Jurnal Algoritma
E-ISSN: 2302-7339 | P-ISSN: 1412-3622
INSTITUT TEKNOLOGI GARUT

Pengenalan Alfabet Sistem Isyarat Bahasa Indonesia (SIBI) Menggunakan *Convolutional Neural Network*

Indra Jiwana Thira[1], Dwiza Riana[2], Azriel Noer Ilhami[3], Brama Rizky Setia Dwinanda[4], Hana Choerunisya[5]

Jurnal Algoritma
Universitas Garut
Jl. Raya Samarang Jl. Hampor No.52A, Tarogong Kaler, Garut, Jawa Barat 44151
Email : rektorat@uniga.ac.id

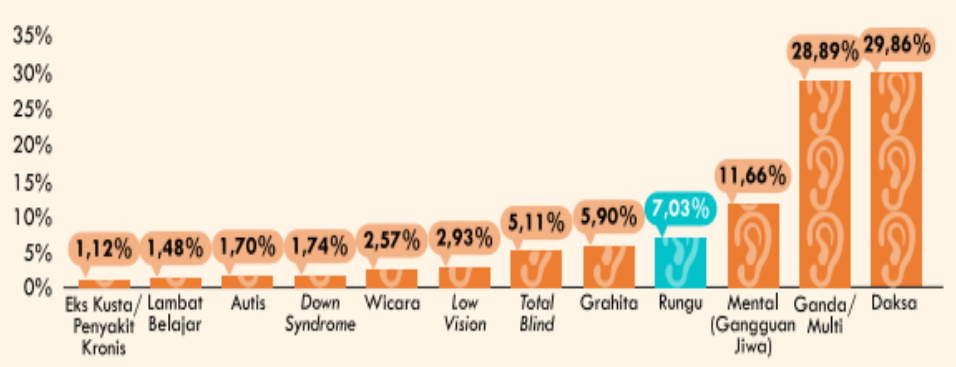
[1]indrajiwana@uniga.ac.id
[2]dwiza@nusamandiri.ac.id
[3]24072121013@fkominfo.uniga.ac.id
[4]24072121005@fkominfo.uniga.ac.id
[5]24072121026@fkominfo.uniga.ac.id

Abstrak **–** Bahasa isyarat, khususnya Bahasa Isyarat Bahasa Indonesia atau lebih dikenal dengan SIBI menjadi Bahasa resmi yang digunakan penyandang tuli dalam berkomunikasi, baik dengan sesama penyandang tuli ataupun dengan orang normal. Namun, kendala muncul karena sebagian kecil orang saja bisa bahasa isyarat. Akibatnya, berkomunikasi dengan penyandang tuli bisa menjadi tantangan. Penelitian ini memiliki tujuan untuk mengklasifikasikan alfabet pada SIBIdengan total 24 kelas, dengan pengecualian huruf J dan Z. Pendekatan klasifikasi ini dilakukan melalui perbandingan tiga arsitektur Convolutional Neural Network (CNN) MobileNet diantaranya adalah MobileNetV2, MobileNetV3Small, dan MobileNetV3Large. Penelitian ini bermaksud menentukan arsitektur yang paling optimal. Hasil penelitian menunjukkan bahwa MobileNetV3Small menghasilkan model yang paling baik. Dalam pengujian menggunakan data tes, model ini mencapai akurasi sebesar 98,81% dengan menggunakan batchsize 32 dan menjalankan proses pelatihan selama 30 epoch.

Kata Kunci **–** Bahasa Isyarat; *Convolutional Neural Network*; Klasifikasi; *MobileNet*; Sistem Isyarat Bahasa Indonesia.

I. PENDAHULUAN

Pada tahun 2019, *World Health Organization (WHO)* memperkirakan terdapat 466 juta orang di Dunia mengalami gangguan pendengaran dan 360 juta diantaranya mengalami ketulian. Orang yang mengalami ketulian, 50% diaantaranya atau sebanyak 180 juta orang berasal dari Asia Tenggara. Diperkirakan pada tahun 2050 terdapat lebih dari 900 juta orang atau satu dari sepuluh penduduk di Dunia akan memiliki gangguan pendengaran [1]. Gambar 1 menunjukan bahwa di Indonesia, tuli berada di urutan keempat dalam daftar penyandang disabilitas dengan 7,03%.



Gambar 1: Presentase Penyandang Disabilitas di Indonesia
Sumber: https://simpd.kemensos.go.id/

Bahasa isyarat adalah bahasa dimana komunikasi antar orang dilakukan dengan cara mentransmisikan pola tanda secara visual untuk mengungkapkan maknanya dan memiliki kosa kata sendiri dan sintaksis yang murni berbeda dari Bahasa lisan/tulisan [2]. Ada dua sistem bahasa isyarat yang digunakan penyandang tuli di Indonesia, yaitu Sistem Isyarat Bahasa Indonesia (SIBI) dan Bahasa Isyarat Indonesia (BISINDO). Namun, pemerintah mengakui SIBI sebagai sistem standar untuk berkomunikasi [3]. SIBI dikeluarkan oleh kementerian Pendidikan dan Kebudayaan dan diterapkan di sekolah – sekolah formal SLB (Sekolah Luar Biasa).

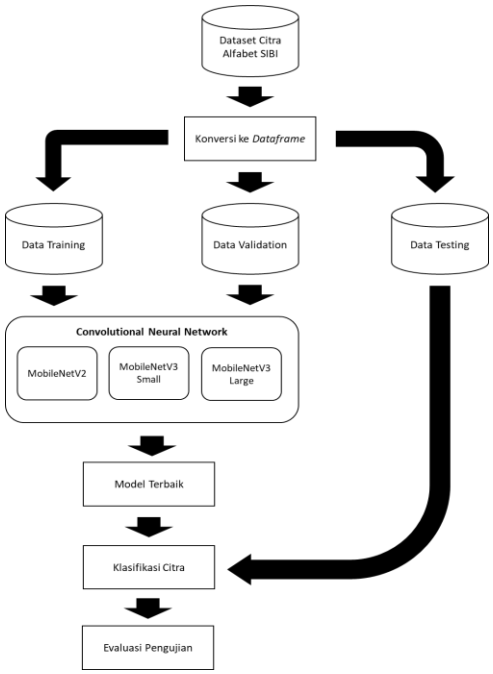
SIBI tidak terlalu banyak dikuasai khususnya oleh orang normal, sehingga menjadi salah satu kendala komunikasi dengan penyandang tuli. Penelitian sebelumnya menunjukan ada beberapa metode yang telah untuk menginterpretasikan Bahasa isyarat. Metode KNN digunakan [4],[5],[6] untuk pengenalan alfabet BISINDO menggunakan dataset dan jumlah dataset yang berbeda pada masing-masing penelitian dengan hasil pengujian rata-rata 91,8%, 88% dan 96,52%. Penelitian lainnya [2],[7],[8],[9] melakukan deteksi gerakan menggunakan Kinect terhadap alfabet dan kata dalam BISINDO dan SIBI dengan hasil yang bervariasi tetapi rata-rata masih di bawah 95%. Hasil yang baik didapatkan pada penelitian yang menggunakan *Convolutional Neural Network* (CNN) dengan objek kata BISINDO sebesar 96% [10], 98% pada pengenalan alfabet Bahasa isyarat Amerika [11] dan 98,75% pada pengenalan Bahasa isyarat Bengali yang mengambil beberapa kelas saja sebagai dataset nya [12].

Melihat perkembangan penelitian penggunaan CNN khususnya pada objek penelitian Bahasa isyarat yang menghasilkan akurasi lebih baik dibanding dengan metode yang lain, maka untuk memecahkan permasalahan di atas akan dilakukan penelitian untuk pengenalan alfabet pada SIBI menggunakan CNN dengan arsitektur *MobileNet*. *MobileNet* merupakan arsitektur CNN yang dapat digunakan untuk klasifikasi, deteksi dan tugas lain di CNN. Arsitektur *MobileNet* juga memiliki latensi dan konsumsi daya yang rendah serta menghasilkan model dengan ukuran kecil tetapi tetap memiliki akurasi yang baik [13]. Pada penelitian ini akan mencari model terbaik untuk pengenalan alfabet SIBI dari tiga arsitektur *MobileNet* yaitu *MobileNetV2*, *MobileNetV3Small* dan *MobileNetV3Large*. *MobileNetV2* telah digunakan untuk pengenalan bahasa isyarat American sign language dengan menghasilkan akurasi 98.67% [14], serta digunakan juga pada bidang lain [15], [16] dan menghasilkan nilai akurasi di atas 95%. Sementara pada penelitian lain, *MobileNet*v3 diklaim memiliki proses lebih cepat dan akurasi lebih baik dari *MobileNetV2* [17]. Sementara penelitian pada sign language lainnya yang menggunakan mobilenet mendapatkan akurasi sebesar 93,48% [18].

II. METODOLOGI PENELITIAN

Penelitian ini akan melakukan klasifikasi alfabet SIBI menggunakan CNN dengan arsitektur *MobileNet*. Ada tiga arsitektur *MobileNet* yang akan digunakan dalam training model pada CNN yaitu *MobileNetV2*, *MobileNetV3Small* dan *MobileNetV3Large*. Ketiga arsitektur tersebut akan dijalankan terpisah dan dicari model yang paling baik dalam melakukan klasifikasi alfabet SIBI. Model yang paling baik akan diuji menggunakan data testing dari dataset dan data uji dari luar dataset untuk klasifikasi alfabet SIBI. Rancangan

penelitian secara keseluruhan mulai dari persiapan dataset sampai proses klasifikasi alfabet SIBI dapat dilihat pada Gambar 2.



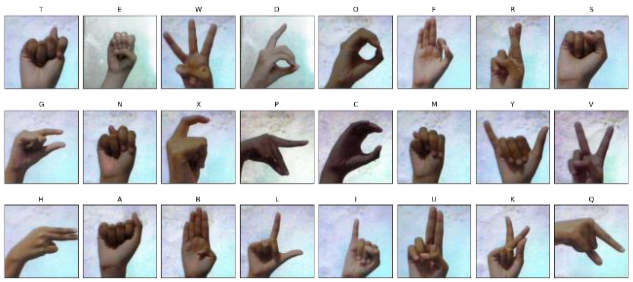
Gambar 2: Rancangan Penelitian

Citra yang digunakan dalam penelitian adalah citra alfabet SIBI yang diperoleh dari www.github.com. Total jumlah citra pada dataset adalah 840 citra dan terdiri dari 24 kelas yang merupakan jumlah alfabet SIBI kecuali huruf J dan Z. Huruf J dan Z tidak masuk kedalam objek penelitian dikarenakan huruf J dan Z pada SIBI merupakan sebuah gerakan dinamis. Daftar kelas pada dataset dapat dilihat pada Tabel 1.

Tabel 1: Jumlah Citra Masing-Masing Kelas

No	Kelas	Jumlah Citra	No	Kelas	Jumlah Citra
1	A	35	13	N	35
2	B	35	14	O	35
3	C	35	15	P	35
4	D	35	16	Q	35
5	E	35	17	R	35
6	F	35	18	S	35
7	G	35	19	T	35
8	H	35	20	U	35
9	I	35	21	V	35
10	K	35	22	W	35
11	L	35	23	X	35
12	M	35	24	Y	35

Semua citra pada dataset memiliki format portable network graphics (PNG) dengan ukuran masing-masing citra 290 x 290 pixels. Dataset berisi citra yang memiliki latar belakang seragam yaitu latar belakang terang dengan tingkat pencahayaan yang juga relatif sama. Dataset ini akan digunakan pada eksperimen training model sebagai data training, data validation dan data testing. Gambar 3 menunjukan contoh citra masing-masing kelas pada dataset.



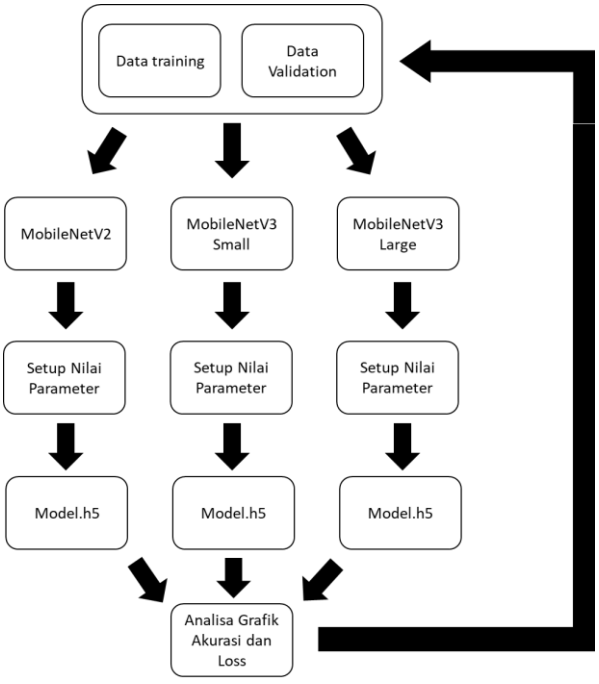
Gambar 3: Contoh Citra Dataset

Langkah pertama yang dilakukan adalah melakukan konversi file dataset ke dalam *DataFrame*. *DataFrame* adalah sebuah koleksi kolom berurutan dengan nama dan jenis, dengan demikian merupakan sebuah tabel yang tampak seperti database dimana sebuah baris tunggal mewakili sebuah contoh tunggal dan kolom mewakili atribut tertentu [19]. Proses konversi file ke *DataFrame* dilakukan sebelum proses pembagian dataset dengan mencatat path masing-masing citra pada dataset. Hal ini dilakukan untuk mempercepat proses pembagian dataset dibanding dengan pembagian dari file secara langsung karena Penggunaan *DataFrame* memungkinkan untuk membaca sebuah file dan menjadikannya sebuah tabel. Pembagian dataset dilakukan setelah file citra dataset di konversi menjadi *DataFrame*, jadi citra dataset tetap pada satu folder yang sama. Pembagian dataset ini bertujuan untuk membagi dataset menjadi data *training*, data *validation* dan data testing. Tabel 2 menunjukan rencana pembagian dataset yang akan dilakukan dengan rasio 7:2:1 untuk data *training*, data *validation* dan data testing.

Tabel 2: Pembagian Dataset

Total Kelas	**Citra *Training***	**Citra *Validation***	**Citra *Testing***	**Total Citra**
24	588	168	84	840

Selanjutnya dilakukan eksperimen pelatihan model terhadap dataset yang telah dibagi menjadi tiga bagian. Pelatihan model akan menggunakan arsitektur *MobileNetV2*, *MobileNetV3Small* dan *MobileNetV3Large* yang dilakukan secara terpisah. Eksperimen dilakukan menggunakan *google colaboratory* dengan mencoba berbagai nilai parameter diantaranya *batch size*, learning rate dan jumlah *epoch* pada data *training* dan data *validation* sampai menemukan model terbaik dari masing-masing arsitektur.



Gambar 4: Alur Eksperimen Pelatihan Model

Eksperimen dilakukan dalam satu *life cycle* penuh pemodelan, sampai model yang dihasilkan bisa disimpan dalam format h5 dan dilakukan analisa grafik akurasi dan loss. Selama grafik belum menunjukan kearah linear eksperimen terus dilakukan dengan mengubah parameter-parameter yang diperlukan. Proses eksperimen digambarkan pada Gambar 4. Model-model yang dihasilkan kemudian akan dilakukan perbandingan pada tahap berikutnya. Perbandingan model dilakukan untuk mencari model terbaik dari semua model yang dihasilkan pada tahap eksperimen. Model terbaik ditentukan dari tingkat akurasi dan loss pada data *training* dan data *validation* alfabet SIBI. Setelah mendapatkan model terbaik, selanjutnya akan dilakukan pengujian terhadap model tersebut menggunakan datas testing. Selain itu Perbandingan juga akan menghasilkan informasi arsitektur *MobileNet* mana yang memiliki performa paling baik dari segi waktu *compile* model untuk klasifikasi alfabet SIBI. Pengujian akan dilakukan terhadap model terbaik yang dihasilkan dari proses ekseperimen. Pengujian dilakukan terhadap data testing yang berasal dari hasil pembagian dataset pada tahap sebelumnya.

III. HASIL DAN PEMBAHASAN

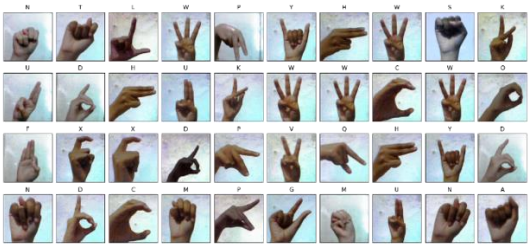
A. Konversi *DataFrame* dan Pembagian Dataset

Pembagian dataset menjadi *training*, *validation* dan *testing* tidak dilakukan dengan pembagian folder dan dataset tetap berada dalam satu folder yang sama. Pembagian dilakukan setelah dataset dikonversi menjadi *DataFrame* terlebih dahulu. Konversi *DataFrame* pada intinya adalah menyimpan masing-masing path file yang ada pada masing dataset dan memberikannya label sesuai dengan label yang telah ditentukan untuk dipelajari oleh arsitektur CNN. Pada saat konversi menjadi *DataFrame* juga dilakukan pengacakan untuk menghindari kebocoran data pada dataset.

Gambar 5 menunjukan sebagian hasil dari konversi *DataFrame* yang terdiri dari filepath dan label yang tidak berurutan karena sudah dilakukan pengacakan pada proses konversi ke *DataFrame*. Hasil konversi *DataFrame* tersebut ditampilkan agar bisa dilakukan pengecekan bahwa konversi berjalan dengan baik dengan bukti filepath dan label yang dihasilkan sudah sesuai. Gambar 6 menunjukan citra dataset yang telah dilakukan pengacakan dan siap untuk dilakukan pembagian menjadi data *training*, *validation* dan *testing*.

	Filepath	Label
0	drive/MyDrive/14002257/citra/dataset/W/1 (1).png	W
1	drive/MyDrive/14002257/citra/dataset/N/3.png	N
2	drive/MyDrive/14002257/citra/dataset/U/1 (1).png	U
3	drive/MyDrive/14002257/citra/dataset/G/8 (1).png	G
4	drive/MyDrive/14002257/citra/dataset/H/10.png	H
5	drive/MyDrive/14002257/citra/dataset/D/16.png	D
6	drive/MyDrive/14002257/citra/dataset/X/2.png	X
7	drive/MyDrive/14002257/citra/dataset/I/5.png	I
8	drive/MyDrive/14002257/citra/dataset/A/18.png	A
9	drive/MyDrive/14002257/citra/dataset/Q/6 (1).png	Q

Gambar 5: Contoh Hasil Konversi ke Dataframe



Gambar 6: Dataset Setelah Dilakukan Pengacakan

Tahap selanjutnya adalah melakukan pembagian dataset dengan rasio 7:2:1, 70% untuk data *training*, 20% untuk data *validation* dan 10% untuk data *testing*. Dari jumlah total dataset 840 citra terbagi menjadi 588 citra untuk data *training*, 168 untuk data *validation* dan 84 untuk data *testing*. Hasil pembagian dataset dapat dilihat pada Gambar 7. Setelah proses pembagian, citra pada dataset telah siap untuk dilakukan pemodelan menggunakan CNN.

```
Total data from training dataframe   : 588
Total data from validation dataframe : 168
Total data from testing dataframe    : 84
```

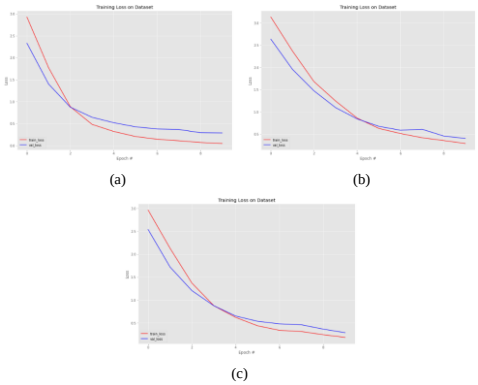
Gambar **7**: Hasil Pembagian Dataset

B. Pelatihan Model CNN

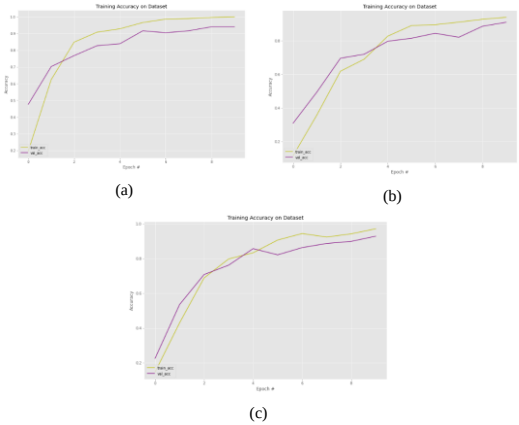
Untuk menghasilkan model dari CNN perlu dilakukan *compile* model menggunakan tiga parameter, yaitu *optimizer, loss* dan *metrics*. *Optimizer* yang digunakan adalah *Adam* dengan *learning rate* 0,001. *Optimizer Adam* digunakan untuk mempercepat pelatihan model dalam mencapai loss minimum [15]. Fungsi *Loss* yang digunakan yaitu *categorical_crossentropy.* Nilai yang semakin kecil menunjukkan performa model yang lebih baik. Parameter *metrics accuracy* yang digunakan untuk melihat hasil akurasi pada saat melatih model.

Tahap selanjutnya adalah melakukan *fit* model untuk melatih data citra ke dalam model. Parameter yang digunakan antara lain *batch_size* dan *epoch*. *Batch_size* yang digunakan sebesar 32 yang merupakan jumlah contoh pelatihan dalam satu *forward/backward pass*. Penentuan *Epoch* dilakukan dengan beberapa percobaan menggunakan kelipatan 10 dengan menganalisa grafik akurasi dan *loss* yang dihasilkan, jika masih ada potensi peningkatan maka *epoch* akan ditingkatkan, tetapi jika sudah menunjukan grafik linier maka *epoch* akan dihentikan. Pada penelitian ini jumlah *epoch* yang digunakan adalah 10, 20 dan 30. Jumlah *epoch* ini berpengaruh pada akurasi dan *loss* model yang dihasilkan. Agar menghasilkan model terbaik, percobaan dilakukan pada tiga arsitektur *MobileNet* yang berbeda yaitu *MobileNetV2*, *MobileNetV3Small* dan *MobileNetV3Large*.

Pada *epoch* 10, *MobileNetV2* menghasilkan model paling baik dibandingkan dengan *MobileNetV3Small* dan *MobileNetV3Large* dengan akurasi pada data *training* sebesar 100% dan *loss* 0,0481 serta akurasi data *validation* sebesar 94,05% dan *loss* 0,2885. Kedua terbaik adalah *MobileNetV3Large* dengan akurasi data *training* sebesar 97,11% dan *loss* 0,1873 serta akurasi data *validation* sebesar 92,86% dan *loss* 0,2899. *MobileNetV3Small* berada pada urutan terakhir dengan akurasi data *training* sebesar 94,05% dan *loss* 0,2915 serta akurasi data *validation* sebesar 91.07% dan *loss* 0,4047. Perbandingan akurasi dan *loss* masing-masing arsitektur dengan jumlah *epoch* 10 dapat dilihat pada Gambar 8 dan Gambar 9



Gambar 8: Grafik Nilai Loss dengan Epoch 10
a: *MobileNetV2*; b: *MobileNetV3Small*; c: *MobileNetV3Large*



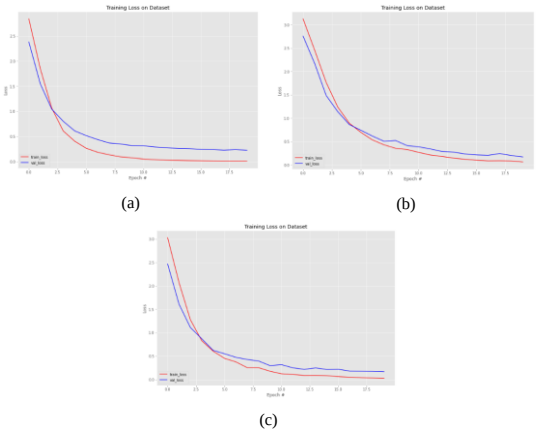
(a) (b)

(c)

Gambar 9: Grafik Nilai Akurasi dengan Epoch 10
a: *MobileNetV2*; b: *MobileNetV3Small*; c: *MobileNetV3Large*

Berdasarkan analisa pada grafik yang dihasilkan dengan jumlah *epoch* 10, dapat disimpulkan bahwa grafik masih bisa bergerak untuk mendapatkan nilai akurasi dan *loss* lebih baik, maka diputuskan untuk melakukan percobaan menggunakan jumlah *epoch* berikutnya yaitu 20.

Pada *epoch* 20, *MobileNetV3Small* menghasilkan model paling baik dibandingkan dengan *MobileNetV2* dan *MobileNetV3Large* dengan akurasi pada data *training* sebesar 99,83% dan *loss* 0,0641 serta akurasi data *validation* sebesar 96,43% dan *loss* 0,1708. Kedua terbaik adalah *MobileNetV3Large* dengan akurasi data *training* sebesar 100% dan *loss* 0,0292 serta akurasi data *validation* sebesar 95,24% dan *loss* 0,1700. *MobileNetV2* berada pada urutan terakhir dengan akurasi data *training* sebesar 100% dan *loss* 0,0144 serta akurasi data *validation* sebesar 94,05% dan *loss* 0,2275. Perbandingan akurasi dan *loss* masing-masing arsitektur dengan jumlah *epoch* 20 dapat dilihat pada Gambar 10 dan Gambar 11.

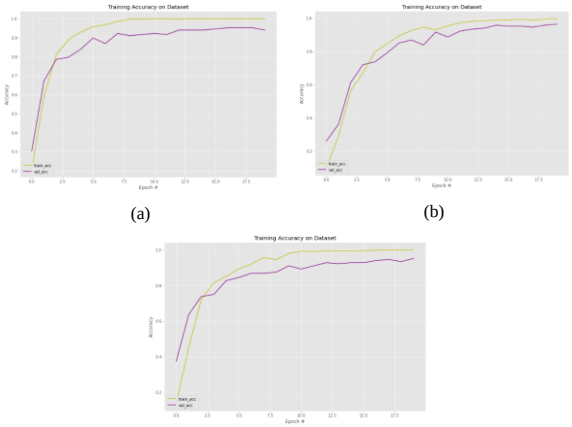


(a) (b)

(c)

Gambar 10: Grafik Nilai Loss dengan Epoch 20
a: *MobileNetV2*; b: *MobileNetV3Small*; c: *MobileNetV3Large*

Berdasarkan analisa pada grafik yang dihasilkan, terdapat grafik yang menunjukan garis linier pada jumlah *epoch* 20 yaitu akurasi dan *loss* untuk *MobileNetV2* serta grafik akurasi untuk *MobileNetV3Large,* tetapi masih ada grafik yang berpotensi untuk terus bergerak. Oleh karena itu, percobaan dilanjutkan pada jumlah *epoch* berikutnya yaitu 30.

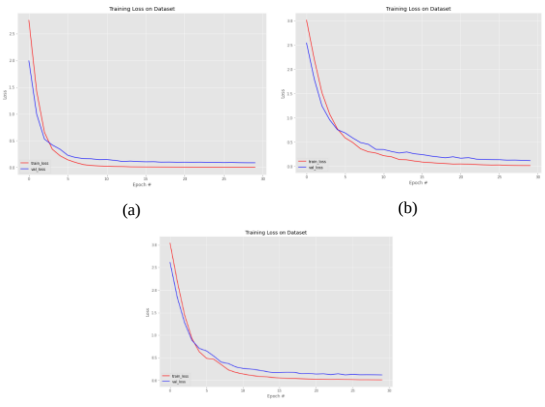


(a) (b)

(c)

Gambar 11: Grafik Nilai Akurasi dengan Epoch 20
a: *MobileNetV2*; b: *MobileNetV3Small*; c: *MobileNetV3Large*

Pada *epoch* 30, *MobileNetV3Small* kembali menghasilkan model paling baik dibandingkan dengan *MobileNetV2* dan *MobileNetV3Large* dengan akurasi pada data *training* sebesar 100% dan *loss* 0,0167serta akurasi data *validation* sebesar 97,62% dan *loss* 0,1193. Kedua terbaik adalah *MobileNetV2* dengan akurasi data *training* sebesar 100% dan *loss* 0,0023 serta akurasi data *validation* sebesar 97,02% dan *loss* 0,0874. *MobileNetV3Large* berada pada urutan terakhir dengan akurasi data *training* sebesar 100% dan *loss* 0,0110 serta akurasi data *validation* sebesar 95,83% dan *loss* 0,1228. Perbandingan akurasi dan *loss* masing-masing arsitektur dengan jumlah *epoch* 30 dapat dilihat pada Gambar 12 dan Gambar 13.

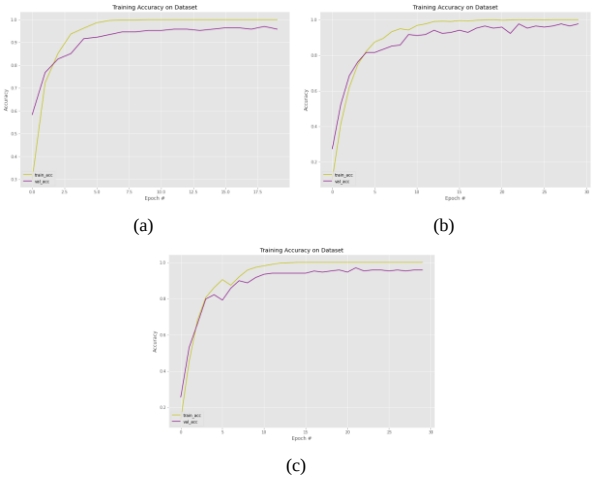


(a) (b)

(c)

Gambar 12: Grafik Nilai Loss dengan Epoch 30
a: *MobileNetV2*; b: *MobileNetV3Small*; c: *MobileNetV3Large*

Berdasarkan analisa pada grafik yang dihasilkan dengan jumlah *epoch* 30, sudah menunjukan garis linier pada semua grafik baik itu akurasi maupun *loss* pada arsitektur *MobileNetV2*, *MobileNetV3Small* dan *MobileNetV3Large*. Melihat hasil tersebut diputuskan untuk menentukan model terbaik dan akan digunakan untuk pengujian pada jumlah *epoch* 30.



(a) (b)

(c)

Gambar 13: Grafik Nilai Akurasi dengan Epoch 30
a: *MobileNetV2*; b: *MobileNetV3Small*; c: *MobileNetV3Large*

C. Model Terbaik

Berdasarkan hasil percobaan, model terbaik yang dihasilkan adalah model dari arsitektur *MobileNetV3Small* karena memiliki akurasi data *validation* tertinggi dengan 97,62% dan *loss* 0.1193 serta akurasi data *training* sebesar 100% dan *loss* 0,0167 dengan jumlah *epoch* 30. Model yang dihasilkan kemudian disimpan dalam format h5. Selain menghasilkan model terbaik, *MobileNetV3Small* juga memiliki waktu eksekusi yang lebih cepat dibanding dua arsitektur pembanding lainnya. *MobileNetV3Small* memerlukan waktu *compile* 8 menit 55 detik untuk *compile* model dengan *epoch* 30. Seperti Akurasi dan *loss*, waktu *compile* juga dipengaruhi oleh jumlah *epoch*. Hasil ini sejalan dengan penelitian yang dilakukan sebelumnya [17]. Perbandingan waktu *compile* pada arsitektur *MobileNet* bisa dilihat pada Tabel 3.

Tabel 3: Perbandingan Waktu *Compile*

ARSITEKTUR CNN	EPOCH		
	10	20	30
	Lama Eksekusi	Lama Eksekusi	Lama Eksekusi
MobileNetV2	9m16s	9m57s	15m52s
MobileNetV3Small	2m37s	5m20s	8m55s
MobileNetV3Large	6m6s	8m15s	16m51s

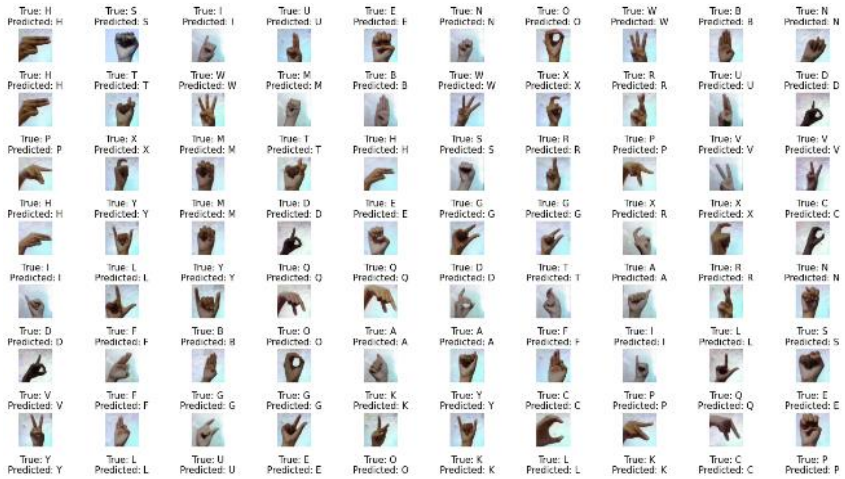
Model terbaik akan diuji pada proses pengujian dengan melakukan klasifikasi citra alfabet pada data *testing* dengan menggunakan model terbaik yang dihasilkan yaitu model dari arsitektur *MobileNetV3Small*.

Tabel 4: Perbandingan Hasil Penelitian

No	Metode	Jumlah Dataset	Jumlah Kelas	Akurasi
1	Gradient-CNN [20]	600	5	98%
2	CNN-*MobileNet* [13]	1800	36	95,71%
3	CNN [12]	25000	46	98,75%
4	Template Macthing -KNN [6]	153	11	96,52%
5	CNN-*MobileNetV2* [14]	4800	24	98,67%
6	CNN [3]	110	3	100%
7	CNN-RNN [21]	2582	26	60,58%
8	CNN-LeNet [22]	450	10	98,89%
9	Fuzzy KNN [5]	130	26	88%
10	Penelitian saat ini	840	24	98,91%

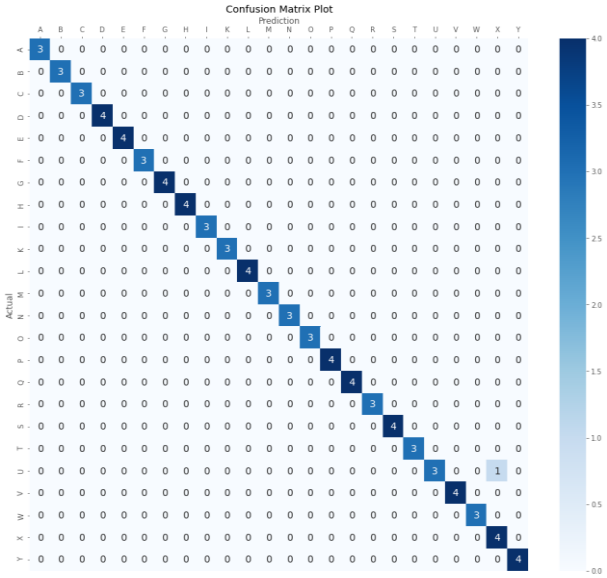
Proses pengujian menggunakan 84 citra yang merupakan 10% dari total dataset awal yang terdiri dari 24 kelas alfabet SIBI. Hasil pengujian terhadap data *testing* menghasilkan akurasi 98,81% dan *loss* 0.0877. Dari total

84 citra uji hanya satu citra yang mendapatkan prediksi salah. Perbandingan akurasi hasil penelitian dengan penelitian sebelumnya dapat dilihat pada Tabel 4.



Gambar 14: Hasil proses prediksi

Untuk mengukur performa pengujian digunakan *classification metrics* yang menghasilkan nilai akurasi 99%. Pengukuran klasifikasi pada tiap kelas juga digambarkan dengan menggunakan confusion matrix yang dapat dilihat pada Gambar 15. Berdasarkan *confusion matrix* dapat disimpulkan hanya satu citra pada kelas huruf U yang di prediksi salah menjadi kelas huruf X.



Gambar 15: *Confusion Matrix*

IV. KESIMPULAN

Perbandingan dari masing-masing Model berhasil dilakukan dan mendapatkan model yang dihasilkan *MobileNetV3Small* dengan jumlah *epoch* 30 menjadi model yang paling baik diantara model lain yang dihasilkan selama penelitian dan memiliki akurasi *training* sebesar 100% dan *validation* sebesar 97,62%. Pengujian model terbaik terhadap 84 data testing mendapatkan akurasi sebesar 98,81% dengan prediksi benar pada 84 citra dan prediksi salah pada satu citra. Hasil evaluasi model terbaik dalam melakukan klasifikasi pada data testing menghasilkan nilai recall 0,99, nilai presisi 0,99 dan nilai f1-score sebesar 0,99. Penelitian

selanjutnya bisa menggunakan dataset beragam terutama dari sisi latar belakang pengambilan citra agar mendapatkan model yang lebih baik lagi dalam melakukan klasifikasi pada berbagai jenis citra alfabet SIBI.

DAFTAR PUSTAKA

[1] A. Harpini, "Infodatin Tunarungu 2019." Kementerian Kesehatan RI, Jakarta, p. 12, 2019.
[2] T. Handhika, R. I. M. Zen, Murni, D. P. Lestari, and I. Sari, "Gesture recognition for Indonesian Sign Language (BISINDO)," *Journal of Physics: Conference Series*, vol. 1028, no. 1, 2018, doi: 10.1088/1742-6596/1028/1/012173.
[3] A. R. Syulistyo, D. S. Hormansyah, and P. Y. Saputra, "SIBI (Sistem Isyarat Bahasa Indonesia) translation using Convolutional Neural Network (CNN)," *IOP Conference Series: Materials Science and Engineering*, vol. 732, no. 1, 2020, doi: 10.1088/1757-899X/732/1/012082.
[4] R. I. Borman, B. Priyopradono, and A. R. Syah, "Klasifikasi Objek Kode Tangan pada Pengenalan Isyarat Alphabet Bahasa Isyarat Indonesia (BISINDO)," no. September, pp. 1–4, 2018, doi: 10.31227/osf.io/c7v2z.
[5] A. A. Gafar and J. Y. Sari, "Sistem Pengenalan Bahasa Isyarat Indonesia dengan Menggunakan Metode Fuzzy K-Nearest Neighbor," *Jurnal ULTIMATICS*, vol. 9, no. 2, pp. 122–128, 2018, doi: 10.31937/ti.v9i2.671.
[6] A. D. Saputra, F. I. Komputer, U. Pembangunan, N. Veteran, and T. Matching, "Klasifikasi Alfabet Bahasa Isyarat Indonesia (Bisindo) Dengan Metode Template Matching Dan K-Nearest Neighbors (Knn)," *Seminar Nasional Mahasiswa Bidang Komputer dan Aplikasinya (SENAMIKA)*, pp. 747–760, 2020.
[7] C. V. Angkoso, M. Fuad, and D. R. Hadiwineka, "Pengenalan Abjad Sistem Isyarat Bahasa Indonesia (Sibi) Berbasis Kamera Depth," *Link*, vol. 24, no. 1, p. 6, 2018, doi: 10.31090/link.v24i1.4.
[8] R. Ridwang, "Pengenalan Bahasa Isyarat Indonesia (SIBI) Menggunakan Leap Motion Controller dan Algoritma Data Mining Naïve Bayes," *Jurnal Insypro (Information System and Processing)*, vol. 2, no. 2, 2017, doi: 10.24252/insypro.v2i2.4070.
[9] T. Handhika, D. P. Lestari, I. Sari, R. I. M. Zen, and Murni, "The generalized learning vector quantization model to recognize Indonesian sign language (BISINDO)," *Proceedings of the 3rd International Conference on Informatics and Computing, ICIC 2018*, 2018, doi: 10.1109/IAC.2018.8780485.
[10] A. Aljabar and Suharjito, "BISINDO (Bahasa isyarat indonesia) sign language recognition using CNN and LSTM," *Advances in Science, Technology and Engineering Systems*, vol. 5, no. 5, pp. 282–287, 2020, doi: 10.25046/AJ050535.
[11] V. Sharma, K. Gupta, and Dr. K. Singh, "Sign Language Detection Using Image Processing," *International Journal of Advanced Research in Science, Communication and Technology*, pp. 100–103, 2021, doi: 10.48175/ijarsct-836.
[12] S. Hossain, D. Sarma, T. Mittra, M. N. Alam, I. Saha, and F. T. Johora, "Bengali Hand Sign Gestures Recognition using Convolutional Neural Network," *Proceedings of the 2nd International Conference on Inventive Research in Computing Applications, ICIRCA 2020*, pp. 636–641, 2020, doi: 10.1109/ICIRCA48905.2020.9183357.
[13] T. M. Angona *et al.*, "Automated bangla sign language translation system for alphabets by means of MobileNet," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 18, no. 3, pp. 1292–1301, 2020, doi: 10.12928/TELKOMNIKA.V18I3.15311.
[14] K. Y. Lum, Y. H. Goh, and Y. Bin Lee, "American sign language recognition based on MobileNetV2," *Advances in Science, Technology and Engineering Systems*, vol. 5, no. 6, pp. 481–488, 2020, doi: 10.25046/aj050657.
[15] M. R. R. Allaam and A. T. Wibowo, "Klasifikasi Genus Tanaman Anggrek Menggunakan Metode Convolutional Neural Network (CNN) Program Studi Sarjana Informatika Fakultas Informatika Universitas Telkom Bandung," vol. 8, no. 2, pp. 3147–3179, 2021.
[16] R. Faurina, E. P. Purwandari, M. T. Pratama, and I. Agustian, "Klasifikasi Level Non-Proliferatif Retinopati Diabetik Dengan Ensemble Convolutional Neural Network," *Pseudocode*, vol. 8, no. 1, pp. 1–10, 2021, doi: 10.33369/pseudocode.8.1.1-10.

[17] A. Howard *et al.*, "Searching for mobileNetV3," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2019-Octob, pp. 1314–1324, 2019, doi: 10.1109/ICCV.2019.00140.
[18] T. N. Abu-Jamie, P. Samy, and S. Abu-Naser, "Classification of Sign-Language Using MobileNet-Deep Learning," 2022. [Online]. Available: www.ijeais.org/ijaisr
[19] I. Mutmainnah, "Mengenal Pandas dalam Python," www.medium.com. Accessed: Jul. 28, 2021. [Online]. Available: https://medium.com/@16611092/mengenal-pandas-dalam-python-cc66d0c5ea40
[20] D. DARMATASIA, "Pengenalan Sistem Isyarat Bahasa Indonesia (Sibi) Menggunakan Gradient-Convolutional Neural Network," *Jurnal INSTEK (Informatika Sains dan Teknologi)*, vol. 6, no. 1, p. 56, 2021, doi: 10.24252/instek.v6i1.18637.
[21] D. Yolanda, K. Gunadi, and E. Setyati, "Pengenalan Alfabet Bahasa Isyarat Tangan Secara Real- Time dengan Menggunakan Metode Convolutional Neural Network dan Recurrent Neural Network," *Infra*, vol. 8, no. 1, pp. 203–208, 2020.
[22] M. Bagus, S. Bakti, and Y. M. Pranoto, "Pengenalan Angka Sistem Isyarat Bahasa Indonesia Dengan Menggunakan Metode Convolutional Neural Network," *Seminar Nasional Inovasi Teknologi*, pp. 11–16, 2019.