Available *online* at : http://bit.ly/InfoTekJar

InfoTekJar : Jurnal Nasional Informatika dan Teknologi Jaringan

ISSN (Print) 2540-7597 | ISSN (Online) 2540-7600

Deep Learning

Implementasi Yolo V8 Dengan Pemanfaatan Apple M2 Untuk Deteksi Perilaku Merokok

Syarifah Syifa Hamidah [1], Ir. Nixon Erzed [2]

Universitas Esa Unggul, Bekasi, Jawa Barat 17214, Indonesia

ARTICLE INFORMATION

Received: February 00, 00
Revised: March 00, 00
Available online: April 00, 00

KEYWORDS

YOLO, Apple Neural Engine, merokok, confusion matrix.

CORRESPONDENCE

Phone: +62 823 - 8388 - 6020

E-mail: syiffassegaf28@gmail.com

A B S T R A C T

Indonesia is one of the countries facing serious problems related to the high number of smokers. Active smokers have a high risk of contracting various serious diseases, such as heart disease, cancer, respiratory diseases, and others. Additionally, exposure to tobacco smoke also has adverse effects on passive smokers, who are often individuals around them who do not smoke but are affected by it. Conventional methods for detecting smokers are often inefficient and require significant manual intervention, thus necessitating a technological solution for automatic and real-time detection to support the enforcement of anti-smoking regulations. Therefore, this research aims to detect smoking behavior using the You Only Look Once (YOLO) version 8 method on Apple M2. YOLO V8 was chosen for its capability in fast and accurate object detection, while the Apple M2 supports real-time processing. The training results showed an accuracy rate of 91.6%, precision of 96.4%, recall of 90.4%, and an F1-Score of 93.2%. During the inference stage, the Apple Neural Engine (ANE) was able to process 21-25 frames per second (fps), demonstrating good capability for real-time object detection. The combination of YOLO V8 and Apple M2 proved effective for detecting smokers in public areas, offering an efficient and effective innovative solution, supporting the creation of a smoke-free environment in Indonesia, and showing great potential for the application of edge computing in similar applications in the future.

INTRODUCTION

Indonesia menghadapi masalah serius dengan tingginya jumlah perokok. Lebih dari 33% populasi dewasa di Indonesia adalah perokok aktif, dan angka ini terus meningkat di kalangan remaja dan anak-anak, yang berpotensi mengakibatkan masalah kesehatan jangka panjang bagi generasi muda. Merokok telah terbukti menyebabkan berbagai penyakit serius seperti kanker paru-paru, penyakit jantung, dan penyakit pernapasan, yang secara signifikan membebani sistem kesehatan dan ekonomi negara. Meskipun berbagai upaya telah dilakukan oleh pemerintah melalui regulasi, kampanye kesehatan, dan kebijakan pajak rokok, penegakan hukum yang lemah dan rendahnya kesadaran masyarakat tetap menjadi tantangan utama.

Pemerintah sudah mengambil tindakan mengenai permasalahan di atas yaitu Aturan tentang rokok sudah tertuang dalam Peraturan Pemerintah Nomor 109 Tahun 2012 tentang Pengamanan Bahan yang Mengandung Zat Adiktif Berupa Produk Tembakau bagi Kesehatan [1]. PP ini menyebut kawasan tanpa rokok meliputi fasilitas pelayanan kesehatan, tempat proses belajar-mengajar, tempat anak bermain, tempat ibadah, angkutan umum, tempat kerja, dan tempat umum atau tempat lain yang ditetapkan.

Salah satu upaya yang dapat dilakukan adalah dengan menciptakan teknologi yang dapat mengimplementasikan regulasi pembatasan merokok. Teknologi *deep learning* telah menjadi pusat perhatian karena kemampuannya yang luar biasa dalam menangani berbagai masalah kompleks, termasuk deteksi objek dan pengenalan pola. Salah satu penerapan mutakhir dari *deep learning* adalah algoritma You Only Look Once (YOLO) dengan versi terbarunya versi 8. YOLO dengan kemampuan deteksi objek dalam sekali melihat yang sangat cocok terhadap deteksi perilaku merokok secara *real time*.

Berdasarkan penelitian terdahulu oleh Redmon dan Farhadi (2016) memperkenalkan YOLO sebagai metode deteksi objek *real-time* yang efisien, yang mampu mendeteksi berbagai objek dalam gambar dengan kecepatan tinggi dan akurasi yang memadai [2]. Penelitian lain oleh Shrey et al. (2021) membandingkan algoritma YOLO, Fast-RCNN, SSD dalam mendeteksi objek, hasil penelitian menunjukkan YOLO v8 memiliki keunggulan dalam hal kecepatan dan akurasi [3].

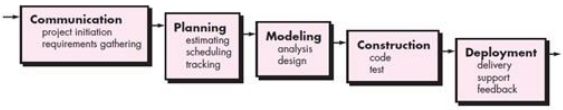
https://doi.org/10.30743/xxxxx

Dengan memanfaatkan *neural engine* pada prosesor Apple M2 menawarkan solusi inovatif. Teknologi *edge computing* memungkinkan pemrosesan data dilakukan dekat dengan sumbernya, mengurangi latensi dan penggunaan *bandwidth*, yang sangat penting untuk aplikasi *real-time*. Sehingga penelitian ini dilakukan dengan menggunakan algoritma YOLO v8 pada perangkat Apple M2. Penelitian ini bertujuan untuk menguji kemampuan apple M2 dalam melakukan *training* dan *inferecing* model menggunakan algoritma YOLO v8 dalam mendeteksi perilaku merokok. Dengan manfaat menambah literatur ilmiah dibidang kecerdasan buatan, selain itu membantu mengimplementasikan regulasi yang ada mengenai pembatasan merokok.

Hasil penelitian ini menunjukkan tingkat akurasi yang sangat baik yaitu 91,6%, %, *precision* 96.4%, recall 90.4%, dan F1-Score 93.2% dan mampu memproses 21 – 25 *frame per second* (fps). Hal ini menunjukkan bahwa performa yang baik dalam mendeteksi perilaku merokok.

METHOD

Methods Pada proses pengerjaan penelitian ini menggunakan metode waterfall sebagai metode pengembangan perangkat lunak. Dalam metode ini, setiap fase dari siklus pengembangan hanya dimulai setelah fase sebelumnya telah diselesaikan sepenuhnya. Berikut merupakan tahapan metode waterfall.

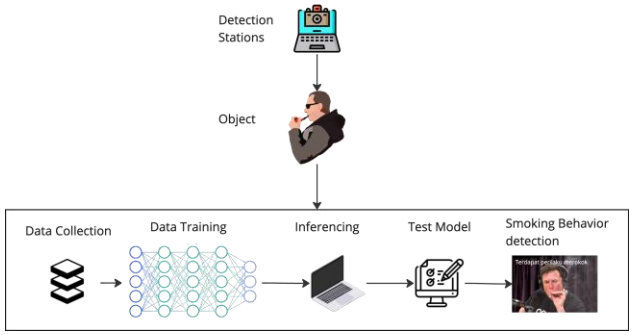


Gambar 1. Arsitektur waterfall

a. *Communication*
   Pada tahap ini peneliti melakukan analisis permasalahan dan mengumpulkan data – data pendukung seperti jurnal, artikel, paper dan sumber terpercaya lainnya. Selain itu penulis melakukan komunikasi dengan pembimbing.
b. *Planning*
   Menyusun perencanaan sistem, menyiapkan sumber data yang diperlukan, hingga membuat jadwal perencanaan penelitian.
c. *Modeling*
   Pada tahap ini membuat perencana-an arsitektur, algoritma program dan perencana-an sistem lainnya.
d. *Deployment*
   Tahap akhir yang berfungsi sebagai implementasi sistem. Pada tahap ini penulis membuat GUI sederhana dalam menampilkan hasil deteksi objek.

Model Konseptual

Model konseptual merupakan representasi dari konsep-konsep kunci dan elemen pada penelitian ini, yang digambarkan sebagai berikut:

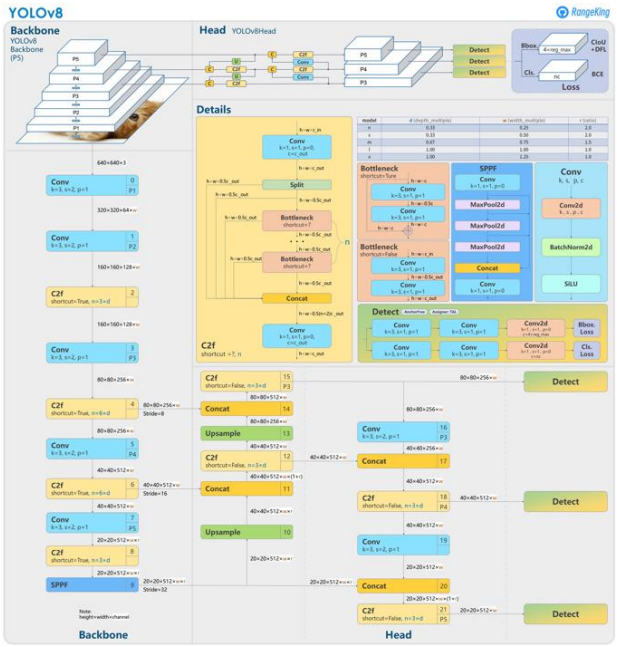


Gambar 2. Model konseptual dari sistem

Berdasarkan ilustrasi di atas, pertama kamera melakukan deteksi terhadap objek pelaku merokok selanjutnya objek yang di tangkap dilakukan kecocokan terhadap model yang sudah dilakukan klasifikasi dan *inferencing*. Apabila objek yang ditangkap terdapat kecocokan maka sistem akan mendeteksi hal tersebut sebagai pelaku merokok.

You Only Look Once (YOLO)

YOLO merupakan algoritma yang dikembangkan untuk mendeteksi sebuah objek secara *real-time* dengan basis *Convolutional Neural Network* (CNN). YOLO merupakan salah satu metode yang cepat dan akurat dalam mendeteksi objek, selain itu memberikan peningkatan pada setiap versinya, dari YOLO v1 hingga saat ini v8 [4].



Gambar 3. Arsitektur YOLO v8

Gambar di atas merepresentasikan arsitektur dari YOLO v8, dengan kerangka yang terdiri dari:

a. Jaringan tulang punggung (*backbone network*)
   Pada *backbone network* menggunakan *feture pyramid network* (FPN) sebagai penghasil fitur dari gambar yang di input.
b. leher (*neck*)
   pada *neck* menggunakan serangkaian *cross layer connection* (CLC) sebagai penyempurnaan fitur dari *backbone*.

 https://doi.org/10.30743/xxxxx

c. kepala (*head*).
Pada *head* menggunakan *feature* yang disempurnakan dan memprediksi *achor box*, skor kelas objek, dan akurasi pada masing-masing objek citra.

Beberapa peningkatan yang dilakukan pada YOLO v8 yaitu, deteksi tanpa *anchor box*, sehingga langsung memprediksi pada pusat objek bukan pada kotak jangkar lagi [5]. Hal ini sangat mengurangi kompleksitas dan meningkatkan generalis objek. Selain itu peningkatan pada efisiensi dan penyebaran hingga kompatibilitas dengan pytorch dan ONNX yang lebih baik.

Apple M2

Bersumber dari web resmi apple meresmikan chip berbasis ARM dengan merilis M1 pada November 2020, dan pada Juni 2022 merilis versi M2. Apple M2 adalah *system on chip (SoC)* dengan berdasarkan arsitektur ARM, sehingga memiliki efisiensi daya dan performa tinggi. Pada apple M2 mengintegrasikan CPU, GPU Neural Engine, kontroler memori, dan komponen lainnya dalam satu chip. Integrasi ini yang mengurangi latensi, meningkatkan efiseinsi daya dan memberikan kinerja secara keseluruhan yang baik [6].

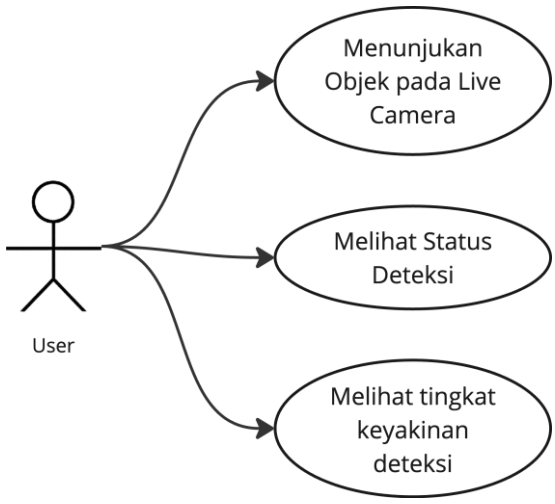
Gambar 4. Arsitektur Chip M2

Saat melatih model ML, developer memanfaatkan dari pelatihan GPU yang dipercepat dengan PyTorch dan TensorFlow dengan memanfaatkan backend Metal Performance Shaders (MPS). Untuk penerapan model terlatih di perangkat Apple, mereka menggunakan coremltools, alat konversi terpadu sumber terbuka Apple, untuk mengonversi model PyTorch dan TensorFlow ke format paket model Core ML.

RANCANGAN SISTEM

Use Case Diagram

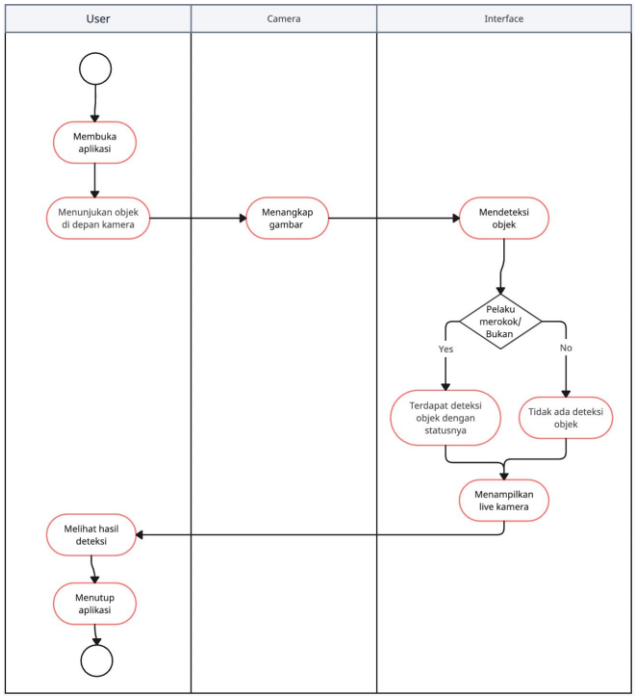
Diagram yang menunjukkan interaksi pengguna dengan sistem, apa saja yang dapat dilakukan oleh pengguna.



Gambar 5. *Use case diagram*

Swimpline

Pemodelan dalam diagram aktivitas menggambarkan aktivitas pada suatu sistem. Berikut aktivitas dari sistem pada penelitian ini.



Gambar 6. Swimpline Diagram

Pengumpulan Data

Pengumpulan data set yang akan digunakan selama pelatihan dan pengujian model. Data yang digunakan pada penelitian dan pengujian terbagi menjadi dua, yaitu:

a. Smoking
Bagian ini terdiri dari citra-citra yang menggambarkan perilaku merokok, seperti contoh berikut:

Gambar 7. Contoh data perilaku merokok

b. Not smoking
Bagian ini terdiri dari citra-citra yang menggambarkan tidak berperilaku merokok, seperti contoh berikut:

Gambar 8. Contoh data perilaku tidak merokok

Pada proses pelatihan digunakan 80% dari data set yang ada, dan 20% lagi digunakan untuk pengujian. Total jumlah data set 3243. Format citra dalam pelatihan ini adalah *selfsupervised* learning dimana citra yang digunakan tidak dilakukan anotasi, namun di input berdasarkan seluruh *frame*.

Pre-Processing

Pada tahap ini dilakukan proses resize atau perubahan ukuran pada seluruh citra menjadi 640 x 640. Proses ini bertujuan untuk mengurangi beban kinerja model sehingga pemrosesan model dapat berjalan lebih ringan dan memakan waktu lebih sedikit. Selain itu penyamaan ukuran seluruh data set juga berfungsi sebagai memastikan konsistensi dan kesesuaian input data dengan arsitektur model yang digunakan.

Pelatihan

Pada proses pelatihan menggunakan Jupyter Notebook sebagai alat utama, bahasa pemrograman Python, dan algoritma YOLOv8. Beberapa *library* yang digunakan meliputi ultralytics untuk implementasi YOLO dan cv2 (OpenCV) untuk pemrosesan gambar.

Tabel 1. Tabel hyperparameter

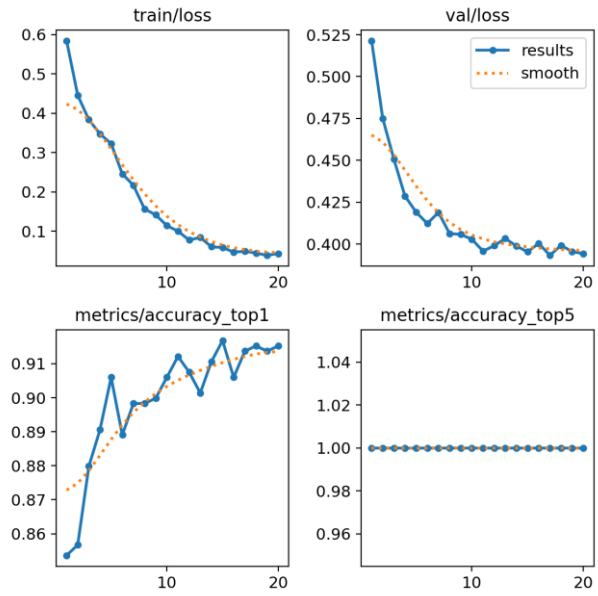
Hyperparameter	**Nilai**
Epoch	20
Batch Size	16
Optimize	SGD
Image size	640
Learning rate	0.01
Seed	42
Drop out	0.3

Proses pelatihan membutuhkan waktu selama 39.315 jam, selama pelatihan penggunaan berfokus pada CPU dan GPU, dan tidak menggunakan kemampuan Apple Neural Engine (ANE).

Dibawah ini merupakan table hasil akurasi dan loss selama pelatihan dan pengujian.

Tabel 2. Tabel hasil akurasi dan loss

Epoch	**Loss**		**Accuracy**	
	Training	**Validation**	**Top 1**	**Top 5**
1	0.584	0.521	0.853	1
2	0.445	0.475	0.856	1
3	0.383	0.450	0.879	1
4	0.347	0.428	0.890	1
5	0.323	0.419	0.906	1
6	0.246	0.412	0.889	1
7	0.217	0.419	0.898	1
8	0.156	0.406	0.898	1
9	0.141	0.406	0.899	1
10	0.114	0.403	0.906	1
11	0.100	0.395	0.912	1
12	0.077	0.399	0.907	1
13	0.084	0.403	0.901	1
14	0.060	0.398	0.910	1
15	0.058	0.395	0.916	1
16	0.047	0.400	0.906	1
17	0.049	0.393	0.913	1
18	0.044	0.399	0.915	1
19	0.038	0.395	0.913	1
20	0.043	0.394	0.915	1



Gambar 9. Grafik hasil akurasi dan loss

Gambar di atas merupakan representasi dari grafik *training* dan *validation*, terdapat sedikit fluktuasi pada *accuracy* top 1. Hasil *training* dievaluasi menggunakan *confusion matrix* yang menunjukkan performa model dalam mendeteksi perilaku

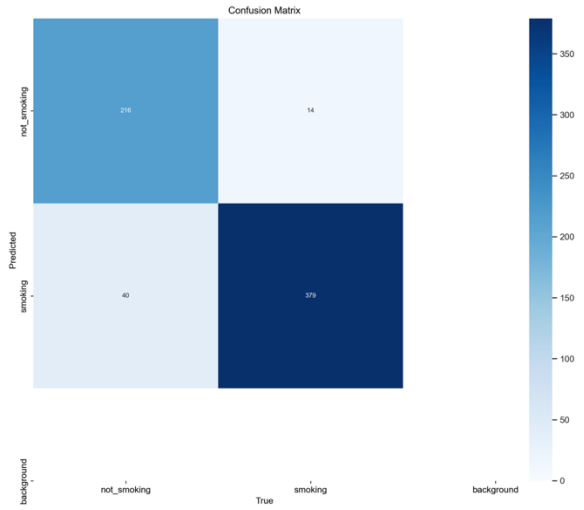
https://doi.org/10.30743/xxxxx

merokok dan tidak. Sehingga dapat dihitung total sampel per detik yang mampu dilatih menggunakan Apple M2 adalah

Total sample = 2594 citra x 20 epoch = 51,880 sampel

Kecepatan = $\frac{Total\ sampel}{Total\ waktu} = \frac{51{,}880}{139{,}352} = 0.372$ sampel per detik.

Bersumber dari https://www.kaggle.com/code/sup41kkk/yolov8-smoking-detection/notebook, pengujian dengan data set dengan kemiripan 97%. Dihasilkan pelatihan 20 epochs selesai dalam waktu 1.097 jam, menggunakan GPU Tesla P100-PCIE-16GB.



Gambar 10. Tabel hasil confusion matrix

Berdasarkan tabel *confusion matrix* dapat dihitung tingkat akurasi, presisi, *recall*, dan F1 – score [7].

Akurasi

Menghitung persentase prediksi yang benar dari keseluruhan data uji, berikut cara menghitung akurasi.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$
$$Accuracy = \frac{379+216}{379+40+14+216}$$
$$Accuracy = \frac{595}{649} = \mathbf{0.916}$$

Hasil akurasi sebesar 91.6% menunjukkan bahwa model ini memiliki kemampuan yang sangat baik dalam memprediksi dengan benar.

Precision

Mengukur seberapa banyak prediksi positif yang benar, berikut cara menghitung persisi.

$$Precission = \frac{TP}{TP+FP}$$
$$Precission = \frac{379}{379+14}$$
$$Precission = \frac{379}{393} = \mathbf{0.964}$$

Hasil *precision* sebesar 96.4% menunjukkan bahwa hampir semua prediksi positif model adalah benar, yang berarti model sudah sangat baik dalam menghindari prediksi positif yang salah

Recall

Mengukur seberapa banyak data positif yang terdeteksi dengan benar.

$$Recall = \frac{TP}{TP+FN}$$
$$Recall = \frac{379}{379+40}$$
$$Recall = \frac{379}{419}$$
$$Recall = \frac{379}{419} = \mathbf{0.904}$$

Hasil *recall* sebesar 90.4% menunjukkan bahwa model ini sangat baik dalam mendeteksi data positif, meskipun ada beberapa data positif yang terlewatkan.

F1-score

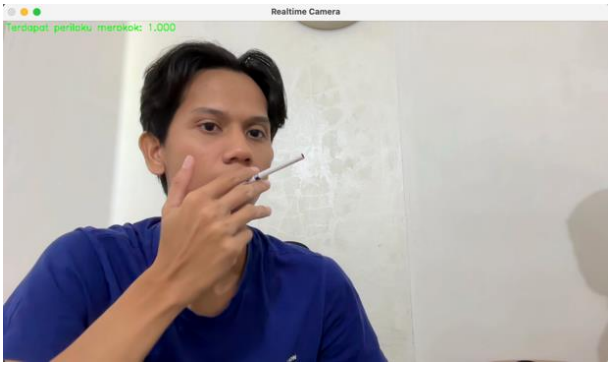
Kombinasi dari *precision* dan *recall* untuk memberikan gambaran yang lebih seimbang mengenai performa model.

$$F1 - Score = 2 . \frac{Precission\ .\ Recall}{Precission+Recall}$$
$$F1 - Score = 2 . \frac{0.964\ .\ 0.9}{0.964\ +0.9}$$
$$F1 - Score = 2 . \frac{0.868}{1.86} = \mathbf{0.932}$$

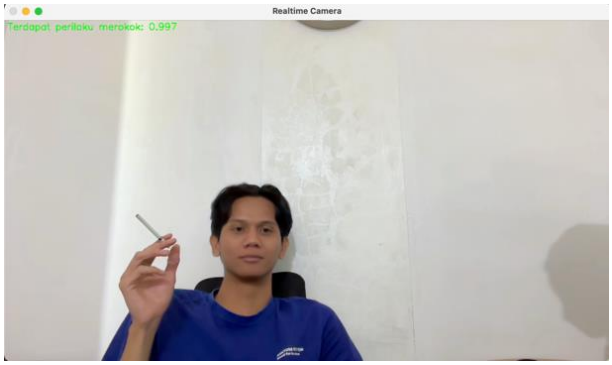
Hasil F1-Score sebesar 93.2% menunjukkan bahwa model ini memiliki keseimbangan yang sangat baik antara *precision* dan *recall*, memberikan gambaran performa model yang sangat baik secara keseluruhan.

Pengujian

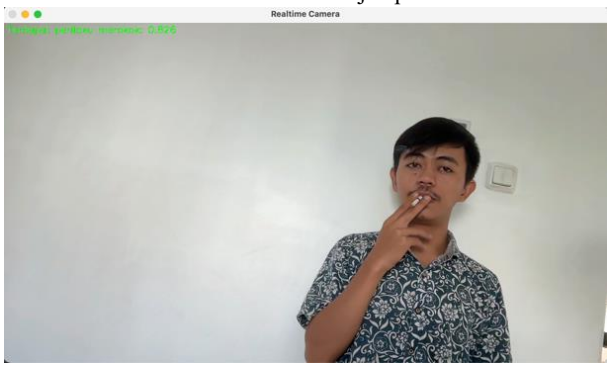
Pengujian pada model menggunakan implementasi GUI sederhana, dimana inisialisasi camera menggunakan OpenCV, dan objek akan dideteksi secara real time. Berikut merupakan beberapa sampel tangkapan layar saat pengujian model secara real time. Selama proses pengujian dilakukan pemantauan kerja ANE dan juga kecepatan deteksi objek setiap framenya.



Gambar 11. Hasil deteksi 1 objek perilaku merokok

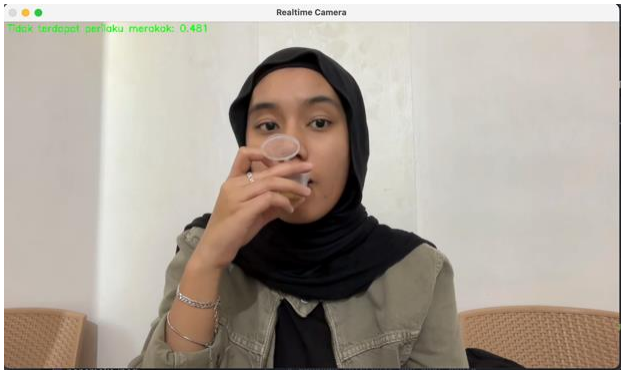


Gambar 12. Hasil deteksi 2 objek perilaku merokok



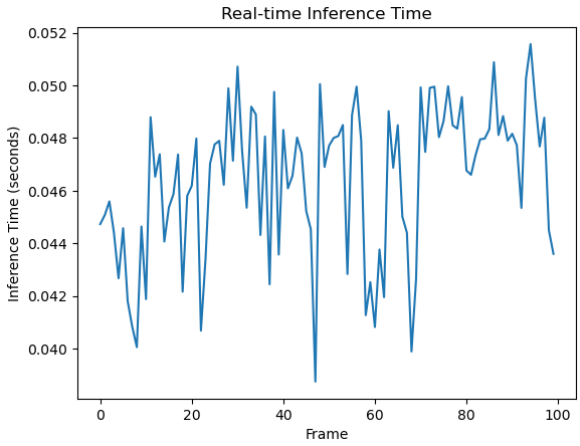
Gambar 13. Hasil deteksi 3 objek perilaku merokok

Gambar 14. Hasil deteksi 1 objek perilaku tidak merokok



Gambar 15. Hasil deteksi 2 objek perilaku tidak merokok

Selama pengujian real-time selama 60 detik, dilakukan pemantauan terhadap kecepatan frame per second untuk mengevaluasi hasil deteksi secara langsung. second sebagai berikut:



Gambar 16. Grafik inference frame per second

Dari grafik tersebut menunjukkan bahwa ANE mampu melakukan inferensi dengan waktu yang stabil dan cepat pada setiap *frame*. Dengan waktu rata – rata sekitar 0.046 detik, ANE mampu memproses 21 – 25 *frame* per *second* (fps). Waktu inferensi yang konsisten dan cepat menunjukkan bahwa ANE efisien dalam menjalankan model pembelajaran.

CONCLUSIONS

Dari hasil penelitian ini dapat disimpulkan bahwa chip M2 dapat melakukan training model untuk mendeteksi perilaku merokok. Hasil training menunjukkan penurunan loss yang konsisten dari sekitar 0.6 menjadi 0.1, hal ini menunjukkan bahwa model belajar dari data pelatihan dan semakin baik dalam meminimalkan kesalahan prediksi. Nilai validation loss juga menurun dari 0.525 menjadi 0.4 yang menunjukkan bahwa model juga baik. Pelatihan ini juga menunjukkan tidak adanya overfitting maupun underfitting. Dari hasil perhitungan didapatkan kecepatan pelatihan model sebesar 0.328 sampel per detik, bila dibandingkan dengan penggunakan chip lain pelatihan ini tergolong lambat, namun tetap menghasilkan akurasi sebesar 91.6%, precision 96.4%, recall 90.4%, dan F1-Score 93.2%.

Selama pelatihan, penggunaan CPU dan GPU lebih dimanfaatkan. Sedangkan selama inferensi, kemampuan tambahan dari Apple Neural Engine (ANE) digunakan. ANE mampu memproses 21 – 25 frame per detik (fps), yang menunjukkan kemampuan yang baik untuk deteksi objek secara real-time. Dengan demikian, dapat disimpulkan bahwa meskipun kecepatan pelatihan menggunakan chip M2 relatif lebih lambat dibandingkan dengan GPU Tesla P100-PCIE, hasil pelatihan menunjukkan performa yang sangat baik dalam mendeteksi perilaku merokok di ruang terbuka. Penggunaan ANE selama inferensi juga memungkinkan deteksi real-time yang efisien, menjadikan chip M2 sebagai pilihan yang solid untuk aplikasi visi komputer yang membutuhkan pemrosesan cepat dan akurat.

REFERENCES

[1] Agung and Pribadi, “Analisis Yuridis Peraturan Pemerintah Nomor 109 Tahun 2012 Tentang Pengamanan Bahan Yang Mengandung Zat Adiktif Berupa Produk Tembakau Bagi Kesehatan,” *Agung Pribadi Azhari*, vol. X, pp. 1–15, 2023.

 https://doi.org/10.30743/xxxxx

[2] M. Gojali and E. Tjiong, "View of Application Development og Cigarette Object Detection and Smoking Activities Using YOLOv3 Algorithm," vol. 10, pp. 1–8, 2023.

[3] S. Srivastava, A. V. Divekar, C. Anilkumar, I. Naik, V. Kulkarni, and V. Pattabiraman, "Comparative analysis of deep learning image detection algorithms," *J Big Data*, vol. 8, no. 1, Dec. 2021, doi: 10.1186/s40537-021-00434-w.

[4] Yanto, F. Aziz, and Irmawati, "Yolo-V8 Peningkatan Algoritma Untuk Deteksi Pemakaian Masker Wajah," pp. 1–8, 2023.

[5] J. Terven and D. Cordova-Esparza, "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS," Apr. 2023, doi: 10.3390/make5040083.

[6] D. Kasperek, P. Antonowicz, M. Baranowski, M. Sokolowska, and M. Podpora, "Comparison of the Usability of Apple M2 and M1 Processors for Various Machine Learning Tasks," *Sensors*, vol. 23, no. 12, Jun. 2023, doi: 10.3390/s23125424.

[7] laila Qadrini, andi Seppewali, and asra Aina, "Decision Tree Dan Adaboost Pada Klasifikasi Penerima Program Bantuan Sosial," vol. 7, pp. 1–8, 2021.