4
#define S3 5
#define Out 6

#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x3F,20,4);
MLP mlp(NET_INPUTS, NET_OUTPUTS, layerSizes, MLP::LOGISTIC, initW, true);

void setup() {
  Serial.begin(9600);
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(S3, OUTPUT);
  pinMode(Out, INPUT_PULLUP);

  // Setting frequency-scaling to 20%
  digitalWrite(S0,HIGH);
  digitalWrite(S1,LOW);
  lcd.begin(20,4); //Pemanggilan LCD
  lcd.init();
```

rgbClassifier | model.h

```
#define NET_INPUTS 3
#define NET_HIDDEN_1 12
#define NET_HIDDEN_2 8
#define NET_OUTPUTS 5

double netInput[] = { -1.00, 0.00, 0.00, 0.00};
int layerSizes[] = {NET_HIDDEN_1, NET_HIDDEN_2, NET_OUTPUTS, -1};

char *Class[] = {"merah", "kuning", "hijau", "biru", "ungu"};
char *mlpClass = "";

double PROGMEM const initW[] = {4.575207471958185,
0.4526222952091551,
0.8650592003943107,
2.0961435887738875,
3.019411872905911,
0.49706470613300213,
0.7364865495048404,
1.228065362672895,
-0.31227934020469655,
1.3437971884997382,
0.7231656194288554,
1.489173047475894,
2.499999242094216,
0.029800469455146656,
```

Gambar 4: Program Mesin Inferensi (*rgbClassifier*)

Program *rgbClassifier* pada Gambar 4 merepresentasikan tahap implementasi mesin inferensi berbasis *Edge AI* dengan memanfaatkan *Neurona Library*. Bobot optimal *Multi Layer Perceptron (MLP)* hasil pelatihan ditanamkan langsung ke dalam sistem. Arsitektur jaringan didefinisikan dengan tiga neuron *input* yang merepresentasikan nilai RGB, dua *hidden layer* dengan jumlah neuron masing-masing 12 dan 8, serta lima neuron *output* yang merepresentasikan kelas warna. Bobot awal hasil pelatihan dimuat melalui berkas model.h dan disimpan dalam memori program untuk efisiensi penggunaan memori. Proses inferensi dilakukan menggunakan fungsi aktivasi sehingga Arduino mampu mengklasifikasikan warna secara *real-time* berdasarkan data sensor dengan hasil prediksi ditampilkan sebagai kelas warna yang sesuai.

2.6 Evaluasi Sistem

Evaluasi sistem dilakukan untuk mengukur kinerja dan keandalan mesin sortir warna yang telah dikembangkan. Parameter evaluasi utama meliputi akurasi klasifikasi, nilai *Mean Squared Error (MSE)*, serta jumlah iterasi pelatihan yang diperlukan untuk mencapai konvergensi. Akurasi sistem dihitung dengan membandingkan hasil prediksi warna terhadap label sebenarnya pada data uji. Dilakukan pula uji fungsionalitas sistem, yaitu pengujian langsung pada prototipe mesin sortir untuk memastikan bahwa sistem mampu membaca warna, melakukan inferensi, dan menghasilkan keluaran sesuai dengan kelas warna yang diharapkan. Keterkaitan antara nilai MSE, jumlah iterasi pelatihan, dan tingkat akurasi dianalisis untuk menilai efektivitas metode *neural network* yang diterapkan pada platform *Edge AI* berbasis Arduino.

3. HASIL DAN PEMBAHASAN

3.1 Karakteristik Dataset

Dataset warna yang diperoleh melalui program *rgbCapture* terdiri dari pengukuran nilai RGB sejumlah 20 sampel untuk setiap kelas warna. Warna yang dilatih yaitu merah, kuning, hijau, biru, dan ungu sehingga total yang digunakan dalam proses pelatihan yaitu sebanyak 100 data. Di sisi lain, sebanyak 50 data uji masing-masing 10 yang digunakan sesuai kelas warna menjadikan total data yang proses yaitu 150. Pengambilan data dilakukan pada kondisi pencahayaan yang terkontrol untuk meminimalkan variasi intensitas cahaya eksternal yang dapat memengaruhi hasil pembacaan sensor. Variasi objek warna juga diperhatikan dengan menggunakan beberapa tingkat intensitas warna, mulai dari cerah hingga gelap, guna memperkaya distribusi data pada setiap kelas. Setiap warna dicetak pada media kertas dengan pola grid yang disesuaikan dengan dimensi sensor untuk memastikan konsistensi pengukuran. Nilai RGB yang diperoleh kemudian dinormalisasi agar sesuai dengan rentang *input* fungsi aktivasi sigmoid. Beberapa contoh sampel data warna terukur dan normalisasi disajikan pada Tabel I.

TABEL I : Sampel Dataset Nilai RGB Terukur dan Normalisasi

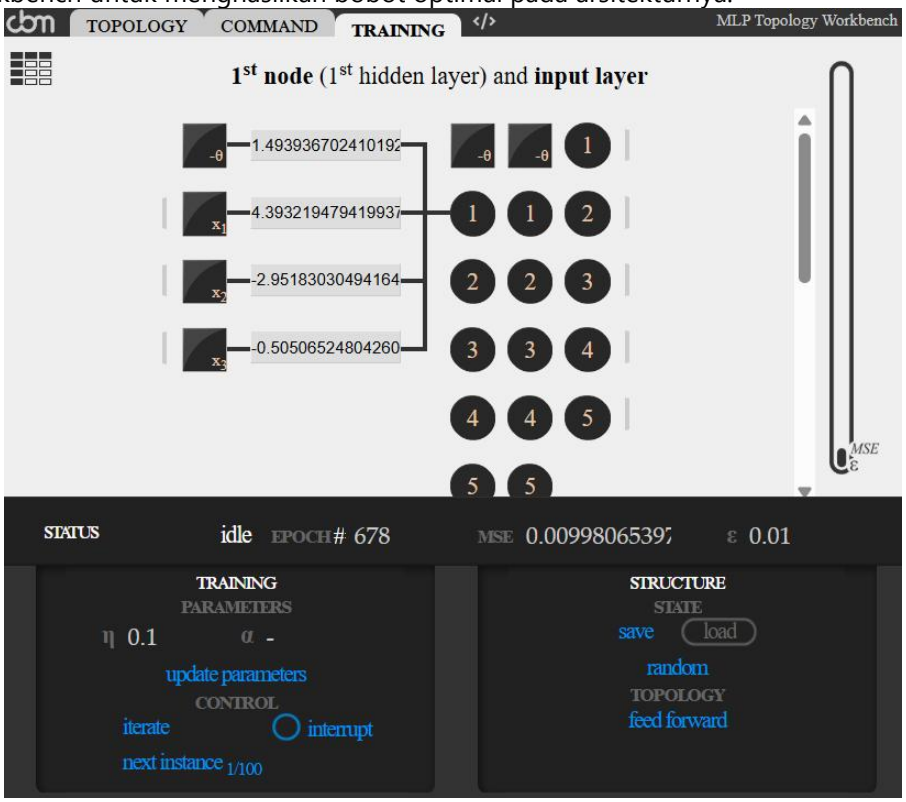
No	Warna	Terukur			Normalisasi		
		R	G	B	R	G	B
1	Merah	66	105	95	0,73	1,17	1,06
		74	106	99	0,82	1,18	1,10
		58	99	83	0,64	1,10	0,92
		73	109	98	0,81	1,21	1,09
		53	94	81	0,59	1,04	0,90
2	Kuning	31	40	68	0,34	0,44	0,76
		31	48	67	0,34	0,53	0,74
		33	56	72	0,37	0,62	0,80
		31	46	69	0,34	0,51	0,77
		36	53	74	0,40	0,59	0,82
3	Hijau	108	81	99	1,20	0,90	1,10
		101	83	101	1,12	0,92	1,12
		112	75	87	1,24	0,83	0,97
		116	94	107	1,29	1,04	1,19
		103	100	114	1,14	1,11	1,27
4	Biru	85	63	72	0,94	0,70	0,80
		79	53	32	0,88	0,59	0,36
		120	89	51	1,33	0,99	0,57
		127	104	71	1,41	1,16	0,79
		89	61	38	0,99	0,68	0,42
5	Ungu	188	248	182	2,09	2,76	2,02
		91	113	79	1,01	1,26	0,88
		76	96	59	0,84	1,07	0,66
		104	103	66	1,16	1,14	0,73
		95	117	83	1,05	1,3	0,92

Dataset menunjukkan bahwa setiap kelas warna memiliki pola distribusi RGB yang berbeda, meskipun beberapa kelas memiliki perbedaan kecil nilai komponen warna. Dataset yang dihasilkan tersebut berasal dari tingkatan warna, mulai dari yang cerah hingga gelap. Variasi warna tersebut bertujuan agar hasil ANN lebih andal dalam memprediksi klasifikasi warna. Komponen RGB terukur kemudian dinormalisasi agar sesuai *range* fungsi aktivasi yang digunakan. Normalisasi data terbukti penting untuk menghindari saturasi fungsi aktivasi sigmoid dan mempercepat proses pelatihan.

4.2 Pelatihan Neural Network

Model *Multi Layer Perceptron* dengan struktur [3–12–8–5] dilatih menggunakan algoritma *backpropagation*. Proses *training* dilakukan hingga mencapai konvergensi pada *Mean Squared Error (MSE)*

yang stabil di bawah ambang batas yang telah ditetapkan (<0.01). *Tool* yang digunakan pada prosesnya yaitu MLP Topology Workbench untuk menghasilkan bobot optimal pada arsitekturnya.



Gambar 5: Hasil *Training Artificial Neural Network*

Berdasarkan Gambar 5, setiap bobot mengalami penyesuaian secara iteratif untuk meminimalkan *error* keluaran. Hasil *training* telah mencapai *epoch* ke-678 dengan nilai MSE sebesar 0,00998 yang berada di bawah ambang batas kesalahan (ε = 0,01). Hal ini mengindikasikan jaringan telah mempelajari pola warna dari data latih secara efektif. *Ouput* dari *training* menghasilkan bobot optimal yang menghubungkan setiap neuron antar layer. Model yang berhasil selanjutnya digunakan pada tahap inferensi di perangkat Arduino sebagai sistem *Edge AI* untuk klasifikasi warna.

4.3 Implementasi Inferensi pada Arduino

Setelah tahap pelatihan, bobot model hasil training disimpan dan dimasukkan ke dalam program Arduino (model.h). Sistem inferensi berjalan secara *real-time*, membaca nilai RGB dari sensor TCS3200, memprosesnya melalui *forward propagation* menggunakan jaringan terlatih, dan kemudian mengklasifikasikan warna. Sebanyak 50 data uji yang digunakan untuk memvalidasi keandalan sistem dalam mengklasifikasi dan memprediksi warna. Data tersebut merupakan komposisi nilai warna primer baru yang telah diatur menjadi warna merah, kuning, hijau, biru, dan ungu. Data warna kemudian dicetak dan dilakukan pengujian seperti pada Gambar 6. Adapun data *confusion matrix* hasil klasifikasi pengujian ditunjukkan pada Tabel II.

TABEL II : *Confusion Matrix* Hasil Pengujian

	Prediksi Merah	**Prediksi Kuning**	**Prediksi Hijau**	**Prediksi Biru**	**Prediksi Ungu**
Aktual Merah	10				
Aktual Kuning		10			
Aktual Hijau			10		
Aktual Biru				10	
Aktual Ungu					10

Tabel II menunjukkan hasil pembacaan terhadap data uji untuk mengevaluasi kinerja sistem pengenalan warna yang terdiri dari lima kelas, yaitu merah, kuning, hijau, biru, dan ungu. Setiap kelas warna memiliki nilai prediksi yang tepat, menunjukkan bahwa seluruh data uji dari masing-masing kelas berhasil dikenali sesuai warna aktualnya. Setiap warna diuji sebanyak 10 sampel, sehingga total ada 50 data pengujian. Dokumentasi pengujian warna dengan memanfaat sistem berbasis *Edge AI* ditampilkan pada Gambar 6.

Gambar 6: Dokumentasi Pengujian 50 Data Warna

Hasil pengujian membuktikan bahwa sistem pengenalan warna yang tertanam pada Arduino mampu bekerja secara optimal dalam melakukan klasifikasi warna secara akurat. Tingkat akurasi yang mencapai 100% pada data uji mengindikasikan bahwa model jaringan syaraf tiruan yang digunakan memiliki kemampuan

generalisasi yang baik terhadap sampel pengujian. Capaian ini dikarenakan karakteristik dataset dan data uji yang digunakan dalam kondisi terkontrol dan homogen, menggunakan objek berupa kertas warna cetak dengan variasi intensitas nilai RGB. Lingkup pengujian yang spesifik ini menyebabkan distribusi data uji relatif konsisten, sehingga model mampu mengenali pola warna dengan sangat baik tanpa gangguan variasi eksternal yang signifikan. Dengan performa tersebut, sistem ini diharapkan dapat dikembangkan lebih lanjut dan diterapkan pada implementasi nyata, seperti mesin sortir otomatis berbasis *Edge AI*.

4.4 Diskusi Analitis

Hasil penelitian ini menunjukkan beberapa poin penting yang perlu perhatikan:

1. Peran Normalisasi: Normalisasi nilai RGB sebelum masuk ke jaringan memiliki dampak signifikan pada stabilitas pelatihan dan akurasi akhir. Tanpa normalisasi, fungsi sigmoid cenderung saturasi, yang memperlambat konvergensi dan menurunkan performa prediksi.
2. Kompleksitas Model vs Resource Arduino: Struktur jaringan yang relatif kecil (total neuron di *hidden layer* 20) dipilih untuk menyeimbangkan kebutuhan representasi pola warna dan keterbatasan memori serta komputasi perangkat Arduino. Hasil membuktikan bahwa model sederhana ini cukup efektif untuk klasifikasi warna sederhana.
3. *Generalizability*: Meskipun akurasi keseluruhan tinggi, kemungkinan kesalahan prediksi kecil masih dapat terjadi. Hal ini terlihat pada beberapa pola komponen warna data *training* yang berbeda. Hal ini dapat menjadi fokus perbaikan lanjutan, misalnya dengan menambah jumlah sampel, memperluas variasi pencahayaan saat *capture* dataset, menggunakan barang lebih variatif untuk klasifikasi berbasis warna atau mengeksplorasi fungsi aktivasi dan arsitektur lainnya.
4. Relevansi Edge AI: Implementasi model langsung pada *embedded system* tanpa koneksi *cloud* membuktikan prinsip *Edge AI* efektif untuk aplikasi *real-time* sederhana. Hal ini selaras dengan tren TinyML di literatur terkini yang menekankan penghematan latensi dan privasi data [23], [24].

4. KESIMPULAN

Penelitian ini berhasil merancang dan mengimplementasikan sistem *Edge AI* berbasis *Neural Network* pada mesin sortir warna menggunakan Arduino. Model klasifikasi warna dibangun menggunakan arsitektur *Multi Layer Perceptron (MLP)* dengan tiga neuron pada *input layer* (R, G, B), dua *hidden layer* masing-masing berisi 12 dan 8 neuron, serta lima neuron pada *output layer* yang merepresentasikan kelas warna. Proses pelatihan menunjukkan konvergensi yang baik dengan nilai MSE di bawah ambang batas ($\varepsilon = 0{,}01$), menandakan bahwa model mampu mempelajari hubungan non linier antara data RGB dan kelas warna secara efektif. Implementasi model hasil pelatihan ke dalam Arduino membuktikan bahwa pendekatan *Edge AI* memungkinkan proses inferensi dilakukan langsung *embedded system* tanpa ketergantungan pada *cloud computing*. Penelitian ini juga menunjukkan kebaruan pada integrasi proses *training neural network* dengan penerapan inferensi langsung di Arduino. Sistem yang dikembangkan memiliki potensi untuk diterapkan pada aplikasi nyata seperti mesin sortir otomatis skala kecil, sistem inspeksi visual sederhana, serta otomasi industri berbasis warna. Pengembangan selanjutnya dapat diarahkan pada peningkatan akurasi melalui optimasi arsitektur jaringan dan penambahan variasi dataset. Penggunaan sensor warna dengan resolusi yang lebih tinggi juga dapat diterapkan untuk meningkatkan keandalan sistem di lingkungan operasional yang dinamis.

DAFTAR PUSTAKA

[1] T. Meuser *et al.*, "Revisiting Edge AI: Opportunities and Challenges," *IEEE Internet Comput.*, vol. 28, no. 4, pp. 49–59, 2024, doi: 10.1109/MIC.2024.3383758.
[2] E. Badidi, K. Moumane, and F. El Ghazi, "Opportunities, Applications, and Challenges of Edge-AI Enabled Video Analytics in Smart Cities: A Systematic Review," *IEEE Access*, vol. 11, pp. 80543–80572, 2023, doi: 10.1109/ACCESS.2023.3300658.
[3] P. Warden and D. Situnayake, *Tinyml: Machine learning with tensorflow lite on arduino and ultra-low-power microcontrollers*. O'Reilly Media, 2019.
[4] C. Banbury *et al.*, "Micronets: Neural network architectures for deploying tinyml applications on commodity microcontrollers," *Proc.*

Mach. Learn. Syst., vol. 3, pp. 517–532, 2021.
[5] M. Rahman, H. A. Hasan, Y. A. Emon, A. Hossain, A. R. Rokon, and M. H. Bhuyan, "Advanced Color Sorting Conveyor System Using Arduino and TCS3200 Color Sensor for Precise Color Classification," in *2025 4th International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)*, IEEE, 2025, pp. 476–481.
[6] F. Arifin and R. J. Surya, "Image-Based Sorting Machine as Training Kit for Artificial Neural Network Learning in Intelligent Control Systems Course," in *The 8th International Conference on Education Innovation (ICEI 2024)*, Atlantis Press, 2025, pp. 1449–1460.
[7] N. M. Dwicahyo, R. Wulanningrum, and R. A. Ramadhani, "Penyortiran Buah Jeruk Dengan Ekraksi Ciri Rgb To Hsv Menggunakan Naïve Bayes," *Pros. SEMNAS INOTEK (Seminar Nas. Inov. Teknol.*, vol. 8, no. 2, pp. 894–901, Jul. 2024, doi: 10.29407/INOTEK.V8I2.5018.
[8] Risaldi, Mukramin, R. Suppa, and V. I. Wahyuni, "Rancang Bangun Alat Pelacak Posisi Kendaraan Berbasis IoT," *JUTINDA (Jurnal Tek. Inform. Unanda)*, pp. 29–33, May 2024.
[9] P. Samad, K. Jayadi, G. C. Arnanto, A. Sabril, and I. Sukma, "Sistem Irigasi Otomatis IoT Off-Grid untuk Optimalisasi Budidaya Cabai di Wilayah Pedesaan Tanpa Listrik," *J. Pengabdi. Masy.*, vol. 3, no. 2, pp. 215–218, 2025.
[10] D. Nofeano Masmur and M. Snae, "Rancang Bangun Sistem Sortir Buah Cabai Berdasarkan Warna Berbasis Arduino UNO," *HOAQ (High Educ. Organ. Arch. Qual. J. Teknol. Inf.*, vol. 15, no. 1, pp. 33–42, May 2024, doi: 10.52972/HOAQ.VOL15NO1.P33-42.
[11] Y. Chen, B. Zheng, Z. Zhang, Q. Wang, C. Shen, and Q. Zhang, "Deep Learning on Mobile and Embedded Devices: State-of-the-art, Challenges, and Future Directions," *ACM Comput. Surv.*, vol. 53, no. 4, Jul. 2021, doi: 10.1145/3398209;ISSUE:ISSUE:DOI.
[12] F. Arifin, M. Habiburrahman, and W. R. Gusti, "Classification of Organic and Inorganic Waste Types Based on Neural Networks," *Elinvo (Electronics, Informatics, Vocat. Educ.*, vol. 8, no. 1, pp. 78–85, Dec. 2023, doi: 10.21831/elinvo.v8i1.53284.
[13] I. R. Fachrezzy SA, "Implementasi Adaptive Neuro-Fuzzy Inference System (ANFIS) untuk Deteksi Otomatis Jenis Buah Berdasarkan Citra Warna dan Bentuk Menggunakan Matlab." Universitas Islam Sultan Agung Semarang, 2025.
[14] M. Lamsani, R. A. Pangestika, M. Cahyanti, and E. R. Swedia, "Sistem identifikasi warna tanah Munsell menggunakan sensor warna TCS3200 dan kelembaban YL-69," *Sebatik*, vol. 27, no. 1, pp. 379–389, 2023.
[15] S. N. M. Al-Faydi and H. N. Y. Al-Talb, "IoT and artificial neural network-based water control for farming irrigation system," in *2022 2nd International Conference on Computing and Machine Intelligence (ICMI)*, IEEE, 2022, pp. 1–5.
[16] M. M. Ali, C. Harshavardhan, G. S. Teja, B. V. Jyothi, and L. S. Kumar, "Intelligent Waste Sorting System: Leveraging Arduino for Automated Trash Identification and Categorization," *Int. J. Comput. Inf. Syst. Ind. Manag. Appl.*, vol. 16, no. 3, p. 16, 2024.
[17] "MLP Topology Workbench." http://www.moretticb.com/MTW/ (accessed Feb. 01, 2026).
[18] Y. Zhang, J. Yu, Y. Chen, W. Yang, W. Zhang, and Y. He, "Real-time strawberry detection using deep neural networks on embedded system (rtsd-net): An edge AI application," *Comput. Electron. Agric.*, vol. 192, p. 106586, 2022.
[19] Z. Zhang and J. Li, "A review of artificial intelligence in embedded systems," *Micromachines*, vol. 14, no. 5, p. 897, 2023.
[20] W. Caesarendra *et al.*, "An embedded system using convolutional neural network model for online and real-time ECG signal classification and prediction," *Diagnostics*, vol. 12, no. 4, p. 795, 2022.
[21] E. Simanjuntak, N. Khairina, Z. Sembiring, R. Muliono, and M. Muhathir, "Analisis Fungsi Aktivasi pada Algoritma Backpropagation dalam Pengenalan Aksara Batak Toba," *JUSTINDO (Jurnal Sist. Dan Teknol. Inf. Indones.*, vol. 8, no. 2, pp. 99–107, 2023.
[22] E. Ashraf, A. E. Kabeel, Y. Elmashad, S. A. Ward, and W. M. Shaban, "Predicting solar distiller productivity using an AI Approach: Modified genetic algorithm with Multi-Layer Perceptron," *Sol. Energy*, vol. 263, p. 111964, 2023.
[23] H. Bourekouche, "TinyML for low-power embedded intelligent systems: A review," *Available SSRN 5431335*, 2025.
[24] A. Patel and F. Al-Fayez, "TinyML on the Edge: Model Compression, On-Device Learning, and Energy–Latency Trade-Offs," *Multidiscip. Eng. Sci. Open*, vol. 2, pp. 1–12, 2025.