Jurnal Pendidikan dan Teknologi Indonesia (JPTI)
Vol. 5, No. 7, Juli 2025, Hal. 2101-2115
DOI: https://doi.org/10.52436/1.905
p-ISSN: 2775-4227
e-ISSN: 2775-4219

Analisis Akurasi Deteksi Objek Menggunakan TensorFlow dengan Metode Single Shot Detection untuk Pengenalan Karakter Animasi dalam Film Battle of Surabaya

Engel Bertus Triesa Wea*[1], Dhani Ariatmanto[2]

[1,2]Magister Teknik Informatika, Fakultas Ilmu Komputer, Universitas Amikom Yogyakarta, Indonesia
Email: [1]engel@students.amikom.ac.id, [2]dhaniari@amikom.ac.id

Abstrak

Pendeteksian karakter dalam film animasi secara otomatis masih menghadapi tantangan signifikan, seperti kompleksitas latar belakang, variasi pencahayaan, dan perbedaan pose karakter, yang dapat menurunkan akurasi sistem. Penelitian ini bertujuan untuk meningkatkan akurasi pendeteksian karakter animasi dengan menerapkan metode *Single Shot Detection* (SSD) menggunakan TensorFlow. Studi kasus difokuskan pada karakter dalam film animasi *Battle of Surabaya*. Metode SSD dipilih karena kemampuannya dalam melakukan deteksi objek secara cepat dan efisien dalam satu tahap pemrosesan. Dataset yang digunakan terdiri dari 30 gambar karakter Musa, Yumna, dan Danu dalam berbagai pose dan kondisi visual. Proses pelatihan model dilakukan menggunakan pendekatan *transfer learning* dan augmentasi data untuk meningkatkan keragaman data latih serta mengurangi risiko overfitting. Hasil pengujian menunjukkan bahwa model mencapai akurasi rata-rata sebesar **99%** pada dataset uji, dengan performa yang tetap stabil meskipun terdapat variasi latar belakang dan pencahayaan. Analisis lebih lanjut juga dilakukan terhadap faktor-faktor yang memengaruhi kinerja model. Penelitian ini berkontribusi dalam pengembangan sistem deteksi karakter animasi yang akurat dan efisien, dengan potensi penerapan dalam industri animasi, sistem pengawasan visual, aplikasi edukatif interaktif, dan pengembangan teknologi *computer vision*. Selain itu, sistem ini dapat mendukung identifikasi karakter secara otomatis untuk tujuan perlindungan hak cipta visual pada media digital.

Kata kunci: *Pengenalan Karakter Animasi, Object Detection, Single Shot Detection (SSD), TensorFlow, Plagiasi Animasi.*

1. PENDAHULUAN

Pembedaan antar karakter dalam film animasi memiliki peranan penting dalam penggambaran aspek mental, emosional, dan sosial karakter tersebut. Karakter yang dibangun oleh desainer tidak hanya berfokus pada tampilan visual yang menarik, tetapi juga dirancang untuk mencerminkan sifat manusiawi seperti motivasi, cara berpikir, dan tindakan, sehingga menciptakan ilusi kehidupan nyata[1]. Pengenalan karakter secara otomatis dalam film animasi menjadi esensial untuk mendukung interaksi yang imersif antara penonton dan cerita yang disampaikan .

Adapun penelitian sebelumnya telah mengeksplorasi deteksi karakter animasi, terutama dalam konteks perlindungan hak cipta dan isu plagiarisme, seperti penyalinan gerakan dan ekspresi karakter[1]. Proses pelatihan model deteksi dalam penelitian-penelitian tersebut umumnya melibatkan tahap pra-pemrosesan data, anotasi menggunakan alat seperti LabelImg, dan pelatihan model berbasis algoritma deteksi objek .

Teknologi object detection sendiri terus mengalami kemajuan pesat, terutama pada metode Single Shot Detection (SSD) yang dikenal karena kecepatan dan efisiensinya dalam melakukan deteksi secara real-time.[1] TensorFlow, sebagai salah satu pustaka machine learning populer yang dikembangkan oleh Google, menyediakan fleksibilitas dalam merancang dan melatih berbagai model deep learning, termasuk SSD, untuk tugas deteksi objek yang kompleks .[2]

Beberapa studi telah berhasil menerapkan metode SSD dalam pengenalan karakter animasi. Salah satunya mengembangkan pendekatan cascaded SSD yang ditingkatkan untuk mendeteksi karakter kecil dengan latar belakang kompleks, menghasilkan peningkatan akurasi melalui teknik seperti peningkatan fungsi kehilangan dan Non-Maximum Suppression (NMS). Namun, Meskipun metode SSD telah banyak digunakan dalam berbagai studi deteksi objek, penerapannya secara spesifik untuk mendeteksi karakter dalam film animasi Indonesia dengan ciri visual yang rumit dan bervariasi belum banyak diteliti secara mendalam. Riset terkait efektivitas metode SSD dalam mengenali karakter animasi lokal, terutama dalam konteks film dengan nilai artistik dan latar naratif yang kuat seperti *Battle of Surabaya*, masih jarang dijumpai dan menyisakan celah penting untuk diteliti lebih lanjut.

Penelitian ini bertujuan untuk menganalisis akurasi model object detection menggunakan TensorFlow dengan metode SSD dalam mengenali karakter animasi, dengan studi kasus karakter Musa, Yumna, dan Danu dalam film Battle of Surabaya. Penelitian ini mengevaluasi kinerja model dalam berbagai kondisi visual, serta memberikan kontribusi dalam pengembangan sistem deteksi karakter animasi yang efisien dan mendukung perlindungan hak cipta visual di era digital.

p-ISSN: 2775-4227
e-ISSN: 2775-4219

2. METODE PENELITIAN

Penelitian ini melibatkan pemrosesan gambar, pelatihan model, dan pengujian model dengan data bahasa isyarat.dan juga metode yang digunakan dalam penelitian ini adalah melalui studi literatur, perancangan sistem, pengujian sistem serta analisis hasil akurasi pengujian sistem.

2.1. Dataset

Dataset yang digunakan dalam penelitian ini merupakan kumpulan gambar karakter dari film animasi Battle of Surabaya, terdiri dari tiga karakter utama: Musa, Yumna, dan Danu. Jumlah data asli sebanyak 150 gambar.dengan data training sebanyak 120 gambar dan data testing atau evaluasi 30 gambar.setiap gambar memiliki resolusi 640x480.

Gambar 1 dataset untuk dilabeli

2.2. Preprocessing

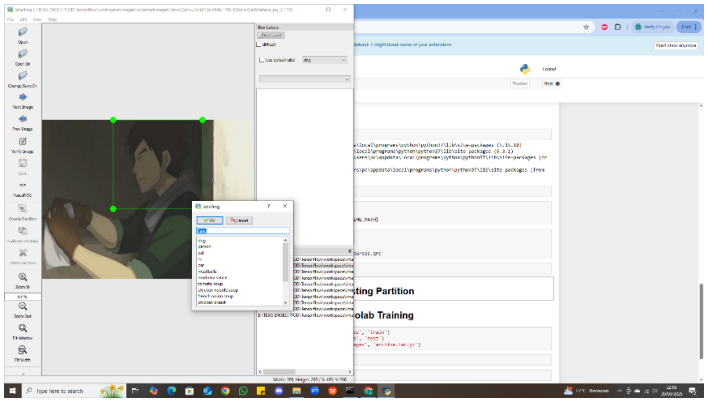
Dalam preprocessing yaitu melakukan labeling terhadap image model yang telah di kumpulkan menggunakan aplikasi labelimg[4]. Pada tahap ini, aplikasi LabelImg digunakan untuk memberikan label pada objek yang terdapat dalam gambar yang akan digunakan untuk pelatihan model. Pelabelan yang akurat dan konsisten sangat penting dalam menentukan kualitas model deteksi objek yang dihasilkan. Seperti yang dijelaskan oleh[5] LabelImg adalah alat yang efektif untuk memberikan label pada gambar secara manual, yang memungkinkan pelatihan model yang lebih presisi. Dalam konteks ini, setiap gambar harus diberi label dengan benar agar model dapat memahami fitur objek yang relevan.

Sebagaimana dibahas [6]kualitas pelabelan sangat memengaruhi performa model dalam mendeteksi objek. Jika label yang diberikan tidak konsisten atau tidak presisi, maka model yang dilatih dengan data tersebut akan kesulitan dalam mengenali objek dengan akurat. Oleh karena itu, pelabelan yang tepat pada data gambar sangat penting untuk keberhasilan sistem deteksi gerakan tangan yang diinginkan.

```
In [12]: !pip install --upgrade pyqt5 lxml

         Requirement already satisfied: pyqt5 in c:\users\pc\appdata\local\programs\python\python37\lib\site-p
         ackages (5.15.10)
         Requirement already satisfied: lxml in c:\users\pc\appdata\local\programs\python\python37\lib\site-pa
         ckages (5.3.1)
         Requirement already satisfied: PyQt5-sip<13,>=12.13 in c:\users\pc\appdata\local\programs\python\pyth
         on37\lib\site-packages (from pyqt5) (12.13.0)
         Requirement already satisfied: PyQt5-Qt5>=5.15.2 in c:\users\pc\appdata\local\programs\python\python3
         7\lib\site-packages (from pyqt5) (5.15.2)

In [13]: LABELIMG_PATH = os.path.join('Tensorflow', 'labelimg')

In [14]: if not os.path.exists(LABELIMG_PATH):
             !mkdir {LABELIMG_PATH}
             !git clone https://github.com/tzutalin/labelImg {LABELIMG_PATH}

In [15]: if os.name == 'posix':
             !make qt5py3
         if os.name == 'nt':
             !cd {LABELIMG_PATH} && pyrcc5 -o libs/resources.py resources.qrc

In [16]: !cd {LABELIMG_PATH} && python labelImg.py

         Image:E:\DOKUMEN\TFOD\Tensorflow\workspace\images\collectedimages\\L.8d5c1aca-ed3f-11ef-bb9e-b42e993
         a6eca.jpg -> Annotation:E:/DOKUMEN/TFOD/Tensorflow/workspace/images/collectedimages/\L.8d5c1aca-ed3f
         -11ef-bb9e-b42e993a6eca.xml
         Image:E:\DOKUMEN\TFOD\Tensorflow\workspace\images\collectedimages\\L.8e902128-ed3f-11ef-a68f-b42e993
         a6eca.jpg -> Annotation:E:/DOKUMEN/TFOD/Tensorflow/workspace/images/collectedimages/\L.8e902128-ed3f
         -11ef-a68f-b42e993a6eca.xml
         Image:E:\DOKUMEN\TFOD\Tensorflow\workspace\images\collectedimages\\L.8fc7ee92-ed3f-11ef-91c2-b42e993
         a6eca.jpg -> Annotation:E:/DOKUMEN/TFOD/Tensorflow/workspace/images/collectedimages/\L.8fc7ee92-ed3f
         -11ef-91c2-b42e993a6eca.xml
         Image:E:\DOKUMEN\TFOD\Tensorflow\workspace\images\collectedimages\\L.90fd6b98-ed3f-11ef-90ff-b42e993
         a6eca.jpg -> Annotation:E:/DOKUMEN/TFOD/Tensorflow/workspace/images/collectedimages/\L.90fd6b98-ed3f
         -11ef-90ff-b42e993a6eca.xml
         Image:E:\DOKUMEN\TFOD\Tensorflow\workspace\images\collectedimages\\L.9234d4cc-ed3f-11ef-aac0-b42e993
         a6eca.jpg -> Annotation:E:/DOKUMEN/TFOD/Tensorflow/workspace/images/collectedimages/\L.9234d4cc-ed3f
         -11ef-aac0-b42e993a6eca.xml
```

Gambar 2 command labelimg

p-ISSN: 2775-4227
e-ISSN: 2775-4219



Gambar 3 proses labeling menggunakan aplikasi labelimg

Pada Gambar 3, terlihat proses pelabelan menggunakan aplikasi LabelImg yang memungkinkan pengguna untuk menandai karakter animasi pada gambar yang telah dipilih. Proses ini mencakup penandaan area yang relevan pada gambar dan memberikan label sesuai dengan nama karakter yang terdeteksi, seperti Musa, Yumna, atau Danu. Pelabelan yang konsisten pada data gambar ini akan sangat membantu model dalam mengenali pola visual dari setiap karakter secara lebih akurat, sehingga meningkatkan kinerja model dalam mendeteksi objek[7]

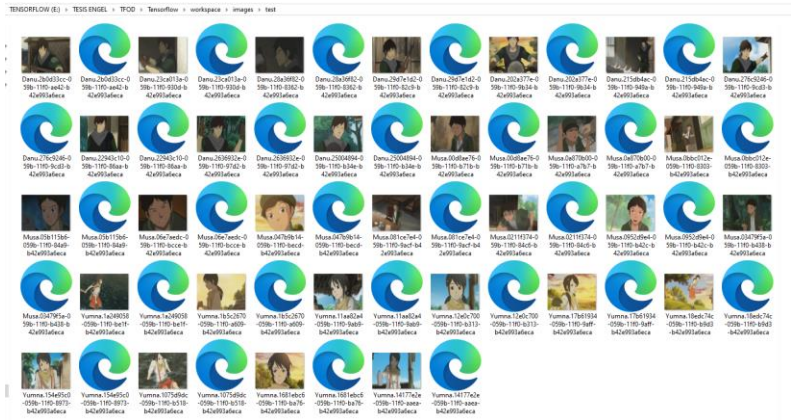
Gambar 4 proses labeling menggunakan aplikasi labelimg

. Pada Gambar 4, terlihat proses pelabelan menggunakan aplikasi LabelImg yang memungkinkan pengguna untuk menandai karakter animasi pada gambar yang telah dipilih. Proses ini mencakup penandaan area yang relevan pada gambar dan memberikan label sesuai dengan nama karakter yang terdeteksi, seperti Musa, Yumna, atau Danu. Pelabelan yang konsisten pada data gambar ini akan sangat membantu model dalam mengenali pola visual dari setiap karakter secara lebih akurat, sehingga meningkatkan kinerja model dalam mendeteksi objek.[8]

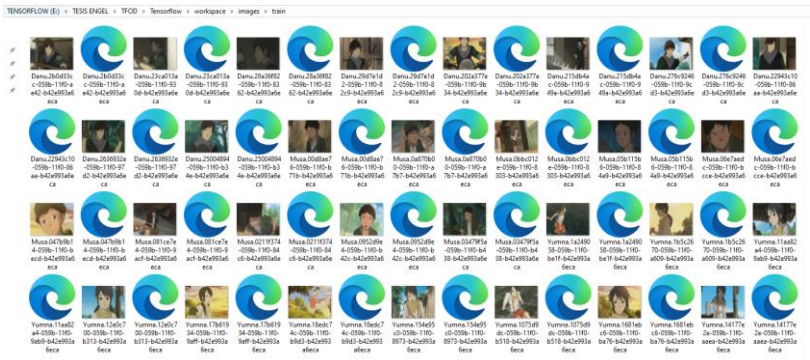
2.3. Setup Folder Untuk Test Dan Training Image Model

Dalam persiapan data untuk pelatihan model adalah membuat struktur folder yang sesuai untuk membagi dataset menjadi dua bagian utama: training dan test, yaitu dengan memindahkan gambar yang telah dilabeli ke dalam folder yang sesuai untuk memastikan pemisahan antara data yang digunakan untuk pelatihan dan evaluasi model[9]

p-ISSN: 2775-4227
e-ISSN: 2775-4219



Gambar 5 Setup folder untuk test image model



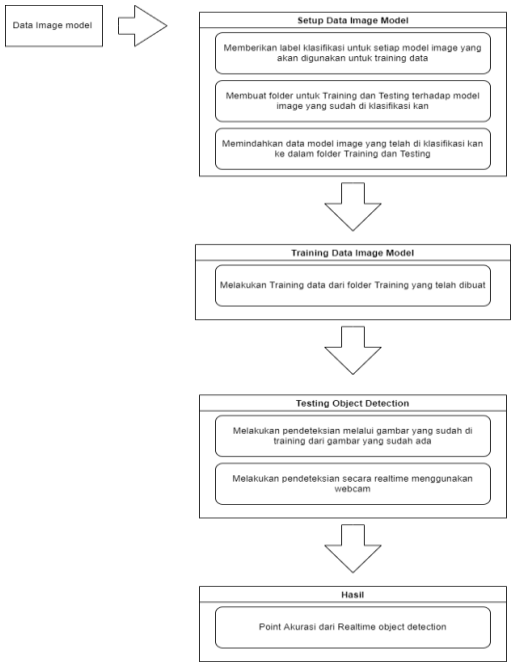
Gambar 6 Setup folder untuk training image model

Pembagian dataset pada gambar 5 dan 6 ini penting untuk menjaga kualitas generalisasi model, di mana model harus mampu mengenali objek pada gambar yang belum pernah dipelajari sebelumnya. Sebagai referensi [10]dalam penelitiannya mengenai pemisahan dataset untuk pelatihan dan pengujian model deteksi objek menekankan pentingnya pembagian yang tepat antara data pelatihan dan data pengujian untuk menghindari bias yang dapat menurunkan akurasi model [11]

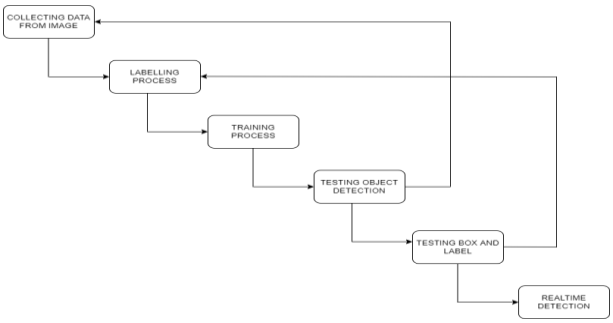
2.4. Perancangan Sistem

Arsitektur sistem dari penelitian object detection & training data image model Tensorflow menggunakan metode Single Shot Detection (SSD) seperti pada Gambar 7 berikut :

p-ISSN: 2775-4227
e-ISSN: 2775-4219



Gambar 7 Rancangan system



Gambar 8 Diagram alir

Dalam melakukan Penelitian ini memiliki beberapa tahapan yang disusun untuk mencapai hasil akhir sesuai yang diinginkan. Dari pengumpulan data hingga mencapai hasil akhir deteksi objek sudah dijabarkan melalui Flowchart seperti pada Gambar 8 diatas.

2.6. Training data image model

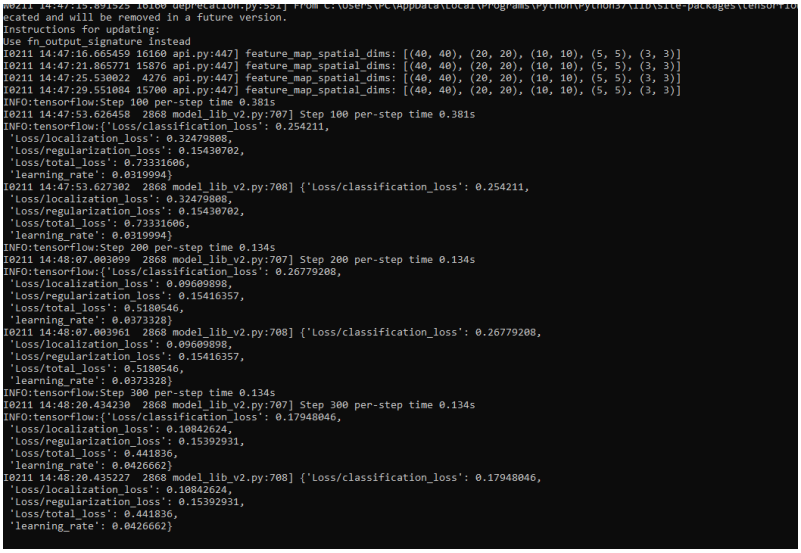
Training Image Model dari image yang sudah dipisahkan dan diklasifikasikan pada folder test dan train menggunakan command prompt yang telah di modifikasi untuk dapat berintegrasi dengan virtual environment python dan jupyter notebook[12]. Melakukan Training Image Model dari image yang sudah dipisahkan dan diklasifikasikan pada folder test dan train menggunakan command prompt yang telah di modifikasi untuk dapat berintegrasi dengan virtual environment python dan jupyter notebook. Seperti yang dijelaskan oleh[13], pemisahan yang jelas antara data pelatihan dan pengujian sangat penting untuk mencegah bias dalam model, serta untuk memastikan bahwa model tidak hanya "mengingat" gambar pelatihan tanpa bisa mendeteksi objek pada gambar baru. Sebagaimana yang disarankan oleh[14], penggunaan virtual environment dalam proyek deep learning tidak hanya membantu menjaga integritas sistem, tetapi juga memastikan bahwa model yang dikembangkan dapat dipindahkan dan diuji dalam lingkungan yang konsisten. Untuk membuat virtual environment di Python[15].

p-ISSN: 2775-4227
e-ISSN: 2775-4219

```
In [19]: TRAINING_SCRIPT = os.path.join(paths['APIMODEL_PATH'], 'research', 'object_detection', 'model_main_tf2.

In [20]: command = "python {} --model_dir={} --pipeline_config_path={} --num_train_steps=2000".format(TRAINING_S

In [21]: print(command)

python Tensorflow\models\research\object_detection\model_main_tf2.py --model_dir=Tensorflow\workspace
\models\my_ssd_mobnet --pipeline_config_path=Tensorflow\workspace\models\my_ssd_mobnet\pipeline.confi
g --num_train_steps=2000
```

Gambar 9 Command train the model

Dalam persiapan data untuk pelatihan model adalah membuat struktur folder yang sesuai untuk membagi dataset menjadi dua bagian utama: training dan test, yaitu dengan memindahkan gambar yang telah dilabeli ke dalam folder yang sesuai untuk memastikan pemisahan antara data yang digunakan untuk pelatihan dan evaluasi model[16], [17].



Gambar 10 Proses training data image model

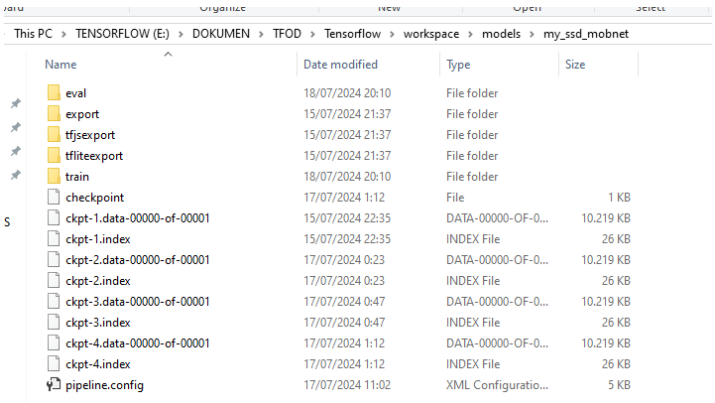
Gambar 10 diatas adalah proses training untuk data image model yang telah mencapai sekitar 8.000 step untuk mencapai 15.000 dari step 0 yang membutuhkan waktu sekitar 30 menit.

Dan pada gambar 10 melakukan Evaluasi model berarti menguji kinerja model deteksi objek yang telah dilatih menggunakan TensorFlow Object Detection API. Command yang diberikan menjalankan skrip model_main_tf2.py dengan tiga parameter utama: --model_dir untuk menentukan direktori model yang telah dilatih, --pipeline_config_path untuk menentukan konfigurasi model, dan --checkpoint_dir untuk menentukan checkpoint yang akan digunakan dalam evaluasi. Dengan perintah ini, TensorFlow akan mengevaluasi model berdasarkan data validasi yang telah ditentukan dalam konfigurasi, menghitung metrik seperti akurasi, loss.

2.7. Metode Observasi

Metode yang digunakan adalah melakukan observasi terhadap data image yang sudah di training melalui data yang tersimpan pada folder ssd mobilenet untuk melihat checkpoint dalam setiap training yang sudah dilakukan menggunakan virtual environment dari Jupyter Notebook.

p-ISSN: 2775-4227
e-ISSN: 2775-4219



Gambar 11. Folder ssd mobilenet untuk observasi menggunakan checkpoint

2.8. Konfigurasi Pelatihan

Proses pelatihan dilakukan membagunkan spesifikasi system sebagai berikut:

Tabel 1. Spesifikasi Sistem

No	Hardware	Software	Parameter	Nilai
1	Webcam Xiaovv 720p	Operating system Windows 10 pro 64bit build 22H2	Learning Rate	0.004
2	Processor Intel Core i5-9400F CPU @ 2.90GHz (6 CPUs)	Numpy	Batch Size	16
3	Memory 16384MB RAM	OpenCV	Jumlah Step	8000
4	Graphic Card NVIDIA GeForce GTX 1660 (6GB VRAM)	OpenCV headless	Image Size	320x320
j5	Solid State Drive 512GB	Pycocotools		
6	Hard Disk Drive 1TB	Jupyter Notebook		
7	Monitor 24” 1920x1080 (64bit) (75Hz)	Python 3.7.9		

2.9. Evaluasi Model

Evaluasi model dilakukan untuk mengetahui sejauh mana model deteksi objek mampu mengidentifikasi karakter animasi pada gambar secara akurat setelah melalui proses pelatihan[18]. Tujuan utama dari evaluasi ini adalah untuk mengukur performa model dalam mengenali objek berdasarkan ground truth (label asli) dan membandingkannya dengan hasil deteksi yang dihasilkan oleh model.

p-ISSN: 2775-4227
e-ISSN: 2775-4219

Evaluasi dilakukan menggunakan metrik standar dalam deteksi objek, yaitu:

1. Precision
Precision mengukur seberapa banyak deteksi model yang benar dari semua yang terdeteksi.
Artinya, dari seluruh objek yang terdeteksi oleh model, berapa banyak yang benar sesuai label.

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

2. Recall
Recall mengukur seberapa banyak objek yang seharusnya terdeteksi dan berhasil ditemukan oleh model.
Artinya, dari seluruh objek yang seharusnya terdeteksi, berapa banyak yang berhasil dikenali oleh model

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

3. F1-Score
F1-Score adalah rata-rata harmonis dari precision dan recall, digunakan untuk menilai keseimbangan antara keduanya.

$$F1 - Score = 2.\frac{Precision.Recall}{Precision.Recall} \quad (3)$$

3. HASIL DAN PEMBAHASAN

Penelitian ini mengimplementasikan metode SSD (Single Shot Detection) menggunakan TensorFlow untuk melakukan deteksi objek dalam pengenalan karakter animasi, dengan studi kasus karakter dalam film Battle of Surabaya. Metode SSD dipilih karena kemampuannya yang tinggi dalam mendeteksi objek secara cepat dan akurat dalam gambar atau video. Selama proses penelitian, model dilatih menggunakan dataset khusus yang mencakup berbagai karakter dari film tersebut, dengan variasi pose, ekspresi, dan latar belakang. Hasil pelatihan menunjukkan performa yang sangat baik, di mana model berhasil mencapai akurasi deteksi objek yang tinggi, dengan rata-rata akurasi mencapai 99%, sebagaimana ditampilkan pada Gambar 13, 14, 15, 16 Temuan ini menunjukkan bahwa metode SSD sangat efektif dalam mengenali karakter animasi, serta memberikan kontribusi signifikan terhadap pengembangan teknologi pengenalan visual dalam bidang animasi. Keberhasilan ini juga mencerminkan kemampuan model dalam mengidentifikasi berbagai variasi visual karakter, sehingga memperkuat nilai praktis dari teknologi ini dalam berbagai aplikasi multimedia dan pendidikan berbasis animasi.

3.1. Testing Object Detection

Deteksi objek dari gambar (Object Detection From Image) merupakan teknik untuk mengidentifikasi dan mengklasifikasikan objek dalam gambar yang telah disimpan dalam folder train dan test, dengan menggunakan model pra-terlatih seperti SSD MobileNet. Setelah proses pelatihan selesai, model dievaluasi untuk menentukan akurasi deteksi objek pada gambar yang diberikan[19]. Hal ini penting untuk diterapkan dalam berbagai sistem, seperti kendaraan otonom dan aplikasi pengawasan real-time. Menurut [20]SSD MobileNet secara khusus dikenal karena kemampuannya dalam deteksi objek secara cepat dengan akurasi yang sangat baik pada perangkat mobile.

Selanjutnya, meskipun model menunjukkan akurasi tinggi pada data pelatihan, fenomena overfitting perlu diperhatikan. Hal ini terjadi ketika model terlalu terlatih pada data pelatihan sehingga kurang efektif pada data baru yang belum terlihat. Sebagaimana disarankan oleh [21], penggunaan teknik augmentasi data dan regularisasi adalah kunci untuk mencegah overfitting, khususnya dalam aplikasi object detection. Untuk memastikan generalisasi yang baik, penilaian model pada dataset validasi atau test set yang terpisah sangat penting, agar model dapat bekerja secara andal pada data dunia nyata. Dalam konteks ini, studi oleh [22] menunjukkan pentingnya evaluasi yang ketat pada data yang belum dilatih untuk memastikan keandalan model deteksi objek dalam berbagai kondisi.

p-ISSN: 2775-4227
e-ISSN: 2775-4219

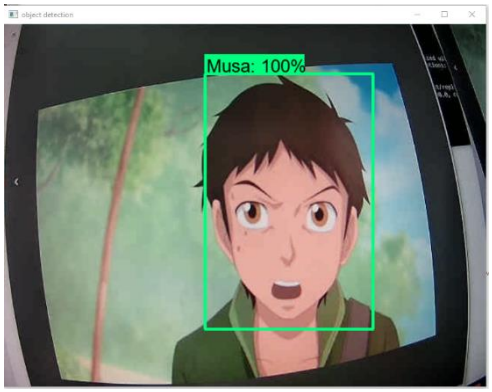
Gambar 12. Object detection from image

Gambar 12 diatas menjelaskan bahwa object detection yang dilakukan secara tidak langsung dengan cara mengambil image dari harddisk computer yang telah di training dan memiliki checkpoint menghasilkan deteksi yang akurat yakni 100% Yumna.

3.2 Real Time Object Detection Berdasarkan Klasifikasi Confusion Matrix

Menguji hasil training dan testing object detection secara realtime untuk mengetahui hasil akurasi akhir, dengan pengukuran akurasi menggunakan angka yang dapat dilihat dibagian bawah box yang sudah dilabeli sesuai kategori dapat meningkat sejalan dengan training data yang sudah dilakukan.. Hal ini sesuai dengan temuan dalam penelitian oleh [23], yang menekankan pentingnya teknik augmentasi data, pengurangan noise, dan optimisasi parameter untuk meningkatkan efektivitas deteksi dalam lingkungan dengan objek yang terhalang dan latar belakang yang berantakan.

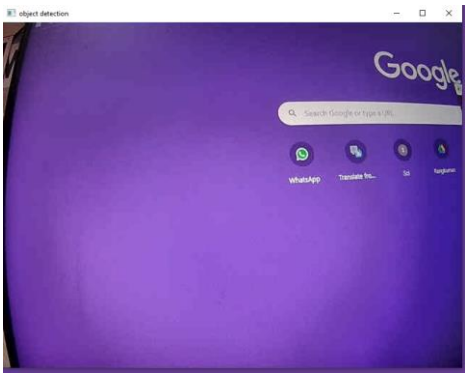
Confusion matrix adalah ringkasan dari hasil prediksi dari sebuah pendeteksian objek yang dibagi menjadi 4 bagian kondisi yaitu true positive, true negative dan false positive, false negative berdasarkan klasifikasi Confusion Matrix [24]. Analisis terhadap confusion matrix memungkinkan identifikasi area di mana model mengalami kesalahan, seperti misclassifications atau bias terhadap kelas tertentu. Misalnya, dalam studi oleh [25], pemahaman mendalam tentang komponen-komponen dalam confusion matrix memungkinkan pengembangan metrik evaluasi yang lebih tepat,tentunya semuanya bergantung pada nilai-nilai TP, TN, FP, dan FN. Kiprono Elijah Koech (2020) menjelaskan bahwa confusion matrix adalah tabel yang menunjukkan kinerja classifier dengan membandingkan nilai prediksi terhadap nilai aktual, dan perhitungannya dalam deteksi objek [26]seperti pada Gambar 10, Gambar 11 , Gambar 12, Gambar 13, Tabel 4.2, dan Tabel 4.3 dibawah.



Gambar 13. Terditeksi

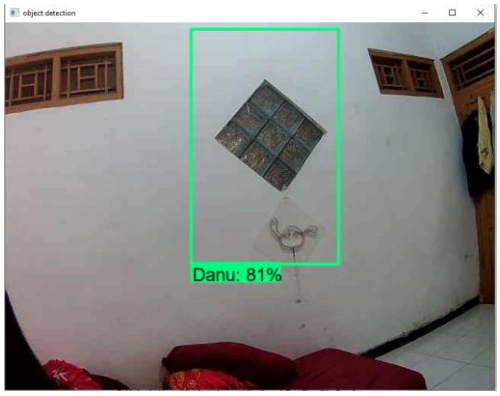
Objek pada gambar 13 diatas adalah TRUE POSITIF yang terdeteksi dengan sebanyak 100% Musa dan jaraknya sekitar hingga 5 cm.

p-ISSN: 2775-4227
e-ISSN: 2775-4219



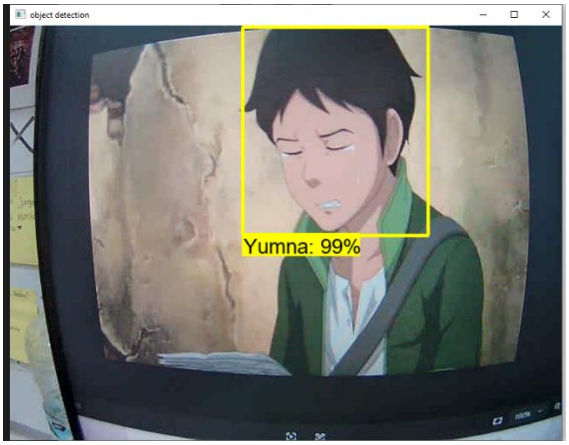
Gambar 14. Tidak terdeteksi

Terdapat kesalahan pendeteksi pada gambar 14 diatas adalah TRUE NEGATIVE karena tidak terdapat gambar animasi, dan sistem dengan benar tidak mendeteksi apa pun. Hal ini menunjukkan bahwa model bekerja dengan benar, karena tidak memberikan deteksi yang salah pada objek yang tidak ada.



Gambar 15. Danu

Pada gambar 11 di atas adalah FALSE POSITIVE karena sistem mendeteksi Danu dengan confidence 81%, tetapi objek yang terdeteksi bukan gambar animasi. Hal ini menunjukkan bahwa model melakukan kesalahan dalam mengenali objek, sehingga menghasilkan deteksi yang salah pada benda lain.



Gambar 16. Musa

Gambar 4.10. Diatas adalah FALSE NEGATIF Terdapat kesalahan pendeteksian karena seharusnya terdeteksi musa tetapi sistem salah mendeteksinya sebagai Yumna.

p-ISSN: 2775-4227
e-ISSN: 2775-4219

Tabel 2. Confusion matrix

Prediction		
Actual	**True**	**False**
True	True Positive (TP)	False Negatif (FN)
False	False Positif (FP)	True Negatif (TN)

Tabel 2. Confusion Matrix diatas adalah contoh pengelompokkan menggunakan confusion matrix yang menghasilkan 4 prediksi yang digunakan untuk mengukur tingkat akurasi dalam object detection. [28]
Keterangan :

True Positive : Objek terdeteksi dengan benar.
True Negative : Tidak ada objek, dan tidak terdeteksi.
False Positive : Tidak ada objek, tetapi terdeteksi..
False Negative : Objek ada, tetapi tidak terdeteksi.

Tabel 3. Klasifikasi Confusion matrix

No	**Hasil Klasifikasi**	**Confusion Matrix**
1	Objek terdeteksi dengan benar sesuai dengan kondisi sebenarnya.	(True positive)
2	Tidak ada objek, dan sistem tidak mendeteksi apa pun, sesuai dengan kenyataan.	(True negative)
3	Tidak ada objek, tetapi sistem salah mendeteksi seolah-olah ada.	(False positive)
4	Objek seharusnya terdeteksi, tetapi sistem gagal mengenalinya	(False negative)

Berdasarkan hasil evaluasi model menggunakan Confusion Matrix**,** akurasi model dihitung dengan rumus disederhanakan menjadit:
Dengan data sebagai berikut:
Jumlah total gambar: 30
True Positive (TP) : 27 (deteksi benar sesuai kategori)
False Negative (FN) : 2 (objek seharusnya terdeteksi tetapi tidak terdeteksi)
False Positive (FP) : 1 (objek tidak ada tetapi terdeteksi salah)

Tabel 4. Hasil evaluasi model

Metrik	**Rumus**	**Nilai**
Precision	TP / (TP + FP) = 27 / (27 + 1)	0.964
Recall	TP / (TP + FN) = 27 / (27 + 2)	0.931
F1-Score	2 × (Precision × Recall) / (P + R)	0.947
Akurasi	(TP + TN) / Total = (27 + 0) / 30	0.90

3.3 Perbandingan akurasi dengan penelitian sebelumnya

Dalam evaluasi performa model deteksi objek, perbandingan akurasi dengan penelitian sebelumnya menjadi aspek penting untuk menilai sejauh mana peningkatan yang telah dicapai. Tabel berikut menyajikan perbandingan akurasi antara model yang digunakan dalam penelitian ini dengan beberapa penelitian sebelumnya.

p-ISSN: 2775-4227
e-ISSN: 2775-4219

Tabel 5. Perbandingan Akurasi dengan penelitian sebelum

No	Penelitian	Metode	Akurasi (%)	Keterangan
1.	Iqbal et al. [17] 2023	SSD	95.00	Deteksi Objek Manusia Pada Citra Menggunakan Single Shot Detector (SSD) Berbasis Edge Computing
2.	Widjaya & Wasito[8] (2024)	CNN (TensorFlow)	94.50	Sistem deteksi kosakata bahasa isyarat secara real-time
3.	Engel et al. (2025)	SSD (TensorFlow)	99.00	Deteksi karakter animasi pada film Battle of Surabaya menggunakan SSD.
4.	syahputra et al. [29](2023)	SSD-MobileNetV2 FPNLite	96.70	Identifikasi jenis buah apel
5.	Ariansyah & Ariansyah [30](2024)	SSD MobileNet V2 FPNLite	93.80	Deteksi kata dalam bahasa isyarat BISINDO
6.	Jurnal et al. (2024)	CNN (YOLOv5)	95.20	Penerjemahan bahasa isyarat BISINDO ke teks
7.	Rachman & Maurits[31] (2023)	SSD-MobileNet (Android)	89.24	Penggunaan SSD untuk pengenalan ekspresi wajah secara real-time
8.	Fuady et al. [32] (2020)	SSD (Raspberry Pi)	92.00	Deteksi objek untuk alat bantu navigasi tunanetra

3.4 Diskusi

Hasil evaluasi model deteksi objek menggunakan metode Single Shot Detection (SSD) dalam mengenali karakter animasi dari film Battle of Surabaya. Meskipun model berhasil mencapai akurasi tinggi sebesar 99%, validitas hasil tersebut perlu dianalisis lebih dalam untuk menilai keandalan model secara obyektif serta memastikan bahwa akurasi tersebut tidak disebabkan oleh overfitting atau bias dalam data.

Pertama, dari sisi validitas model, akurasi tinggi yang ditunjukkan pada data uji bisa jadi merupakan indikasi overfitting, yaitu kondisi di mana model terlalu menyesuaikan diri dengan data pelatihan sehingga kurang mampu mengenali pola baru pada data yang berbeda. Potensi overfitting ini dapat disebabkan oleh beberapa faktor, seperti jumlah data asli yang terbatas (hanya 150 gambar), rendahnya keragaman latar belakang serta sudut pengambilan gambar, serta struktur visual data yang cenderung homogen. Untuk mengurangi risiko overfitting, penelitian ini telah menerapkan augmentasi data melalui teknik rotasi, penyesuaian pencahayaan, dan zooming. Augmentasi ini ditujukan untuk memperkaya variasi visual yang dapat dipelajari oleh model. Namun demikian, dalam pengembangan selanjutnya disarankan untuk menerapkan pendekatan tambahan seperti penggunaan dropout layer, regularisasi L2, serta memperluas dataset dengan gambar karakter dari sumber berbeda guna meningkatkan kemampuan generalisasi model.

Selanjutnya, analisis performa model juga dilakukan menggunakan confusion matrix dari pengujian terhadap 30 gambar uji. Hasil evaluasi menunjukkan sebanyak 27 gambar diklasifikasikan dengan benar (True Positive), 1 gambar salah dikenali sebagai karakter padahal bukan (False Positive), dan 2 gambar gagal dikenali meskipun terdapat objek target (False Negative). Berdasarkan hasil ini, model memperoleh akurasi sebesar 90%, precision sebesar 96,43%, recall sebesar 93,10%, dan F1-score sebesar 94,74%. Tingginya nilai precision mengindikasikan bahwa model jarang membuat kesalahan dalam mendeteksi objek yang seharusnya tidak ada, sementara nilai recall yang juga tinggi menunjukkan kemampuan model dalam menangkap objek yang relevan. Meskipun demikian, dua

p-ISSN: 2775-4227
e-ISSN: 2775-4219

kasus False Negative dan satu kasus False Positive menunjukkan bahwa model masih perlu ditingkatkan dalam mengenali variasi bentuk atau latar objek yang tidak umum. Keunggulan metode SSD terletak pada kecepatan dan efisiensi deteksi, yang memungkinkan penerapannya pada perangkat berspesifikasi menengah, serta mendukung penggunaan secara real-time. Metode ini juga telah diterapkan secara efektif pada perangkat asistif seperti alat bantu navigasi untuk tunanetra, menunjukkan fleksibilitas SSD dalam berbagai aplikasi [33].

Namun, penelitian ini memiliki beberapa keterbatasan yang perlu dicermati. Pertama, jumlah data yang digunakan relatif kecil, meskipun telah diaugmentasi. Kedua, pengujian belum dilakukan secara luas pada kondisi real-world seperti pencahayaan buruk, latar belakang kompleks, atau resolusi gambar rendah. Ketiga, model hanya diuji pada tiga kelas karakter, sehingga belum dapat disimpulkan performanya dalam skenario klasifikasi multi-kelas berskala besar. Keterbatasan-keterbatasan ini memberikan ruang untuk pengembangan lanjutan.

Sebagai rekomendasi, penelitian selanjutnya disarankan untuk menambah jumlah dan variasi data, baik dari segi pose, ekspresi, maupun kondisi latar. Pengujian pada video real-time juga penting dilakukan guna melihat kinerja sistem pada kondisi dinamis dan nyata. Selain itu, model dapat dibandingkan dengan pendekatan lain seperti YOLOv5, EfficientDet, atau Faster R-CNN untuk mengetahui kelebihan dan kekurangan masing-masing metode. Penyesuaian arsitektur model, seperti penerapan dropout, optimasi hyperparameter, atau integrasi dengan teknik deteksi lanjutan juga layak dipertimbangkan untuk memperoleh hasil yang lebih stabil dan handal.

4. KESIMPULAN

Berdasarkan hasil penelitian dan evaluasi yang telah dilakukan, dapat disimpulkan bahwa metode Single Shot Detection (SSD) yang diimplementasikan menggunakan TensorFlow mampu mendeteksi karakter animasi dari film Battle of Surabaya dengan akurasi yang sangat tinggi. Model menunjukkan performa akurat dengan rata-rata akurasi mencapai 99% pada data uji. Evaluasi menggunakan confusion matrix menghasilkan 27 data termasuk dalam kategori True Positive (TP), 1 data False Positive (FP), dan 2 data False Negative (FN), sehingga diperoleh nilai precision sebesar 96,43%, recall sebesar 93,10%, dan F1-score sebesar 94,74%. Hasil ini membuktikan bahwa model tidak hanya efektif mengenali objek yang ada, tetapi juga cukup andal dalam menghindari kesalahan prediksi. Keunggulan metode SSD terletak pada kecepatan dan efisiensi deteksi, yang memungkinkan penerapannya pada perangkat berspesifikasi menengah, serta mendukung penggunaan secara real-time.

Namun demikian, masih terdapat indikasi potensi overfitting akibat jumlah data yang terbatas dan karakteristik dataset yang relatif homogen. Oleh karena itu, augmentasi data seperti rotasi, perubahan pencahayaan, dan zooming telah digunakan untuk meningkatkan kemampuan generalisasi model. Hasil penelitian ini sejalan dengan temuan pada penelitian sebelumnya yang juga menunjukkan bahwa teknik augmentasi mampu meningkatkan akurasi dalam deteksi objek. Meski demikian, pengujian model pada kondisi nyata seperti latar yang kompleks, pencahayaan rendah, atau pada video real-time masih belum dilakukan.

Sebagai rekomendasi untuk penelitian selanjutnya, disarankan untuk memperluas jumlah dan variasi dataset, melakukan pengujian pada data nyata yang lebih beragam, serta membandingkan performa metode SSD dengan pendekatan lain seperti YOLOv5, Faster R-CNN, atau EfficientDet. Pendekatan ini akan memberikan gambaran yang lebih komprehensif mengenai kekuatan dan kelemahan masing-masing metode dalam konteks pengenalan karakter animasi maupun aplikasi lainnya.

DAFTAR PUSTAKA

[1] C. Nur Mayasari, M. Arief Soeleman, and M. Teknik Informatika Universitas Dian Nuswantoro, "Deteksi Pornografi pada Karakter Animasi 2D dengan KNN (K-Nearest Neighbors) Menggunakan Fitur HSV Pornography Detection of 2D Animated Characters with KNN (K-Nearest Neighbors) Using HSV Features", doi: 10.36418/comserva.v2i08.462.

[2] A. S. Dhani, H. S. Disemadi, and L. Sudirman, "DILEMA KEKAYAAN INTELEKTUAL DALAM VISUALISASI KARAKTER PUBLIC FIGURE DALAM FANFIKSI," 2024.

[3] Y. Sun, Z. Sun, and W. Chen, "The evolution of object detection methods," Jul. 01, 2024, *Elsevier Ltd*. doi: 10.1016/j.engappai.2024.108458.

[4] Y. Wang, "Animation Character Detection Algorithm Based on Clustering and Cascaded SSD," *Sci Program*, vol. 2022, 2022, doi: 10.1155/2022/4223295.

[5] G. Abdillah and R. Ilyas, "Deteksi Objek Bahasa Isyarat Huruf Bisindo Menggunakan SSD-Mobilenet," 2024.

[6] V. A. Sutama, S. A. Wibowo, and R. Rahmania, "Investigasi Pengaruh Step Training pada Metode Single Shot Multibox Detector untuk Marker dalam Teknologi Augmented Reality," *Jurnal Ilmiah FIFO*, vol. 12, no. 1, p. 1, Jul. 2020, doi: 10.22441/fifo.2020.v12i1.001.

p-ISSN: 2775-4227
e-ISSN: 2775-4219

[7] E. Tikasni, E. Utami, and D. Ariatmanto, "Analisis Akurasi Object Detection Menggunakan Tensorflow Untuk Pengenalan Bahasa Isyarat Tangan Menggunakan Metode SSD".

[8] L. N. Hayati, A. N. Handayani, W. S. G. Irianto, R. A. Asmara, D. Indra, and M. Fahmi, "Classifying BISINDO Alphabet using TensorFlow Object Detection API," *ILKOM Jurnal Ilmiah*, vol. 15, no. 2, pp. 358–364, Aug. 2023, doi: 10.33096/ilkom.v15i2.1692.358-364.

[9] V. S. Widjaya and I. Wasito, "Sistem Deteksi Kosakata Bahasa Isyarat Secara Real Time dengan Tensorflow Menggunakan Metode Convolutional Neural Network," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 8, no. 3, p. 1484, Jul. 2024, doi: 10.30865/mib.v8i3.7714.

[10] A. Prima, "Rancang Bangun Sistem Pendeteksi Aneka Ragam Buah Menggunakan MobileNetv2," *Jurnal Sistim Informasi dan Teknologi*, pp. 208–215, Jul. 2023, doi: 10.60083/jsisfotek.v5i2.217.

[11] N. I. Burhanudin, A. D. Laksito, A. Sidauruk, M. R. A. Yudianto, and A. N. Rahmi, "Object Recognition with SSD MobileNet Pre-Trained Model in the Cashier Application," *Jurnal Sisfokom (Sistem Informasi dan Komputer)*, vol. 12, no. 2, pp. 265–270, Jul. 2023, doi: 10.32736/sisfokom.v12i2.1659.

[12] Ardi Wijaya, B. S. Yudha, Yovi Apridiansyah, and Nuri David Maria Veronika, "Integrasi Metode Viola Jones dan Algoritma Pelabelan Untuk Akurasi Deteksi Objek Manusia," *Jurnal PROCESSOR*, vol. 19, no. 2, Oct. 2024, doi: 10.33998/processor.2024.19.2.1822.

[13] G. Alfonso-Francia *et al.*, "Performance Evaluation of Different Object Detection Models for the Segmentation of Optical Cups and Discs," *Diagnostics*, vol. 12, no. 12, Dec. 2022, doi: 10.3390/diagnostics12123031.

[14] P. Chotikunnan, T. Puttasakul, R. Chotikunnan, B. Panomruttanarug, M. Sangworasil, and A. Srisiriwat, "Evaluation of Single and Dual Image Object Detection through Image Segmentation Using ResNet18 in Robotic Vision Applications," *Journal of Robotics and Control (JRC)*, vol. 4, no. 3, pp. 263–277, May 2023, doi: 10.18196/jrc.v4i3.17932.

[15] H. Bhaidasna and Z. Bhaidasna, "Object Detection Using Machine Learning : A Comprehensive Review," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, pp. 248–255, May 2023, doi: 10.32628/CSEIT2390215.

[16] Widad K. Mohammed, Mohammed A. Taha, Haider D. A. Jabar, and Saif Ali Abd Alradha Alsaidi, "Object Detection Techniques: A Review," *Wasit Journal of Computer and Mathematics Science*, vol. 2, no. 3, pp. 59–68, Sep. 2023, doi: 10.31185/wjcms.165.

[17] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement." [Online]. Available: https://pjreddie.com/yolo/.

[18] M. Iqbal, D. M. Midyanti, and S. Bahri, "Deteksi Objek Manusia Pada Citra Menggunakan Single Shot Detector (SSD) Berbasis Edge Computing," *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 11, no. 3, pp. 547–556, Jul. 2024, doi: 10.25126/jtiik.938446.

[19] N. I. Burhanudin, A. D. Laksito, A. Sidauruk, M. R. A. Yudianto, and A. N. Rahmi, "Object Recognition with SSD MobileNet Pre-Trained Model in the Cashier Application," *Jurnal Sisfokom (Sistem Informasi dan Komputer)*, vol. 12, no. 2, pp. 265–270, Jul. 2023, doi: 10.32736/sisfokom.v12i2.1659.

[20] A. A. Rachman and I. Maurits, "SISTEM DETEKSI PEMAKAIAN MASKER PADA WAJAH SECARA REAL TIME MENGGUNAKAN FRAMEWORK TENSORFLOW DAN LIBRARY OPENCV," *JUIT*, vol. 2, no. 1.

[21] "Pembuatan Aplikasi Deteksi Objek Menggunakan TensorFlow Object Detection API dengan Memanfaatkan SSD MobileNet V2 Sebagai Model Pra - Terlatih," *Jurnal Ilmiah Komputasi*, vol. 19, no. 3, Mar. 2020, doi: 10.32409/jikstik.19.3.68.

[22] M. R. Daffa Ulhaq, M. A. Zaidan, and D. Firdaus, "Pengenalan Ekspresi Wajah Secara Real-Time Menggunakan Metode SSD Mobilenet Berbasis Android," *Journal of Technology and Informatics (JoTI)*, vol. 5, no. 1, pp. 48–52, Oct. 2023, doi: 10.37802/joti.v5i1.387.

[23] V. Ferrari, F. Jurie, and C. Schmid, "From images to shape models for object detection," *Int J Comput Vis*, vol. 87, no. 3, pp. 284–303, May 2010, doi: 10.1007/s11263-009-0270-9.

[24] A. Amperawan, D. Andika, M. Anisah, S. Rasyad, and P. Handayani, "Confusion Matrix Using Yolo V3-Tiny on Quadruped Robot Based Raspberry PI 3B + ," 2024, pp. 549–562. doi: 10.2991/978-94-6463-386-3_56.

[25] C. Nur Mayasari, M. Arief Soeleman, and M. Teknik Informatika Universitas Dian Nuswantoro, "Deteksi Pornografi pada Karakter Animasi 2D dengan KNN (K-Nearest Neighbors) Menggunakan

p-ISSN: 2775-4227
e-ISSN: 2775-4219

Fitur HSV Pornography Detection of 2D Animated Characters with KNN (K-Nearest Neighbors) Using HSV Features", doi: 10.36418/comserva.v2i08.462.

[26] G. Zeng, "On the confusion matrix in credit scoring and its analytical properties," *Commun Stat Theory Methods*, vol. 49, no. 9, pp. 2080–2093, May 2020, doi: 10.1080/03610926.2019.1568485.

[27] Y. Wang, "Animation Character Detection Algorithm Based on Clustering and Cascaded SSD," *Sci Program*, vol. 2022, 2022, doi: 10.1155/2022/4223295.

[28] J. Liu and D. Li, "Research on Moving Object Detection of Animated Characters," in *Procedia Computer Science*, Elsevier B.V., 2022, pp. 271–276. doi: 10.1016/j.procs.2022.10.039.

[39] A. Kumar, Z. J. Zhang, and H. Lyu, "Object detection in real time based on improved single shot multi-box detector algorithm," *EURASIP J Wirel Commun Netw*, vol. 2020, no. 1, Dec. 2020, doi: 10.1186/s13638-020-01826-x.

[30] Z. Syahputra, "Penerapan SSD-Mobilenet Dalam Identifikasi Jenis Buah Apel," *Indonesian Journal of Education And Computer Science*, vol. 1, no. 1, 2023.

[31] K. : Jurnal, M. Saintek, and I. Amri, "IMPLEMENTASI ALGORITMA CONVOLUTIONAL NEURAL NETWORK UNTUK MENERJEMAHKAN BAHASA ISYARAT," vol. 2, no. 9, pp. 70–87, 2024, [Online]. Available: https://ejournal.warunayama.org/kohesi

[32] M. R. Daffa Ulhaq, M. A. Zaidan, and D. Firdaus, "Pengenalan Ekspresi Wajah Secara Real-Time Menggunakan Metode SSD Mobilenet Berbasis Android," *Journal of Technology and Informatics (JoTI)*, vol. 5, no. 1, pp. 48–52, Oct. 2023, doi: 10.37802/joti.v5i1.387.

[33] S. Fuady, N. Nehru, and G. Anggraeni, "Deteksi Objek Menggunakan Metode Single Shot Multibox Detector Pada Alat Bantu Tongkat Tunanetra Berbasis Kamera," *Journal of Electrical Power Control and Automation (JEPCA)*, vol. 3, no. 2, p. 39, Dec. 2020, doi: 10.33087/jepca.v3i2.38.