JURNAL SINTAKS LOGIKA Vol. 4 No. 2, Mei 2024, pp. 121-133, **eISSN : 2775-412X**
https://jurnal.umpar.ac.id/index.php/sylog

Aplikasi Penerjemah Huruf *Braille* Berbasis Android

Nur Azizah Ramadana[1*], Ahmad Selao[3]

[1,2]Program Studi Teknik Informatika, Universitas Muhammadiyah Parepare, Indonesia*
**Email :219280039nurazizahramadana@gmail.com*

Abstract: Reading braille requires sensitivity of the hand, memorizing every combination of dots that are formed. Getting to know Braille takes a relatively long time. The aim of this research is to understand Braille letters using an Android-based smartphone.The research method used is descriptive quantitative using the Java, Kotlin, Python programming languages, and the Convolutional Neural Network (CNN) algorithm with the MobilenetV2 architectural model. The results of testing the CNN algorithm with the MobilenetV2 architecture model have an accuracy value of 0.9183. This shows that the application detection performance is less than optimal.

Keywords: Braille; Convolutional Neural Network; Android

1. PENDAHULUAN

Para penyandang tuna netra sangat sulit memperoleh informasi maupun pengetahuan. Ini disebabkan mereka memerlukan alat bantu untuk membaca maupun menulis dengan huruf *braille* (Ronando & Sudaryanto, 2018). Huruf *braille* adalah suatu kode berbentuk titik-titik timbul, dimana titik tersebut kombinasi enam titik yang ditonjolkan pada kertas (Ramiati et al., 2020).

Menurut (Ilham & Rochmawati, 2020) *Convolution Neural Network* (CNN) adalah jaringan berlapis-lapis yang mempelajari, mengekstrak, dan mengklasifikasikan fitur secara otomatis. Kelebihan dari metode CNN adalah dapat secara otomatis mengekstraksi ciri penting dari setiap citra tanpa bantuan manusia, selain itu metode CNN juga lebih efisien dibandingkan metode neural network lainnya terutama untuk memori dan kompleksitas. Menurut (Ilham & Rochmawati, 2020) *Convolution Neural Network* (CNN) adalah metode pembelajaran mendalam yang dapat digunakan untuk mengklasifikasikan gambar, mengelompokkannya berdasarkan kesamaan, dan melakukan pengenalan gambar dalam adegan. CNN memiliki beberapa arsitektur umum yang sering digunakan seperti LeNet-5, AlexNet, Mobilenet, GoogleNet, VGGNet dan ResNet. Masing-masing arsitektur mempunyai kelebihan dan kekurangan masing masing. Penggunaan arsitektur yang digunakan pada CNN sangat mempengaruhi hasil dari klasifikasi.

Menurut (Syarif, 2021) tensorflow *lite* adalah suatu library yang dikembangkan oleh perusahaan Google pada tahun 2018. Tensorflow *lite* dapat mengurangi ukuran file dan meningkatkan kecepatan eksekusi tanpa mengurangi akurasi, sehingga dapat diterapkan ke ponsel android.

Menurut (Sulastio et al., 2021) android adalah sistem operasi berbasis Linux bagi telepon seluler seperti telepon pintar dan komputer tablet. Android juga menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri yang akan digunakan untuk berbagai macam piranti gerak.

Beberapa penelitian yang telah dilakukan sebelumnya (Sulviyani & Novamizanti, 2017) yaitu perancangan konversi *braille* ke teks berbasis android menggunakan *K-Nearst Neighbor* (KNN) sebagai metode klasifikasinya. Inputan citra pada penelitian tersebut berupa file citra *braille* dalam format bitmap. Penelitian selanjutnya yaitu pengenalan karakter *braille* dengan metode *Celullar Neural Network* menggunakan bahasa pemrograman python. Penelitian ini ditujukan untuk membantu tuna netra dalam mempelajari huruf *braille* (Ramadhan, A. F., Putra, A. D., & Surahman, 2021). Selanjutnya penelitian yang dilakukan oleh (Yunus & Anwar, 2022) aplikasi penerjemah bahasa *isyarat* Indonesia ke dalam Huruf abjad. Penelitian ini ditujukan untuk membantu penyandang disabilitas tuna rungu. Penelitian ini menggunakan metode *Convolutional Neural Network* (CNN) dan menggunakan arsitektur mobilenetv2. Peneliti ingin membuat suatu aplikasi pembelajaran agar masyarakat bisa belajar lebih mengenal bahasa isyarat indonesia. Penelitian selanjutnya yang dilakukan (Herlambang et al., 2021) yaitu pengenalan karakter *braille* dengan metode CNN menguji akurasi pengenalan karakter jika akuisisi citra menggunakan *smartphone* dengan kemiringan 0 hingga 4 derajat. Hasil perhitungan secara keseluruhan sistem, didapatkan nilai akurasi sebesar 81.54%.

Dari beberapa penelitian diatas, maka peneliti mengajukan suatu pendekatan baru untuk membuat aplikasi penerjemah huruf *braille* menggunakan huruf abjad sebagai inputan dan outputnya huruf *braille*. Aplikasi ini memanfaatkan *class hashmap* pada bahasa pemrograman java dan *Convolutional Neural Network* model arsitekturnya MobileNetV2.

Penelitian ini bertujuan untuk menerjemahkan huruf *braille* sekaligus menjadi media pembelajaran untuk mengetahui huruf *braille* ke abjad, oleh karena itu dibutuhkan sebuah sistem yang dapat menerjemahakan huruf *braille* ke dalam huruf abjad, sistem itu dapat dipakai melalui *smartphone* android dengan menggunakan metode *Convolutional Neural Network* dan menggunakan model arsitektur MobileNetV2.

2. METODOLOGI PENELITIAN

2.1 Jenis penelitian

Jenis penelitian yang digunakan yaitu kuantitatif deskriptif dan menggunakan bahasa pemrograman Java, Kotlin, dan Python. Penelitian dilakukan melalui sumber data informasi mengenai huruf *braille* dan algoritma *Convolutional Neural Network* (CNN).

2.2 Lokasi dan waktu penelitian

Penelitian ini dilakukan di kota Parepare, adapun waktu yang dipergunakan untuk pelaksanaan penelitian ini adalah selama tiga bulan dari bulan Oktober-desember 2023.

2.3 Alat dan bahan penelitian

a. Alat penelitian
   a) *Reglet* dan *stylus* (alat tulis *braille*)
   b) *Hardware* (Perangkat keras)

Perangkat keras yang digunakan dalam pembuatan aplikasi terjemahan bahasa isyarat ke dalam huruf abjad dapat dilihat dari tabel berikut :

Tabel 1. Spesifikasi *hardware*

Spesifikasi *hardware*	
Laptop/PC	Toshiba Satelite l40-A
Processor laptop	Intel(R) CPU 1037U @1.80GHz (2 CPUs), ~1.8 GHz
RAM laptop	8 GB

   c) *Software* (Perangkat lunak)

Perangkat lunak yang digunakan dalam pembuatan aplikasi terjemahan *braille* ke dalam teks bahasa indonesia dapat dilihat dari tabel sebagai berikut :

Tabel 2. Spesifikasi *software*

Spesifikasi software	
Sistem operasi laptop	Windows 10
Tool IDE	Android studio
Bahasa pemrograman	Java & Kotlin

b. Bahan penelitian

Berupa data-data yang didapat dari perpustakaan, internet maupun referensi-referensi lain mengenai huruf *braille.*

2.4 Tahap Penelitian

Tahap penelitian dilakukan dalam enam tahap, yaitu tahap penelitian, pengumpulan data, analisis, perancangan, pengujian, dan implementasi. Uraian dari keenam tahapan tersebut adalah sebagai berikut :

a. Pengumpulan data, pada tahap ini dilakukan pengumpulan data-data yang menunjang keberhasilan dari penelitian, seperti melakukan studi literatur tentang aplikasi penerjemah huruf *braille*.

b. Analisis data, peneliti melakukan analisa terhadap sistem yang di terapkan sekarang berdasarkan kemudian merumuskan masalah yang menjadi pokok penelitian sehingga dapat dibuat alternatif pemecahan masalah.
c. Perancangan aplikasi, pada tahap ini yang dilakukan perancangan *flowchart*, *UML*, dan desain dari aplikasi.
d. Pembuatan aplikasi, aplikasi dibangun menggunakan *android studio*.
e. Pengujian aplikasi, aplikasi yang telah dibangun akan diuji keberhasilannya dalam melakukan penerjemahan huruf *braille* ke dalam huruf abjad.
f. Implementasi, implementasi akan dilakukan setelah pengujian terhadap aplikasi berhasil dilakukan.

2.4 Metode Pengujian

a. *White box testing*

Whitebox adalah pengujian yang dikembangkan berdasarkan pada kode program. Penguji dalam white box testing harus memiliki pengetahuan tentang kode dan penulisan kasus uji dengan parameter yang sesuai (Nugraha, 2022).

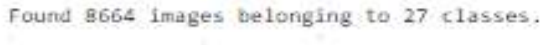
b. *Black box*

Semua navigasi dan tombol fasilitas dan proses program lain berjalan tanpa *error* sesuai dengan hasil pengujian yang dilakukan, namun aplikasi telah menetapkan aturan dan harus dipatuhi, karena jika diabaikan, sistem akan menolak perintah kepada pengguna yang tidak mengikuti aturan sistem yang berjalan persyaratan untuk pemrosesan lebih lanjut yang tidak sesuai seperti kesalahan dalam memberikan informasi kepada pengguna karena data yang tidak lengkap atau tidak sesuai yang akan dimasukkan oleh sistem. Pengujian *black box* memiliki peran penting dalam pengujian perangkat lunak yaitu untuk memvalidasi fungsi keseluruhan sistem apakah telah bekerja dengan baik (Parlika et al., 2020).

3. HASIL DAN PEMBAHASAN

3.1. Rancangan Training Data

Proses training diawali dengan membuat model dasar dengan metode *image processing* atau *image classifier*. Model arsitektur yang digunakan adalah MobileNetV2. Dataset *image* yang digunakan berjumlah 8664 *image* dengan jumlah *class* 27. *Class* merupakan sebuah *object* yang akan dideteksi jumlah huruf *braille*.



Gambar1. Dataset *image*

Training dilakukan beberapa percobaan dengan parameter berbeda yaitu data *epoch* untuk melakukan 50 putaran proses training pada data set, *Learning Rate* atau waktu proses training pada dataset, dan *batch size* 32 untuk jumlah sampel yang akan ditraining. Hingga kemudian mendapatkan hasil *accuracy* mencapai 0.9183 dengan nilai loss 0.2284.

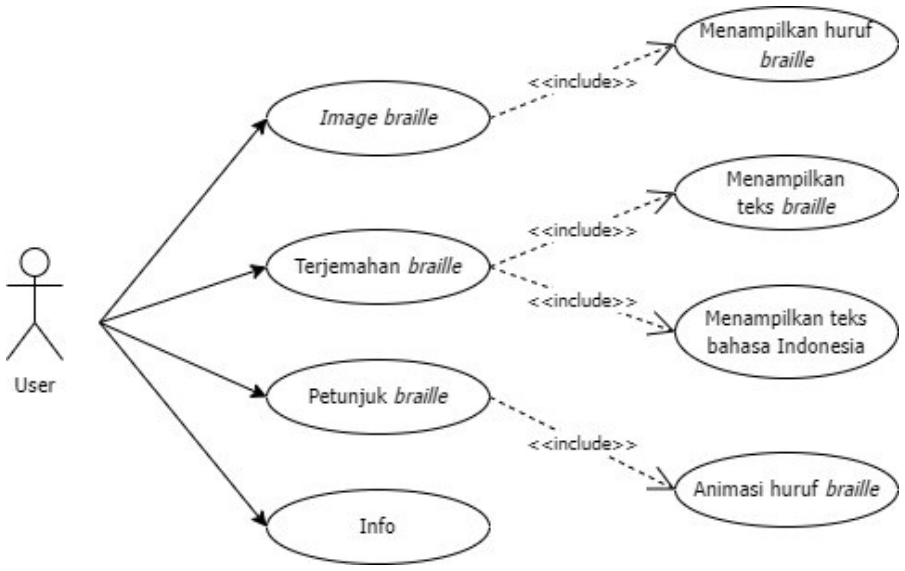
```
271/271 [==============================] - 462s 2s/step - loss: 0.2721 - accuracy: 0.9102 - val_loss: 0.3981 - val_accuracy: 0.8922
Epoch 35/50
271/271 [==============================] - 483s 2s/step - loss: 0.2720 - accuracy: 0.9054 - val_loss: 0.4076 - val_accuracy: 0.8851
Epoch 36/50
271/271 [==============================] - 449s 2s/step - loss: 0.2894 - accuracy: 0.8994 - val_loss: 0.3797 - val_accuracy: 0.8970
Epoch 37/50
271/271 [==============================] - 477s 2s/step - loss: 0.2758 - accuracy: 0.9058 - val_loss: 0.3804 - val_accuracy: 0.8891
Epoch 38/50
271/271 [==============================] - 469s 2s/step - loss: 0.2569 - accuracy: 0.9111 - val_loss: 0.3645 - val_accuracy: 0.9002
Epoch 39/50
271/271 [==============================] - 451s 2s/step - loss: 0.2517 - accuracy: 0.9112 - val_loss: 0.3555 - val_accuracy: 0.8899
Epoch 40/50
271/271 [==============================] - 430s 2s/step - loss: 0.2454 - accuracy: 0.9152 - val_loss: 0.3695 - val_accuracy: 0.8922
Epoch 41/50
271/271 [==============================] - 426s 2s/step - loss: 0.2478 - accuracy: 0.9147 - val_loss: 0.3717 - val_accuracy: 0.8994
Epoch 42/50
271/271 [==============================] - 457s 2s/step - loss: 0.2499 - accuracy: 0.9149 - val_loss: 0.3783 - val_accuracy: 0.8962
Epoch 43/50
271/271 [==============================] - 462s 2s/step - loss: 0.2510 - accuracy: 0.9108 - val_loss: 0.3839 - val_accuracy: 0.8954
Epoch 44/50
271/271 [==============================] - 421s 2s/step - loss: 0.2424 - accuracy: 0.9176 - val_loss: 0.3652 - val_accuracy: 0.8994
Epoch 45/50
271/271 [==============================] - 458s 2s/step - loss: 0.2322 - accuracy: 0.9199 - val_loss: 0.3751 - val_accuracy: 0.9010
Epoch 46/50
271/271 [==============================] - 433s 2s/step - loss: 0.2204 - accuracy: 0.9232 - val_loss: 0.3810 - val_accuracy: 0.9017
Epoch 47/50
271/271 [==============================] - 425s 2s/step - loss: 0.2204 - accuracy: 0.9224 - val_loss: 0.3912 - val_accuracy: 0.8946
Epoch 48/50
271/271 [==============================] - 428s 2s/step - loss: 0.2367 - accuracy: 0.9177 - val_loss: 0.4008 - val_accuracy: 0.8954
Epoch 49/50
271/271 [==============================] - 440s 2s/step - loss: 0.2263 - accuracy: 0.9215 - val_loss: 0.4022 - val_accuracy: 0.8954
Epoch 50/50
271/271 [==============================] - 414s 2s/step - loss: 0.2284 - accuracy: 0.9183 - val_loss: 0.3733 - val_accuracy: 0.8946
```

Gambar 2. Hasil training data

Hasil training model inilah yang akan disimpan dalam model tensorflow, kemudian ubah model tersebut menjadi sebuah model tensorflow *lite*. Tensorflow *lite* ini digunakan karena dapat mengurangi ukuran file dan meningkatkan kecepatan eksekusi tanpa mengurangi akurasi.

3.2. Rancangan Sistem yang diusulkan

a. *Use Case*



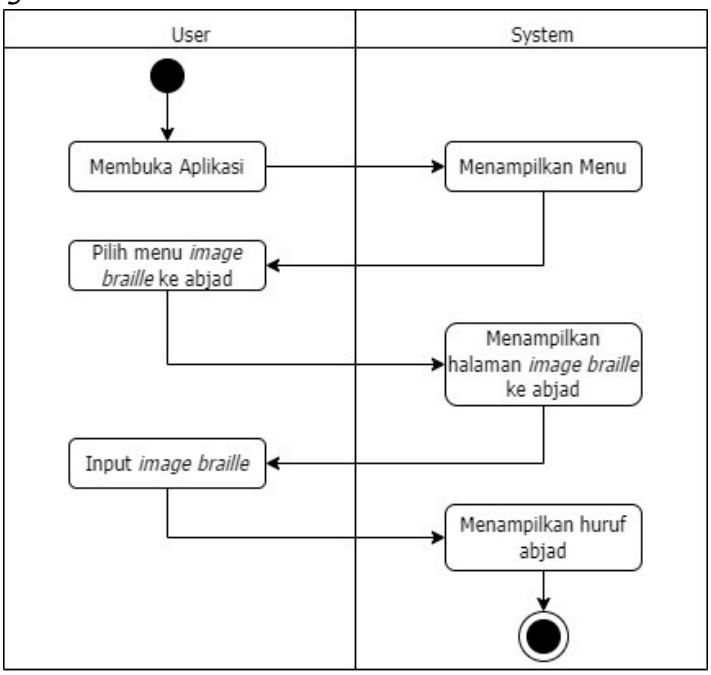
Gambar 3. *Use case diagram*

Tabel 3. Penjelasan *use case diagram*

Nama *use case*	**Deskripsi *use case***
Image huruf *braille*	*Use case* ini menjelaskan bagaimana proses untuk membuka suatu

	aplikasi untuk mengubah gambar huruf *braille* menjadi huruf abjad.
Petunjuk *braille*	*Use case* ini pada bagian ini menampilkan halaman suatu list button huruf abjad dari A-Z.
Informasi aplikasi	*Use case* ini menjelaskan proses menampilkan tentang aplikasi.
Terjemahan *braille*	*Use case* ini menjelaskan huruf *braille* di ubah ke teks bahasa Indonesia dan mengubah teks bahasa Indonesia menjadi huruf *braille*.
Animasi huruf *braille*	*Use case* ini digunakan untuk menampilkan suatu animasi huruf *braille*.

b. *Activity diagram*



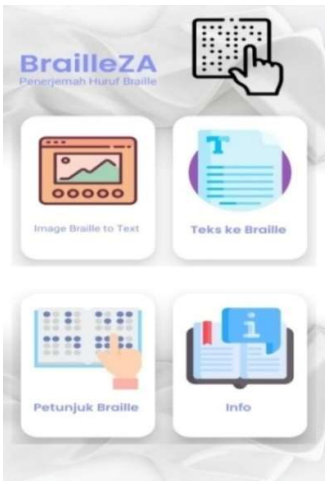
Gambar 4. Activity diagram menu *image braille* ke abjad

Activity diagram di atas menjelaskan bahwa ketika *user* membuka aplikasi ini akan menampilkan menu kemudian user memilih menu *image braille* ke abjad setelah itu menampilkan halaman *image braille* ke abjad kemudian *user* menginput gambar dari galeri *smartphone* android apabila menekan tombol *load image* maka akan menampilkan huruf abjad.

3.3.Tampilan Aplikasi

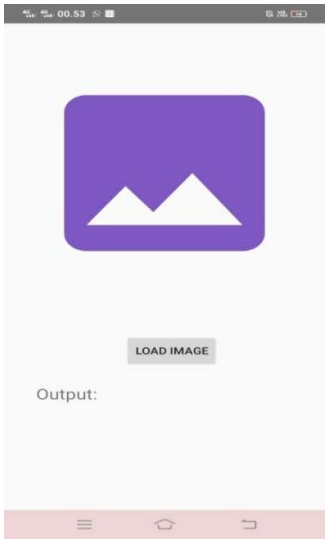
Gambar 6. Merupakan halaman *splashcreen* yaitu tampilan awal halaman aplikasi dan pada gambar 7. merupakan halaman menu utama pada aplikasi penerjemah huruf *braille* berbasis android

Gambar 6. S*plashscreen*



Gambar 7. Menu awal aplikasi

Pada gambar 8. Merupakan halaman *image braille* ke abjad inputannya berupa gambar huruf *braille*, outputnya berupa huruf abjad dan gambar 9. merupakan menu halaman penerjemah huruf *braille*.

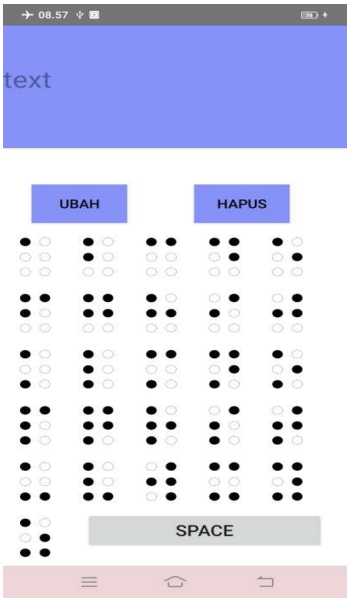


Gambar 8. *Image braille* ke teks

Gambar 9. Penerjemah huruf *braille*

Pada gambar 10. merupakan halaman konversi huruf *braille* ke teks, inputannya huruf *braille* yang terdapat pada aplikasi setelah *user* menginput klik tombol ubah maka huruf *braille* tersebut akan di ubah menjadi bahasa Indonesia dan pada gambar 11.

merupakan halaman konversi teks ke *braille*, inputannya berupa huruf abjad ketika *user* menekan tombol ubah maka huruf abjad tersebut berubah menjadi huruf *braille*.



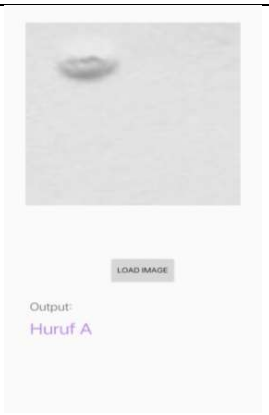
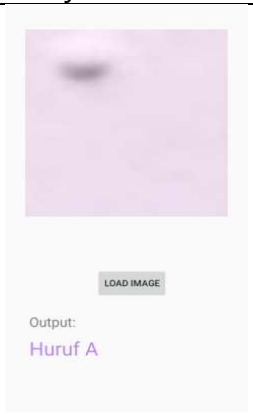
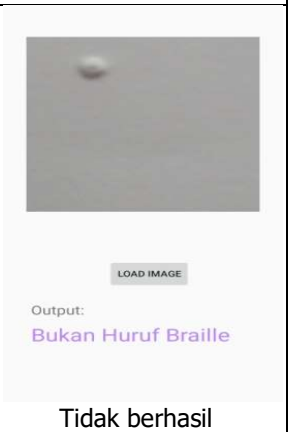
Gambar 10. Huruf *braille* ke teks

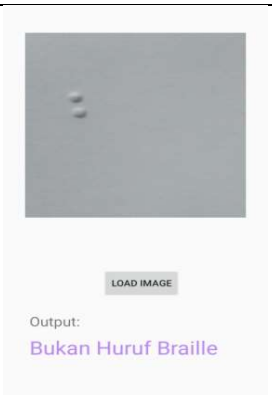
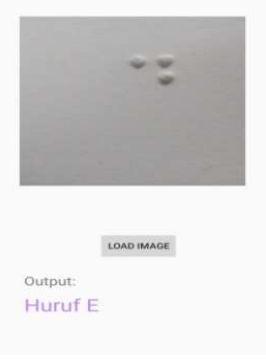
Gambar 11. Teks ke *braille*

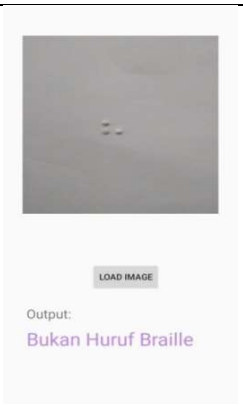
3.4.Pengujian aplikasi

a. Pengujian huruf *braille*

Tabel 4. Pengujian huruf *braille*

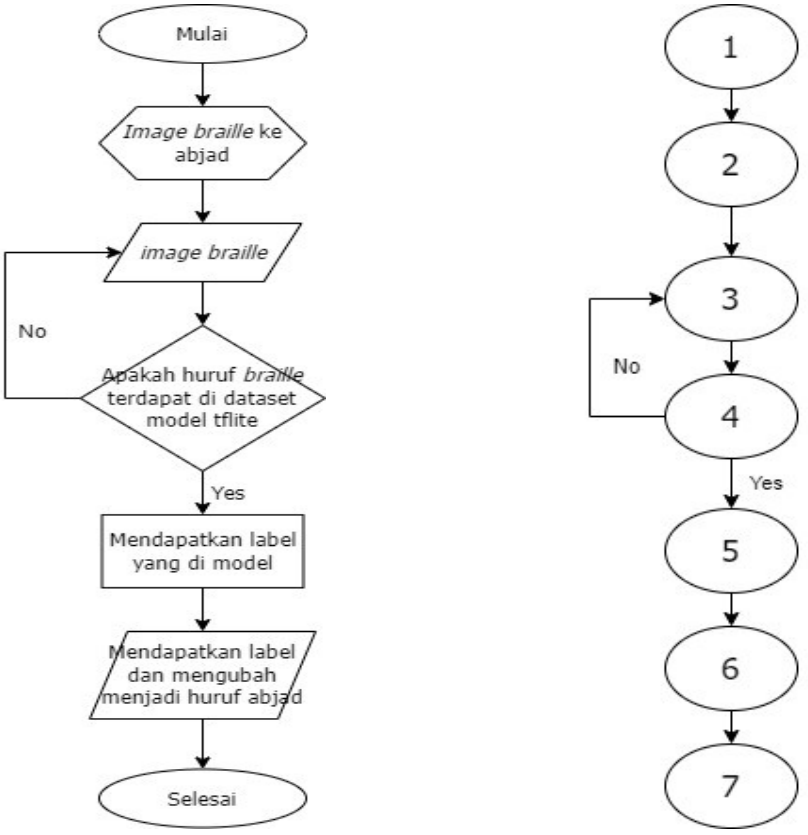
No	Huruf	Gambar I Dengan jarak 3 cm	Gambar II Berwarna dengan jarak 3 cm	Gambar III Dengan jarak 5 cm
1	A	 Berhasil, karena outputnya sudah benar yaitu huruf A	 Berhasil, karena outputnya sudah benar	 Tidak berhasil

2	B	 Berhasil, karena outputnya sudah benar	 Berhasil dengan menggunakan *background* berwarna	 Tidak berhasil
3	C	 Berhasil, karena outputnya sudah benar	 Berhasil dengan menggunakan *background* berwarna	 Tidak berhasil
4	D	 Berhasil, karena outputnya sudah benar	 Berhasil, karena outputnya sudah benar	 Tidak berhasil

5	H			
		Berhasil, karena outputnya sudah benar	Tidak berhasil	Tidak berhasil

Tabel diatas merupakan pengujian *image braille* ke abjad terdapat 15 percobaan yang dimana pengujian gambarnya ada 3 macam yaitu gambar *braille* dengan jarak 3 cm, gambar *braille background* berwarna dengan jarak 3 cm dan pengujian gambar *braille* dengan jarak 5 cm. Faktor yang mempengaruhi huruf braille tidak terdeteksi yaitu karna faktor pencahayaan pada saat pengambilan gambar huruf *braille* dan jarak saat pengambilan huruf *braiile* juga sangat mempengaruhi.

a. Pengujian *whitebox*



Gambar 12. *Flowchart* & *flowgraph image* huruf *braille* ke abjad

Gambar *flowchart* di atas menjelaskan tentang *image braille* ke abjad dengan menginput *image braille* dan apakah huruf *braille* terdapat di dataset model tflite jika iya maka akan mendapatkan label yang di model tflite dan outputnya berubah huruf abjad.

Dari *flowgraph image* huruf *braille* ke abjad di atas maka dapat dilakukan proses perhitungan sebagai berikut:

1. Menghitung *cyclomatic complexcity* V(G) asal *Egde* dan *Node*:
   Menggunakan rumus berikut : V(G) = E – N + 2
   E (*edge*) = 7
   N (*Node*) = 7
   P (Predikat *Node*) = 1
   Penyelesaian : V(G) = E – N + 2
   = 7 – 7 + 2 = 2
   Predikat (P) = P + 1
   = 1 + 1 = 2
2. Memiliki *region* = 2 menurut perhitungan *cyclomatic complexcity* dari grafik aliran di atas.
3. Pada diagram alur di atas, jalur independennya adalah:
   Path 1 = 1 – 2 – 3 – 4 – 5 – 6 – 7
   Path 2 = 1 – 2 – 3 – 4 – 3 – 4 – 5 – 6 – 7

Tabel 5. Grafik matriks *image braille* ke abjad

	1	2	3	4	5	6	7	E-1
1		1						1-1=0
2			1					1-1=0
3				1				1-1=0
4			1		1			2-1=1
5						1		1-1=0
6							1	1-1=0
7								0
	SUM (E+1)							1+1=2

4. KESIMPULAN

Hasil penelitian memperlihatkan bahwa aplikasi penerjemah huruf *braille* berbasis *android* dapat membantu *user* mempelajari dan mengetahui huruf *braille*. Pada penerjemahan *image braile* ke huruf abjad menggunakan metode klasifikasi *Convolutional Neural Network* dengan menggunakan model arsitektur MobileNetV2 yang terdiri dari 27 *class* dengan jumlah 8664 *image*, menghasilkan tingkat akurasi mencapai 0.9183. Sehingga performa dari deteksi aplikasi tersebut belum optimal atau belum akurat.

REFERENSI

Ariska, A., & Wahyuddin, W. (2022). Penerapan Kriptografi Menggunakan Algoritma Des (Data Encryption Standard). Jurnal Sintaks Logika, 2(2), 9-19.

Ayu, A. N. S. (2023). Aplikasi Pembaca Nilai Resistor Berbasis Android. Jurnal Sintaks Logika, 3(1), 17-22.

Ferdy, F., & Wahyuddin, W. (2024). Aplikasi Game Edukasi Mitigasi Bencana Alam (Gempa Bumi Dan Tsunami) Mengunakan Metode Waterfall Berbasis Android. Jurnal Sintaks Logika, 4(1), 1-6.

Herlambang, M. F., Hermana, A. N., & Putra, K. R. (2021). Pengenalan Karakter Huruf Braille dengan Metode Convolutional Neural Network. Systemic: Information System and Informatics Journal, 6(2), 20–26. https://doi.org/10.29080/systemic.v6i2.969

Ilham, F., & Rochmawati, N. (2020). Transliterasi Aksara Jawa Tulisan Tangan ke Tulisan Latin Menggunakan CNN. Journal of Informatics and Computer Science (JINACS), 1(04), 200–208. https://doi.org/10.26740/jinacs.v1n04.p200-208

Nugraha, W. A. (2022). Pengujian White Box Berbasis Path Pada Form Autentikasi Berbasis Mobile. Jurnal Siliwangi Seri Sains Dan Teknologi, 8(2), 42–47. https://doi.org/10.37058/jssainstek.v8i2.4098

Parlika, R., Nisaa', T. A., Ningrum, S. M., & Haque, B. A. (2020). Studi Literatur Kekurangan Dan Kelebihan Pengujian Black Box. Teknomatika, 10(02), 131–140.

Ramadhan, A. F., Putra, A. D., & Surahman, A. (2021). Aplikasi Pengenalan Perangkat Keras Komputer Berbasis Android Menggunakan Augmented Reality (AR). Jurnal Teknologi Dan Sistem Informasi (JTSI), 2(2), 24–31.

Ramiati, R., Aulia, S., & Lifwarda, L. (2020). Aplikasi Identifikasi Huruf Braille Menggunakan Computer Vision Berbasis Raspberry Pi. Jurnal Nasional Teknik Elektro, 9(1), 12. https://doi.org/10.25077/jnte.v9n1.707.2020

Ronando, E., & Sudaryanto, A. (2018). Sistem Pengenalan Pola Huruf Braille Berbasis Audio Menggunakan Metode Naïve Bayes. Jurnal Ilmu Komputer Dan Desain Komunikasi Visual, 3(1), 43–52.

Sulastio, B. S., Anggono, H., Putra, A. D., Teknik, F., & Indonesia, U. T. (2021). RAWAN MACET DI JAM KERJA PADA KOTA BANDARLAMPUNG PADA BERBASIS ANDROID. 2(1), 104–111.

Sulviyani, I., & Novamizanti, L. (2017). Perancangan Konversi Braille Ke Teks Berbasis Android. ReTII.

SYARIF, A. K. (2021). Sistem Klasifikasi Penyakit Tanaman Cabai Menggunakan Metode Deep Learning DenganLibrary Tensorflow Lite. Universitas Hasanuddin.

Yunus, M., & Anwar, Y. (2022). Aplikasi Penerjemah Bahasa Isyarat Indonesia Ke

Dalam Huruf Abjad. Jurnal Sintaks Logika, 2(1), 257–262. https://doi.org/10.31850/jsilog.v2i1.1726

Wahyuddin, W., & Hasim, A. (2023). Aplikasi Ekstraksi Data Kartu Vaksin Berbasis Web Menggunakan Metode Ocr. Jurnal Sintaks Logika, 3(2), 53-57.

Wahyuddin, W., & Saputra, A. (2021). Aplikasi schedule pengerjaan proyek online dinas PU Kab. Sidrap. Jurnal Sintaks Logika, 1(2), 54-61.

Wahyuddin, W., & As, K. (2022). Pengembangan Aplikasi Risalah Tuntunan Shalat Secara Lengkap Berbasis Android. Jurnal Sintaks Logika, 2(1), 248-256.

Wahyuddin, W., & Wafiah, A. (2022). Aplikasi Pemesanan Menu Pada Warkop Shearlock Berbasis Abdroid. Jurnal Sintaks Logika, 2(3), 11-16.

Wahyuddin, W., Alam, S., & Said, I. R. (2021). E-COMMERCE BUMBU MASAKAN KELOMPOK TANI KWT (KELOMPOK WANITA TANI) SETIA DESA PAKKODI KAB. ENREKANG. Jurnal Sintaks Logika, 1(3), 209-214.