

Pengembangan Algoritma Pengambilan Keputusan Untuk Menentukan Status Anak Pada Aplikasi Deteksi Stunting

1st Fariz Rahman Ramadhan

Fakultas Teknik Elektro

Universitas Telkom

Bandung, Indonesia

farizrahman@student.telkomuniversity.ac.id

2nd Casi Setianingsih

Fakultas Teknik Elektro

Universitas Telkom

Bandung, Indonesia

setiacasie@telkomuniversity.ac.id

3rd Ashri Dinimaharwati

Fakultas Teknik Elektro

Universitas Telkom

Bandung, Indonesia

ashridini@telkomuniversity.ac.id

Abstrak—Stunting merupakan masalah serius dalam kesehatan masyarakat Indonesia, khususnya di daerah-daerah yang masih sulit mendapatkan akses terhadap makanan-makanan bergizi, daerah dengan sanitasi yang kurang memadai, dan daerah dengan tingkat kesadaran yang rendah mengenai pentingnya gizi dan kesehatan anak sehingga diperlukan aplikasi yang dapat melakukan *early prediction* terhadap status anak agar orang tua mengetahui anak tersebut normal atau kekurangan nutrisi. Dalam membuat aplikasi tersebut tentu saja diperlukan algoritma pengambilan keputusan pada aplikasi tersebut, untuk mendapatkan model K-Nearest Neighbors (KNN) yang baik, dibutuhkan metode *split dataset*, pengujian, nilai K, dan pengujian dataset. Dengan begitu akan ditemukan kekurangan pada kinerja aplikasi NutriScan dalam pengambilan keputusan.

Kata Kunci: Stunting, Aplikasi deteksi, Algoritma pengambilan keputusan

I. PENDAHULUAN

Stunting, merupakan sebuah kondisi di mana pertumbuhan anak balita terhambat, telah menjadi permasalahan kesehatan global yang cukup signifikan. Meskipun telah ada penurunan prevalensi stunting secara global dalam beberapa dekade terakhir, permasalahan ini masih menciptakan dampak buruk pada pengembangan sosial dan ekonomi di berbagai negara, terutama di negara-negara berkembang. Kementerian Kesehatan mengumumkan hasil Survei Status Gizi Indonesia (SSGI) pada Rapat Kerja Nasional BKKBN, Rabu (25/1) dimana prevalensi stunting di Indonesia turun dari 24,4% di tahun 2021 menjadi 21,6% di 2022 [1].

Fakta bahwa ada sebelas wilayah di Jawa Barat yang memiliki prevalensi stunting lebih tinggi daripada rata-rata provinsi (20,2%) menunjukkan bahwa upaya pencegahan dan penanganan stunting yang lebih komprehensif diperlukan di wilayah-wilayah tersebut untuk mencapai target global WHO yang menetapkan prevalensi stunting di bawah 20% [2]. Oleh karena itu, identifikasi dini stunting menjadi kunci dalam pencegahan dan intervensi yang efektif.

Namun langkah tersebut masih menjadi tantangan tersendiri, di banyak daerah, terutama daerah terpencil dan memiliki akses kesehatan yang terbatas seperti Kecamatan Bojongsoang, melakukan deteksi dini stunting adalah hal yang sulit. Adanya kesenjangan informasi, pengetahuan, dan akses

teknologi antara wilayah-wilayah ini membuat kompleksitas masalah bertambah.

Karena itu, pengembangan aplikasi deteksi stunting, merupakan sebuah langkah yang baik untuk memberikan sarana yang memudahkan identifikasi stunting pada tahap awal, sehingga langkah-langkah intervensi yang tepat dapat segera dilakukan. Pendekatan ini melibatkan berbagai disiplin ilmu, mulai dari pengembangan teknologi, analisis data, hingga pengaplikasian pengetahuan medis dan psikologi anak, membentuk solusi kompleks untuk masalah yang juga kompleks.

Agar semua hal tersebut dapat berjalan dengan baik diperlukan pembuatan aplikasi menggunakan algoritma *image processing* untuk mengukur tinggi badan, lingkaran lengan, dan lingkaran kepala balita serta algoritma pengambilan keputusan untuk menentukan apakah balita tersebut terkena stunting atau tidak.

II. KAJIAN TEORI

Pembuatan aplikasi NutriScan membutuhkan algoritma *Decision Making* untuk pengambilan keputusan. Dari angka-angka yang didapatkan dari algoritma *image processing*, data diolah menggunakan algoritma KNN untuk menentukan apakah balita yang sudah diperiksa tersebut masuk kedalam kategori balita normal atau balita yang stunting.

K-Nearest Neighbors (KNN) merupakan algoritma klasifikasi yang bersifat non-parametrik dalam pembelajaran mesin. Algoritma ini didasarkan pada konsep bahwa data dengan atribut yang serupa cenderung berkumpul dalam ruang fitur yang sama. Menentukan nilai K yang optimal adalah langkah penting dalam pengujian algoritma KNN. Nilai K mewakili jumlah tetangga terdekat yang akan digunakan untuk melakukan klasifikasi atau regresi[3]. Pemilihan nilai K yang tepat dapat mempengaruhi akurasi model secara signifikan. Nilai K yang memberikan performa terbaik rata-rata di seluruh lipatan dianggap sebagai nilai K optimal. Metode ini membantu mengurangi variabilitas hasil dan memberikan estimasi yang lebih robust mengenai performa model.

Metode *split dataset* adalah teknik yang umum digunakan untuk menguji dan mengevaluasi kinerja model KNN. Dalam metode ini, dataset dibagi menjadi dua subset: subset pelatihan (*training set*) dan subset pengujian (*testing set*). Subset pelatihan digunakan untuk melatih model, sementara

subset pengujian digunakan untuk menguji performa model yang telah dilatih [8]. Proporsi pembagian dataset biasanya bervariasi, dengan pembagian umum seperti 70% untuk pelatihan dan 30% untuk pengujian, atau 80% untuk pelatihan dan 20% untuk pengujian. Teknik ini membantu menghindari overfitting dan memberikan indikasi yang lebih realistis mengenai kinerja model pada data yang belum pernah dilihat sebelumnya.

III. METODE

Dalam pembuatan aplikasi NutriScan untuk mendeteksi stunting pada anak menggunakan algoritma *image processing* dan pengambilan keputusan, metode yang dimulai dari:

A. Pembuatan dan Pengujian Algoritma *Decision Making*

Dari angka-angka yang didapatkan dari algoritma *image processing*, data diolah menggunakan algoritma K-Nearest Neighbors (KNN) untuk menentukan apakah balita yang sudah diperiksa tersebut masuk kedalam kategori balita normal atau balita yang stunting. Setelah itu Model KNN diinisialisasi dengan menggunakan `KNeighborsClassifier` dari `scikit-learn`. `n_neighbors=1` menunjukkan bahwa model akan mempertimbangkan hanya satu tetangga terdekat untuk melakukan prediksi.

B. Pengujian Algoritma K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) merupakan algoritma klasifikasi yang bersifat *non-parametrik* dalam pembelajaran mesin. Algoritma ini didasarkan pada konsep bahwa data dengan atribut yang serupa cenderung berkumpul dalam ruang fitur yang sama. Implementasi KNN pada kodingan melibatkan beberapa langkah. Pertama-tama, memisahkan fitur (X) dan label (y) dari dataframe.

Pengujian algoritma dilakukan dengan beberapa metode untuk mendapatkan model KNN yang baik diantaranya, Metode split dataset yang umum digunakan untuk menguji dan mengevaluasi kinerja model KNN, menentukan nilai K yang optimal adalah langkah penting dalam pengujian algoritma KNN. Nilai K mewakili jumlah tetangga terdekat yang akan digunakan untuk melakukan klasifikasi atau regresi [3], dan pengujian dataset, untuk memastikan hasil pengujian stunting yang dilakukan pada aplikasi memiliki hasil yang sama dengan data yang ada pada dataset. Tujuan dari pengujian ini untuk memastikan apakah dataset yang digunakan tersebut dapat digunakan dalam pengujian secara *real time* dan hasil pengujian tersebut sesuai dengan dataset.

IV. HASIL DAN PEMBAHASAN

A. Pembuatan dan Pengujian Algoritma *Decision Making*

Dari angka-angka yang didapatkan dari algoritma *image processing*, data diolah menggunakan algoritma KNN untuk menentukan apakah balita yang sudah diperiksa tersebut masuk kedalam kategori balita normal atau balita yang stunting.

KNN adalah salah satu metode klasifikasi yang paling sederhana dan mudah dipahami dalam machine learning. Dalam KNN, prediksi kelas suatu data baru didasarkan pada mayoritas kelas dari k tetangga terdekatnya dalam ruang fitur. Proses ini dilakukan dengan mengukur jarak antara data baru dan setiap titik data dalam dataset, lalu memilih k titik data terdekat tersebut.

Kelas yang paling sering muncul di antara k tetangga tersebut kemudian ditetapkan sebagai prediksi kelas untuk data baru tersebut. Meskipun sederhana dalam konsepnya, KNN telah terbukti efektif dalam berbagai aplikasi, terutama ketika pola kelas relatif mudah dipisahkan dalam ruang fitur. Namun, pemilihan nilai k yang optimal dan efisiensi komputasinya menjadi perhatian penting dalam penerapan praktisnya [7].

K-Nearest Neighbors (KNN) merupakan algoritma klasifikasi yang bersifat *non-parametrik* dalam pembelajaran mesin. Algoritma ini didasarkan pada konsep bahwa data dengan atribut yang serupa cenderung berkumpul dalam ruang fitur yang sama. Implementasi KNN pada kodingan melibatkan beberapa langkah. Pertama-tama, memisahkan fitur (X) dan label (y) dari dataframe. Fitur adalah semua kolom kecuali 'status', sedangkan label adalah kolom 'status'. Selanjutnya, fitur-fitur dalam X dinormalisasi menggunakan `StandardScaler` dari `scikit-learn`. Ini dilakukan untuk memastikan bahwa semua fitur memiliki skala yang serupa, yang dapat meningkatkan kinerja beberapa model. Rumus untuk normalisasi fitur menggunakan `StandardScaler` adalah:

$$X_{Scaled} = \frac{x - \mu}{\sigma}$$

Rumus menjelaskan bahwa x adalah nilai asli dari fitur, μ adalah rata-rata dari fitur, dan σ adalah standar deviasi dari fitur.

Selanjutnya Model KNN diinisialisasi dengan menggunakan `KNeighborsClassifier` dari `scikit-learn`. `n_neighbors=1` menunjukkan bahwa model akan mempertimbangkan hanya satu tetangga terdekat untuk melakukan prediksi. Intinya dengan K = 1, model KNN akan menggunakan hanya satu tetangga terdekat untuk membuat prediksi kelas untuk titik data baru, jika K dibuat menjadi 3 atau = 3 maka KNN akan menggunakan 3 tetangga terdekat untuk membuat prediksi. K diatur menjadi 1 karena merupakan nilai dengan akurasi yang paling besar jika dibandingkan dengan nilai K ganjil yang lainnya.



GAMBAR 1
Perbandingan nilai K

TABLE 1
Percobaan Nilai K dan Akurasinya

Nama K=	Nilai Akurasi
1	0,85
3	0,65
5	0,40
7	0,45
9	0,50
11	0,50
13	0,45
15	0,45

17	0,35
19	0,40
21	0,55

Data dibagi menjadi data latih dan data uji. KNN kemudian mengukur jarak antara setiap titik data uji dengan semua titik data latih menggunakan metrik jarak seperti *Euclidean distance* atau metrik lainnya. Pengukuran jarak ini membantu menemukan tetangga terdekat dari setiap titik data uji.

Setelah nilai k ditentukan, algoritma KNN menghitung jarak antara titik data yang ingin diprediksi dengan semua titik data dalam set pelatihan. Jarak Euclidean sering digunakan sebagai metrik untuk mengukur seberapa dekat satu titik data dengan titik data lainnya. Rumus jarak Euclidean antara dua titik data x dan y adalah:

$$\text{Euclidean Distance} = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

Di mana x dan y adalah koordinat fitur untuk dua titik data yang dibandingkan dan n adalah jumlah fitur.

```
point1 = X_scaled[0] point2
= X_scaled[1]
euclidean_distance = np.sqrt(np.sum((point1 -
point2)**2))
print("Euclidean Distance between point1 and
point2:", euclidean_distance)
```

Kodingan diatas digunakan untuk menghitung Euclidean Distance antara dua titik data dalam dataset yang telah dinormalisasi. Setelah dilakukan nilai Euclidean Distance antara *point1* and *point2*: 2.0782. Jarak Euclidean antara dua titik data pertama dalam dataset menunjukkan bahwa meskipun ada perbedaan antara titik-titik ini, data relatif dekat satu sama lain dalam ruang fitur yang dinormalisasi. Ini memberikan indikasi bahwa data cukup terkluster dengan baik.

Setelah itu, KNN menggunakan metode voting mayoritas untuk memprediksi label kelas dari setiap titik data uji. Dalam konteks kodingan di atas, algoritma mempertimbangkan label mayoritas dari 1 tetangga terdekat untuk memprediksi label kelas. Setelah prediksi dilakukan, kinerja model dievaluasi menggunakan matrikskebingungan dan laporan klasifikasi. Model dievaluasi menggunakan *confusion matrix* dan *classification report*. *Confusion matrix* memberikan informasi tentang jumlah prediksi yang benar dan salah, sedangkan *classification report* memberikan metrik evaluasi seperti *precision*, *recall*, dan *f1-score* untuk setiap kelas.

Confusion matrix dapat dihitung dengan cara menghitung jumlah true positive (TP), false positive (FP), true negative (TN), dan false negative (FN). Rumusnya tidak langsung terlibat dalam langkah ini, tetapi perhitungan dilakukan berdasarkan prediksmodel dan label sebenarnya. Sedangkan *Classification Report*: Rumus untuk metrik evaluasi seperti *precision*, *recall*, dan *f1-score* adalah:

$$\begin{aligned} \text{Presisi} &= \frac{TP}{TP+FP} \\ \text{Recall} &= \frac{TP}{TP+FN} \\ \text{F1-Score} &= 2 \times \frac{\text{Presisi} \times \text{Recall}}{\text{Presisi} + \text{Recall}} \end{aligned}$$

Gambar 2 Rumus Confusion matrix

Di sini, TP (*True Positive*) adalah jumlah prediksi positif yang benar, FP (*False Positive*) adalah jumlah prediksi positif yang salah, dan FN (*False Negative*) adalah jumlah prediksi negatif yang salah.

KNN memiliki kelebihan karena sifatnya yang sederhana dan mudah diimplementasikan, namun juga memiliki kelemahan seperti sensitivitas terhadap skala fitur dan memerlukan penyimpanan data pelatihan lengkap di memori. Oleh karena itu, pemilihan parameter yang tepat seperti jumlah tetangga dan metrik jarak, serta pemahaman tentang karakteristik dataset, sangat penting untuk mendapatkan hasil yang optimal dengan KNN [6].

B. Pengujian Algoritma K-Nearest Neighbors (KNN)

KNN menggunakan metode voting mayoritas untuk memprediksi label kelas dari setiap titik data uji. Algoritma mempertimbangkan label mayoritas dari 1 tetangga terdekat untuk memprediksi label kelas. Setelah prediksi dilakukan, kinerja model dievaluasi menggunakan matriks kebingungan dan laporan klasifikasi. Model dievaluasi menggunakan *confusion matrix* dan *classification report*. *Confusion matrix* memberikan informasi tentang jumlah prediksi yang benar dan salah, sedangkan *classification report* memberikan metrik evaluasi seperti *precision*, *recall*, dan *f1-score* untuk setiap kelas.

KNN memiliki kelebihan karena sifatnya yang sederhana dan mudah diimplementasikan, namun juga memiliki kelemahan seperti sensitivitas terhadap skala fitur dan memerlukan penyimpanan data pelatihan lengkap di memori. Oleh karena itu, pemilihan parameter yang tepat seperti jumlah tetangga dan metrik jarak, serta pemahaman tentang karakteristik dataset, sangat penting untuk mendapatkan hasil yang optimal dengan KNN [5].

1. Split Database

Metode split dataset adalah teknik yang umum digunakan untuk menguji dan mengevaluasi kinerja model KNN. Dalam metode ini, dataset dibagi menjadi dua subset: subset pelatihan (training set) dan subset pengujian (testing set). Subset pelatihan digunakan untuk melatih model, sementara subset pengujian digunakan untuk menguji performa model yang telah dilatih[3]. Proporsi pembagian dataset biasanya bervariasi, dengan pembagian umum seperti 70% untuk pelatihan dan 30% untuk pengujian, atau 80% untuk pelatihan dan 20% untuk pengujian. Teknik ini membantu menghindari overfitting dan memberikan indikasi yang lebih realistis mengenai kinerja model pada data yang belum pernah dilihat sebelumnya.

Classification Report:				
	precision	recall	f1-score	support
konsul	0.81	0.96	0.88	84
microcephaly	0.95	0.81	0.87	95
normal	0.74	0.89	0.81	87
stunting	0.53	0.51	0.52	95
underweight	0.83	0.78	0.80	94
wasting	0.89	0.81	0.85	93
accuracy			0.79	548
macro avg	0.79	0.79	0.79	548
weighted avg	0.79	0.79	0.79	548

GAMBAR 3

Hasil Split Data Dengan 80% Data Pengujian

Gambar di atas menunjukkan pada percobaan dengan data dibagi menjadi 60% untuk pengujian dan 40% untuk pelatihan, dengan akurasi mencapai 90%. Meskipun lebih banyak data digunakan untuk pengujian, model tetap memberikan hasil yang sangat baik.

TABLE 2
Percobaan test_size dan Akurasinya

Test Size	Presisi	Recall	F1-Score	Akurasi
0,1	0,92	0,92	0,92	0,90
0,2	0,92	0,92	0,92	0,91
0,3	0,92	0,92	0,92	0,91
0,4	0,91	0,91	0,91	0,91
0,5	0,87	0,86	0,87	0,86
0,6	0,91	0,90	0,90	0,90
0,7	0,87	0,86	0,87	0,86
0,8	0,79	0,79	0,79	0,79
0,9	0,64	0,64	0,62	0,64

Performa model KNN dievaluasi berdasarkan berbagai ukuran metrik seperti presisi, recall, F1-score, dan akurasi untuk berbagai ukuran dataset pengujian (test size) yang berbeda. Semakin besar test size, artinya semakin besar proporsi data yang digunakan untuk pengujian, dan semakin representatif hasil evaluasi terhadap data yang sebenarnya. Presisi, recall, F1-score, dan akurasi cenderung stabil dan tinggi saat test size berada di sekitar 0,1 hingga 0,4, dengan. Ketika test size lebih besar dari 0,4, terjadi fluktuasi kecil dalam performa model, meskipun tetap relatif tinggi.

2. Pengujian Nilai K

Menentukan nilai K yang optimal adalah langkah penting dalam pengujian algoritma KNN. Nilai K mewakili jumlah tetangga terdekat yang akan digunakan untuk melakukan klasifikasi atau regresi[4]. Pemilihan nilai K yang tepat dapat mempengaruhi akurasi model secara signifikan. Nilai K yang memberikan performa terbaik rata-rata di seluruh lipatan dianggap sebagai nilai K optimal. Metode ini membantu mengurangi variabilitas hasil dan memberikan estimasi yang lebih robust mengenai performa model.

Classification Report:				
	precision	recall	f1-score	support
konsul	1.00	1.00	1.00	18
microcephaly	0.92	0.96	0.94	23
normal	0.96	0.87	0.92	31
stunting	0.76	0.76	0.76	25
underweight	1.00	1.00	1.00	16
wasting	0.88	0.96	0.92	24
accuracy			0.91	137
macro avg	0.92	0.92	0.92	137
weighted avg	0.91	0.91	0.91	137

GAMBAR 4

Hasil Pengujian Nilai K Dengan K=1

Pengujian pada model KNN dengan K = 1 memberikan akurasi yang cukup bagus, yaitu 0,91. Ini menunjukkan efektivitas model dalam memprediksi data uji hanya berdasarkan tetangga terdekatnya, memberikan indikasi kuat bahwa fitur yang digunakan relevan dan mampu membedakan kelas dengan jelas.

Classification Report:				
	precision	recall	f1-score	support
konsul	1.00	1.00	1.00	18
microcephaly	0.92	1.00	0.96	23
normal	0.97	0.97	0.97	31
stunting	0.91	0.84	0.87	25
underweight	1.00	1.00	1.00	16
wasting	0.96	0.96	0.96	24
accuracy			0.96	137
macro avg	0.96	0.96	0.96	137
weighted avg	0.96	0.96	0.96	137

GAMBAR 5

Hasil Pengujian Nilai K Dengan K=2

Pada nilai K = 2, model mencapai akurasi tertinggi yaitu 96%. Ini menunjukkan bahwa mempertimbangkan dua tetangga terdekat memberikan keseimbangan optimal antara sensitivitas dan generalisasi, menghasilkan prediksi yang paling akurat.

TABLE 3
Percobaan Nilai K dan Akurasinya

Nilai K	Presisi	Recall	F1-Score	Nilai Akurasi
1	0,92	0,92	0,92	0,91
2	0,96	0,96	0,96	0,96
3	0,94	0,94	0,93	0,93
4	0,95	0,95	0,95	0,95
5	0,91	0,92	0,91	0,91
6	0,92	0,93	0,92	0,92
7	0,90	0,92	0,90	0,91

8	0,93	0,93	0,92	0,93
9	0,88	0,89	0,88	0,88
10	0,86	0,88	0,86	0,87
11	0,86	0,87	0,84	0,86

Nilai K 1 hingga 3 menunjukkan presisi dan recall yang relatif tinggi, namun F1-score dan akurasi cenderung stabil di sekitar 0,93. Pada nilai K 4, terjadi peningkatan yang signifikan dalam F1-score dan akurasi, mencapai 0,97, sementara presisi dan recall tetap tinggi. Nilai K yang lebih besar dari 4, seperti K = 5 hingga 8, menunjukkan presisi dan recall yang menurun sedikit, tetapi F1-score dan akurasi tetap relatif stabil di sekitar 0,93 sampai 0,94.

a. Pengujian Dataset

Pengujian dataset dilakukan dengan memastikan hasil pengujian stunting yang dilakukan pada aplikasi memiliki hasil yang sama dengan data yang ada pada dataset. Tujuan dari pengujian ini untuk memastikan apakah dataset yang digunakan tersebut dapat digunakan dalam pengujian secara real time dan hasil pengujian tersebut sesuai dengan dataset.



GAMBAR 6
Data Balita 1

Dari data pertama dapat dilihat balita bernama Anita dengan jenis kelamin perempuan yang berusia 25 bulan terdeteksi terkena stunting karena sesuai dataset yang didapat dan sesuai dengan WHO dengan tinggi 76,76 cm, z-scorenya itu berada di bawah -2. Dengan tinggi yang di bawah -2 itu sudah bisa dipastikan bahwa balita tersebut terkena stunting.

Selain tingginya, dengan berat badan 7 kg di umur 25 bulan, z-scorenya juga berada di bawah -2. Tanpa perlu pengecekan lanjutan sebenarnya sudah dapat dibilang balita tersebut terkena stunting.



GAMBAR 7
Data Balita 2

Dari data kedua dapat dilihat balita bernama Toni dengan jenis kelamin laki-laki yang berusia 33 bulan terdeteksi terkena underweight karena sesuai dataset yang didapat dan sesuai dengan WHO dengan tinggi 86,98 cm, z-scorenya itu berada di atas -2. Dengan tinggi yang di atas -2 balita tersebut sudah bisa dibilang tidak terkena stunting tapi masih bisa terkena kurang nutrisi lainnya.

Selanjutnya dilihat dari berat badannya yaitu 8,6 kg dan menurut dataset dan WHO berat badan tersebut pada balita berusia 33 bulan itu nilai z-scorenya berada di bawah -2. Karena tinggi badan balita tersebut normal namun berat badannya memiliki z-score di bawah -2 maka Toni mengalami *underweight*.

b. Pengujian Lapangan

Pengujian lapangan adalah metode evaluasi yang dilakukan di lokasi sebenarnya di mana produk atau sistem akan digunakan. Tujuannya adalah untuk mengamati kinerja, keandalan, dan interaksi produk atau sistem dengan lingkungan nyata serta pengguna sebenarnya. Pengujian langsung dilakukan ke beberapa balita di sekitar bojongsong bersama para staf puskesmas untuk mencoba aplikasi secara langsung dan melihat tingkat keakuratan aplikasi serta mendapatkan feedback dari orang tua balita serta para staf puskesmas.



GAMBAR 8
Mencoba Aplikasi Pada Balita

Pengujian dilakukan dengan memperlihatkan cara penggunaan aplikasi Nutriscan kepada orang tua, Bu Reni, dan staf puskesmas, pengujian aplikasi dilakukan dengan melakukan pengukuran tinggi, lingkaran kepala, dan lingkaran lengan menggunakan aplikasi dan berat badan dari hasil

timbangan terakhir di Puskesmas. Hasil pengukuran tersebut dibandingkan dengan hasil pengukuran yang dilakukan secara manual pada puskesmas dan dilihat perbedaannya.

Setelah semua pengukuran selesai dilakukan dilanjutkan dengan pengecekan status balita apakah terkena stunting atau tidak, didapatkan ada 1 balita stunting dan beberapa balita lainnya ada yang terkena *underweight* dan *wasting* dikarenakan semua balita yang diperiksa pada program pembagian PMT dari puskesmas ini memiliki kekurangan nutrisi.

Pengujian pada balita ini dikatakan cukup sulit dikarenakan kebanyakan dari balita tersebut tidak mau berdiri dan hanya mau digendong oleh ibunya jadi didapatkan sedikit *error* dalam pengujian tinggi badannya. Setelah semua pengukuran berhasil dilakukan dilanjutkan dengan pemberian jajanan kepada balita sebagai hadiah kepada mereka.

V. KESIMPULAN

Dari penelitian, pengembangan, dan pengujian yang dilakukan dapat diambil kesimpulan bahwa aplikasi deteksi stunting NutriScan masih butuh pengembangan lebih lanjut untuk mengoptimalkan fungsi deteksi tinggi badan, lebar lingk kepala, dan lebar lingk lengan untuk mendapatkan data yang valid.

Selain itu dari hasil uji lapangan, untuk menggunakan aplikasi pada balita perlu upaya khusus agar balita bisa di analisis menggunakan aplikasi NutriScan, karena itu aplikasi masih membutuhkan pengembangan terhadap algoritma *Image Processing* dan *Decision Making* untuk mendapatkan hasil yang sesuai.

REFERENSI

- [1] Rokom. (2023, January 25). *sehatNegeriku*. Diambil kembali dari sehatNegeriku Kementerian Kesehatan Republik Indonesia: <https://sehatnegeriku.kemkes.go.id/baca/rilis-media/20230125/3142280/pr-evalensi-stunting-di-indonesia-turun-ke-216-dari-244/>
- [2] Fitriani, N. (2023, August 4). *Open Data Jabar*. Diambil kembali dari opendata.jabarprov.go.id: <https://opendata.jabarprov.go.id/id/artikel/data-terbaru-prevalensi-stunting-di-jabar-menurun-43-pencapaian-target-who-semakin-dekat>
- [3] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830.
- [4] Beizer, B. (1995). *Black-Box Testing: Techniques for Functional Testing of Software and Systems*. John Wiley & Sons.
- [5] Parelta, JH. (2023). *Microservice APIs: Using Python, Flask, FastAPI, OpenAPI and More*.
- [6] Zhang, C., & Ma, Y. (2012). k-Nearest neighbor classification algorithm based on attribute weighting. *Pattern Recognition Letters*, 33(11), 1436-1441.
- [7] Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media.
- [8] Han, J., Kamber, M., & Pei, J. (2011). *Data Mining: Concepts and Techniques*. Morgan Kaufmann.