Institut Riset dan Publikasi Indonesia (IRPI)
MALCOM: Indonesian Journal of Machine Learning and Computer Science
Journal Homepage: https://journal.irpi.or.id/index.php/malcom
Vol. 5 Iss. 1 January 2025, pp: 391-404
ISSN(P): 2797-2313 | ISSN(E): 2775-8575

Fire Detection Using Single Shoot MultiBox Detector Algorithm with RGB Rule and YCbCr Rule

Deteksi Kebakaran Menggunakan Algoritma Single Shoot MultiBox Detector dengan Rule RGB dan Rule YcbCr

Baharuddin[1], Yuyun[2], Nasrullah[3]

[1,2,3]Department of Computer System, Handayani University Makassar, Indonesia

E-mail: baharanthyqu@gmail.com[1],
yuyun010@brin.go.id[2], nasrullah@handayani.ac.id[3]

Received Nov 10th 2024; Revised Jan 13th 2025; Accepted Jan 22th 2025; Available Online Jan 25th 2025, Published Jan 30th 2025
Corresponding Author: Baharuddin

Abstract

This research develops a fire detection system based on computer vision using the Single Shot Multibox Detector (SSD) algorithm to address the limitations of conventional fire detection systems, which are often less responsive and less accurate under varying lighting conditions and distances. The system employs an SSD MobileNetV2 model trained on an augmented fire image dataset, enabling reliable detection across diverse scenarios. The results indicate that integrating RGB and YCbCr rules significantly enhances detection accuracy, particularly by reducing the false positive rate commonly encountered in previous methods. Additionally, the system improves detection efficiency by achieving a more optimal fire-to-frame detection ratio, ensuring rapid real-time response. With high accuracy and real-time detection capabilities, this system proves effective for practical applications such as fire monitoring in industrial and public areas, providing additional protection and supporting faster preventive actions against potential fires.

Keywords: Computer Vision, Fire Detection, RGB Rule, Single Shot Multibox Detector (SSD), YCbCr Rule

Abstrak

Penelitian ini mengembangkan sistem deteksi api berbasis computer vision menggunakan algoritma Single Shot Multibox Detector (SSD) untuk mengatasi keterbatasan sistem deteksi api konvensional yang umumnya kurang responsif dan kurang akurat dalam kondisi pencahayaan dan jarak yang bervariasi. Sistem ini menerapkan model SSD MobileNetV2 yang telah dilatih menggunakan dataset gambar api yang di-augmentasi, memungkinkan deteksi yang andal dalam berbagai skenario. Hasil penelitian menunjukkan bahwa penggunaan kombinasi aturan RGB dan YCbCr secara signifikan meningkatkan akurasi deteksi, khususnya dalam mengurangi tingkat false positive yang sering terjadi pada metode sebelumnya. Selain itu, sistem ini meningkatkan efisiensi waktu deteksi dengan rasio deteksi api terhadap frame yang lebih optimal, sehingga dapat memberikan respons real-time yang cepat. Dengan tingkat akurasi yang tinggi dan kemampuan deteksi real-time, sistem ini efektif untuk aplikasi praktis seperti pemantauan kebakaran di area industri dan publik, memberikan perlindungan tambahan, serta mendukung tindakan preventif yang lebih cepat dalam menghadapi potensi kebakaran.

Kata Kunci: *Computer Vision, Fire Detection, RGB Rule, Single Shot Multibox Detector (SSD), YCbCr Rule*

1. PENDAHULUAN

Kebakaran adalah salah satu bencana yang memiliki dampak signifikan terhadap lingkungan, ekonomi, dan kehidupan manusia. Deteksi dini terhadap kebakaran menjadi sangat penting, terutama dalam konteks pencegahan kebakaran di lingkungan industri, hutan, dan pemukiman. Di sektor industri, kebakaran dapat mengakibatkan kerugian finansial yang besar, sementara kebakaran hutan dapat menyebabkan kerusakan ekosistem yang luas serta dampak kesehatan dari asap. Di pemukiman, kebakaran bisa mengakibatkan kehilangan tempat tinggal dan jiwa. Oleh karena itu, sistem deteksi dini yang efektif dan andal menjadi krusial dalam upaya mitigasi bencana kebakaran. Sistem deteksi dini dapat mengurangi dampak kebakaran hingga 40% jika diimplementasikan dengan benar [1].

Dalam lima tahun terakhir, Indonesia telah mengalami berbagai musibah kebakaran yang menyebabkan kerugian besar. Berdasarkan data dari Badan Nasional Penanggulangan Bencana (BNPB), kebakaran hutan dan lahan (karhutla) telah menghanguskan jutaan hektar lahan setiap tahunnya, dengan puncaknya terjadi pada tahun 2019 yang menghanguskan lebih dari 1,6 juta hektar. Di sektor pemukiman, menurut data Dinas Pemadam Kebakaran DKI Jakarta, terdapat lebih dari 1.500 kejadian kebakaran setiap tahunnya di Jakarta saja, yang mayoritas disebabkan oleh hubungan arus pendek listrik. Data ini menunjukkan betapa pentingnya deteksi dini untuk mengurangi dampak dari kebakaran di berbagai sektor. Kebakaran di Jakarta menyebabkan kerugian ekonomi sebesar lebih dari Rp 500 miliar setiap tahunnya [2].

Seiring dengan perkembangan teknologi, berbagai pendekatan baru telah diusulkan untuk mendeteksi kebakaran secara otomatis dan lebih cepat. Salah satu tren yang berkembang pesat adalah penggunaan teknik *computer vision* dalam deteksi kebakaran. Teknologi ini memungkinkan sistem untuk mengenali api melalui analisis gambar secara *real-time*, sehingga memungkinkan respon yang lebih cepat dibandingkan metode tradisional. Penelitian menunjukkan bahwa teknik *computer vision* dapat digunakan untuk mendeteksi api dengan akurasi yang tinggi, terutama ketika dikombinasikan dengan teknik *machine learning* dan *deep learning* [3][4], [5]. Hal ini didukung oleh penelitian lainnya yang menemukan bahwa integrasi teknik *deep learning* dengan *computer vision* dapat meningkatkan akurasi deteksi api hingga 95% [6].

Salah satu algoritma yang banyak digunakan dalam deteksi objek, termasuk api, adalah *Single Shot Multibox Detector* (SSD). Algoritma ini dikenal karena kemampuannya mendeteksi objek dalam waktu nyata dengan tingkat akurasi yang baik. SSD bekerja dengan membagi gambar menjadi grid dan memprediksi *bounding box* serta klasifikasi untuk setiap grid. Dalam konteks deteksi kebakaran, SSD dapat digunakan untuk mendeteksi nyala api secara cepat dan akurat, bahkan dalam kondisi yang kompleks seperti kebakaran hutan. Penelitian sebelumnya menunjukkan bahwa SSD dapat diimplementasikan secara efektif untuk mendeteksi api dalam berbagai kondisi pencahayaan dan latar belakang yang berbeda [7]. Selain itu, penelitian terbaru menunjukkan bahwa kombinasi SSD dengan teknik augmentasi data dapat lebih meningkatkan ketepatan deteksi pada kondisi lingkungan yang bervariasi [8][9].

Penggunaan metode SSD dalam deteksi objek, termasuk deteksi api, telah menjadi fokus penelitian beberapa peneliti. Penelitian oleh Zheng et al. (2020) menggunakan SSD untuk mendeteksi kebakaran hutan dengan tujuan meningkatkan kecepatan dan akurasi deteksi. Mereka menemukan bahwa SSD mampu mencapai akurasi sebesar 87% dalam mendeteksi api di area hutan yang luas, meskipun dalam kondisi pencahayaan rendah [10]. Penelitian lainnya oleh Nguyen et al. (2021) berfokus pada penggunaan SSD untuk mendeteksi nyala api pada kapal laut, dengan akurasi sebesar 85,7%, meskipun tantangan besar yang dihadapi adalah deteksi di tengah gangguan visual seperti air dan asap [11]. Penelitian lain oleh Safarov et al. (2024) mengeksplorasi penerapan SSD untuk mendeteksi api di area industri yang padat, dengan akurasi 90,8% dalam mendeteksi sumber api kecil [12]. Liu et al. (2016) menggunakan SSD dalam penelitian untuk mendeteksi api dalam skenario kebakaran hutan, mencapai akurasi sebesar 91,5% dengan penggunaan teknik augmentasi data yang membantu model beradaptasi dengan kondisi cuaca dan cahaya yang berubah-ubah [13]. Penelitian oleh Pincott et al. (2022) juga menggunakan SSD dalam mendeteksi kebakaran pada bangunan, dengan hasil akurasi sebesar 88,6%, dan metode ini terbukti efektif dalam mengurangi tingkat false positive [14].

Meskipun berbagai penelitian telah dilakukan dalam bidang deteksi kebakaran menggunakan *computer vision*, masih terdapat beberapa *research gap* yang perlu diatasi. Salah satunya adalah tantangan dalam mendeteksi api di lingkungan yang memiliki banyak *noise* visual, seperti asap tebal atau latar belakang yang kompleks. Penelitian ini berusaha untuk mengisi gap tersebut dengan mengembangkan model SSD yang lebih adaptif terhadap kondisi lingkungan yang beragam. Kebaharuan dari penelitian ini terletak pada optimasi model SSD untuk mendeteksi api dengan lebih baik di berbagai kondisi, serta integrasinya dengan teknik pra-pemrosesan gambar yang lebih canggih untuk meningkatkan akurasi deteksi. Hal ini sejalan dengan temuan terbaru, yang menyarankan penggunaan filter visual tambahan untuk meminimalkan noise pada deteksi api [15].

2. METODOLOGI DAN LITERATUR REVIEW

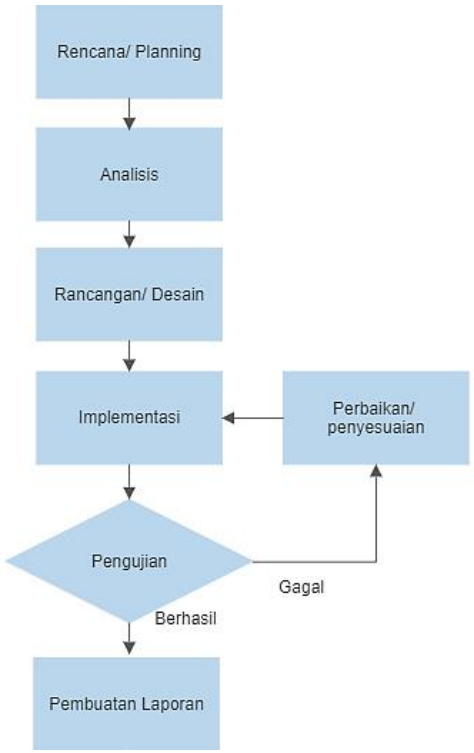
2.1. Metodologi Penelitian

Pendekatan yang digunakan dalam penelitian ini adalah pendekatan kuantitatif Jenis penelitian yang digunakan adalah eksperimen pada penelitian terapan, dalam penelitian terapan sering dilakukan dengan eksperimen dan implementasi, pembuktian prinsip dan mengumpulkan pengalaman dari pengguna.
Rancangan penelitian ini dapat digambarkan sesuai pada gambar 1.

Rancangan penelitian ini melibatkan beberapa tahapan utama yang dapat dilihat pada gambar 1. Tahap pertama adalah rencana/planning, di mana dilakukan pengumpulan data dan alat yang diperlukan untuk merancang sistem deteksi api berbasis computer vision menggunakan algoritma SSD, termasuk penggunaan PC, web camera, Anaconda, Python, dan TensorFlow, serta studi literatur terkait. Tahap kedua, analisis, melibatkan kajian teknis dan permasalahan melalui tinjauan literatur untuk memahami gambaran sistem yang akan dibangun. Selanjutnya, tahap rancangan/desain meliputi pembuatan dataset, pelatihan model, dan

konversi hasil pelatihan ke TensorFlow Lite untuk meningkatkan efisiensi. Sistem menggunakan kamera yang mendeteksi api berdasarkan tingkat rasio objek api terhadap frame video, dengan berbagai level peringatan.

Pada tahap implementasi, sistem diuji coba secara langsung di lokasi tertentu, misalnya di ruang tamu dengan satu kamera terpasang. Tahap pengujian dilakukan untuk memastikan fungsi sistem, termasuk analisis jarak, pencahayaan, dan perbandingan antara penggunaan rule RGB dan YCbCr. Evaluasi kinerja model diukur dengan metrik seperti presisi, recall, F1-score, dan confusion matrix. Terakhir, tahap pembuatan laporan mencakup dokumentasi seluruh proses penelitian dalam bentuk laporan resmi yang diserahkan ke pihak kampus.



Gambar 1. Rancangan Penelitian

Metode Pengumpulan Data diantaranya observasi yang dilakukan dengan mengumpulkan data dan informasi yang dibutuhkan dalam merancang dan membangun sistem deteksi api berbasis computer vision menggunakan algoritma SSD dalam hal ini dalam membangun sistem ini peneliti akan melakukan observasi terhadap sistem yang dibangun dan melakukan pencatatan secara sistematis untuk kemudian dilaporkan dalam laporan hasil penelitian. Eksperimen atau uji coba dari sistem yang dibangun akan dilakukan secara langsung dan hasilnya akan dilakukan pencatatan secara sistematis agar diperoleh data-data yang dapat menunjukkan efektivitas sistem yang dibangun, hasil dari uji coba tersebut akan dilaporkan dalam laporan hasil penelitian.

2.2. Algoritma Single Shot Multibox Detector (SSD)

Algoritma Single Shot Multibox Detector (SSD) merupakan salah satu metode deteksi objek yang populer dalam berbagai aplikasi, termasuk pendeteksian kebakaran berbasis visi komputer. SSD dirancang untuk mendeteksi objek dalam satu langkah (single shot) dengan memprediksi bounding box dan kelas objek secara langsung dari gambar input. Keunggulan SSD terletak pada kecepatan dan akurasinya, yang membuatnya sangat cocok untuk aplikasi real-time seperti sistem keamanan dan pengawasan.

SSD bekerja dengan memanfaatkan beberapa lapisan konvolusi dalam jaringan saraf tiruan atau Convolutional Neural Network (CNN) untuk menghasilkan prediksi pada berbagai skala objek. Setiap lapisan menghasilkan kumpulan bounding box dengan keyakinan tertentu, sehingga memungkinkan deteksi objek dengan ukuran yang bervariasi dalam satu gambar. Dibandingkan dengan metode seperti R-CNN dan YOLO, SSD menawarkan keseimbangan yang baik antara akurasi dan kecepatan, menjadikannya pilihan ideal untuk sistem yang membutuhkan respons cepat, seperti pendeteksian kebakaran.

Pada arsitektur SSD MobileNetV2, yang digunakan dalam penelitian ini, input gambar dengan dimensi 300x300x3 diproses melalui backbone MobileNetV2 untuk mengekstraksi fitur. Fitur ini kemudian diproses melalui lapisan tambahan (SSD extra feature layers) yang secara bertahap mengurangi dimensi fitur:

38x38x512, 19x19x1024, 10x10x512, hingga 1x1x256. Pada tahap akhir, bounding box dan klasifikasi objek dihasilkan, dan Non-Maximum Suppression (NMS) digunakan untuk mengeliminasi prediksi yang tumpang tindih, menghasilkan deteksi akhir.

Keunggulan utama arsitektur ini adalah efisiensi komputasi dan latensi rendah, yang sangat penting dalam aplikasi berbasis perangkat dengan sumber daya terbatas, seperti perangkat mobile. Menurut Liu et al. (2016), arsitektur SSD yang menggunakan backbone ringan seperti MobileNetV2 mampu mencapai kinerja yang kompetitif dengan kecepatan lebih tinggi dibandingkan arsitektur yang lebih kompleks seperti ResNet. Selain itu, kombinasi antara MobileNetV2 dan SSD memungkinkan deteksi real-time dengan akurasi tinggi, yang sangat relevan untuk pendeteksian kebakaran berbasis kamera [16].

2.3. Rule RGB dan Deteksi Kebakaran

Ruang warna Red Green Blue (RGB) adalah model warna aditif yang menggunakan tiga komponen utama: merah (Red), hijau (Green), dan biru (Blue). Dalam deteksi kebakaran, Rule RGB memanfaatkan dominasi intensitas warna merah pada api. Piksel api biasanya memiliki nilai komponen merah (R) yang lebih tinggi dibandingkan komponen hijau (G) dan biru (B). Hal ini memungkinkan identifikasi area api secara efektif.

Celik et al. (2009) mengembangkan pendekatan berbasis Rule RGB yang menunjukkan efektivitas dalam mendeteksi api pada berbagai kondisi pencahayaan [17]. Pendekatan ini didasarkan pada karakteristik distribusi warna api yang khas di ruang warna RGB, dengan dominasi warna merah dan kuning. Penelitian ini menunjukkan bahwa Rule RGB cukup andal untuk deteksi kebakaran pada kondisi cahaya alami. Munshi (2021) menyempurnakan metode ini dengan menambahkan pra-pemrosesan gambar untuk mengurangi noise, yang secara signifikan meningkatkan akurasi deteksi api, bahkan dalam kondisi visual yang menantang[18].

Penggunaan ruang warna merupakan teknik penting dalam deteksi kebakaran berbasis visi komputer, yang memanfaatkan karakteristik warna api untuk memisahkan area api dari latar belakang. Dua ruang warna yang sering digunakan adalah RGB dan YCbCr, masing-masing memiliki keunggulan tersendiri.

2.4. Rule YCbCr

Ruang warna YCbCr memisahkan informasi warna (chrominance) dari kecerahan (luminance), sehingga lebih tahan terhadap perubahan pencahayaan. Komponen Y mewakili kecerahan, sedangkan Cb dan Cr merepresentasikan perbedaan warna biru dan merah. Dalam deteksi kebakaran, Rule YCbCr digunakan untuk memisahkan piksel yang mengandung warna api berdasarkan nilai Cb dan Cr yang khas. Secara keseluruhan, kombinasi Rule RGB dan YCbCr sering digunakan untuk meningkatkan akurasi deteksi kebakaran. Rule RGB berguna untuk mendeteksi warna dominan api, sementara Rule YCbCr membantu mengatasi tantangan pencahayaan yang kompleks. Kombinasi ini memberikan pendekatan yang lebih andal untuk mendeteksi kebakaran dalam berbagai kondisi lingkungan, menjadikannya bagian integral dalam sistem pendeteksian kebakaran berbasis visi komputer.

2.5. Penelitian Terdahulu

Melihat dari sejumlah judul dan tema yang berkaitan dengan penelitian yang dilakukan, diperoleh persamaan dan perbedaan dari penelitian yang akan dilakukan. Tujuannya membuktikan bahwa penulisan penelitian asli dan bukan duplikasi dari tugas akhir penelitian lain, seperti table 1

Tabel 1. Penelitian Terdahulu

No	Peneliti, Judul dan Tahun	Persamaan	Perbedaan
1	Pu Li & Wangda Zhao (2022) *Image fire detection algorithms based on convolutional neural networks*	Sama - sama fokus pada deteksi kebakaran dan menggunakan algoritma dengan dasar CNN.	Penelitian ini membandingkan beberapa algoritma CNN, dengan YOLO v3 sebagai yang terbaik, sementara penelitian kami fokus pada optimasi SSD untuk deteksi kebakaran dalam berbagai kondisi lingkungan.
2	Mohamed Chetoui & Moulay A. Akhloufi (2024) *Fire and Smoke Detection Using Fine-Tuned YOLOv8 and YOLOv7 Deep Models*	Kedua penelitian menggunakan metode *deep learning* untuk deteksi kebakaran berbasis gambar, dengan fokus pada pengembangan dan optimalisasi model deteksi objek.	Penelitian ini menggunakan dan membandingkan beberapa versi YOLO (v6, v7, v8) serta model lain, sementara penelitian kami berfokus pada penggunaan algoritma SSD untuk deteksi kebakaran.
3	Arafat Islam & Md. Imtiaz Habib (2023) *Fire Detection from Image and Video Using YOLOv5*	Kedua penelitian bertujuan meningkatkan akurasi deteksi kebakaran berbasis *deep learning* dan menargetkan deteksi waktu nyata. Keduanya juga mengandalkan pengembangan model deteksi objek untuk mencapai hasil yang lebih baik.	Penelitian ini berfokus pada peningkatan model YOLOv5, khususnya untuk deteksi target api kecil dalam situasi kompleks, sementara penelitian kami menggunakan algoritma SSD untuk deteksi api.
4	Muhammad Salman Farhan, Febryanti Sthevanie & Kurniawan Nur Ramadhani (2022)	Kedua penelitian sama-sama menggunakan pendekatan *deep learning* dan model deteksi objek untuk deteksi api berbasis visi komputer. Keduanya	Penelitian ini menggunakan model YOLOv4, sedangkan penelitian kami menggunakan algoritma SSD. Meskipun sama-sama berbasis visi komputer, penelitian kami fokus pada

No	Peneliti, Judul dan Tahun	Persamaan	Perbedaan
	Video Based Fire Detection Method Using CNN and YOLO Version 4	juga bertujuan meningkatkan akurasi dan kecepatan deteksi untuk penerapan waktu nyata.	penggunaan SSD, yang berbeda dari YOLOv4 dalam hal pendekatan dan arsitektur deteksi objek.
5	Aum Shiva Rama Bishoyi, Raghav Goel, Vansh Batra, Kiran Thomas Jacob, Shashwat Agarwal, Sriram M, Chunduru Sri Abhijit, and Rohith G (2022) *A Deep Learning approach for fire object detection in Autonomous vehicles*	Keduanya menggunakan *deep learning* untuk mendeteksi api secara *real-time* dan menekankan pentingnya akurasi dan kecepatan dalam deteksi objek untuk mencegah kebakaran.	Penelitian ini berfokus pada penerapan deteksi api dalam konteks kendaraan otonom, sedangkan penelitian kami lebih fokus pada algoritma SSD dalam deteksi api berbasis visi komputer tanpa keterkaitan dengan aplikasi kendaraan otonom.
6	Fathorazi Nur Fajri, Syaiful & Wahyu Galih Priambodo (2024) *Fire and Smoke Object Detection Using Mask R-CNN*	Keduanya menggunakan teknologi *computer vision* untuk deteksi api dengan model *deep learning*, dan berfokus pada pengembangan akurasi deteksi.	Penelitian ini menggunakan metode Mask R-CNN dan dataset yang lebih besar serta lebih terfokus pada kebakaran hutan, sedangkan penelitian kami menggunakan algoritma SSD dan memiliki konteks aplikasi yang berbeda seperti deteksi api dalam ruangan.
7	Wu, H., Wu, D., & Zhao, J. (2019) *An intelligent fire detection approach through cameras based on computer vision methods*	Keduanya fokus pada penggunaan teknologi *computer vision* untuk deteksi kebakaran dengan tujuan meningkatkan respons cepat dan akurasi deteksi.	Penelitian ini menggunakan pendekatan berbasis kamera video dengan tiga langkah deteksi, sedangkan penelitian kami fokus pada algoritma SSD untuk deteksi objek dalam konteks deteksi api. Penelitian ini juga lebih terfokus pada aplikasi industri kimia dan petrolium, sementara penelitian kami dapat mencakup konteks yang lebih luas atau berbeda.
8	Rahayu, E., Isnomo, Y. H. P., & Anshori, Moh. A. (2023) *Automatic Early Warning System Design with Firefighter Synchronization Based on Internet of Things (IoT)*	Kedua penelitian berfokus pada peningkatan deteksi kebakaran untuk mengurangi kerusakan akibat kebakaran dengan pendekatan teknologi terbaru.	Penelitian ini menggunakan sensor suhu dan gas untuk mendeteksi indikasi awal kebakaran, sementara penelitian kami memanfaatkan algoritma deteksi objek berbasis *deep learning* (SSD) untuk mendeteksi api melalui video. Sistem dalam penelitian ini lebih menekankan pada deteksi berbasis sensor dan notifikasi awal, sedangkan penelitian kami berfokus pada analisis gambar dan deteksi objek dalam konteks video.
9	Lee, H., Kang, S., & Chung, K. (2022) *Object Detection with Dataset Augmentation for Fire Images Based on GAN*	Kedua penelitian bertujuan untuk meningkatkan akurasi deteksi kebakaran dengan menggunakan teknologi canggih.	Penelitian ini fokus pada augmentasi dataset dengan menghasilkan gambar baru menggunakan GAN untuk melatih model deteksi kebakaran, sementara penelitian kami menggunakan algoritma SSD untuk mendeteksi api langsung dari video. Penelitian ini mengatasi kekurangan dataset dan anotasi, sedangkan penelitian kami fokus pada penerapan dan evaluasi algoritma deteksi api berbasis *deep learning*.
10	Daoud, Z., ben Hamida, A., & ben Amar, C. (2024) *Fire Object Detection and Tracking Based on Deep Learning Model and Kalman Filter*	Kedua penelitian bertujuan meningkatkan akurasi deteksi kebakaran dengan menggunakan teknologi deteksi canggih berbasis *deep learning*.	Penelitian ini fokus pada kombinasi detektor YOLOv3 dan pelacak Kalman untuk meningkatkan akurasi dan mengurangi *false positives*, sedangkan penelitian kami menggunakan SSD untuk mendeteksi kebakaran secara langsung dari video dan menggunakan rule RGB dan YCbCr. Penelitian ini menekankan pada pemrosesan dan pelacakan waktu nyata, sedangkan penelitian kami lebih fokus pada penerapan dan evaluasi performa SSD dalam deteksi api.

3. HASIL DAN ANALISIS

3.1. Hasil Perancangan

Sistem pendeteksi api ini dirancang melalui beberapa tahap yaitu perancangan sistem, pebuatan dataset, *training* dataset dan implementasi model dari hasil *training* dataset. Dari beberapa tahap tersebut, telah dihasilkan tujuan yang ingin dicapai yaitu Sistem deteksi api berbasis computer vision menggunakan algoritma SSD.

3.2. Perancangan Sistem

Perancangan sistem ini melibatkan beberapa komponen perangkat keras dan perangkat lunak yang saling terintegrasi. Untuk perangkat keras, sistem ini hanya memerlukan sebuah komputer yang terhubung dengan kamera, yang berfungsi untuk menangkap data gambar atau video. Sedangkan untuk perangkat lunak, sistem ini menggunakan beberapa aplikasi yang memiliki peran penting dalam pengembangan dan pengoperasian sistem. Sistem operasi seperti Windows, Macintosh, atau Linux digunakan sebagai penghubung antara perangkat keras dan perangkat lunak aplikasi. Untuk mendukung pengembangan machine learning, digunakan Anaconda sebagai distribusi Python/R dengan manajemen paket dan lingkungan yang

baik. Bahasa pemrograman Python juga digunakan karena memiliki ekosistem yang luas dan mendukung berbagai aplikasi. TensorFlow, pustaka pembelajaran mesin yang digunakan, memungkinkan pembuatan dan pelatihan model AI dengan skalabilitas yang tinggi. Selain itu, OpenCV digunakan sebagai pustaka untuk aplikasi pengolahan gambar dan video, yang mendukung fitur komputer visi dalam sistem ini. Semua perangkat lunak ini bekerja secara sinergis untuk mendukung kinerja sistem yang optimal.

3.3. Pembuatan Dataset

Pada tahap pembuatan dataset untuk klasifikasi, digunakan platform Roboflow untuk mempermudah proses anotasi, preprocessing, dan augmentasi data. Dataset yang digunakan berawal dari 1.295 gambar visualisasi kebakaran yang dikumpulkan dari berbagai sumber dengan tujuan untuk mencakup beragam situasi kebakaran, yang nantinya akan memperkuat kemampuan model dalam mendeteksi api dalam berbagai konteks. Proses anotasi dimulai dengan memberi label atau tanda pada gambar untuk menunjukkan area yang relevan, yaitu api, menggunakan bounding box yang mengelilingi area api pada gambar. Keakuratan anotasi menjadi faktor penting karena akan mempengaruhi seberapa baik model dapat mengenali api pada gambar yang berbeda.

Setelah anotasi selesai, tahap berikutnya adalah preprocessing, di mana gambar-gambar diubah ukurannya menjadi 320x320 piksel sesuai dengan kebutuhan model SSD MobileNetV2. Model ini memerlukan input gambar dengan ukuran tertentu untuk menjalankan deteksi objek secara cepat dan efisien. Selain itu, untuk memperkaya dataset dan mengurangi risiko overfitting, dilakukan augmentasi data. Dengan augmentasi, gambar-gambar asli dimodifikasi untuk menciptakan variasi tanpa mengubah labelnya. Teknik augmentasi seperti flip horizontal dan vertikal, rotasi dengan sudut antara -15° hingga +15°, serta penambahan blur hingga 1,5 piksel digunakan untuk mensimulasikan kondisi nyata yang dapat ditemui model saat diimplementasikan, sehingga meningkatkan kemampuan model untuk bekerja dengan baik pada data baru.

Setelah proses augmentasi selesai, jumlah gambar dalam dataset bertambah menjadi 3.185 gambar. Dataset yang telah diproses kemudian diunduh dalam format Pascal VOC dan disiapkan untuk digunakan pada tahap pelatihan model di Google Colab. Pembuatan dataset ini, yang mencakup anotasi, preprocessing, dan augmentasi, merupakan tahap krusial dalam pengembangan model deteksi kebakaran. Kualitas dan keberagaman data yang digunakan akan sangat mempengaruhi performa akhir model dalam mendeteksi kebakaran pada gambar.

3.4. *Training* Dataset

Pelatihan model ini bertujuan untuk mendeteksi objek api menggunakan arsitektur SSD MobileNetV2, dengan pendekatan yang dioptimalkan sesuai keterbatasan komputasi yang tersedia. Proses ini dimulai dengan mempersiapkan lingkungan di Google Colab, yang dipilih karena menyediakan GPU untuk mempercepat komputasi. Dataset yang digunakan terdiri dari 3.185 gambar hasil augmentasi yang telah diproses sebelumnya. Dataset ini diunggah ke Google Colab dan dimasukkan ke dalam *pipeline* pelatihan untuk melatih model dalam mendeteksi objek api secara akurat.

Tabel 2. Pembagian data *training*, validasi dan test

Total *Images*	*Training* (80%)	Validasi (10%)	Test (10%)
3.185	2.548	318	319

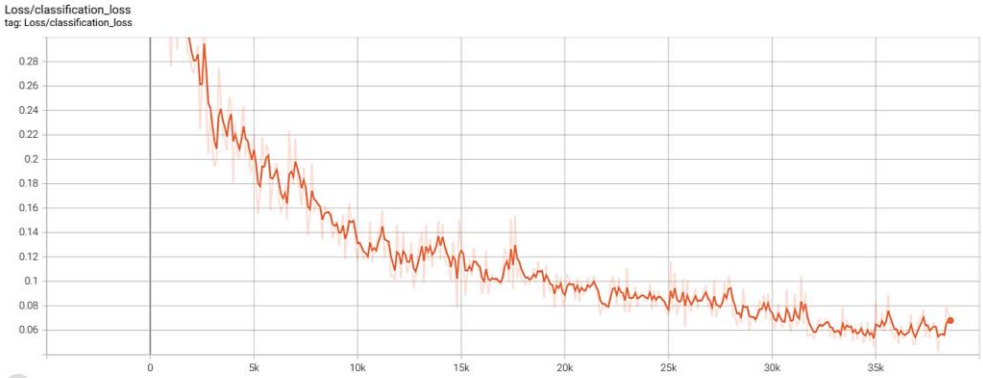
Tahap pertama melibatkan pengimporan berbagai *library* penting seperti TensorFlow, yang merupakan framework utama dalam *machine learning* dan *deep learning*, serta beberapa *library* lainnya yang diperlukan untuk manipulasi data dan *preprocessing*. Setelah itu, dataset yang sudah disiapkan dimuat dan disesuaikan dengan *pipeline* pelatihan. Gambar dalam dataset di-*preprocess* sesuai dengan *input* model SSD MobileNetV2, yaitu dengan mengubah ukuran gambar menjadi 320x320 piksel, sesuai dengan konfigurasi arsitektur model.

Tabel 3. Hyperparameter Training Dataset

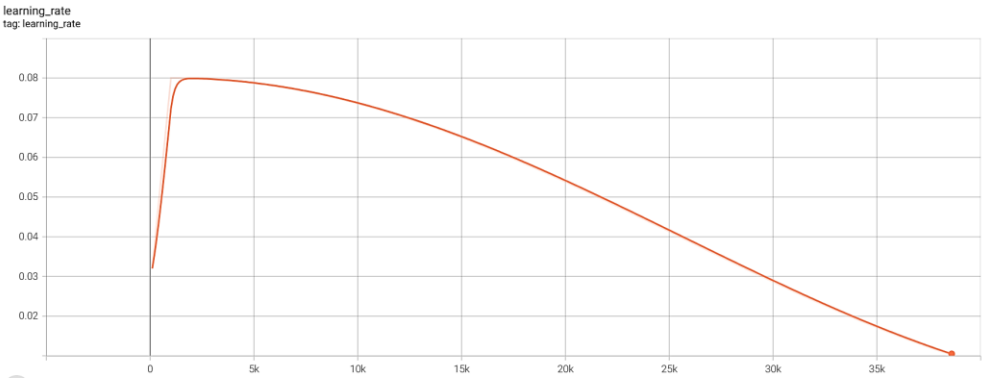
Jumlah Kelas	*Input Size*	*Batch Size*	*Steps*	*Learning Rate*	*Epoch*	Fungsi Aktivasi
1	320x320	16	40.000	0.08 (base), 0.026666 (warmup learning rate)	Tidak menggunakan epoch, tetapi menggunakan steps	ReLU_6 (Rectified Linear Unit 6)

Proses pelatihan model menggunakan metode *steps*, di mana model dilatih berdasarkan sejumlah langkah tertentu, bukan berdasarkan jumlah *epoch*. Setiap langkah mencakup beberapa iterasi gambar dari dataset, yang digunakan untuk memperbarui bobot model. Selama pelatihan, model mengambil *batch* data,

melakukan *forward pass* untuk mendapatkan prediksi, lalu menghitung *loss* berdasarkan perbedaan antara prediksi dan label sebenarnya. Setelah itu, dilakukan *backpropagation* untuk memperbarui bobot model. Selama pelatihan, metrik penting seperti akurasi dan loss dipantau untuk memastikan model belajar dengan baik. Hyperparameter Training Dataset yang dihunakan dapat dilihat pada tabel 3.



Gambar 2. Grafik *Loss Training* Dataset

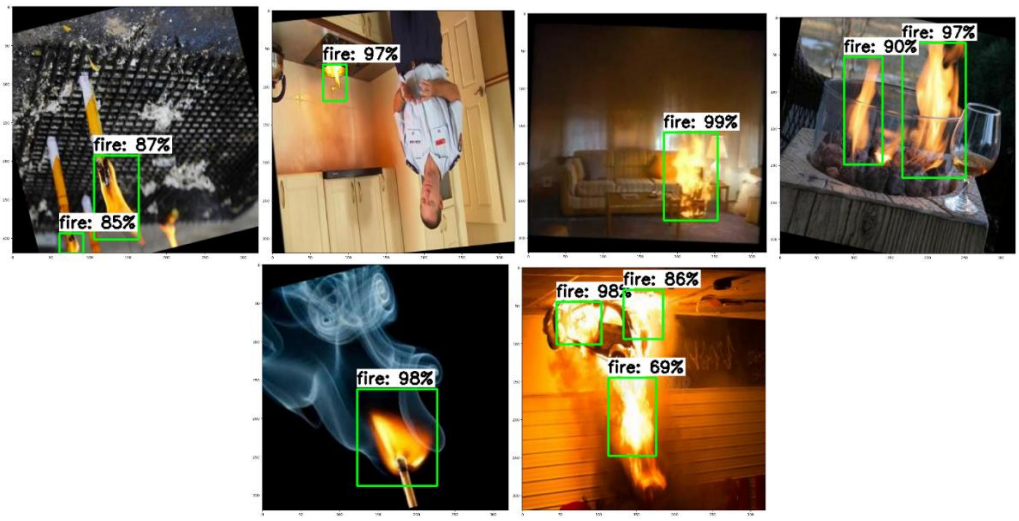


Gambar 3. Grafik *Learning Rate Training*

Tabel 4. Hasil Training Loss dan Learning Rate

No	Steps	Classification Loss	Localization Loss	Regularization Loss	Total Loss	Learning Rate
1	5.000	0,22050	0,20780	0,16310	0,59140	0,07869
2	10.000	0,11640	0,11000	0,16150	0,38790	0,07352
3	15.000	0,13130	0,09120	0,15020	0,37270	0,06494
4	20.000	0,08530	0,04890	0,13650	0,27070	0,05381
5	25.000	0,07234	0,03912	0,12370	0,23516	0,04128
6	30.000	0,07105	0,03746	0,11320	0,22171	0,02862
7	35.000	0,08269	0,02843	0,10560	0,21672	0,01712
8	40.000	0,06676	0,03010	0,10220	0,19906	0,01022

Hasil pelatihan model yang ditampilkan dalam Tabel 4 mencakup metrik penting seperti Classification Loss, Localization Loss, Regularization Loss, Total Loss, dan Learning Rate pada berbagai tahap langkah pelatihan (steps). Classification Loss melatih model untuk mengklasifikasikan jenis objek target, sementara Localization Loss berperan dalam menyesuaikan prediksi bounding box dengan nilai sebenarnya agar lebih presisi. Regularization Loss digunakan untuk mencegah overfitting, meningkatkan kemampuan generalisasi model terhadap data baru. Total Loss, yang merupakan rata-rata dari ketiga loss tersebut, terus menurun seiring bertambahnya langkah pelatihan, menunjukkan peningkatan kinerja model dalam mempelajari data. Selain itu, Learning Rate, yang mengatur besarnya perubahan parameter selama pelatihan, secara bertahap menurun untuk mendukung stabilitas pembaruan parameter, terutama pada tahap akhir pelatihan, sehingga membantu mencapai hasil yang lebih optimal. Gambar 2 dan 3 merupakan grafik dari *Loss Training* Dataset dan Grafik *Learning Rate Training.*



Gambar 4. Hasil Test Dataset

Setelah sejumlah langkah pelatihan selesai, model yang terlatih disimpan dalam format yang siap digunakan untuk proses inferensi pada gambar baru. Model yang disimpan ini kemudian diuji pada dataset validasi untuk mengevaluasi kinerjanya. Kemudian dihitung mAP untuk melihat kinerja *training* dataset, dalam hal ini nilai mAP nya adalah 66,76%. Pada gambar 4 merupakan hasil test dataset.

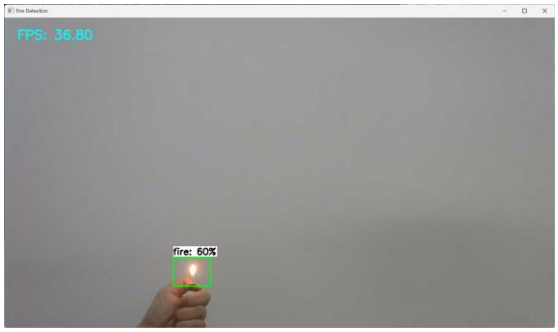
3.5. Implementasi Model

Implementasi model deteksi api berbasis TFLite ini dilakukan dengan tujuan agar model yang telah dilatih pada Google Colab dapat dijalankan secara lokal, serta dioptimalkan untuk perangkat *edge computing*. *Edge computing* merupakan konsep di mana pemrosesan data dilakukan lebih dekat ke sumber data, seperti pada perangkat *Internet of Things* (IoT) atau sensor, sehingga mempercepat respons dan mengurangi latensi. Konversi model ke dalam format TensorFlow Lite (TFLite) bertujuan agar model dapat berjalan dengan lebih efisien pada perangkat yang memiliki sumber daya terbatas, seperti *smartphone*, Raspberry Pi, atau perangkat *edge* lainnya.

Dalam proses konversi, model TensorFlow yang awalnya digunakan untuk pelatihan di Google Colab diubah menjadi model TFLite menggunakan TensorFlow Lite Converter. Proses ini melibatkan optimasi ukuran model dan komputasi agar sesuai dengan kemampuan perangkat yang lebih kecil. Setelah konversi selesai, model TFLite ini dapat diintegrasikan ke dalam aplikasi lokal yang menggunakan bahasa pemrograman Python dan pustaka OpenCV untuk mendeteksi api secara *real-time* dari *input* video, seperti kamera *webcam*.

Pada implementasinya, kode program dimulai dengan impor berbagai pustaka yang dibutuhkan, seperti cv2 dari OpenCV untuk pemrosesan gambar, numpy untuk operasi numerik, dan argparse untuk menangani argumen masukan. Pustaka tflite_runtime.interpreter digunakan untuk memuat dan menjalankan model TFLite yang sudah dikonversi.

Untuk menjalankan deteksi api, aplikasi ini memanfaatkan *VideoStream class* yang dirancang untuk menangani *streaming* video dari kamera secara efisien. Video *frame* yang diambil dari kamera kemudian diproses dengan model TFLite untuk mendeteksi objek yang sesuai dengan klasifikasi api. Pada proses deteksi ini, model akan mengidentifikasi *bounding box* atau kotak pembatas di sekitar area api yang terdeteksi, serta memberikan skor keyakinan (*confidence score*) yang menunjukkan seberapa yakin model terhadap hasil deteksi tersebut.



Gambar 5. Sistem Deteksi Api Berbasis *Computer Vision* Menggunakan Algoritma SSD

Selain itu, dalam kode program ini juga disediakan fungsi untuk mendeteksi api berdasarkan aturan warna menggunakan ruang warna RGB dan YCbCr. Ruang warna ini digunakan karena api memiliki karakteristik warna tertentu yang dapat dikenali, seperti warna merah-oranye pada RGB dan nilai chroma pada YCbCr. Kombinasi deteksi berbasis warna dan deteksi berbasis model *deep learning* ini bertujuan untuk meningkatkan akurasi dalam mengidentifikasi api. Sistem Deteksi Api Berbasis *Computer Vision* Menggunakan Algoritma SSD dapat dilihat pada gambar 5.

Tabel 5. Ruang Warna Api pada Rule RGB dan YcbCr

Warna	Ruang Warna	Batas Bawah	Batas Atas
Api	RGB	[0, 0, 100]	[100, 100, 255]
	YCbCr	[0, 135, 85]	[255, 180, 135]

Tabel 5 menunjukkan batasan ruang warna yang digunakan untuk mendeteksi api berdasarkan aturan Rule RGB dan YCbCr. Pada ruang warna RGB (Red, Green, Blue), deteksi api dilakukan jika nilai komponen merah (R) berada dalam rentang tinggi, dimulai dari nilai minimal 100, sementara komponen hijau (G) dan biru (B) memiliki nilai yang relatif rendah hingga menengah. Batas bawah [0, 0, 100] menunjukkan bahwa api teridentifikasi saat intensitas warna merah lebih dominan dibandingkan hijau dan biru. Sedangkan batas atas [100, 100, 255] memungkinkan nilai merah mencapai intensitas maksimum (255), dengan tetap menjaga nilai hijau dan biru dalam rentang yang lebih kecil. Sebaliknya, pada ruang warna YCbCr (Luminance, Chrominance-Blue, Chrominance-Red), deteksi dilakukan berdasarkan pemisahan kecerahan (Y) dan komponen warna (Cb dan Cr). Batas bawah [0, 135, 85] menunjukkan bahwa api dapat terdeteksi pada kondisi pencahayaan tertentu dengan dominasi chroma warna merah (nilai Cr tinggi). Batas atas [255, 180, 135] memastikan warna api tetap berada dalam rentang yang sesuai, dengan nilai Cr dan Cb yang menggambarkan dominasi warna merah dan minimnya pengaruh warna biru. Pendekatan ini memungkinkan algoritma mendeteksi api secara andal, bahkan dalam kondisi pencahayaan yang bervariasi.

Untuk mencatat hasil deteksi, program ini juga dilengkapi dengan fitur *logging*. Setiap kali terjadi deteksi api yang skornya melebihi ambang batas (*threshold*) tertentu, program akan mencatat informasi seperti FPS (*frame per second*), akurasi, dan waktu deteksi ke dalam file CSV. Log ini dapat digunakan untuk analisis lebih lanjut, misalnya untuk mengevaluasi performa sistem deteksi api dalam kondisi nyata. *Log File* Sistem dapat dilihat pada gambar 6.

	A	B	C	D	E
1	Date	Time	FPS	Acc	Detect Time
2	21/08/2024	22:19:53	41.51	97.45	0.02376
3	21/08/2024	22:19:54	39.71	97.45	0.02291
4	21/08/2024	22:19:54	41.05	97.45	0.02419
5	21/08/2024	22:19:54	39.19	64.63	0.02235
6	21/08/2024	22:19:54	41.99	64.63	0.02382
7	21/08/2024	22:19:54	39.76	88.51	0.02172
8	21/08/2024	22:19:54	43.35	88.51	0.02156
9	21/08/2024	22:19:54	43.74	90.45	0.022
10	21/08/2024	22:19:54	42.84	90.45	0.02198
11	21/08/2024	22:19:54	42.66	90.45	0.02306
12	21/08/2024	22:19:54	40.92	89.49	0.02215

Gambar 6. *Log File* Sistem

3.6. Pengujian Sistem

Dalam pengembangan sistem deteksi kebakaran berbasis visi komputer, evaluasi kinerja sistem sangatlah penting untuk memastikan bahwa sistem mampu mendeteksi kebakaran secara akurat dan efisien dalam berbagai kondisi. Tahap pengujian dilakukan untuk mengukur seberapa efektif model yang telah dilatih dalam mendeteksi api, baik dalam kondisi yang diharapkan maupun dalam skenario yang lebih menantang. Pengujian ini juga bertujuan untuk meminimalkan kemungkinan kesalahan deteksi atau deteksi negatif palsu, yang dapat mengakibatkan peringatan yang salah atau bahkan kegagalan dalam mendeteksi api yang sebenarnya.

Pada pengujian sistem ini, terdapat dua fokus utama yang akan dievaluasi. Pertama, perbandingan kinerja sistem ketika menggunakan aturan warna RGB dan YCbCr dengan kondisi tanpa aturan tersebut, yang mencakup pengujian jarak, pencahayaan, serta deteksi *false negative*. Kedua, pengujian rasio deteksi api terhadap *frame* video, di mana sistem diharapkan mampu memberikan label deteksi yang sesuai dengan tingkat keberadaan api di dalam *frame* video. Pengujian ini diharapkan dapat memberikan gambaran yang jelas tentang sejauh mana sistem mampu beroperasi secara efektif dan dapat diandalkan dalam berbagai situasi.

3.7. Perbandingan menggunakan *Rule* RGB dan YCbCr dan tidak

3.7.1. Uji Jarak

Pada uji jarak, pengujian dilakukan dengan menggunakan lilin sebagai sumber api pada jarak 2 meter, 4 meter, dan 6 meter. Tujuannya adalah untuk mengukur efektivitas sistem dalam mendeteksi api pada berbagai jarak, baik dengan aturan RGB dan YCbCr maupun tanpa aturan tersebut. Hasil pengujian dapat dilihat dari tabel berikut.

Table 6. Hasil Pengujian Dengan Jarak

Rule RGB dan YCbCr	Jarak	Terdeteksi	Rata-rata CS	Rata-rata DT
Ya	2 Meter	Ya	95,52	0,02198
	4 Meter	Ya	88,51	0,02291
	6 Meter	Ya	64,63	0,02382
Tidak	2 Meter	Ya	67,01	0,02374
	4 Meter	Ya	53,61	0,02387
	6 Meter	Tidak	-	-

Hasil pengujian deteksi api dengan berbagai jarak pada tabel 6 menunjukkan perbedaan kinerja yang signifikan antara sistem yang menggunakan aturan RGB dan YCbCr dibandingkan dengan yang tidak menggunakannya. Pada pengujian dengan aturan RGB dan YCbCr, sistem menunjukkan kinerja yang konsisten dalam mendeteksi api, baik pada jarak dekat maupun jarak menengah, dengan tingkat keyakinan yang cukup tinggi dan waktu deteksi yang relatif cepat. Meskipun terjadi penurunan dalam keyakinan deteksi seiring dengan bertambahnya jarak, sistem masih mampu mendeteksi api hingga jarak yang lebih jauh.

Sebaliknya, tanpa penggunaan aturan RGB dan YCbCr, sistem mengalami penurunan performa yang lebih drastis. Pada jarak yang sama, sistem tanpa aturan ini menunjukkan tingkat keyakinan yang lebih rendah dan waktu deteksi yang sedikit lebih lama. Pada jarak yang lebih jauh, sistem bahkan gagal mendeteksi api sama sekali, menandakan ketergantungan yang cukup besar pada aturan RGB dan YCbCr untuk mendukung deteksi yang akurat.

Secara keseluruhan, penggunaan aturan RGB dan YCbCr terbukti meningkatkan efektivitas sistem dalam mendeteksi api pada berbagai jarak, terutama dalam situasi yang lebih menantang seperti pada jarak yang lebih jauh. Tanpa aturan ini, kemampuan deteksi sistem berkurang secara signifikan, terutama pada jarak yang lebih jauh dari sumber api.

3.7.2. Uji Pencahayaan

Selanjutnya, pada uji pencahayaan, sistem diuji dalam kondisi pencahayaan yang bervariasi, yaitu 50-100 lux, 100-200 lux, 200-500 lux, dan 500-1000 lux. Pengujian ini bertujuan untuk melihat apakah sistem tetap dapat mendeteksi api dengan benar pada kondisi pencahayaan yang berbeda-beda, serta untuk membandingkan hasil deteksi antara penggunaan aturan RGB dan YCbCr dengan metode tanpa aturan. Hasil pengujian dapat dilihat dari tabel 7.

Hasil pengujian deteksi api dengan variasi pencahayaan menunjukkan bahwa penggunaan aturan RGB dan YCbCr memberikan hasil yang lebih baik dalam berbagai kondisi pencahayaan dibandingkan dengan yang tidak menggunakannya. Pada pengujian dengan aturan RGB dan YCbCr, sistem berhasil mendeteksi api dalam semua kondisi pencahayaan yang diuji, meskipun terjadi penurunan tingkat keyakinan seiring dengan peningkatan intensitas pencahayaan. Namun, waktu deteksi tetap stabil meskipun pencahayaan meningkat.

Tabel 7. Hasil Pengujian Dengan Pencahayaan

Rule RGB dan YCbCr	Pencahayaan	Terdeteksi	Rata-rata CS	Rata-rata DT
Ya	50 - 100 lux	Ya	98,22	0,02152
	100 - 200 lux	Ya	89,71	0,02251
	200 - 500 lux	Ya	77,21	0,02344
	500 - 1000 lux	Ya	68,53	0,02389
Tidak	50 - 100 lux	Ya	85,82	0,02361
	100 - 200 lux	Ya	77,24	0,02395
	200 - 500 lux	Ya	65,89	0,02421
	500 - 1000 lux	Tidak	-	-

Di sisi lain, tanpa aturan RGB dan YCbCr, sistem juga berhasil mendeteksi api dalam kondisi pencahayaan rendah hingga sedang, tetapi dengan tingkat keyakinan yang lebih rendah dibandingkan dengan penggunaan aturan. Pada intensitas pencahayaan yang lebih tinggi, sistem tanpa aturan ini mulai menunjukkan keterbatasan, hingga akhirnya gagal mendeteksi api pada pencahayaan yang sangat terang.

Secara keseluruhan, hasil ini menegaskan bahwa aturan RGB dan YCbCr membantu sistem untuk tetap akurat dan konsisten dalam mendeteksi api, terutama dalam kondisi pencahayaan yang menantang.

Tanpa aturan ini, sistem menunjukkan penurunan kinerja, terutama dalam kondisi pencahayaan yang lebih terang, di mana deteksi api menjadi tidak akurat atau bahkan gagal sama sekali.

3.7.3. Uji *False Positive*

Kemudian, uji *false positive* dilakukan untuk mengevaluasi apakah sistem menghasilkan deteksi positif palsu ketika diuji dengan objek yang bukan api, seperti cahaya terang, gambar atau video api, serta objek atau warna yang menyerupai api. Jika sistem mendeteksi objek tersebut sebagai api, maka hasil pengujian dianggap gagal. Pengujian ini dilakukan baik dengan menggunakan aturan RGB dan YCbCr maupun tanpa aturan tersebut. Hasil pengujian dapat dilihat dari tabel 8.

Tabel 8. Hasil Pengujian *False Positive*

Rule RGB dan YCbCr	Pengujian	Terdeteksi
Ya	Cahaya	Ya
	Video/ Gambar	Tidak
	Warna	Tidak
Tidak	Cahaya	Ya
	Video/ Gambar	Ya
	Warna	Tidak

Hasil pengujian *False Positive* menunjukkan bahwa penggunaan aturan RGB dan YCbCr memberikan hasil yang lebih baik dalam mengurangi deteksi positif palsu dibandingkan dengan yang tidak menggunakan aturan tersebut. Pada pengujian dengan aturan RGB dan YCbCr, sistem tidak berhasil membedakan dengan baik antara cahaya nyata dan api, tetapi tidak terpengaruh oleh video atau gambar api maupun warna yang mirip dengan api. Ini menunjukkan bahwa aturan tersebut cukup efektif dalam meminimalkan kesalahan deteksi.

Sebaliknya, ketika aturan RGB dan YCbCr tidak digunakan, sistem juga tidak berhasil mendeteksi cahaya dengan benar sebagai bukan api, juga terjadi kesalahan deteksi pada video atau gambar api yang dianggap sebagai api nyata. Hal ini menunjukkan kelemahan sistem tanpa aturan RGB dan YCbCr dalam membedakan antara cahaya dan api juga gambar statis atau video dengan api nyata. Namun, seperti pada pengujian dengan aturan, sistem tetap tidak terpengaruh oleh warna yang mirip api dalam kedua kasus.

Secara keseluruhan, aturan RGB dan YCbCr terbukti lebih andal dalam mengurangi kesalahan deteksi pada skenario yang melibatkan gambar, dan warna yang mirip dengan api, memastikan bahwa hanya api nyata yang terdeteksi oleh sistem.

3.8. Rasio Deteksi Api terhadap *Frame*

Selain itu, dilakukan juga pengujian rasio deteksi api terhadap *frame* video. Pada pengujian ini, proporsi deteksi api dalam setiap frame dikategorikan ke dalam empat tingkatan, yaitu kurang dari 1/10 *frame* yang terdeteksi sebagai api diberi label "*Fire*", antara 1/10 hingga kurang dari 3/10 diberi label "Bahaya Kebakaran", antara 3/10 hingga kurang dari 5/10 diberi label "AWAS Bahaya Kebakaran!!!" dan suara serine putus-putus akan dibunyikan, dan lebih dari atau sama dengan 5/10 *frame* yang terdeteksi sebagai api diberi label "Kebakaran!!!" dan suara serine panjang akan dibunyikan. Pengujian ini hanya dilakukan dengan aturan RGB dan YcbCr, dan keberhasilannya diukur dari seberapa akurat sistem memberikan label yang sesuai dengan kategori rasio deteksi api dalam *frame*. Jika label yang diberikan sesuai dengan proporsi deteksi, maka hasil pengujian dianggap berhasil. Hasil Uji Rasio Deteksi Api Terhadap *Frame* dapat dilihat pada tabel 9.

Tabel 9. Hasil Uji Rasio Deteksi Api Terhadap *Frame*

Rasio *Frame*	Label	Sesuai/ Tidak Sesuai
< 1/10	*fire*	Sesuai
1/10 < 3/10	Bahaya Kebakaran	Sesuai
3/10 < 5/10	AWAS Bahaya Kebakaran!!!	Sesuai
=> 5/10	KEBAKARAN!!!	Sesuai

Hasil pengujian rasio deteksi api terhadap *frame* menunjukkan bahwa sistem telah sesuai dalam memberikan peringatan. Ini menunjukkan bahwa pengaturan pada sistem telah sesuai dengan kondisi yang diinginkan.

3.9. Perhitungan *Confusion Matrix*

Untuk melakukan analisa terhadap hasil pengujian, maka digunakan *Confusion Matrix* agar dapat diketahui akurasi dari sistem yang dibuat. *Confusion Matrix* adalah tabel dengan 4 kombinasi berbeda dari nilai prediksi dan nilai aktual. Terdapat 4 istilah sebagai representasi hasil proses klasifikasi pada *confusion*

matrix dapat dilihat pada tabel 10. (1) *True Positive* (TP), (2) *True Negative* (TN), (3) *False Positive* (FP), dan (4) *False Negative* (FN).

Tabel 10. Parameter *Confusion Matrix*

Nilai Prediksi	Nilai Sebenarnya	
	TRUE	*FALSE*
TRUE	TP (*True Positive*)	FP (*False Positive*)
FALSE	FN (*False Negative*)	TN (*True Negative)*

Dimana TP (*True Positive*) merupakan jumlah data yang dideteksi dengan benar, FP (*False Positive*) merupakan jumlah data yang harusnya dideteksi benar, namun oleh sistem dideteksi salah; FN (*False Negative*) merupakan jumlah data yang harusnya dideteksi salah, namun oleh sistem dideteksi benar; TN (*True Negative*) merupakan jumlah data yang dideteksi salah.

Nilai *recall* menggambarkan keberhasilan model dalam menemukan kembali sebuah informasi, nilai *precision* menggambarkan akurasi antara data yang diminta dengan hasil prediksi yang diberikan oleh model dan nilai *accuracy* menunjukkan seberapa akurat model dalam mengklasifikasikan dengan benar. Dalam hal melihat performa suatu model harus di cocokkan dengan tujuan model itu dibuat, contohnya dalam hal deteksi kebakaran ini tentunya kita menghindari terjadinya *False Negative* untuk itu selain *accuracy* maka *recall* juga akan menjadi acuan kita. Tabel 11 adalah table konversi hasil pengujian proses sistem ke *Confusion Matrix*.

Tabel 11. Konversi hasil pengujian ke *Confusion Matrix*

No	Pengujian	Parameter	Hasil Seharusnya	Hasil Pengujian	Konversi ke *Confusion Matrix*			
					True Positive	*False Positive*	*False Negative*	*True Negative*
1	Jarak	2 Meter	Ya	Ya	√			
2		4 Meter	Ya	Ya	√			
3		6 Meter	Ya	Ya	√			
4	Pencahayaan	50 - 100 lux	Ya	Ya	√			
5		100 - 200 lux	Ya	Ya	√			
6		200 - 500 lux	Ya	Ya	√			
7		500 - 1000 lux	Ya	Ya	√			
8	*True Negative*	Cahaya	Tidak	Ya		√		
9		Video/ Gambar	Tidak	Tidak				√
10		Warna	Tidak	Tidak				√
11	Rasio *Frame*	< 1/10	Ya	Ya	√			
12		1/10 < 3/10	Ya	Ya	√			
13		3/10 < 5/10	Ya	Ya	√			
14		=> 5/10	Ya	Ya	√			

True Positive (TP) = 11
False Positive (FP) = 1
False Negative (FN) = 0
True Negative (TN) = 2
Recall = TP / (TP+FN) = 11 / (11+0) = 11 / 11 = 1
Precision = TP / (TP+FP) = 11 / (11+1) = 11 / 12 = 0,92
Accuracy = (TP+TN) / (TP+TN+FP+FN) = (11 +2) / (11+2+1+0) = 13 / 14 = 0,93

Dapat dilihat hasil perhitungan *confusion matrix* di atas bahwa *accuracy* dari sistem yang dibangun adalah 0,93 dengan nilai *recall* adalah 1 dan *precision* 0,92, hal ini menunjukkan bahwa sistem yang dibuat sudah memiliki *accuracy* yang cukup tinggi dengan nilai *recall* yang sangat baik dan berhasil menghindari terjadinya *false negative* pada sistem.

3.10. Pembahasan

Penelitian ini bertujuan untuk menguji efektivitas penggunaan aturan RGB dan YCbCr dalam mendeteksi api di berbagai kondisi, termasuk pengujian jarak, pencahayaan, dan *false negative*. Hasil dari pengujian ini menunjukkan bahwa penerapan aturan RGB dan YCbCr memberikan hasil yang lebih baik dalam hal akurasi deteksi dibandingkan tanpa aturan tersebut. Pada pengujian jarak, penggunaan aturan RGB dan YCbCr berhasil mendeteksi api dengan akurasi yang lebih tinggi, bahkan pada jarak 6 meter, dibandingkan dengan pengujian tanpa aturan yang gagal mendeteksi api pada jarak yang sama. Demikian pula, dalam pengujian pencahayaan, aturan RGB dan YCbCr memberikan hasil deteksi yang konsisten pada

berbagai tingkat pencahayaan, sementara tanpa aturan, sistem mulai gagal mendeteksi api pada pencahayaan tinggi (500-1000 lux).

Hasil ini sejalan dengan penelitian yang dilakukan oleh Pritam et al. (2017) yang menemukan bahwa penggunaan kombinasi model warna dalam deteksi api secara signifikan meningkatkan akurasi dibandingkan dengan model tunggal. Dalam penelitian Zhang et al., penggunaan model RGB dan YCbCr dalam deteksi api terbukti efektif dalam mengurangi *false positive* yang sering kali disebabkan oleh cahaya terang atau objek berwarna merah [19]. Selain itu, penelitian ini juga konsisten dengan hasil yang diperoleh oleh Lu et al. (2016) yang menunjukkan bahwa integrasi berbagai model warna dapat membantu dalam mengidentifikasi karakteristik api yang lebih kompleks, seperti variasi intensitas cahaya dan jarak [20].

Namun, ada perbedaan signifikan dalam hal kecepatan deteksi (*detection time*). Penelitian sebelumnya, seperti yang dilakukan oleh Gragnaniello et al. (2025), melaporkan bahwa penggunaan model RGB saja dapat memberikan waktu deteksi yang lebih cepat dibandingkan dengan kombinasi model, terutama pada perangkat dengan kemampuan komputasi terbatas [21]. Dalam penelitian ini, penerapan aturan RGB dan YCbCr memang menghasilkan akurasi yang lebih tinggi, namun waktu deteksi sedikit lebih lambat dibandingkan tanpa aturan, khususnya pada jarak yang lebih jauh dan pencahayaan yang lebih tinggi. Perbedaan ini dapat disebabkan oleh kompleksitas tambahan dalam pemrosesan gambar yang diperlukan untuk menerapkan kedua aturan tersebut secara bersamaan.

4. KESIMPULAN

Berdasarkan dari hasil dan pembahasan dari pembuatan Sistem deteksi api berbasis *computer vision* menggunakan algoritma SSD yang telah dilakukan, maka dapat ditarik kesimpulan bahwa penelitian ini berhasil menemukan metode penerapan algoritma *Single Shot Multibox Detector* (SSD) dalam melakukan deteksi api dengan efisien. Melalui penggunaan model SSD MobileNetV2 yang telah dilatih dengan dataset yang terdiri dari gambar-gambar api yang telah di-augmentasi, sistem ini mampu mendeteksi api dalam berbagai kondisi, baik dari segi jarak maupun pencahayaan. Pengujian menunjukkan bahwa penerapan aturan RGB dan YCbCr lebih efektif dalam meningkatkan akurasi deteksi api, terutama dalam kondisi pencahayaan yang bervariasi dan jarak yang lebih jauh. Penggunaan kombinasi model warna ini menjadi solusi optimal dalam mendeteksi ancaman kebakaran secara *real-time*, sekaligus mengurangi *false positive*.

Selain itu implementasi sistem deteksi api yang dibangun dalam penelitian ini terbukti mampu mendeteksi ancaman insiden kebakaran dengan tingkat akurasi yang tinggi. Sistem ini tidak hanya efektif dalam mendeteksi api pada berbagai skenario pengujian, tetapi juga mampu memberikan respons yang cepat dan tepat dalam menentukan level ancaman kebakaran berdasarkan rasio deteksi api terhadap *frame*. Dengan demikian, sistem ini memiliki potensi besar untuk digunakan dalam aplikasi praktis seperti pemantauan kebakaran di lingkungan industri atau area publik, memberikan perlindungan yang lebih baik dan memungkinkan tindakan preventif yang cepat.

Untuk meningkatkan kinerja sistem deteksi api, pengembangan lebih lanjut bisa difokuskan pada penggunaan algoritma deep learning yang lebih kompleks, seperti model Convolutional Neural Networks (CNN) dengan arsitektur yang lebih dalam atau kombinasi beberapa model untuk meningkatkan akurasi deteksi dalam situasi yang lebih variatif. Selain itu, peningkatan pada dataset, baik dalam hal variasi gambar maupun jumlah data, akan membantu model untuk lebih baik dalam generalisasi pada kondisi nyata, sehingga mengurangi tingkat kesalahan deteksi, terutama dalam mengatasi false negative dan false positive.

Pengembangan selanjutnya dapat mengarah pada peningkatan adaptabilitas sistem di berbagai kondisi lingkungan, seperti lingkungan dengan variasi cuaca ekstrem atau lokasi dengan banyak interferensi visual seperti pantulan cahaya atau asap tebal. Selain itu, integrasi sistem dengan teknologi IoT (Internet of Things) dapat meningkatkan fleksibilitas penggunaan sistem di lapangan. Misalnya, sistem deteksi api dapat dihubungkan dengan sensor suhu atau kelembaban untuk memberikan informasi yang lebih komprehensif dan memperkuat sistem deteksi dini. Pengembangan antarmuka pengguna yang lebih ramah, serta penambahan fitur notifikasi real-time melalui aplikasi mobile, akan semakin mempermudah pengguna dalam memantau dan merespons ancaman kebakaran dengan cepat dan efisien.

REFERENSI

[1] E. Rahayu, Y. H. P. Isnomo, and Moh. A. Anshori, "Automatic Early Warning System Design With Firefighter Synchronization Based on Internet of Things (IoT)," *Jurnal Jartel Jurnal Jaringan Telekomunikasi*, vol. 13, no. 1, pp. 103–108, 2023, doi: 10.33795/jartel.v13i1.416.

[2] S. Suhardono *et al.*, "Human activities and forest fires in Indonesia: An analysis of the Bromo incident and implications for conservation tourism," *Trees, Forests and People*, vol. 15, p. 100509, 2024, doi: https://doi.org/10.1016/j.tfp.2024.100509.

[3] A. Bayegizova *et al.*, "Fire detection using deep learning methods," *International Journal of Electrical and Computer Engineering*, vol. 14, no. 1, pp. 547–555, Feb. 2024, doi: 10.11591/ijece.v14i1.pp547-555.

[4] D. Tribuana, H. Hazriani, and A. Latief Arda, "Face recognition for smart door security access with convolutional neural network method," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 22, no. 3, pp. 702–710, Jun. 2024, doi: 10.12928/telkomnika.v22i3.25946.

[5] D. Tribuana, H. Hazriani, and A. Latief Arda, "Image Preprocessing Approaches Toward Better Learning Performance with CNN," *JURNAL RESTI (Rekayasa Sist. Teknol. Inf.)*, vol. 10, no. 1, pp. 1–9, 2024, doi: 10.29207/resti.v8i1.5417.

[6] S. Chitram, S. Kumar, and S. Thenmalar, "Enhancing Fire and Smoke Detection Using Deep Learning Techniques †," *Engineering Proceedings*, vol. 62, no. 1, 2024, doi: 10.3390/engproc2024062007.

[7] H. Zhan, X. Pei, T. Zhang, and L. Zhang, "Research on flame detection method based on improved SSD algorithm," *Journal of Intelligent & Fuzzy Systems*, vol. 45, no. 4, pp. 6501–6512, Oct. 2023, doi: 10.3233/JIFS-232645.

[8] F. Xu, X. Zhang, T. Deng, and W. Xu, "An Image-Based Fire Monitoring Algorithm Resistant to Fire-like Objects," *Fire*, vol. 7, no. 1, Jan. 2024, doi: 10.3390/fire7010003.

[9] Riswanto, A. Ahmad, Hazriani, and D. Tribuana, "Calorie Detection of Traditional Indonesian Food Using the Single Shot Multibox Detector (SSD) Method," *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 4, no. 3, pp. 819–829, 2024, doi: 10.57152/malcom.v4i3.1332.

[10] X. Zheng, F. Chen, L. Lou, P. Cheng, and Y. Huang, "Real-Time Detection of Full-Scale Forest Fire Smoke Based on Deep Convolution Neural Network," *Remote Sens (Basel)*, vol. 14, no. 3, Feb. 2022, doi: 10.3390/rs14030536.

[11] A. Q. Nguyen, H. T. Nguyen, V. C. Tran, H. X. Pham, and J. Pestana, "A Visual Real-time Fire Detection using Single Shot MultiBox Detector for UAV-based Fire Surveillance," in *2020 IEEE Eighth International Conference on Communications and Electronics (ICCE)*, 2021, pp. 338–343. doi: 10.1109/ICCE48956.2021.9352080.

[12] F. Safarov, S. Muksimova, M. Kamoliddin, and Y. I. Cho, "Fire and Smoke Detection in Complex Environments," *Fire*, vol. 7, no. 11, Nov. 2024, doi: 10.3390/fire7110389.

[13] W. Liu *et al.*, "SSD: Single Shot MultiBox Detector," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., in Lecture Notes in Computer Science, vol. 9905. Springer, 2016, pp. 21–37. doi: 10.1007/978-3-319-46448-0_2.

[14] J. Pincott, P. W. Tien, S. Wei, and J. K. Calautit, "Indoor fire detection utilizing computer vision-based strategies," *Journal of Building Engineering*, vol. 61, p. 105154, 2022, doi: https://doi.org/10.1016/j.jobe.2022.105154.

[15] D. Lazzaro, F. Sthevanie, and K. N. Ramadhani, "Enhancing Fire Detection in Images using Faster R-CNN with Gaussian Filtering and Contrast Adjustment," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 7, no. 3, p. 1572, Jul. 2023, doi: 10.30865/mib.v7i3.6486.

[16] W. Liu *et al.*, "SSD: Single Shot MultiBox Detector," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., Cham: Springer International Publishing, 2016, pp. 21–37.

[17] T. Çelik and H. Demirel, "Fire detection in video sequences using a generic color model," *Fire Saf J*, vol. 44, no. 2, pp. 147–158, 2009, doi: 10.1016/j.firesaf.2008.05.005.

[18] A. A. Munshi, "Fire Detection Methods Based on Various Color Spaces and Gaussian Mixture Models," *Advances in Science and Technology Research Journal*, vol. 15, no. 3, pp. 197–214, 2021, doi: 10.12913/22998624/138924.

[19] D. Pritam and J. H. Dewan, "Detection of fire using image processing techniques with LUV color space," in *2017 2nd International Conference for Convergence in Technology (I2CT)*, 2017, pp. 1158–1162. doi: 10.1109/I2CT.2017.8226309.

[20] Q. Lu, J. Yu, and Z. Wang, "A Color Model Based Fire Flame Detection System," in *Communications in Computer and Information Science*, Singapore: Springer Nature, 2016, pp. 474–485. doi: 10.1007/978-981-10-3002-4_40.

[21] D. Gragnaniello, A. Greco, C. Sansone, and B. Vento, "FLAME: fire detection in videos combining a deep neural network with a model-based motion analysis," *Neural Comput Appl*, 2025, doi: 10.1007/s00521-024-10963-z.