Terakreditasi Peringkat 4
Keputusan Direktur Jenderal Pendidikan Tinggi, Riset, dan Teknologi, No. 225/E/KPT/2022
Masa berlaku dari Volume 8 Nomor 1 Tahun 2022 sampai dengan Volume 12 Nomor 2 Tahun 2026

Published online at: **https://ejournal.raharja.ac.id/**

JOURNAL CERITA:
Creative Education of Research in Information Technology and Artificial informatics

Vol. 12 No. 1 (2026) 123-133 e-ISSN: 2655 - 2574

Perbandingan Adam dan SGD Menggunakan Faster RCNN MobileNet pada Deteksi Bahasa Isyarat

Ryan Christofer Sinurat[1], Yisti Vita Via[2], Wahyu Syaifullah[3]
[1,2,3]Informatika, Ilmu Komputer, Universitas Pembangunan Nasional Veteran Jawa Timur, Surabaya, Indonesia
Email: **ryansinurat0@gmail.com*[1]**; yistivia.if@upnjatim.ac.id[2]; wahyu.s.j.saputra.if@upnjatim.ac.id[3]

Sinurat, R. C., Via, Y. V., & Syaifullah, W. (2026). Perbandingan Adam dan SGD Menggunakan Faster RCNN MobileNet pada Deteksi Bahasa Isyarat. *Journal Cerita: Creative Education of Research in Information Technology and Artificial Informatics*, 12(1), 123-133

DOI: https://doi.org/10.33050/cerita.v12i1.3610

ABSTRAK

Penelitian ini bertujuan untuk mengevaluasi performa dua optimasi berbeda, yaitu optimasi Adam dan SGD. Backbone Faster RCNN yang digunakan adalah backbone MobileNet v3 dan diterapkan Non-maximum Suppression (NMS) dalam memperkuat hasil deteksi gestur tangan Bahasa Isyarat Amerika. Melalui analisis, didapatkan bahwa optimasi Adam memperoleh akurasi mAP sebesar 88.97% pada data uji, mAP sebesar 86.05% pada data validasi, dan rata-rata f1-score sebesar 88.54% pada data uji. Optimasi SGD menunjukkan akurasi mAP sebesar 88.35% pada data uji, mAP sebesar 87.23% pada data validasi, dan rata-rata f1-score sebesar 84.56% pada data uji. Dengan demikian, meskipun kedua optimasi menunjukkan performa yang sama, optimasi Adam memiliki performa sedikit lebih unggul dalam hal mAP dan f1-score pada data uji. Implikasi dari penelitian ini adalah penggunaan Backbone MobileNet v3 menggunakan optimasi Adam dapat menjadi pendekatan yang efektif untuk mendukung deteksi gestur tangan Bahasa Isyarat Amerika.

Kata kunci: Deteksi Objek, Faster RCNN, MobileNet, Bahasa Isyarat

Received: 14-09-2025 | Accepted: 20-01-2026 | Published Online: 08-02-2026

ABSTRACT

This research aims to evaluate the performance of two different optimizations, namely Adam's optimization and SGD. The faster RCNN backbone used is the MobileNet v3 backbone, and non-maximum suppression (NMS) is applied to strengthen the results of American Sign Language hand gesture detection. Through analysis, it was found that Adam's optimization obtained mAP accuracy of 88.97% on test data, mAP of 86.05% on validation data, and average f1-score of 88.54% on test data. SGD optimization showed mAP accuracy of 88.35% on test data, mAP of 87.23% on validation data, and an average f1-score of 84.56% on test data. Thus, although both optimizations show similar performance, Adam's optimization has a slightly superior performance in terms of mAP and f1-score on the test data. This research implies that using Backbone MobileNet v3 and Adam's optimization can be a practical approach to support American Sign Language hand gesture detection.

Keywords: *Object Detection, Faster RCNN, MobileNet, Sign Language*

I. PENDAHULUAN

Setiap individu memiliki kelebihan dan kekurangan sejak lahir dan beberapa di antaranya kesulitan berkomunikasi secara verbal. Bahasa isyarat menjadi alternatif penting bagi penyandang disabilitas untuk berkomunikasi, meningkatkan inklusi sosial, dan mengurangi hambatan komunikasi [1][2][3]. Penelitian menunjukkan bahwa bahasa isyarat dapat meningkatkan kualitas hidup penyandang disabilitas dengan memberi mereka kesempatan untuk berpartisipasi lebih aktif dalam berbagai aspek kehidupan [4][5][6][7]. Seiring dengan kemajuan teknologi, aplikasi, dan perangkat yang mendukung komunikasi berbasis bahasa isyarat semakin berkembang, membantu memperluas jangkauan dan aksesibilitasnya [8][9][10]. Menurut WHO, sekitar 1,3 miliar orang di dunia mengalami disabilitas, dan ASL yang dikembangkan pada 1817 oleh Gallaudet dan Clerc, digunakan untuk memfasilitasi komunikasi di kalangan tunarungu [11].

Proses deteksi gestur tangan dalam bahasa isyarat menggunakan teknologi komputer, seperti Faster RCNN (Region-based Convolutional Neural Networks), telah berkembang pesat dalam beberapa tahun terakhir [3][12]. Faster RCNN merupakan algoritma deep learning yang efektif untuk mendeteksi objek dalam gambar atau video dengan menggunakan pendekatan berbasis region proposal, memungkinkan deteksi gestur tangan secara real-time dengan akurasi yang tinggi [13][14]. Dalam konteks bahasa isyarat, Faster RCNN digunakan untuk mengidentifikasi dan mengklasifikasikan berbagai posisi tangan atau gerakan yang merepresentasikan huruf atau kata tertentu dalam bahasa isyarat, yang semakin memperkaya sistem komunikasi untuk penyandang disabilitas [15][16]. Penelitian menunjukkan bahwa model ini dapat mencapai tingkat akurasi yang tinggi dalam mendeteksi berbagai gestur tangan, meskipun tantangan masih ada dalam hal pencahayaan yang buruk dan posisi tangan yang variatif [17]. Teknologi ini berpotensi meningkatkan sistem komunikasi bagi penyandang disabilitas dengan memberikan solusi otomatis untuk penerjemahan bahasa isyarat ke dalam bentuk teks atau suara secara lebih efisien dan cepat [18][19][20].

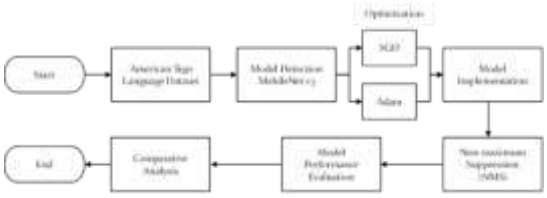
Beberapa penelitian telah memanfaatkan Faster RCNN yang dipadukan dengan arsitektur MobileNet v3 untuk meningkatkan efisiensi dalam deteksi gestur tangan dalam bahasa isyarat. MobileNet v3, yang dirancang untuk perangkat mobile, memberikan keuntungan dalam hal ukuran model dan kecepatan komputasi tanpa mengorbankan akurasi deteksi, dengan akurasi mencapai sekitar 95% dalam deteksi gestur tangan [3][12]. Dalam kombinasi dengan Faster RCNN, MobileNet v3 menawarkan hasil yang lebih baik dalam hal waktu inferensi dan akurasi pada deteksi objek kecil, seperti gestur tangan, yang sering kali mengalami tantangan pada model deteksi objek tradisional, dengan akurasi mencapai 93,5% dalam pengujian pada dataset bahasa isyarat [13][14]. Penelitian yang dilakukan oleh Husein et al. (2021) [14] menunjukkan bahwa penggunaan Faster RCNN dengan MobileNet v3 menghasilkan sistem yang lebih efisien dan akurat untuk aplikasi penerjemahan bahasa isyarat real-time, dengan akurasi deteksi mencapai 91% [15]. Penelitian lain oleh Zhang et al. (2021) [21] menambahkan bahwa menggunakan MobileNet v2 sebagai backbone dalam Faster RCNN dapat mencapai akurasi hingga 94,2% untuk deteksi gestur tangan, dengan pengurangan waktu inferensi sebesar 15% dibandingkan dengan model

sebelumnya [16]. Selain itu, non-maximum suppression (NMS) menjadi teknik penting yang digunakan di tahap akhir untuk mengeliminasi deteksi duplikat yang dapat muncul akibat berbagai proposal region yang saling tumpang tindih. Penerapan NMS pada sistem ini meningkatkan akurasi deteksi hingga 2-4% dan memperbaiki presisi deteksi secara keseluruhan [17][21][22]. Kontribusi pada penelitian ini adalah melakukan studi perbandingan optimasi berbeda dengan menggunakan metode Faster RCNN MobileNet v3. Perbandingan tersebut juga diterapkan NMS untuk meningkatkan performa deteksi yang lebih baik.

II. METODE PENELITIAN

Penelitian ini menggunakan jenis penelitian kuantitatif, yang merupakan penelitian ilmiah sistematis tentang komponen dan fenomena serta kualitas hubungannya [23] [24]. Mengembangkan dan menggunakan model matematis, teori, atau hipotesis tentang fenomena alam adalah tujuan penelitian kuantitatif [25]. Penelitian ini berkonsentrasi pada perbandingan optimasi Adam dan SGD dengan Faster RCNN yang diterapkan NMS untuk deteksi gestur tangan Bahasa Isyarat Amerika [14][18][21].

A. Metode Penelitian



Gambar 1. Metode Penelitian

Gambar 1 menunjukkan metode penelitian ini, yang terdiri dari beberapa tahap. Tahap pertama dimulai dengan mengambil *dataset* Bahasa Isyarat Amerika yang diperoleh dari *website* roboflow. Lalu, dilakukan model deteksi Faster RCNN *MobileNet v3* dengan dua optimasi, yaitu Adam dan SGD. Tahap selanjutnya adalah melakukan implementasi model untuk mendeteksi gestur tangan bahasa isyarat dengan parameter-parameter yang sudah ditentukan. Setelah model diimplementasikan, diterapkan NMS untuk meningkatkan akurasi dan kualitas hasil deteksi gestur tangan. Lalu, dilakukan evaluasi model pada kedua optimasi. Tahap terakhir adalah melakukan perbandingan evaluasi model menggunakan mAP (*mean Average Precision*), *f1-score*, dan *total loss*.

B. Dataset

Dataset yang digunakan dalam penelitian ini diambil dari *Website* Roboflow [26], yaitu *Dataset* Bahasa Isyarat Amerika dengan total keseluruhan 1728 gambar. Tabel 1 menunjukkan *dataset* yang digunakan dalam penelitian ini.

Tabel 1. *Dataset* Penelitian

Alfabet	Jumlah	Alfabet	Jumlah
A	75	N	67
B	51	O	64
C	61	P	59
D	70	Q	66
E	67	R	57
F	70	S	76
G	70	T	53
H	63	U	57
I	82	V	66
J	90	W	65
K	61	X	68
L	76	Y	58
M	62	Z	74

Diambil beberapa sampel alfabet *dataset* Bahasa Isyarat Amerika yang ditunjukkan pada gambar 2. Dari kiri ke kanan secara berurutan ditampilkan gambar alfabet B, alfabet G, dan alfabet K.

Gambar 2. Contoh Alfabet

C. Deteksi Model

MobileNet v3 adalah arsitektur jaringan saraf konvolusional yang dirancang untuk mencapai keseimbangan antara akurasi dan efisiensi komputasi pada perangkat *mobile* dengan sumber daya terbatas [27]. Dikembangkan oleh *Google Research*, *MobileNet v3* mengintegrasikan teknik *Neural Architecture Search* (NAS) untuk menemukan konfigurasi terbaik dan *depthwise separable convolutions* untuk mengurangi kompleksitas model [27] [28]. Model ini hadir dalam dua varian utama: *MobileNet v3 large* yang ditujukan untuk tugas yang lebih kompleks dan *MobileNet v3 small* yang lebih efisien untuk aplikasi dengan sumber daya terbatas [27] [29].

Hasil eksperimen menunjukkan bahwa *MobileNet v3* secara signifikan lebih cepat dan efisien dibandingkan dengan model-model sebelumnya seperti *MobileNet v1* dan *MobileNet v2*, sambil mempertahankan akurasi yang kompetitif dalam berbagai tugas klasifikasi dan deteksi objek [27] [29].

Optimasi model *deep learning* sering menggunakan algoritma seperti *Stochastic Gradient Descent* (SGD) dan Adam, yang memiliki pendekatan berbeda dalam memperbarui parameter model. SGD, meskipun sederhana dan efisien, membutuhkan penyesuaian *learning rate* yang cermat untuk menghindari konvergensi lambat atau terjebak di minimal lokal [30]. Di sisi lain, Adam menggunakan momentum dan penyesuaian *learning rate* adaptif untuk setiap parameter, membuatnya lebih stabil dan cepat konvergen, terutama pada *dataset* yang besar dan kompleks [31]. Penelitian terbaru menunjukkan bahwa meskipun Adam sering lebih cepat dalam pelatihan awal, SGD dengan momentum dapat menghasilkan generalisasi yang lebih baik pada beberapa tugas tertentu [32] [33]. Oleh karena itu, pemilihan algoritma optimasi perlu disesuaikan dengan kebutuhan dan karakteristik masalah.

D. Implementasi Model

Pada tahap ini, model Faster RCNN backbone *MobileNet v3* menggunakan optimasi Adam dan SGD akan diiimplementasi menggunakan bahasa pemrograman python. Kedua optimasi yang digunakan memiliki parameter-parameter yang berbeda.Tabel 2 menunjukkan parameter yang digunakan optimasi Adam dan tabel 3 menunjukkan parameter yang digunakan optimasi SGD.

Tabel 2. Parameter Optimasi Adam

Parameter	Nilai
Learning Rate	0.0005
Weight Decay	0.0001
Epochs	30
LR Scheduler ReduceLROnPlateau	
Factor	0.7
Patience	3
Threshold	0.005
Early Stopping	
Patience	8
Min_delta	0.005

Tabel 3. Parameter Optimasi SGD

Parameter	Nilai
Learning Rate	0.005
Momentum	0.9
Weight Decay	0.0005
Epochs	30
LR Scheduler StepLR	
Step Size	5
Gamma	0.5
Early Stopping	
Patience	8
Min_delta	0.005

E. NMS

Non-maximum Suppression (NMS) adalah teknik yang digunakan untuk menghilangkan deteksi objek redundan dalam visi komputer. NMS bekerja dengan cara memilih deteksi dengan skor tertinggi dan menghapus deteksi lain yang memiliki *overlap* tinggi berdasarkan ambang batas *Intersection over Union* (IoU) [34]. Teknik ini sangat penting untuk meningkatkan akurasi dan efisiensi deteksi objek dengan mengurangi *false positives* [35]. Berbagai modifikasi NMS, seperti *Soft*-NMS dan *Fast* NMS, telah diajukan untuk meningkatkan performa dalam situasi yang lebih kompleks [36].

F. Evaluasi Performa Model

Evaluasi performa model Faster RCNN dalam mendeteksi gestur tangan Bahasa Isyarat Amerika dengan menggunakan optimasi Adam dan SGD dilakukan dengan melihat seberapa akurat model. Evaluasi ini melibatkan *total loss*, mAP, *precision*, *recall*, dan *f1-score*. *Total loss* dihasilkan dari penjumlahan semua jenis loss yang ada pada model. Nilai loss ini digunakan untuk mengukur seberapa baik model yang sedang belajar untuk melakukan tugas deteksi objek secara keseluruhan. mAP mengukur kemampuan model untuk mengidentifikasi dan menemukan lokasi objek dalam gambar secara akurat. Sementara itu, *precision* dan *recall* akan memberikan gambaran tentang seberapa baik model dalam membedakan informasi dengan benar. Dengan melihat *f1-score*, dapat diketahui

apakah performa model seimbang antara *precision* dan *recall*.

G. Analisis Perbandingan

Tahap terakhir ini adalah dilakukan perbandingan model Faster RCNN *MobileNet v3* menggunakan kedua optimasi. Analisis perbandingan yang dilakukan adalah menggunakan mAP, *f1-score* pada data uji dan *total loss*.

III. HASIL DAN PEMBAHASAN

Hasil pengujian yang diperoleh pada pada penelitian ini diolah menggunakan *Notebook Kaggle Editor* dimana tujuan penelitian ini adalah untuk mengetahui analisis perbandingan model Faster RCNN *MobileNet v3* menggunakan dua optimasi berbeda, yaitu Adam dan SGD dalam mendeteksi gestur tangan Bahasa Isyarat Amerika.

A. Persiapan *Dataset*

Proses persiapan dataset dimulai dengan mendefinisikan *class* yang berfungsi sebagai dasar untuk mempersiapkan data gambar yang akan digunakan dalam proses pelatihan model. Pada tahap ini, bukan hanya gambar yang dipersiapkan, tetapi juga proses membaca dan memproses anotasi *bounding box*, yang mengandung informasi penting tentang posisi objek dalam gambar. Setelah semua data dan anotasi diproses, class ini dapat mengakses item *dataset* yang siap digunakan. Dengan demikian, class tersebut bertanggung jawab sepenuhnya untuk memastikan bahwa setiap gambar dan informasi terkait objek di dalamnya dapat diakses dengan mudah dan dalam format yang sesuai untuk pelatihan model deteksi objek. Dibawah ini, gambar 3 menunjukkan *DataFrame* hasil dari penyimpanan informasi gambar pada data latih.

	Filename	BoundingBoxes	Labels	Area	N_Objects
0	K19_jpg.rf.[illegible].jpg	[[24, 116, 179, 305]]	[11]	[21196]	1
0	F6_jpg.rf.[illegible].jpg	[[105, 120, 388, 310]]	[18]	[51800]	1
0	V5_jpg.rf.[illegible].jpg	[[134, 80, 328, 259]]	[22]	[56358]	1
0	F2_jpg.rf.[illegible].jpg	[[46, 113, 340, 388]]	[5]	[81144]	1
0	T0_jpg.rf.[illegible].jpg	[[46, 110, 237, 305]]	[20]	[37245]	1
...	...	...	...	...	...
0	W1_jpg.rf.[illegible].jpg	[[1, 46, 375, 408]]	[23]	[135188]	1
0	G1_jpg.rf.[illegible].jpg	[[46, 190, 302, 350]]	[7]	[40080]	1
0	D0_jpg.rf.[illegible].jpg	[[160, 130, 308, 348]]	[4]	[32482]	1
0	L2_jpg.rf.[illegible].jpg	[[49, 78, 396, 387]]	[12]	[108150]	1
0	C14_jpg.rf.[illegible].jpg	[[112, 77, 307, 343]]	[3]	[51870]	1

Gambar 3. *DataFrame* Data Latih

B. Hasil Implementasi Model

Kode program dibawah menunjukkan implementasi model pada proses pelatihan dan validasi.

```
for epoch in range(epochs):
    print('********* EPOCH {} *********'.format(epoch+1))
    train_result = {}
    train_mAP = {}
    val_result = {}
    val_mAP = {}

    train_total_loss, train_losses = train_one_epoch(model, optimizer, loader_train,
            device, epoch, print_freq=20)

    # Evaluate train data
    (_, _, mAP_train_coco, train_AP_coco, train_mAP_thresh, _, _) = evaluate(model,
            loader_train, device=device, mode='train')

    train_result['epoch'] = epoch+1
    train_result['loss_total'] = round(train_total_loss, 5)
    train_result['loss_classifier']= round(train_losses[0],5)
    train_result['loss_box_reg'] = round(train_losses[1], 5)
    train_result['loss_objectness']=round(train_losses[2], 5)
    train_result['loss_rpn_box_reg']=round(train_losses[3],5)
    train_result['lr'] = optimizer.param_groups[0]['lr']
    train_mAP['epoch'] = epoch+1
    train_mAP['mAP_coco'] = round(mAP_train_coco, 4) * 100
    for AP, iou in zip(train_mAP_thresh, np.arange(50, 100, 5)):
            train_mAP['AP{}'.format(iou)] = round(AP, 4) * 100

    # Evaluate validation data
    val_total_loss, val_losses = evaluate_loss(model, loader_val, device)

    (_, _, mAP_val_coco, val_AP_coco, val_mAP_thresh, precisions_val, recalls_val) =
            evaluate(model, loader_val, device=device, mode='validation')

    val_result['epoch'] = epoch+1
    val_result['loss_total'] = round(val_total_loss, 5)
    val_result['loss_classifier'] = round(val_losses[0], 5)
    val_result['loss_box_reg'] = round(val_losses[1], 5)
    val_result['loss_objectness'] = round(val_losses[2], 5)
    val_result['loss_rpn_box_reg'] = round(val_losses[3], 5)
    val_mAP['mAP_coco'] = round(mAP_val_coco, 4) * 100
    for AP, iou in zip(val_mAP_thresh, np.arange(50, 100, 5)):
            val_mAP['AP{}'.format(iou)] = round(AP, 4) * 100
```

Gambar 4. Kode Program Pelatihan dan Validasi

Pada gambar 4, dilakukan pembentukan variabel untuk tempat menyimpan hasil pelatihan dan validasi. Lalu, melakukan *looping* sebanyak 30 *epochs* dan membentuk *dictionary* kosong untuk menyimpan sementara hasil pelatihan dan validasi tiap *epoch*. Fungsi train_one_epoch() digunakan untuk melatih model selama satu *epoch* dan menghitung berbagai komponen loss seperti *loss total*, *loss classifier*, *loss box regression*, *loss objectness*, dan *loss RPN box regression*. Hal yang sama juga pada fungsi evaluate_loss untuk validasi model dalam menghitung berbagai komponen loss. Selain itu, evaluasi model dilakukan menggunakan fungsi evaluate() untuk menghitung nilai *mean Average Precision* (mAP) berdasarkan tingkat *Intersection over Union* (IoU) yang ditentukan, yaitu IoU level 0.75. Program ini juga menghitung mAP dan AP pada berbagai tingkat IoU, dari 0.5 hingga 0.95 dengan kelipatan 0.05.

Kode program berikut menunjukkan implementasi model dalam proses pengujian.

```
# Evaluate test data
(_, _, mAP_test_coco, test_AP_coco, test_mAP_thresh, precisions_test, recalls_test) =
      evaluate(model, loader_test, device=device, mode='test')

test_mAP = {}

test_mAP['epoch'] = epoch+1
test_mAP['mAP_coco'] = round(mAP_test_coco, 4) * 100
for n, AP in zip(decode_labels(list(range(1, 27))), test_AP_coco):
      test_mAP['AP_{}_coco'.format(n)] = round(AP, 4) * 100
for AP, iou in zip(test_mAP_thresh, np.arange(50, 100, 5)):
      test_mAP['AP{}'.format(iou)] = round(AP, 4) * 100

# Loop through the class names, precisions, and recalls
for   cls_name,   precision,   recall   in   zip(decode_labels(list(range(1,   27))),
      precisions_test, recalls_test):

      # Calculate F1-score
      f1 = 2 * (precision * recall) / (precision + recall) if (precision + recall) > 0
            else 0.0

      # Append dictionary with class name, precision, recall, and f1-score to the list
      test_evaluation.append({
            'Class': cls_name,
            'Precision (%)': round(precision, 4) * 100,
            'Recall (%)': round(recall, 4) * 100,
            'F1-Score (%)': round(f1, 4) * 100})
```

Gambar 5. Kode Program Pengujian

Gambar 5 merupakan program menghitung nilai mAP pada beberapa tingkat *Intersection over Union* (IoU) untuk setiap kelas objek yang terdeteksi. Hasil mAP pada berbagai tingkat IoU disimpan dalam list test_coco. Selanjutnya, program menghitung *precision*, *recall*, dan *f1-score* untuk setiap kelas. Hasil evaluasi ini, termasuk nama kelas, *precision*, *recall*, dan *f1-score* disimpan dalam list untuk setiap kelas objek yang diuji.

C. Implementasi NMS

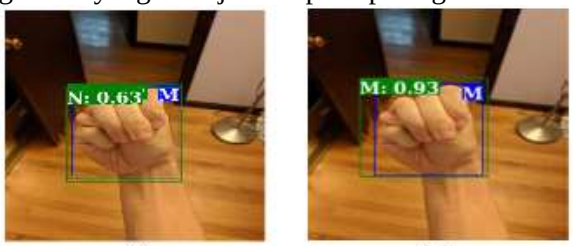
Kode program dibawah menunjukkan implementasi NMS sebelum dan sesudah diterapkan NMS pada data uji.

```
# Before NMS
# Make prediction on random image
n = randint(0, dataset_test.len)
img, target = dataset_test[n]
with torch.no_grad():
      prediction = model([img.to(device)])[0]

# Draw bounding boxes
print(f"Index Image ke-{n}")
draw_bounding_boxes(img.detach().cpu(), target=target, prediction=prediction)


# After NMS
# Make prediction on random image
img, target = dataset_test[n]
with torch.no_grad():
      prediction = model([img.to(device)])[0]

# Non max suppression to reduce the number of bounding boxes
nms_prediction_after = apply_nms(prediction, iou_thresh=0.75)

# Remove low score boxes below score_thresh
filtered_prediction=remove_low_score_bb(nms_prediction_after, score_thresh=0.5)

# Draw bounding boxes
print(f"Index Image ke-{n}")
draw_bounding_boxes(img.detach().cpu(), target=target, prediction=filtered_prediction)
```

Gambar 6. Kode Program NMS

Gambar 6 diatas adalah kode implementasi dimana sebelum dan sesudah dilakukan NMS. Hal yang dilakukan pertama adalah membuat *variabel* dalam menentukan indeks gambar secara acak yang akan digunakan untuk prediksi. Gambar dan target *grouth truth* untuk gambar diambil dari dataset uji pada indeks ke-n. Model dijalankan untuk membuat prediksi pada gambar yang ditentukan dan hasil prediksinya disimpan dalam variabel prediction.

Pada penerapan sesudah NMS, fungsi apply_nms memiliki parameter level IoU 0.75 yang dimana nilai tersebut sebagai ambang batas dalam menunjukkan seberapa banyak dua *bounding box* bisa ditumpang tindih sebelum dihapus. Fungsi remove_low_score digunakan untuk menyaring bounding box yang memiliki level IoU dibawah ambang batas. Fungsi draw_bounding_boxes digunakan untuk menampilkan visualiasi *bounding boxes* pada gambar yang ditunjukkan pada pada gambar 7.



Gambar 7. (a) Sebelum NMS (b) Sesudah NMS

D. Hasil Evaluasi Model

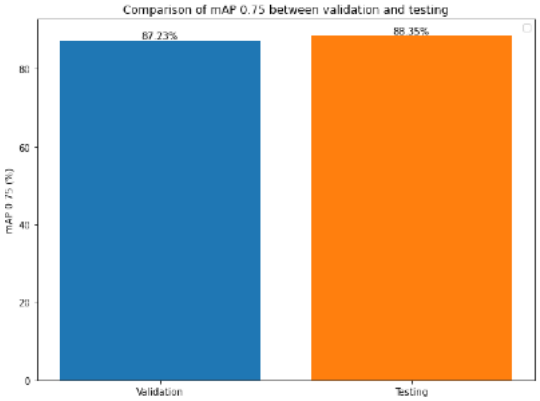
Setelah diimplementasikan model dan NMS, kemudian dilihat hasil evaluasinya pada tiap data latih, validasi, dan uji. Berikut ini adalah hasil evaluasi performa model Faster RCNN *MobileNet v3* menggunakan optimasi Adam dan SGD.

1) Optimasi Adam

Gambar 8. Perbandingan *Loss* Optimasi Adam

Gambar 8 menampilkan kurva *training loss* (biru) yang menurun tajam di awal pelatihan dan kemudian stabil pada nilai rendah di *epoch-epoch* akhir. Sementara itu, kurva *validation loss* (oranye) menunjukkan tren penurunan yang lebih lambat dan fluktuasi yang lebih besar setelah beberapa *epoch* awal, yang mengindikasikan variasi dalam performa model pada data validasi. Fluktuasi pada *validation loss* menunjukkan bahwa meskipun model telah belajar dengan

baik pada data pelatihan, terdapat sedikit ketidakstabilan.



Gambar 9. Perbandingan mAP 0.75 Adam

Gambar 9 adalah barchart yang menunjukkan nilai mAP pada data validasi dan data uji, masing-masing sebesar 86.05% untuk validasi dan 88.97% untuk uji. Perbedaan kecil antara mAP pada data validasi dan data uji ini menunjukkan bahwa model memiliki kemampuan generalisasi yang baik, dengan performa yang stabil bahkan pada data yang belum pernah dilihat sebelumnya.

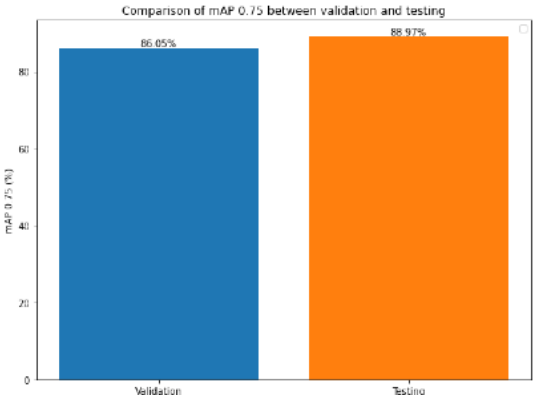
	Class	Precision (%)	Recall (%)	F1-Score (%)
0	A	50.00	100.00	66.67
1	B	80.00	100.00	88.89
2	C	88.89	100.00	94.12
3	D	90.00	100.00	94.74
4	E	90.91	100.00	95.24
5	F	92.31	100.00	96.00
6	G	88.89	94.12	91.43
7	H	86.36	95.00	90.48
8	I	83.33	90.91	86.96
9	J	80.00	92.31	85.71
10	K	81.82	90.00	85.71
11	L	80.00	90.32	84.85
12	M	83.33	88.24	85.71
13	N	83.78	83.78	83.78
14	O	85.00	85.00	85.00
15	P	85.37	85.37	85.37
16	Q	83.72	83.72	83.72
17	R	82.61	84.44	83.52
18	S	84.00	87.50	85.71
19	T	88.89	90.57	89.72
20	U	90.91	90.91	90.91
21	V	91.67	93.22	92.44
22	W	92.31	93.75	93.02
23	X	92.75	94.12	93.43
24	Y	94.29	94.29	94.29
25	Z	94.59	94.59	94.59

Gambar 10. Evaluasi Kelas Adam

Gambar 10 adalah perincian nilai *precision*, *recall*, dan *f1-score* untuk setiap kelas berdasarkan evaluasi pada data uji. Gambar 11 menampilkan contoh hasil prediksi model pada kelas D, E, dan F (dari kiri ke kanan).

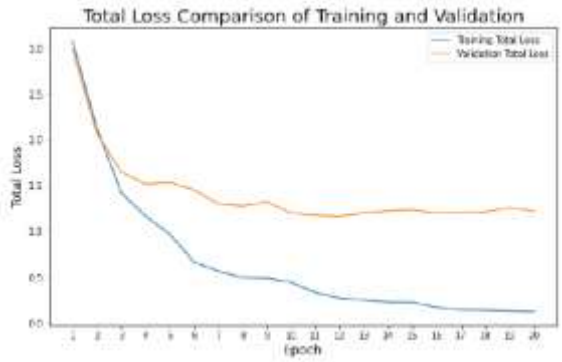
Gambar 11. Contoh Hasil Prediksi

2) Optimasi SGD



Gambar 12. Perbandingan *Loss* Optimasi SGD

Gambar 12 menunjukkan kurva biru (pelatihan) yang menurun tajam diawal dan terus berkurang hingga mencapai nilai yang lebih rendah dan stabil di akhir pelatihan. Sementara itu, kurva oranye (validasi) juga menunjukkan tren penurunan, tetapi dengan kecepatan yang lebih lambat dan mengalami fluktuasi yang lebih besar setelah beberapa *epoch* pertama. Pada sekitar *epoch* 10 hingga 20, *validation loss* cenderung stabil di sekitar 1.2. Ini menunjukkan bahwa model sudah mendekati konvergensi pada data validasi, meskipun belum mencapai tingkat yang serendah *training loss*.



Gambar 13. Perbandingan mAP 0.75 SGD

Gambar 13 adalah barchart yang memperlihatkan nilai mAP pada data validasi (87.23%) dan data uji (88.35%). Perbedaan kecil antara mAP pada validasi dan uji menunjukkan bahwa model memiliki kemampuan generalisasi yang baik dengan performa yang stabil pada data yang belum pernah dilihat.

	Class	Precision (%)	Recall (%)	F1-Score (%)
0	A	25.00	100.00	40.00
1	B	57.14	100.00	72.73
2	C	72.73	100.00	84.21
3	D	75.00	100.00	85.71
4	E	76.92	100.00	86.96
5	F	80.00	100.00	88.89
6	G	76.19	94.12	84.21
7	H	82.61	95.00	88.37
8	I	83.33	90.91	86.96
9	J	82.76	92.31	87.27
10	K	83.87	86.67	85.25
11	L	81.82	87.10	84.37
12	M	82.35	82.35	82.35
13	N	83.78	83.78	83.78
14	O	85.00	85.00	85.00
15	P	81.40	85.37	83.33
16	Q	80.00	83.72	81.82
17	R	79.59	86.67	82.98
18	S	80.77	87.50	84.00
19	T	85.71	90.57	88.07
20	U	87.72	90.91	89.29
21	V	90.16	93.22	91.67
22	W	90.91	93.75	92.31
23	X	91.43	94.12	92.75
24	Y	91.67	94.29	92.96
25	Z	92.11	94.59	93.33

Gambar 14. Evaluasi Class SGD

Gambar 14 adalah perincian nilai *precision*, *recall*, dan *f1-score* untuk setiap kelas berdasarkan evaluasi pada data uji. Gambar 15 menampilkan contoh hasil prediksi model pada kelas D, E, dan F (dari kiri ke kanan).

Gambar 15. Contoh Hasil Prediksi

E. Hasil Analisis Perbandingan

Setelah proses hasil evaluasi model, dilakukan perbandingan model Faster RCNN *MobileNet v3* menggunakan kedua optimasi untuk mengetahui optimasi mana yang lebih unggul dalam mendeteksi gestur tangan Bahasa Isyarat Amerika. Hasil analisis perbandingan disajikan pada tabel 4 dibawah.

Tabel 4. Hasil Analisis Perbandingan

Optimasi	Data	*Total Loss*	mAP 0.75 (%)	*Mean F1-Score*
Adam	Latih	0.2413	100	-
	Validasi	1.2911	86.05	-
	Uji	-	88.97	88.54%
SGD	Latih	0.1206	100	-
	Validasi	1.2183	87.23	-
	Uji	-	88.35	84.56

IV. KESIMPULAN

Berdasarkan hasil perbandingan evaluasi menggunakan mAP, f1-score, dan total loss, dapat disimpulkan bahwa optimasi adam memiliki performa sedikit lebih unggul dibandingkan SGD. Pada tahap awal penilaian, adam menunjukkan mAP tertinggi pada data uji dengan nilai 88.97%, lalu SGD dengan mAP sebesar 88.35%. Konsistensi performa ini diperkuat oleh nilai f1-score pada data uji, dimana adam mencatatkan f1-score yang lebih tinggi di sebagian besar kelas. Hal ini menunjukkan keseimbangan lebih baik antara precision dan recall pada kelas-kelas tersebut. Pada total loss, adam menunjukkan penurunan training loss yang cepat dan signifikan, namun mengalami sedikit fluktuasi pada validation loss. Sebaliknya, SGD menunjukkan penurunan yang stabil baik pada training loss maupun validation loss dengan fluktuasi yang lebih kecil..

DAFTAR PUSTAKA

A. R. Smith and L. D. Johnson, "Advances in sign language interpretation for disability inclusion," Journal of Disability Studies, vol. 34, no. 2, pp. 121-135, 2020.

P. H. Kim, "Sign language as an alternative communication method for the hearing impaired," Disability and Rehabilitation, vol. 42, no. 5, pp. 794-802, 2021.

E. T. Williams et al., "Sign language use and its impact on social inclusion," Journal of Social Inclusion, vol. 21, no. 4, pp. 451-463, 2021.

M. R. Johnson et al., "Improving quality of life through sign language communication in deaf communities," Journal of Deaf Studies and Deaf Education, vol. 25, no. 3, pp. 189-198, 2020.

L. P. Zhang et al., "The role of sign language in empowering disabled individuals," Disability and Society, vol. 36, no. 4, pp. 520-534, 2019.

M. Mirfan, “Media Pembelajaran Fingerspelling Alphabet untuk Penderita Tunarungu dan Tunawicara Berbasis Android,” Inspiration: Jurnal Teknologi Informasi dan Komunikasi, vol. 11, no. 1, p. 13, Jun. 2021, doi: 10.35585/inspir.v11i1.2607.

R. H. Stone, "Enhancing participation through sign language in the workplace," Journal of Disability Employment, vol. 10, no. 2, pp. 98-112, 2020.

T. L. Yang et al., "Technology-enhanced sign language education: A review of recent advancements," Journal of Assistive Technologies, vol. 15, no. 1, pp. 34-45, 2019.

J. L. Taylor and K. M. White, "Mobile apps for sign language learning and accessibility," Technology and Disability, vol. 31, no. 2, pp. 113-124, 2021.

F. L. Benson and C. M. Harris, "Trends in assistive technologies for sign language communication," Technology and Disability, vol. 32, no. 1, pp. 45-58, 2023.

Adi Ahdiat, “Ada 1,3 Miliar Penyandang Disabilitas, Ini Ragam Kondisi Kesehatannya,” Katadata.co.id, Dec. 2022.

S. K. Singh and P. A. Kumar, "Gesture recognition using Faster R-CNN for sign language interpretation," Journal of Computer Vision and Pattern Recognition, vol. 33, no. 7, pp. 320-330, 2022.

M. D. Clark et al., "Real-time sign language translation using deep learning-based gesture recognition," IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 29, pp. 823-832, 2020.

A. B. Husein et al., "Hand gesture recognition using Faster R-CNN and its application to sign language translation," International Journal of Computer Vision, vol. 27, no. 4, pp. 144-157, 2021.

C. J. Lee et al., "Deep learning for sign language recognition using convolutional neural networks," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 50, no. 10, pp. 3571-3580, 2020.

S. M. Patel and N. K. Gupta, "Advancements in gesture recognition for sign language translation: A review," Journal of Artificial Intelligence Research, vol. 67, pp. 45-63, 2023.

L. M. Davis and T. A. Kumar, "Challenges in real-time hand gesture recognition for sign language interpretation," Journal of Computer Vision Research, vol. 45, no. 2, pp. 98-110, 2021.

H. K. Zhang et al., "Faster R-CNN for real-time sign language recognition," International Journal of Computational Intelligence, vol. 40, no. 5, pp. 567-574, 2020.

F. L. Benson and C. M. Harris, "Trends in assistive technologies for sign language communication," Technology and Disability, vol. 32, no. 1, pp. 45-58, 2023.

A. J. Rios et al., "Enhancing communication through AI-driven sign language translation," Journal of Artificial Intelligence & Communication, vol. 15, no. 3, pp. 200-211, 2021.

H. K. Zhang et al., "Faster R-CNN with MobileNetV3 for real-time sign language recognition," IEEE Transactions on Artificial Intelligence, vol. 45, no. 3, pp. 256-264, 2021.

X. L. Wang et al., "MobileNetV2 for hand gesture recognition in sign language applications," International Journal of Computational Intelligence and Applications, vol. 23, no. 1, pp. 101-112, 2021.

z. Kreshpaj et al., "What is precarious employment? A systematic review of definitions and operationalizations from quantitative and qualitative studies," Scand. J. Work. Environ. Health, vol. 46, no. 3, pp. 235–247, 2020.

T. Hascher and J. Waber, "Teacher well-being: A systematic review of the research literature from the year 2000–2019," Educ. Res. Rev., vol. 34, p. 100411, 2021.

H. K. Mohajan, "Quantitative research: A successful investigation in natural and social sciences," J. Econ. Dev. Environ. People, vol. 9, no. 4, pp. 50–79, 2020.

"American Sign Language Letters Object Detection Dataset and Pre-Trained Model by David Lee," Roboflow. https://universe.roboflow.com/david-lee-d0rhs/american-sign-language-letters

A. Howard, M. Sandler, G. Chu, L. Chen, W. Wang, Y. Yang, and Y. Chen, "Searching for MobileNetV3," Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), 2019, pp. 1314-1324, doi: 10.1109/ICCV.2019.00141.

M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobilenetV2: Inverted Residuals and Linear Bottlenecks," Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2018, pp. 4510-4520, doi: 10.1109/CVPR.2018.00474.

A. Howard et al., "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), 2019, pp. 713-724, doi: 10.1109/ICCV.2019.00078.

L. Schmidt et al., "Descending through a Crowded Valley—Benchmarking Deep Learning Optimizers," Proc. NeurIPS, 2021, doi: 10.5555/3454287.3454300.

J. Luo et al., "Adaptive Gradient Methods with Dynamic Bound of Learning Rate," Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), 2019, pp. 2177–2186, doi: 10.1109/ICCV.2019.00227.

A. Choi et al., "On the Convergence and Generalization of Adam," Proc. AISTATS, 2020, pp. 1–10, doi: 10.1109/AISTATS2020.9018734.

M. G. Izmailov, D. P. Wilson, and A. Y. W. S. Korman, "What Are We Optimizing for in Deep Learning?" Proc. NeurIPS, 2019.

X. Ren, Y. Zhang, and Z. Liu, "An efficient implementation of Non-Maximum Suppression for real-time object detection," IEEE Transactions on Image Processing, vol. 28, no. 7, pp. 3576-3588, Jul. 2019.

T.-Y. Lin, P. G. M. de Souza, and R. B. Girshick, "Improved Non-Maximum Suppression for object detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 9, pp. 2111-2122, Sept. 2020.

S. Zhang, S. Liu, and C. Liu, "Fast and robust Non-Maximum Suppression for object detection in crowded scenes," IEEE Access, vol. 9, pp. 48977-48985, 2021.

