

Fire Detection Using Logistic Regression with GLCM, RGB Ratio, RGB Intersection, and Color Moments

Pieter Dickens*¹, Tedy Matius Surya Mulyana²

^{1,2}Informatics, Faculty of Technology and Design, Universitas Bunda Mulia, Indonesia.

Email: ¹pieter.dickens@yahoo.com, ²tmulyana@bundamulia.ac.id

Received : Jan 25, 2025; Revised : Aug 15, 2025; Accepted : Aug 16, 2025; Published : Aug 28, 2025

Abstract

Fires pose a significant threat to human safety and property, particularly in densely populated urban environments where rapid and accurate early detection is critical. This study proposes an automated fire detection system based on computer vision and Logistic Regression classification, utilizing a combination of texture and color-based features to improve detection performance. The proposed approach integrates Gray-Level Co-occurrence Matrix (GLCM), RGB Ratio, RGB Intersection, and Color Moments to extract discriminative features from fire and non-fire images. The dataset, obtained from Kaggle, was preprocessed through HSV-based color segmentation to isolate candidate fire regions before manual annotation. The extracted features were then used to train a Logistic Regression model with hyperparameter tuning of the *max_iter* parameter to achieve optimal convergence. Experimental results show that the proposed model achieved an accuracy of 86% and a recall of 84% on the training dataset, and an accuracy of 87% with a recall of 82% on the test dataset. Despite these promising results, some false negatives were observed, indicating the need for further refinement to improve sensitivity. Comparative evaluation with a Convolutional Neural Network (CNN) demonstrated that the Logistic Regression approach achieved higher average processing speed, reaching up to 16.2 FPS for video input, compared to 11 FPS for CNN, making it more suitable for real-time applications. Overall, the integration of multi-feature extraction with Logistic Regression offers a balance between accuracy and computational efficiency for early fire detection in real-world scenarios.

Keywords : *Color Moments, Fire Detection, GLCM, Logistic Regression, RGB Ratio, RGB Intersection*

This work is an open access article licensed under a Creative Commons Attribution 4.0 International License.



1. PENDAHULUAN

Kebakaran adalah sesuatu yang terjadi akibat adanya api yang tidak ditangani dengan baik dan cepat [1]. Deteksi api kebakaran merupakan aspek kritis dalam menjaga keamanan dan mengurangi risiko kerugian materi dan nyawa. Sepanjang tahun 2023, Dinas Gulkarmat Provinsi DKI Jakarta mencatat bahwa terjadi total 2.286 kejadian kebakaran di wilayah tersebut. Jakarta Timur mencatat jumlah kebakaran tertinggi dengan 594 kejadian, disusul oleh Jakarta Selatan dengan 573 kejadian, Jakarta Barat dengan 484 kejadian, Jakarta Utara dengan 379 kejadian, dan Jakarta Pusat dengan 256 kejadian. Jenis objek yang paling sering terbakar meliputi bangunan perumahan (637 kejadian), instalasi luar gedung (480 kejadian), sampah (267 kejadian), tumbuhan (215 kejadian), kendaraan (118 kejadian), lapak (40 kejadian), bangunan industri (32 kejadian), serta kejadian lainnya sebanyak 156 kali [2].

Berbagai metode telah dikembangkan untuk mendeteksi kebakaran sejak dini, mulai dari penggunaan sensor asap dan panas hingga sistem pengawasan berbasis kamera. Pendekatan berbasis sensor memiliki keterbatasan, seperti sensitivitas yang tinggi terhadap kondisi lingkungan dan ketidakmampuan dalam mendeteksi sumber api pada jarak jauh. Sementara itu, metode berbasis *deep learning*, seperti Convolutional Neural Network (CNN) atau YOLO, menawarkan akurasi tinggi namun membutuhkan sumber daya komputasi yang besar, sehingga kurang optimal untuk implementasi *real-time* pada perangkat dengan keterbatasan *hardware*.

Metode yang efektif untuk mendeteksi kebakaran secara cepat dan akurat sangat diperlukan dalam lingkungan yang padat, seperti gedung perkantoran atau area perkotaan yang padat penduduknya. Salah satu pendekatan yang digunakan adalah menggunakan teknologi citra untuk mendeteksi api secara otomatis. Teknologi ini memanfaatkan analisis citra untuk mengidentifikasi pola visual yang merupakan tanda-tanda kebakaran, seperti cahaya yang intens dan pola warna yang khas. Dalam kasus ini, penggunaan *Logistic Regression* sebagai model klasifikasi dan Algoritma deteksi objek berbasis warna *HSV* untuk segmentasi objek dalam citra dapat meningkatkan akurasi dan kecepatan deteksi kebakaran. Dimana kombinasi kedua pendekatan ini diharapkan dapat memberikan deteksi api kebakaran yang handal dan responsif.

Dibutuhkan alat yang sekaligus bisa mendeteksi api dan mengetahui bagaimana kondisi keadaan saat terjadi kebakaran [1]. Dimana penggunaan *computer vision* di kasus ini diharapkan dapat meningkatkan akurasi pendeteksi api yang telah dikembangkan, karena seperti yang kita ketahui tidak hanya api saja yang memiliki asap dan suhu yang tinggi, tetapi juga fenomena lain seperti proses industri atau aktivitas manusia yang bisa menyerupai tanda-tanda kebakaran. Dengan menggunakan *computer vision*, detektor dapat dilatih untuk mengenali pola-pola visual khusus yang secara konsisten terkait dengan kebakaran, seperti warna api yang khas dan struktur area yang terbakar. Hal ini memungkinkan detector untuk lebih akurat membedakan antara situasi kebakaran sebenarnya dengan kejadian lain yang dapat menimbulkan alarm palsu, seperti debu industri atau proses pemanasan yang normal. Dengan demikian, integrasi *computer vision* dalam pengembangan model deteksi kebakaran diharapkan tidak hanya meningkatkan akurasi dalam mendeteksi kebakaran, tetapi juga mengurangi kemungkinan kejadian yang tidak perlu diprioritaskan atau direspon oleh petugas pemadam kebakaran, sehingga meningkatkan efisiensi dan efektivitas keseluruhan dalam respons terhadap keadaan darurat.

Sejumlah penelitian terdahulu menunjukkan bahwa kombinasi teknik *feature extraction* berbasis tekstur dan warna mampu meningkatkan akurasi dalam klasifikasi citra api. Fitur tekstur seperti Gray-Level Co-occurrence Matrix (GLCM) dapat menggambarkan pola spasial pada citra, sedangkan fitur berbasis warna seperti RGB Ratio, RGB Intersection, dan Color Moments efektif untuk membedakan karakteristik visual api dari objek lain. Namun, penelitian yang memadukan seluruh fitur ini dengan algoritma *machine learning* yang relatif ringan, seperti Logistic Regression, masih terbatas.

Pemilihan Logistic Regression dalam penelitian ini didasarkan pada kemampuannya melakukan klasifikasi biner secara efisien dengan kebutuhan komputasi yang rendah, serta kemudahan interpretasi model. Dengan mengintegrasikan empat jenis fitur :GLCM, RGB Ratio, RGB Intersection, dan Color Moments, diharapkan model mampu mencapai keseimbangan antara akurasi tinggi dan kecepatan pemrosesan yang memadai untuk deteksi kebakaran secara *real-time*.

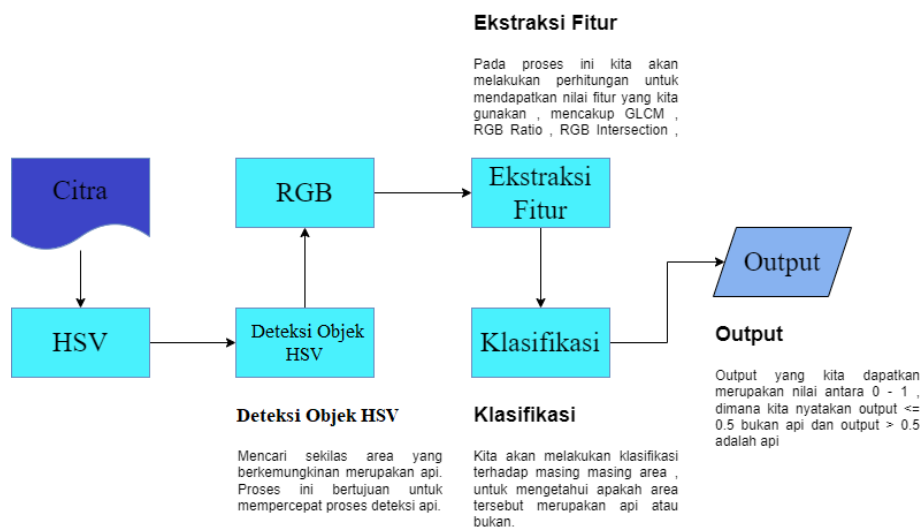
Penelitian ini bertujuan untuk mengembangkan dan mengevaluasi model deteksi kebakaran berbasis *computer vision* menggunakan Logistic Regression yang didukung oleh kombinasi fitur tekstur dan warna, yaitu GLCM, RGB Ratio, RGB Intersection, dan Color Moments. Model ini diharapkan mampu memberikan akurasi deteksi yang tinggi serta kinerja pemrosesan yang cepat, sehingga dapat diimplementasikan secara efektif dalam sistem peringatan dini kebakaran di lingkungan nyata.

2. METODE

Penelitian ini menggunakan *Logistic Regression* untuk mengklasifikasikan area kebakaran berdasarkan fitur-fitur visual seperti *GLCM*, *RGB Ratio*, *RGB Intersection*, dan *Color Moments*. Berdasarkan Gambar 1, penelitian diawali dengan proses konversi citra ke ruang warna *HSV* untuk mendeteksi area yang berpotensi merupakan api melalui analisis awal. Selanjutnya, data citra dalam ruang warna *RGB* diolah melalui proses ekstraksi fitur, meliputi perhitungan *GLCM*, *RGB Ratio*, *RGB Intersection*, dan *Color Moments*. Hasil ekstraksi fitur ini digunakan sebagai input pada tahap klasifikasi

menggunakan algoritma *Logistic Regression*, yang kemudian menghasilkan output berupa probabilitas apakah area tersebut merupakan api atau bukan.

Berdasarkan Gambar 1, alur kerja sistem deteksi kebakaran diawali dengan input citra yang dikonversi ke ruang warna HSV. Konversi ini bertujuan untuk memudahkan proses deteksi awal objek api, karena ruang warna HSV lebih efektif dalam memisahkan informasi warna dibandingkan RGB, khususnya dalam kondisi pencahayaan yang bervariasi. Pada tahap *Deteksi Objek HSV*, sistem mencari area yang secara visual berpotensi merupakan api, sehingga proses ini dapat mempercepat deteksi dengan hanya memproses wilayah yang relevan. Setelah area kandidat api teridentifikasi, citra dikembalikan ke format RGB untuk tahap *ekstraksi fitur*.



Gambar 1. *Flowchart* Penelitian

Tahap *Ekstraksi Fitur* melibatkan perhitungan berbagai parameter yang digunakan sebagai masukan model, mencakup fitur tekstur dan warna seperti Gray-Level Co-occurrence Matrix (GLCM), RGB Ratio, dan RGB Intersection. Hasil ekstraksi fitur kemudian dimasukkan ke tahap *klasifikasi*, di mana algoritma Logistic Regression digunakan untuk memprediksi apakah area tersebut mengandung api atau tidak. Proses klasifikasi menghasilkan output dalam bentuk nilai probabilitas antara 0 hingga 1. Nilai $\leq 0,5$ diinterpretasikan sebagai non-api, sedangkan nilai $> 0,5$ dikategorikan sebagai api. Dengan alur ini, sistem mampu melakukan deteksi api secara efisien, menggabungkan kecepatan deteksi awal berbasis HSV dan akurasi klasifikasi melalui kombinasi fitur yang kaya.

2.1. Pengumpulan Data

Data yang digunakan dalam penelitian ini diperoleh dari platform *Kaggle*, berupa gambar mentah yang mencakup objek api dan non-api dalam satu gambar. Untuk memisahkan area relevan, dilakukan deteksi objek berbasis warna HSV dengan *rules* yang ditunjukkan pada Tabel 1 untuk menghasilkan region proposal yang hanya mencakup satu jenis objek [3]. Setelah segmentasi, citra diberi anotasi manual dan dikelompokkan ke dalam folder api dan non-api untuk pelatihan model klasifikasi *Logistic Regression*.

Tabel 1. *Range* Api Merah dalam Model HSV

Channel	Lebih Besar Dari	Lebih Kecil Dari
Hue	5	90
Saturation	40	255
Value	220	255

Tabel 1 menunjukkan *range* nilai komponen warna pada model HSV yang digunakan untuk mendeteksi area kandidat api. Tiga komponen yang dianalisis adalah Hue, Saturation, dan Value. Pada komponen Hue, nilai yang dipilih berada di antara 5 hingga 90, yang merepresentasikan rentang warna kemerahan hingga kekuningan yang umum terdapat pada nyala api. Saturation dibatasi antara 40 hingga 255 untuk memastikan area yang terdeteksi memiliki tingkat kejenuhan warna yang cukup tinggi, sehingga mampu membedakan warna api dari objek lain dengan rona pucat atau kurang jenuh. Komponen Value berada pada rentang 220 hingga 255, yang menunjukkan tingkat kecerahan tinggi, sesuai karakteristik visual api yang biasanya memiliki intensitas cahaya kuat.

Dengan menerapkan *threshold* HSV ini, sistem dapat melakukan segmentasi awal untuk memisahkan area yang memiliki kemungkinan besar merupakan api dari latar belakang atau objek lain. Proses ini tidak hanya meningkatkan akurasi deteksi pada tahap selanjutnya, tetapi juga mempercepat pemrosesan karena hanya area hasil segmentasi yang akan diekstraksi fiturnya dan diklasifikasikan oleh model Logistic Regression.

2.2. Gray-Level Co-occurrence Matrix (GLCM)

Gray-Level Co-occurrence Matrix (GLCM) adalah sebuah matriks yang digunakan dalam pengolahan citra untuk menggambarkan distribusi spasial dari pasangan nilai intensitas piksel [4][5].

$$Dissimilarity = \sum_{i=1}^N \sum_{j=1}^N |i - j| \cdot P(i, j) \quad (1)$$

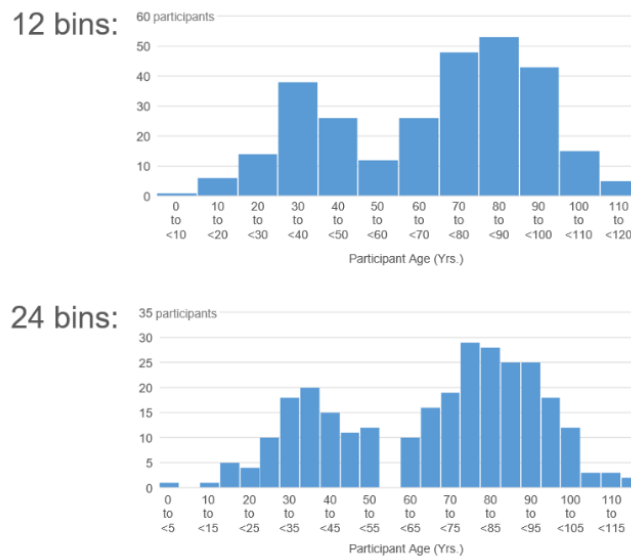
$$Correlation = \sum_{i=1}^N \sum_{j=1}^N P(i, j) \left[\frac{(i - \mu_x)(j - \mu_y)}{\sigma_x \sigma_y} \right] \quad (2)$$

Persamaan (1) digunakan untuk menghitung *Dissimilarity*, yang menggambarkan tingkat variasi atau ketidakseragaman pola tekstur dalam citra. Nilai yang lebih tinggi menunjukkan pola tekstur yang lebih tidak seragam. Sementara itu, Persamaan (2) menghitung *Correlation*, yang menunjukkan tingkat keterkaitan pola antara pasangan piksel pada citra. Nilai korelasi yang lebih tinggi mencerminkan adanya keteraturan atau pola tekstur tertentu. Kedua persamaan ini digunakan untuk mendapatkan fitur tekstur dari citra api dan non-api, yang kemudian membantu dalam proses klasifikasi untuk membedakan kedua jenis objek tersebut.

2.3. Rasio RGB

RGB Histogram adalah representasi sebaran piksel dalam sebuah gambar yang memberikan informasi apakah gambar tersebut cenderung terang atau gelap dan juga informasi lainnya. Setiap perubahan pada warna dan pencahayaan gambar akan tercermin dalam histogram, sehingga memungkinkan analisis visual terhadap citra dengan menggunakan histogram [6]. Sedangkan *RGB Histogram Ratio* merupakan teknik analisis yang digunakan untuk membandingkan distribusi warna antara dua gambar dengan menghitung rasio histogram warna RGB setelah melalui proses binning dan normalisasi. Histogram warna *RGB* menggambarkan frekuensi kemunculan setiap nilai warna dalam gambar, dengan saluran merah (R), hijau (G), dan biru (B) masing-masing mewakili bagian dari informasi warna.

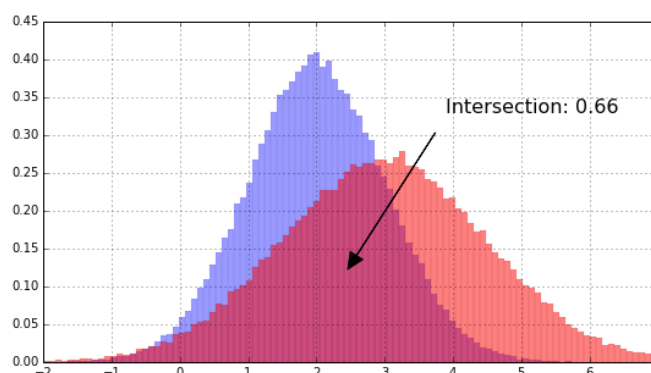
Gambar 2 menggambarkan proses *histogram binning*, yaitu teknik untuk mengelompokkan nilai intensitas warna ke dalam jumlah *bin* (interval) yang lebih sedikit guna menyederhanakan distribusi warna pada citra. Pada bagian bawah gambar ditampilkan histogram awal dengan 24 *bin*, di mana setiap *bin* merepresentasikan jumlah piksel pada rentang intensitas warna tertentu. Histogram ini memuat detail distribusi warna yang cukup tinggi, namun jumlah *bin* yang terlalu banyak dapat membuat proses perhitungan menjadi lebih kompleks dan memerlukan waktu komputasi yang lebih lama.

Gambar 2. Ilustrasi *Histogram Binning* (Sumber : [7])

Proses *binning* kemudian dilakukan dengan menggabungkan beberapa *bin* menjadi *bin* yang lebih besar, sehingga menghasilkan histogram di bagian atas gambar dengan 12 *bin*. Penyederhanaan ini bertujuan untuk mengurangi kompleksitas data tanpa kehilangan informasi penting terkait distribusi warna. Dalam konteks penelitian ini, *histogram binning* digunakan pada perhitungan RGB Ratio untuk memudahkan perbandingan distribusi warna antara citra api dan non-api. Dengan jumlah *bin* yang lebih sedikit, proses perhitungan rasio menjadi lebih efisien, sekaligus mempertahankan kemampuan model dalam membedakan karakteristik visual api berdasarkan distribusi warna.

2.4. RGB Intersection

RGB Histogram Intersection adalah teknik yang digunakan untuk menilai kesamaan antara dua gambar berdasarkan distribusi warna mereka seperti yang ditampilkan pada Gambar 3. Merupakan cara paling sederhana untuk merepresentasikan informasi warna gambar disediakan oleh histogram [8].

Gambar 3. Ilustrasi *Histogram Intersection* (Sumber : [9])

Dalam praktiknya, akan dimiliki tiga histogram untuk masing-masing gambar: satu untuk *channel* merah (R), *channel* hijau (G), dan *channel* biru (B). Proses *histogram intersection* dilakukan dengan membandingkan histogram dari gambar pertama dengan histogram dari gambar kedua untuk setiap *channel*. Untuk setiap *channel*, nilai *intersection* dihitung dengan menjumlahkan nilai minimum dari

frekuensi pada setiap bin yang tumpang tindih antara kedua histogram seperti yang ditampilkan pada persamaan (3), dan persamaan (4) sebagai proses normalisasi terhadap nilai *Intersection*.

$$I = \sum_{i=1}^n \min(H_1(i), H_2(i)) \quad (3)$$

$$I_{norm} = \frac{I}{\min(Total_{H_1}, Total_{H_2})} \quad (4)$$

2.5. Color Moments

Color moments adalah teknik statistik yang digunakan untuk menggambarkan distribusi warna dalam citra. Konsep ini berfokus pada karakteristik warna dari gambar dengan menghitung momen dari nilai intensitas warna dalam setiap channel *RGB*. Momen ini sering sekali meliputi tiga tingkat perhitungan [10] :

a. Mean

$$\mu = \frac{1}{M \times N} \sum_{i=1}^M \sum_{j=1}^N P_{ij} \quad (5)$$

Mean pada persamaan (5) memberikan gambaran umum tentang warna dominan dalam gambar. Nilai *mean* yang lebih tinggi menunjukkan dominasi warna yang lebih terang, sementara nilai yang lebih rendah menunjukkan warna yang lebih gelap.

b. Standard Deviation

$$\sigma = \sqrt{\frac{1}{M \times N} \sum_{i=1}^M \sum_{j=1}^N (P_{ij} - \mu)^2} \quad (6)$$

Standard Deviation pada persamaan (6) memberikan informasi tentang seberapa jauh nilai piksel menyebar dari rata-ratanya. Nilai deviasi standar yang tinggi menunjukkan bahwa ada variasi yang signifikan dalam intensitas piksel, sedangkan nilai yang rendah menunjukkan bahwa piksel-piksel cenderung memiliki intensitas yang mirip.

c. Skewness

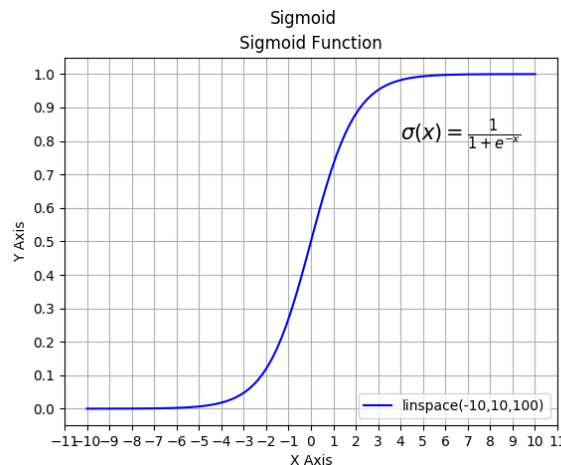
$$\theta = \frac{1}{M \times N} \sum_{i=1}^M \sum_{j=1}^N \left(\frac{(P_{ij} - \mu)}{\sigma} \right)^3 \quad (7)$$

Skewness pada persamaan (7) memberikan informasi tentang arah dan derajat asimetri distribusi warna. Nilai *skewness* yang positif menunjukkan bahwa distribusi cenderung lebih condong ke kiri (banyak nilai yang lebih rendah), sementara nilai negatif menunjukkan distribusi yang condong ke kanan (banyak nilai yang lebih tinggi). Jika *skewness* mendekati nol, ini menunjukkan bahwa distribusi dianggap simetris.

2.6. Logistic Regression

Logistic regression adalah salah satu teknik dalam statistika yang digunakan untuk memodelkan hubungan antara satu atau beberapa variabel independen (prediktor) dengan variabel dependen biner (hasil yang bersifat biner, seperti ya/tidak, sukses/gagal, atau 0/1) [11]. Logistic Regression merupakan algoritma klasifikasi biner yang digunakan untuk memprediksi probabilitas suatu data masuk ke dalam salah satu dari dua kelas, misalnya api (1) atau non-api (0). Berbeda dengan regresi linear yang menghasilkan nilai kontinu, Logistic Regression memanfaatkan fungsi *sigmoid* untuk memetakan output menjadi nilai probabilitas antara 0 dan 1.

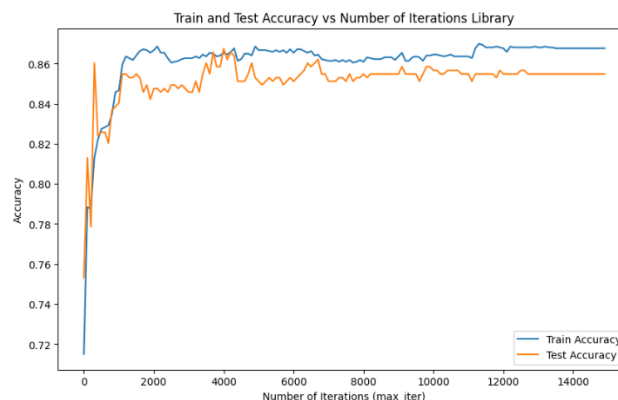
Gambar 4 merupakan kurva matematis yang penting dalam *logistic regression* karena mengilustrasikan bagaimana model tersebut mengubah nilai-nilai prediktor menjadi probabilitas. Dalam grafik ini, sumbu x mewakili nilai dari fungsi linear yang disebut logit, sedangkan sumbu y menunjukkan probabilitas kejadian positif yang diprediksi oleh model.



Gambar 4. Grafik Fungsi Sigmoid (Sumber : [12])

2.7. Hyperparameter Tuning max_iter Logistic Regression

Parameter *max_iter* dalam algoritma *Logistic Regression* mengacu pada jumlah iterasi maksimum yang diperbolehkan oleh algoritma optimasi untuk menemukan solusi yang konvergen. *Logistic Regression* menggunakan metode optimasi, seperti *Gradient Descent* atau *Newton-Raphson*, untuk meminimalkan fungsi *log-loss* atau *cross-entropy loss*. Jika jumlah iterasi yang diizinkan terlalu kecil, algoritma mungkin berhenti sebelum mencapai solusi optimal, yang menyebabkan model tidak konvergen dan menghasilkan hasil yang kurang akurat. Sebaliknya, jika nilai *max_iter* terlalu tinggi, model mungkin memerlukan waktu yang lebih lama untuk berlatih, meskipun hasil akhirnya tidak selalu lebih baik. Oleh karena itu, memilih nilai *max_iter* yang tepat sangat penting untuk mencapai akurasi yang paling optimal.



Gambar 5. Train Test Accuracy

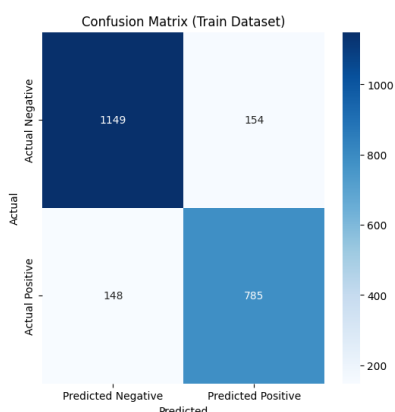
Pemilihan nilai *max_iter* dilakukan untuk memperoleh akurasi pelatihan (*Train Accuracy*) dan akurasi pengujian (*Test Accuracy*) yang optimal. Berdasarkan hasil eksperimen yang ditampilkan pada Gambar 5, dapat dilihat bahwa rentang nilai *max_iter* yang menghasilkan akurasi tinggi pada kedua data, yaitu *Train Accuracy* dan *Test Accuracy*, berada pada kisaran antara 3.500 hingga 4.000. Dari

rentang tersebut, nilai `max_iter` yang paling optimal dipilih pada titik 3.700, karena memberikan hasil akurasi yang baik untuk kedua data tersebut. Akurasi Train Data yang diperoleh adalah 86%, sementara akurasi Test Data mencapai 87%. Oleh karena itu, nilai `max_iter` = 3.700 dipilih sebagai nilai optimal untuk digunakan dalam model regresi logistik pada penelitian ini.

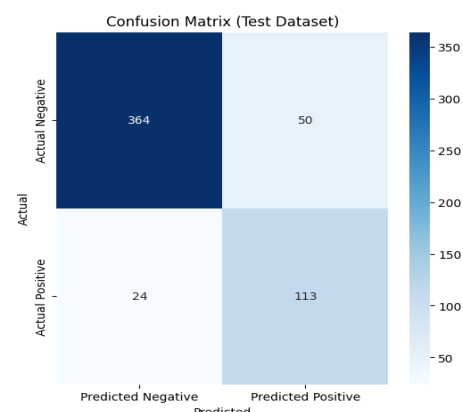
3. HASIL

Bab ini menyajikan hasil implementasi dan evaluasi model deteksi kebakaran berbasis *computer vision* menggunakan algoritma Logistic Regression dengan kombinasi fitur GLCM, RGB Ratio, RGB Intersection, dan Color Moments. Proses pengujian dilakukan pada dataset citra api dan non-api yang telah melalui tahap segmentasi berbasis HSV dan ekstraksi fitur. Evaluasi kinerja model meliputi pengukuran akurasi, *recall*, serta analisis *confusion matrix* pada data latih dan data uji. Selain itu, dilakukan pula perbandingan kinerja dengan model Convolutional Neural Network (CNN) untuk menilai keunggulan dari sisi kecepatan pemrosesan (*frames per second*). Penyajian hasil pada bab ini bertujuan untuk memberikan gambaran kuantitatif dan kualitatif terhadap kemampuan model dalam mendeteksi api, sekaligus mengidentifikasi potensi kelemahan yang dapat menjadi dasar pengembangan lebih lanjut.

Gambar 6 dan Gambar 7 menunjukkan *Confusion Matrix* yang dihasilkan untuk model deteksi api pada *Train Dataset* dan *Test Dataset*. *Confusion matrix* ini menggambarkan distribusi hasil prediksi model terhadap data yang sebenarnya. Pada matriks ini, terdapat empat nilai yang mencerminkan hasil klasifikasi: **True Positives (TP)**, **False Positives (FP)**, **True Negatives (TN)**, dan **False Negatives (FN)**. Nilai-nilai ini penting untuk memahami seberapa baik model dalam memprediksi kedua kelas (positif dan negatif).



Gambar 6. *Confusion Matrix Train Dataset*



Gambar 7. *Confusion Matrix Test Dataset*

Gambar 6 memiliki **True Positives (TP)** yang berjumlah 785, model berhasil mengidentifikasi kebakaran dengan benar. Ini menunjukkan bahwa model dapat mengenali kebakaran dengan baik. Namun, meskipun model berhasil mendeteksi sebagian besar kebakaran, terdapat **False Negatives (FN)** sebanyak 148. Nilai **False Negatives (FN)** menunjukkan bahwa masih ada peluang untuk meningkatkan performa model, khususnya dalam hal Recall. Recall adalah metrik yang mengukur kemampuan model dalam mendeteksi kebakaran yang sebenarnya terjadi. Dalam kasus ini, meskipun model dapat mendeteksi kebakaran dengan baik pada sebagian besar kejadian, masih ada sejumlah kebakaran yang tidak terdeteksi, yang bisa menjadi masalah dalam konteks sistem deteksi kebakaran yang harus bekerja dengan sangat presisi.

Gambar 7 memiliki **True Positives (TP)** yang berjumlah 113, model berhasil mengidentifikasi kebakaran dengan benar. Ini menunjukkan bahwa ada 113 kejadian kebakaran yang diprediksi dengan

tepat sebagai positif oleh model. Namun, pada **False Negatives (FN)** yang berjumlah 24, terdapat 24 kejadian kebakaran yang tidak terdeteksi oleh model, yang bisa berisiko tinggi dalam konteks aplikasi deteksi kebakaran karena kebakaran yang sebenarnya terjadi tidak dikenali. Di sisi lain, **True Negatives (TN)** sebanyak 364 menunjukkan bahwa model berhasil memprediksi dengan benar situasi yang tidak ada kebakaran. Namun, **False Positives (FP)** yang berjumlah 50 menunjukkan adanya 50 prediksi kebakaran yang salah pada data yang sebenarnya tidak ada kebakaran. Meskipun angka *False Positives* ini relatif lebih kecil dibandingkan dengan *True Negatives*, kesalahan ini tetap perlu diperhatikan untuk menghindari peringatan palsu yang tidak perlu. Dimana *False Positives* sering terjadi pada data yang memiliki bentuk visual yang identik dengan api.

Tabel 2. Hasil Prediksi *Testing Data*

<i>Data</i>	<i>Ground-Truth</i>	<i>Predict</i>
	Api	Api
	Api	Non-Api
	Api	Api
	Api	Api
	Api	Api
.....		
	Non-Api	Non-Api
	Non-Api	Api
	Non-Api	Non-Api
	Non-Api	Non-Api

Tabel 2 menunjukkan bahwa model masih menghasilkan kesalahan prediksi berupa *False Positive (FP)* dan *False Negative (FN)*. Salah satu contoh terlihat pada baris ke-2, di mana model salah memprediksi gambar yang sebenarnya merupakan api (*True Class: Api*) sebagai gambar non-api (*Predicted Class: Non-Api*). Kesalahan ini termasuk kategori *False Negative*, yang berarti model gagal mengenali keberadaan api dalam gambar tersebut.

Kesalahan ini menunjukkan bahwa meskipun model memiliki tingkat akurasi yang cukup baik, ada kasus tertentu di mana fitur yang diekstraksi tidak cukup untuk membedakan karakteristik api dari

non-api. Hal ini menyoroti masih diperlukan pengembangan lanjut, baik dalam proses ekstraksi fitur maupun parameterisasi model, untuk meningkatkan sensitivitas dan mengurangi risiko kesalahan seperti False Negatives, yang berpotensi menyebabkan keterlambatan dalam mendeteksi kebakaran.

Tabel 3. Perbandingan *Logistic Regression* dan *CNN*

Durasi Video	<i>Logistic Regression (Average FPS)</i>	<i>CNN (Average FPS)</i>
12 Detik	11.3	5.9
22 Detik	12.7	7
27 Detik	16.2	10.4
71 Detik	12.6	7.4
156 Detik	15.7	11

Tabel 3 menunjukkan hasil testing yang dilakukan pada beberapa video dengan durasi berbeda dengan menggunakan laptop yang berspesifikasi : *Processor* : Intel i7-10750H, *GPU* : Nvidia GeForce GTX 1650 4 GB, *RAM* : 16 GB. Hasil *testing* menunjukkan penggunaan *Logistic Regression* memiliki jumlah *Average FPS* yang lebih tinggi jika dibandingkan dengan menggunakan *CNN* pada beberapa video yang dilakukan *testing*.

4. PEMBAHASAN

Hasil pengujian menunjukkan bahwa model deteksi kebakaran berbasis *computer vision* dengan algoritma *Logistic Regression* dan kombinasi fitur GLCM, RGB Ratio, RGB Intersection, serta Color Moments mampu memberikan kinerja yang cukup baik, dengan akurasi 86% pada data latih dan 87% pada data uji. Nilai *recall* sebesar 84% (train) dan 82% (test) menunjukkan kemampuan model yang konsisten dalam mengenali citra yang benar-benar mengandung api. Meski demikian, masih terdapat *false negatives* yang dapat berisiko pada implementasi di dunia nyata, karena kejadian kebakaran yang sebenarnya bisa tidak terdeteksi.

Keunggulan utama dari metode ini terletak pada kecepatan pemrosesan. Dibandingkan dengan *Convolutional Neural Network (CNN)*, *Logistic Regression* menghasilkan rata-rata *frames per second (FPS)* yang lebih tinggi, yaitu hingga 16,2 FPS pada video berdurasi 27 detik. Kecepatan ini sangat relevan untuk kebutuhan *real-time monitoring* pada perangkat dengan keterbatasan sumber daya, seperti CCTV berbasis *embedded system* atau *edge devices*. Hal ini sejalan dengan penelitian Ryu dan Kwak (2022) yang menggarisbawahi pentingnya pra-pemrosesan berbasis warna untuk mempercepat deteksi api pada sistem pengawasan.

Jika dibandingkan dengan penelitian yang menggunakan pendekatan *deep learning* murni, seperti YOLO atau Faster R-CNN (I Gede et al., 2022), metode yang diajukan dalam studi ini memang sedikit lebih rendah dalam hal akurasi absolut. Namun, *Logistic Regression* dengan multi-feature extraction memberikan kompromi yang ideal antara akurasi dan efisiensi komputasi. Pendekatan ini mengisi celah penelitian (*research gap*) pada deteksi kebakaran yang mengutamakan *low-latency inference* tanpa mengorbankan terlalu banyak kinerja deteksi.

Dari perspektif informatika dan ilmu komputer, penelitian ini memberikan kontribusi pada pengembangan sistem deteksi berbasis visi komputer yang *resource-efficient*. Penerapan metode ini berpotensi besar pada sistem peringatan dini kebakaran yang terintegrasi dengan IoT, di mana perangkat di lapangan memiliki keterbatasan daya komputasi dan kebutuhan untuk merespons secara instan. Selain itu, penggunaan kombinasi fitur tekstur dan warna membuktikan bahwa teknik *feature engineering* tradisional tetap relevan di era dominasi *deep learning*, khususnya untuk aplikasi yang membutuhkan efisiensi tinggi.

Meskipun hasil yang diperoleh cukup menjanjikan, masih terdapat ruang untuk peningkatan, terutama dalam mengurangi jumlah *false negatives*. Beberapa pendekatan yang dapat dipertimbangkan

untuk penelitian lanjutan antara lain menggabungkan Logistic Regression dengan *ensemble methods*, menerapkan *adaptive thresholding* pada segmentasi HSV, atau memadukannya dengan *lightweight CNN* untuk meningkatkan sensitivitas tanpa mengorbankan kecepatan.

5. CONCLUSION

Berdasarkan rumusan masalah yang bertujuan untuk mengevaluasi kemampuan kombinasi fitur *Grey Level Co-occurrence Matrix* (GLCM), Rasio *RGB*, *RGB Intersection*, dan *Color Moments* dalam mendukung algoritma *Logistic Regression* untuk mengenali adanya api, dapat disimpulkan bahwa kombinasi fitur tersebut memberikan hasil yang cukup baik dalam deteksi kebakaran. Dengan hasil penelitian menunjukkan bahwa kombinasi fitur pada model deteksi kebakaran memiliki akurasi dan *recall* yang cukup tinggi. Pada *Train Dataset*, model mencapai akurasi 86% dan *recall* 84%, sementara pada *Test Dataset*, akurasi 87% dan *recall* 82%.

CONFLICT OF INTEREST

The authors declares that there is no conflict of interest between the authors or with research object in this paper.

REFERENCES

- [1] I. Gede *et al.*, “Deteksi api kebakaran berbasis computer vision dengan algoritma YOLO,” *Journal of Applied Mechanical Engineering and Green Technology*, vol. 3, pp. 53–58, 2022, [Online]. Available: <https://ojs2.pnb.ac.id/index.php/JAMETECH>
- [2] pemadam.jakarta.go.id, “Kaleidoskop Kejadian Kebakaran di Provinsi DKI Jakarta Selama Tahun 2023,” pemadam.jakarta.go.id.
- [3] J. Ryu and D. Kwak, “A Method of Detecting Candidate Regions and Flames Based on Deep Learning Using Color-Based Pre-Processing,” *Fire*, vol. 5, no. 6, Dec. 2022, doi: 10.3390/fire5060194.
- [4] F. Feiters Tampinongkol, C. Herdian, H. Basri, and L. Halim, “Identifikasi Penyakit Daun Tomat Menggunakan Gray Level Co-occurrence Matrix (GLCM) dan Support Vector Machine (SVM),” 2023. [Online]. Available: <https://www.kaggle.com/>
- [5] F. Feiters Tampinongkol, J. A. Ginting, C. Herdian, Y. Purnomo, H. Basri, and L. Halim, “Perbandingan Metode GLCM dan DWT Dalam Mengekstraksi Ciri Penyakit pada Daun Tomat,” 2023. [Online]. Available: <https://www.kaggle.com/>.
- [6] M. Ikhsan and A. Wiranda Hakiki, “Analisis Perbandingan Metode Histogram Equalization dan Gaussian Filter untuk Perbaikan Kualitas Citra,” 2024. [Online]. Available: <http://jurnal.goretanpena.com/index.php/JSSR>
- [7] practicalreporting.com, “How many bins should my histogram have?,” practicalreporting.com.
- [8] A. Barla, F. Odone, and A. Verri, “Histogram intersection kernel for image classification,” in *IEEE International Conference on Image Processing*, IEEE Computer Society, 2003, pp. 513–516. doi: 10.1109/icip.2003.1247294.
- [9] blog.datadive.net, “Histogram Intersection for Change Detection,” blog.datadive.net.
- [10] M. Abdan Mulia and Y. Arum Sari, “Klasifikasi Citra Jenis Makanan dengan Color Moments, Morphological Shape Descriptors, dan Gray Level Cooccurrence Matrix menggunakan Neighbor Weight K-Nearest Neighbor,” 2019. [Online]. Available: <http://j-ptiik.ub.ac.id>
- [11] R. Herina Situngkir and P. Sembiring, “Analisis Regresi Logistik Untuk Menentukan Faktor-Faktor Yang Mempengaruhi Kesejahteraan Masyarakat Kabupaten/Kota Di Pulau Nias,” vol. 6, no. 1, pp. 25–31, 2023.
- [12] Palak Jain, “Activation Functions in Neural Networks - Part 2,” medium.com.

This Page is Blank