E-ISSN : 2988-1986
https://ejournal.cahayailmubangsa.institute/index.php/kohesi

Kohesi: Jurnal Multidisiplin Saintek
Volume 11 No 1 Tahun 2026

IMPLEMENTASI DETEKSI EMOSI WAJAH REAL-TIME BERBASIS TENSORFLOW LITE DALAM LINGKUNGAN DOCKER CONTAINER

Muhammad Dimas Putra Sudrajat
Program Studi Teknik Informatika, STMIK Tazkia, Bogor
251552010031dimasputra@student.stmik.tazkia.ac.id

ABSTRAK

Implementasi sistem kecerdasan buatan berbasis *deep learning* yang dilakukan pada perangkat keras standar mengalami dua hambatan utama: beban komputasi tinggi dan kompleksitas pengelolaan dependensi perangkat lunak lintas platform. Penelitian ini bertujuan membangun sistem deteksi emosi wajah *real-time* yang ringan dan mudah dengan menggunakan *TensorFlow Lite* (TFLite) dan *Docker Container*. Model *Convolutional Neural Network* (CNN) dilatih pada dataset FER-2013 (28.709 citra latih, 7.178 citra uji, 7 kelas emosi) kemudian dikompresi ke format TFLite. Evaluasi menghasilkan akurasi keseluruhan 51,94% dengan kelas *Happy* mencapai F1-*Score* tertinggi (72,73%). Latensi inferensi rata-rata 0,309 ms per citra dengan *throughput* 3.241 FPS membuktikan kemampuan *real-time* yang melampaui standar 24 FPS. Kompresi TFLite mereduksi ukuran model 97% (50–100 MB menjadi 1,4 MB). Sistem berhasil dieksekusi pada laptop standar dalam lingkungan Docker tanpa akselerator perangkat keras khusus.

Kata kunci: *Deteksi emosi wajah; TensorFlow Lite; Docker Container; FER-2013; Convolutional Neural Network*

ABSTRACT

The implementation of a deep learning-based artificial intelligence system on standard hardware faces two major challenges: high computational load and the complexity of managing cross-platform software dependencies. This study aims to develop a lightweight and user-friendly real-time facial emotion detection system using TensorFlow Lite (TFLite) and Docker containers. A Convolutional Neural Network (CNN) model was trained on the FER-2013 dataset (28,709 training images, 7,178 test images, 7 emotion classes) and then compressed into the TFLite format. Evaluation yielded an overall accuracy of 51.94%, with the "Happy" class achieving the highest F1-Score (72.73%). The average inference latency of 0.309 ms per image, with a throughput of 3,241 FPS, demonstrates real-time capability that exceeds the 24 FPS standard. TFLite compression reduced the model size by 97% (from 50–100 MB to 1.4 MB). The system ran successfully on a standard laptop in a Docker environment without a dedicated hardware accelerator.

Keywords: *Facial emotion detection; TensorFlow Lite; Docker Container; FER-2013; Convolutional Neural Network*

Article History
Received: Juni 2026
Reviewed:Juni 2026
Published: Juni 2026

Plagiarism Checker
No 237.u873247.fh8374

Copyright : Author
Publish by : Kohesi

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License

1. PENDAHULUAN

Menjalankan program berbasis kecerdasan buatan di laptop dengan spesifikasi standar bukanlah hal yang mudah. Pengalaman ini dirasakan langsung oleh penulis saat mencoba menjalankan model deteksi emosi wajah di laptop pribadi yang prosesnya sangat lambat, laptop

sering mengalami *freeze* hingga harus di-*restart*, dan hasil deteksinya pun jauh dari akurat. Keterbatasan RAM dan GPU menjadi hambatan nyata yang tidak bisa diabaikan begitu saja.

Sebagai mahasiswa Teknik Informatika yang ingin mendalami *machine learning* dengan keterbatasan perangkat keras, penulis harus mencari solusi alternatif. Dua teknologi yang kemudian dieksplorasi adalah TensorFlow Lite (TFLite) dan Docker Container. TFLite terbukti mampu menjalankan model AI dengan konsumsi *resource* yang jauh lebih ringan [5], sementara Docker memungkinkan seluruh lingkungan program dikemas dalam satu wadah yang mudah dipindahkan tanpa khawatir masalah kompatibilitas [6].

Salah satu tantangan tambahan adalah keterbatasan pilihan dataset. FER-2013 dipilih bukan semata karena kualitasnya, melainkan karena merupakan dataset publik dengan ukuran file yang masih terjangkau untuk diunduh dan diproses pada perangkat standar [3]. Dataset dengan cakupan lebih luas umumnya memiliki ukuran yang jauh lebih besar dan tidak praktis untuk kondisi seperti ini.

Penelitian ini hadir sebagai eksplorasi nyata bagaimana mahasiswa dengan keterbatasan *hardware* namun tetap dapat membangun dan mengevaluasi sistem *machine learning* secara mandiri. Kontribusi utamanya meliputi: (1) implementasi sistem deteksi emosi berbasis TFLite dalam Docker Container yang dapat berjalan di laptop standar, (2) evaluasi kuantitatif mencakup akurasi, F1-*Score* per kelas, dan latensi inferensi, serta (3) analisis jujur terhadap keterbatasan sistem sebagai bahan perbaikan ke depan.

2. TINJAUAN PUSTAKA

Dari berbagai dokumentasi dan referensi yang ditemukan selama pengerjaan proyek ini, satu benang merah yang terus muncul adalah soal kebutuhan komputasi. Melatih model *machine learning* dengan jumlah data yang besar memang pada dasarnya membutuhkan perangkat keras yang mumpuni karena prosesnya berat dan menghabiskan banyak *resource*. Ini bukan opini, melainkan kondisi yang diakui oleh para peneliti sebelumnya [2][7].

Ekman dan Friesen [9] sejak lama sudah membuktikan bahwa ekspresi emosi manusia bersifat universal lintas budaya, yang menjadi landasan ilmiah mengapa deteksi emosi lewat wajah layak diteliti lebih jauh. Sariyanidi et al. [1] kemudian memetakan perkembangan riset di bidang ini dan mencatat CNN sebagai pendekatan yang paling banyak digunakan. Li dan Deng [14] dalam survei yang lebih baru melaporkan bahwa akurasi model CNN standar pada FER-2013 berkisar antara 45–65%, sementara model yang lebih canggih berbasis *transformers* bisa mencapai 88%, tapi dengan konsekuensi kebutuhan komputasi yang jauh lebih besar, sesuatu yang tidak realistis untuk perangkat standar.

Dataset FER-2013 sendiri diperkenalkan oleh Goodfellow et al. [3] dan sejak itu menjadi tolok ukur umum dalam penelitian deteksi emosi. Dataset ini memiliki kelemahan yang sudah diidentifikasi oleh Pramerdorfer dan Kampel [10] distribusi kelasnya tidak seimbang, dan beberapa kelas seperti *Fear* dan *Surprise* memiliki kemiripan visual yang membingungkan model.

Soal solusi untuk keterbatasan perangkat keras, beberapa penelitian sudah menunjukkan arah yang menarik. LeCun et al. [7] menegaskan CNN sebagai fondasi *deep learning* untuk pengenalan visual. Howard et al. [11] lalu mengembangkan MobileNet arsitektur yang sengaja dirancang lebih ringan tanpa kehilangan banyak akurasi. David et al. [5] membuktikan TFLite bisa berjalan bahkan di mikrokontroler dengan memori di bawah 256 KB. Wibowo et al. [12] mengimplementasikannya di Raspberry Pi, dan Prasetyo dan Santoso [13] melaporkan efisiensi energi 68% menggunakan TFLite untuk kamera pengawas. Untuk deteksi wajahnya sendiri,

metode *Haar Cascade* dari Viola dan Jones [8] masih relevan sampai sekarang karena kecepatannya di lingkungan berbasis CPU.

Di sisi pengelolaan lingkungan program, Boettiger [4] dan Merkel [6] sudah menunjukkan bahwa Docker secara nyata mengurangi kerumitan konfigurasi dan meningkatkan reprodusibilitas. Sculley et al. [2] bahkan mengidentifikasi manajemen dependensi sebagai salah satu utang teknis tersembunyi yang paling sering diabaikan dalam sistem *machine learning*.

Dari semua referensi tersebut, satu celah yang belum terjawab adalah: bagaimana jika semua teknologi ini digabungkan dalam satu sistem yang bisa dijalankan langsung di laptop standar, lengkap dengan evaluasi kuantitatif yang terukur? Sebagian besar penelitian sebelumnya menguji TFLite di perangkat ARM atau mikrokontroler belum ada yang secara eksplisit mengintegrasikannya dengan Docker untuk skenario laptop biasa. Penelitian ini mencoba menjawab celah tersebut.

3. METODE PENELITIAN

Dataset

Saat pertama kali mengunduh dan mengekstrak dataset FER-2013, penulis cukup terkejut melihat struktur data latih untuk model *machine learning* — ribuan gambar wajah berukuran kecil yang sudah dikelompokkan per kelas emosi. Karena jumlahnya sangat banyak dan tidak memungkinkan untuk difilter secara manual, seluruh data langsung digunakan tanpa penyaringan tambahan. Dataset ini terdiri dari 28.709 citra latih dan 7.178 citra uji pada tujuh kelas emosi [3]. Distribusi kelasnya tidak merata — kelas *Happy* mendominasi dengan 7.215 citra (25,1%), sedangkan *Disgust* hanya memiliki 436 citra (1,5%). FER-2013 dipilih bukan semata karena kualitasnya, melainkan karena ukuran filenya masih terjangkau untuk diunduh dan diproses pada laptop dengan spesifikasi standar.

Tabel 1. Distribusi Kelas Dataset FER-2013

Kelas Emosi	Latih	% Latih	Uji	% Uji
Angry	3.995	13,9%	958	13,3%
Disgust	436	1,5%	111	1,5%
Fear	4.097	14,3%	1.024	14,3%
Happy	7.215	25,1%	1.774	24,7%
Neutral	4.965	17,3%	1.233	17,2%
Sad	4.830	16,8%	1.247	17,4%
Surprise	3.171	11,0%	831	11,6%
Total	**28.709**	**100%**	**7.178**	**100%**

Arsitektur Model CNN

Pelatihan model dilakukan di Google Colaboratory memanfaatkan akselerator GPU yang tersedia secara gratis. Hasilnya cukup mengejutkan — proses training yang pada laptop biasa memakan waktu sangat lama, di Colab selesai hanya dalam 20–30 detik. Arsitekturnya terdiri dari tiga blok konvolusi bertingkat yang masing-masing memiliki filter sebanyak 32, 64, dan 128 dengan ukuran kernel 3×3. Setiap blok menyusun empat lapisan secara berurutan: Conv2D

dengan aktivasi ReLU, BatchNormalization, MaxPooling2D berukuran 2×2, dan Dropout 0,25. Setelah ketiga blok konvolusi, output diratakan lewat Flatten lalu diteruskan ke Dense 128 neuron dengan Dropout 0,5, dan diakhiri lapisan Dense 7 neuron ber-aktivasi Softmax sebagai output prediksi emosi. Total keseluruhan sekitar 1,1 juta parameter [7].

Konversi ke TensorFlow Lite

Hasil training menghasilkan file model berformat .h5. Yang menarik, ukurannya tidak sebesar yang dibayangkan mengingat jumlah data yang dilatih. Namun setelah dikonversi ke format TFLite menggunakan TFLiteConverter dengan teknik kuantisasi dinamis, ukurannya menyusut drastis menjadi hanya 1,4 MB — turun 97% dari ukuran semula. Teknik kuantisasi dinamis ini mengubah bobot model dari float32 ke int8 tanpa perlu data kalibrasi tambahan [5]. Pada saat inferensi, bobot dikembalikan ke float32 secara otomatis sehingga hasil prediksinya tetap akurat.

Preprocessing Citra

Kondisi pencahayaan ruangan ternyata berpengaruh besar terhadap hasil deteksi. Pengujian dilakukan di ruangan dengan cahaya terang dan merata tanpa bayangan, yang membantu sistem bekerja lebih stabil. Untuk mengantisipasi variasi pencahayaan yang mungkin terjadi, setiap ROI wajah yang terdeteksi diproses menggunakan tiga tahap: pertama, *resize* ke dimensi 48×48 piksel menggunakan interpolasi bilinear; kedua, peningkatan kontras lokal menggunakan CLAHE dengan *clipLimit* 2,0 dan *tileGridSize* 8×8 — teknik ini bekerja per region kecil pada gambar sehingga area yang gelap dan terang sama-sama bisa terbaca; ketiga, normalisasi nilai piksel ke rentang [0,0; 1,0].

Kontainerisasi dengan Docker

Seluruh komponen sistem dikemas dalam Docker image berbasis python:3.9-slim. Meski terlihat kompleks di awal, perintah Docker yang digunakan ternyata cukup sederhana dan mudah dihafalkan. Untuk menjalankan aplikasi dengan akses kamera dari dalam container di sistem Linux, hanya perlu dua tambahan perintah: *--device=/dev/video0* untuk memetakan kamera fisik, dan *-e DISPLAY=$DISPLAY -v /tmp/.X11-unix:/tmp/.X11-unix* untuk meneruskan tampilan grafis X11 [4][6].

Prosedur Evaluasi

Evaluasi dilakukan pada seluruh 7.178 citra uji FER-2013 secara individual menggunakan script evaluate.py. Setiap citra diproses dengan pipeline yang sama seperti saat inferensi real-time resize, CLAHE, normalisasi, lalu diumpankan ke model TFLite. Metrik yang diukur meliputi akurasi keseluruhan, precision, recall, dan F1-Score per kelas emosi, confusion matrix, serta latensi inferensi rata-rata menggunakan time.perf_counter() Python. Seluruh pengujian dilakukan pada CPU laptop standar tanpa GPU.

4. HASIL DAN PEMBAHASAN

Performa Kompresi Model

Konversi dari format Keras .h5 ke TFLite menghasilkan perubahan yang cukup signifikan. File model yang sebelumnya berukuran 50–100 MB berhasil dipangkas menjadi hanya 1,4 MB,

reduksi sebesar 97%. Dalam praktiknya, perbedaan ini langsung terasa: model TFLite jauh lebih responsif saat dijalankan di laptop standar dibanding model aslinya yang kerap membuat sistem melambat (Tabel 2).

Tabel 2. Perbandingan Format Model AI

Format Model	Ukuran File	Performa Kamera	Keterangan
Keras (.h5)	~50–100 MB	Lambat / Lag	Model penuh, float32
TFLite (.tflite)	1,4 MB	Real-time (3.241 FPS)	Dikuantisasi, int8→float32

Hasil Evaluasi Akurasi

Saat hasil evaluasi pertama kali muncul di terminal, reaksi pertama penulis adalah kecewa, akurasi 51,94% masih jauh dari target awal yang diharapkan mencapai 70%. Saat dicoba langsung di depan kamera pun hasilnya tidak selalu konsisten; beberapa ekspresi yang ditunjukkan tidak terdeteksi, dan tidak jarang ekspresi A malah terbaca sebagai ekspresi B. Tabel 3 menyajikan rincian metrik per kelas emosi.

Kelas *Happy* menjadi yang paling mudah dikenali sistem dengan F1-*Score* 72,73% dan *recall* 85,01%. Hal ini masuk akal karena senyum adalah ekspresi yang paling khas secara visual dan paling banyak tersedia dalam data latih (7.215 citra). Sebaliknya, kelas *Disgust* mencatatkan nilai nol di semua metrik. Dari 111 citra uji *Disgust*, tidak satu pun yang berhasil dikenali dengan benar, mayoritas malah diklasifikasikan sebagai *Angry*. Ini bukan kesalahan model semata, melainkan konsekuensi langsung dari data latih yang terlalu sedikit, hanya 436 citra atau 1,5% dari total [3][10].

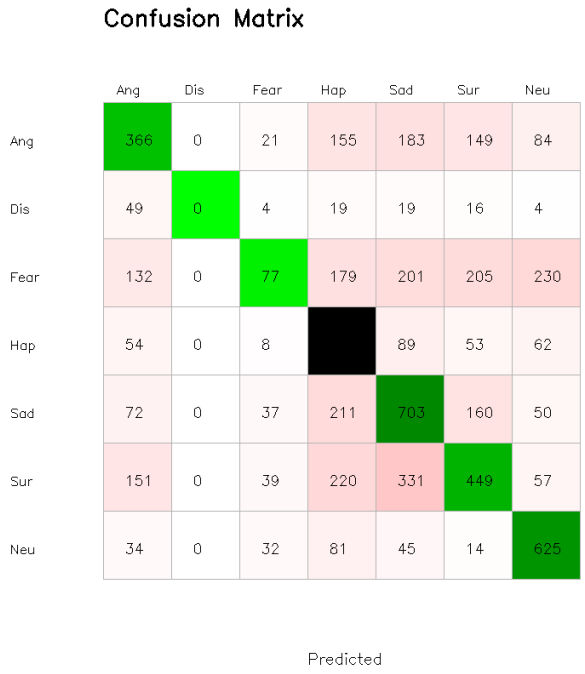
Kelas *Fear* juga bermasalah dengan *recall* hanya 7,52%. Artinya dari 1.024 citra ekspresi takut, lebih dari 90%-nya salah dikenali, sebagian besar dikira *Neutral* atau *Surprise*. Secara visual memang ekspresi *Fear* dan *Surprise* mirip: mata sama-sama terbuka lebar, mulut sedikit terbuka. Pada resolusi 48×48 piksel, perbedaan kecil seperti itu sulit ditangkap model [10].

Tabel 3. Hasil Evaluasi Per Kelas Emosi pada Data Uji FER-2013

Kelas	Precision (%)	Recall (%)	F1-Score (%)	Support	Rank
Angry	42,66	38,20	40,31	958	4
Disgust	0,00	0,00	0,00	111	7
Fear	35,32	7,52	12,40	1.024	6
Happy	63,55	85,01	72,73	1.774	1
Neutral	44,75	57,02	50,14	1.233	3
Sad	42,93	36,01	39,16	1.247	5
Surprise	56,21	75,21	64,33	831	2
Makro	**40,77**	**42,71**	**39,87**	7.178	-

Analisis Confusion Matrix

Gambar 1 memperlihatkan pola kesalahan klasifikasi secara keseluruhan. Diagonal utama yang berwarna hijau menunjukkan prediksi yang benar, sementara sel-sel di luar diagonal adalah kesalahan. Terlihat jelas bahwa kolom *Happy* menjadi "tempat buang" banyak prediksi salah, model terlalu sering menebak *Happy* bahkan untuk ekspresi lain, kemungkinan karena dominasinya dalam data latih membuat model bias ke kelas ini.



Gambar 1. Confusion Matrix Evaluasi pada Data Uji FER-2013 (7.178 Citra)

Analisis Performa Inferensi

Yang mengejutkan justru bukan akurasinya melainkan kecepatannya. Latensi inferensi rata-rata hanya 0,309 ms per citra dengan estimasi *throughput* 3.241 FPS (Tabel 4). Angka ini jauh melampaui standar *real-time* 24 FPS. Ironisnya, sistem yang sangat cepat ini menghasilkan prediksi yang tidak selalu tepat, bukti bahwa kecepatan komputasi dan akurasi adalah dua hal yang berbeda dan tidak otomatis berjalan beriringan. Standar deviasi yang kecil (0,089 ms) menunjukkan performa inferensi yang konsisten dan stabil tanpa lonjakan yang berarti.

Tabel 4. Statistik Latensi Inferensi Model TFLite (n=7.178 citra)

Metrik	Nilai
Rata-rata latensi	0,309 ms/citra
Median latensi	0,294 ms/citra
Standar deviasi	0,089 ms
Latensi minimum	0,284 ms/citra
Latensi maksimum	2,143 ms/citra
Estimasi throughput	3.241 FPS

5. KESIMPULAN

Penelitian ini membuktikan bahwa keterbatasan perangkat keras bukan halangan mutlak untuk membangun sistem *machine learning* yang fungsional. Dengan menggabungkan TFLite

dan Docker Container, model deteksi emosi wajah berhasil dijalankan secara *real-time* di laptop standar yang sebelumnya menyebabkan laptop *freeze* saat dicoba tanpa optimasi.

Tiga hal yang dapat disimpulkan. Pertama, kompresi TFLite memangkas ukuran model 97% (menjadi 1,4 MB) dengan latensi rata-rata 0,309 ms per citra, jauh melampaui kebutuhan *real-time* 24 FPS. Kedua, akurasi keseluruhan 51,94% masih di bawah ekspektasi awal penulis yang menargetkan 70%. Kelas *Happy* mencatatkan F1-*Score* terbaik (72,73%), sementara kelas *Disgust* tidak berhasil dikenali sama sekali akibat data latih yang terlalu sedikit. Ketiga, Docker Container mempermudah pengelolaan lingkungan program, meskipun pengujian lintas perangkat belum sempat dilakukan dalam penelitian ini.

Untuk ke depannya, prioritas utama adalah mengganti FER-2013 dengan dataset yang lebih baru dan memiliki distribusi kelas lebih merata. Eksplorasi arsitektur seperti MobileNetV2 [11] juga menjadi agenda berikutnya, dengan harapan sistem yang dibangun kelak dapat mencapai akurasi yang lebih memuaskan tanpa mengorbankan efisiensi komputasi.

DAFTAR PUSTAKA

[1] E. Sariyanidi, H. Gunes, and A. Cavallaro, "Automatic analysis of facial affect: A survey of registration, representation, and recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 6, pp. 1113–1133, Jun. 2015, doi: 10.1109/TPAMI.2014.2366127.

[2] D. Sculley et al., "Hidden technical debt in machine learning systems," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2015, vol. 28, pp. 2503–2511.

[3] I. J. Goodfellow et al., "Challenges in representation learning: A report on three machine learning contests," Neural Netw., vol. 64, pp. 59–63, Apr. 2015, doi: 10.1016/j.neunet.2014.09.005.

[4] C. Boettiger, "An introduction to Docker for reproducible research," ACM SIGOPS Oper. Syst. Rev., vol. 49, no. 1, pp. 71–79, Jan. 2015, doi: 10.1145/2723872.2723882.

[5] R. David et al., "TensorFlow Lite Micro: Embedded machine learning for TinyML systems," Proc. Mach. Learn. Syst. (MLSys), vol. 3, pp. 800–811, 2021.

[6] D. Merkel, "Docker: Lightweight Linux containers for consistent development and deployment," Linux J., vol. 2014, no. 239, Art. 2, Mar. 2014.

[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[8] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2001, vol. 1, pp. I-511–I-518, doi: 10.1109/CVPR.2001.990517.

[9] P. Ekman and W. V. Friesen, "Constants across cultures in the face and emotion," J. Pers. Soc. Psychol., vol. 17, no. 2, pp. 124–129, 1971, doi: 10.1037/h0030377.

[10] C. Pramerdorfer and M. Kampel, "Facial expression recognition using convolutional neural networks: State of the art," arXiv:1612.02903, Dec. 2016. [Online]. Available: https://arxiv.org/abs/1612.02903

[11] A. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv:1704.04861, Apr. 2017. [Online]. Available: https://arxiv.org/abs/1704.04861

[12] A. Wibowo, A. Santoso, and H. Nugroho, "Implementasi TensorFlow Lite untuk klasifikasi gestur tangan pada Raspberry Pi berbasis kamera," J. Tek. Inform. dan Sist. Inf., vol. 8, no. 2, pp. 145–156, 2022.

[13] B. Prasetyo and L. Santoso, "Optimasi deteksi objek real-time menggunakan TFLite pada sistem pengawasan bertenaga baterai," J. Nas. Tek. Elektro Teknol. Inf. (JNTETI), vol. 12, no. 1, pp. 45–53, 2023.

[14] S. Li and W. Deng, "Deep facial expression recognition: A survey," IEEE Trans. Affect. Comput., vol. 13, no. 3, pp. 1195–1215, 2022, doi: 10.1109/TAFFC.2020.2981446.

[15] Z. Zhao, Q. Liu, and F. Zhou, "Robust lightweight facial expression recognition network with label distribution training," in Proc. AAAI Conf. Artif. Intell., 2021, vol. 35, no. 4, pp. 3510–3519, doi: 10.1609/aaai.v35i4.16467.