Klasifikasi Api Pada Klip Pendek Menggunakan HSV Rule dan Ekstraksi Fitur LBP-TOP

Azril Nurfaizi Adlin
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
azzril@students.telkomuniversity.ac.id

Febryanti Sthevanie, S.T., M.T.
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
sthevanie@telkomuniversity.ac.id

Kurniawan Nur Ramadhani, S.T., M.T.
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
kurniawannr@telkomuniversity.ac.id

***Abstrak* — Bencana kebakaran dapat menyebabkan kerusakan yang signifikan terhadap properti dan infrastruktur, serta menimbulkan ancaman berat bagi kehidupan manusia. Dibutuhkan suatu sistem yang mampu mendeteksi api sejak dini agar tindakan pencegahan dapat dilakukan. Sistem deteksi api konvensional yang sudah banyak digunakan biasanya dibagun untuk mendeteksi asap atau suhu untuk memicu alarm dimana respon sistem akan terlambat pada saat api yang dideteksi menghasilkan asap dan suhu yang minimal. Untuk mengatasi masalah tersebut, penelitian ini membangun sebuah sistem klasifikasi api berbasis video dengan menggunakan kombinasi metode ruang warna HSV, *Local Binary Patterns* pada tiga bidang ortogonal (LBP-TOP), dan *Support Vector Machine*(SVM). Dataset yang digunakan berjumlah 26 video dimana 10 video berlabel api dan 16 video berlabel bukan api. Penelitian ini menghasilkan skor akurasi tertinggi sebesar 90.9%**

***Kata kunci*— Deteksi api, HSV, LBP-TOP, SVM**

I. PENDAHULUAN

Latar Belakang

Deteksi api merupakan aspek penting dalam manajemen keamanan diberbagai industri. Kebakaran dapat menyebabkan kerusakan yang signifikan terhadap properti dan infrastruktur, serta menimbulkan ancaman berat bagi kehidupan manusia, Oleh karena itu dibutuhkan sistem deteksi api yang cepat dan akurat sangatlah penting untuk meminimalkan kerugian serta kerusakan yang dapat timbul dari bencana kebakaran.

Sudah banyak sistem deteksi api konvensional yang digunakan dalam bangunan skala besar maupun kecil diantaranya seperti pendeteksi suhu, pendeteksi asap, dan lain-lain. Sistem deteksi api konvensional populer digunakan karena keterjangkauan, kesederhanaan, dan keandalan yang diberikan oleh sistem tersebut, namun bukan berarti sistem ini tidak memiliki keterbatasan. Keterbatasan utama pada sistem deteksi api konvensional adalah sistem ini hanya dapat memberikan indikasi lokasi umum api tanpa bisa memberikan lokasi persis dari api dan juga kecepatan respon dari sistem ini tergantung dari seberapa dekat titik api dengan alat karena sistem ini bekerja dengan mendeteksi asap dan suhu yang dihasilkan oleh api.

Untuk mengatasi keterbatasan sistem pendeteksi konvensional, banyak penelitian yang mengajukan metode deteksi api berdasarkan rekaman video dengan memanfaatkan kemampuan machine learning. Metode yang diajukan mulai dari menggunakan informasi warna api[6], mendeteksi gerakan api untuk meningkatkan akurasi dan memisahkan objek api sesungguhnya dengan objek bukan api tetapi berwarna api[5][7], serta menggunakan metode berbasis CNN untuk mendeteksi area api[8].

Untuk mengatasi permasalahan sistem deteksi api konvensional, maka pada penelitian ini akan dibuat sebuah sistem deteksi-klasifikasi api yang presisi dan responsif agar dapat mendeteksi api secara dini dengan menggunakan masukan berupa video.

Topik dan Batasannya

Topik dan batasan dari penelitian ini adalah:

1. Fokus dari penelitian ini adalah melakukan deteksi dan klasifikasi api menggunakan masukan data video yang kemudian diproses menjadi klip pendek.
2. Jenis api yang digunakan pada penelitian ini adalah api yang memiliki suhu diantara 1100 – 2200 derajat celcius yang memiliki karakteristik warna merah, oranye atau kuning.

Tujuan

Tujuan dari penelitian ini adalah:

1. Mengimplementasikan sistem untuk mendeteksi api dengan menggunakan kombinasi metode pembelajaran mesin ruang warna HSV, *Local Binary Patterns* pada tiga bidang ortogonal(LBP-TOP), dan *Support Vector Machine*(SVM)
2. Mengevaluasi kinerja sistem yang sudah diimplementasikan.

II. KAJIAN TEORI

A. Ruang Warna HSV

Ruang Warna HSV mempersepsikan warna mirip dengan bagaimana manusia mempersepsikan warna, sifat ini yang

membuat ruang warna HSV sangat cocok digunakan dalam suatu sistem yang bertujuan memproses properti warna pada citra.

Dalam ruang warna HSV terdapat tiga parameter yang digunakan untuk merepresentasikan warna yaitu *hue*, *saturation* dan *value*. *hue* merupakan distribusi warna berdasarkan panjang gelombang dari warna, *saturation* merupakan derajat dimana warna mengandung cahaya putih, dan *value* merupakan ukuran intensitas dari pencahayaan yang membuat ruang warna ini tahan akan pencahayaan yang berganti-ganti[9].

B. Local Binary Pattern Three Orthogonal Planes

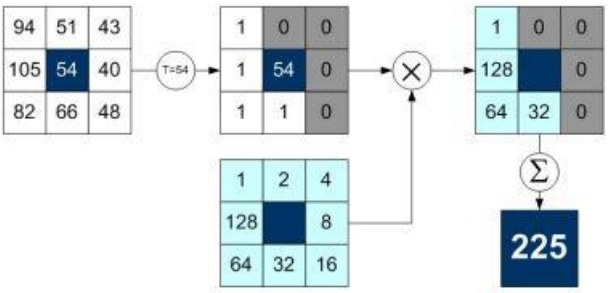
Local Binary Pattern(LBP) merupakan metode yang simpel dan efficient untuk mendeskripsi fitur sebuah citra. Secara umum LBP bekerja dengan cara menghitung setiap piksel pada citra dengan melakukan *thresholding* piksel tengah(*center pixel*) dengan piksel tetangganya (*neighbour pixel*) yang hasilnya akan dibentuk kedalam *binary pattern*, kemudian akan dibentuk histogram yang menggambarkan berapa banyaknya suatu fitur muncul[2]. Persamaan 1 digunakan untuk menghitung nilai LBP.

$$LBP(x_c, y_c) = \sum_{n=0}^{n-1} g(I_n - I(x_c, y_c))2^n \qquad (1)$$

dimana fungsi g(x) atau threshold untuk pixel tetangga diekspresikan persamaan 2 sebagai berikut:

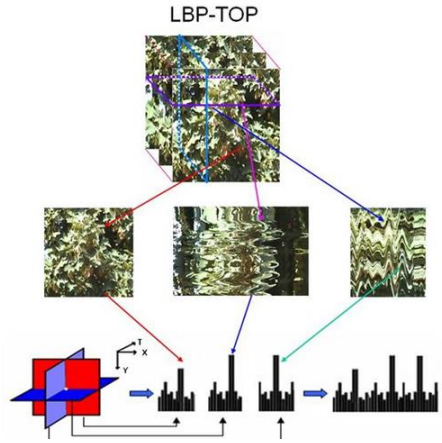
$$g(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases} \qquad (2)$$

Contoh penggunaan persamaan 1 dan 2 dapat dilihat pada gambar 1 berikut. gambar 1 merupakan matriks 3x3 yang memiliki piksel tengah bernilai 54 yang akan dipilih sebagai nilai ambang batas (*threshold value*). Nilai piksel tetangga akan dikurangi dengan nilai ambang batas, hasil dari pengurangan akan mengikuti aturan pada persamaan 2. Seluruh piksel tetangga yang sudah diproses kemudian akan dilakukan penjumlahan seperti pada gambar 1[10].



Gambar 1 Contoh LBP matriks 3x3

LBP-TOP memiliki prinsip kerja yang sama dengan LBP, hanya saja karena informasi spatial-temporal berada dalam ranah 3D, LBP-TOP harus melakukan dekomposisi 3D volume ke dalam tiga orthogonal plane: XY,XT,YT[2]. Cara kerja LBP-TOP secara umum dapat dilihat pada gambar 2.



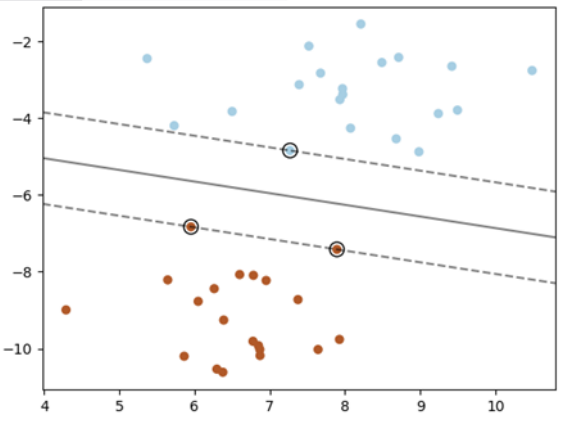
Gambar 2 LBP-TOP

C. Support Vector Machine

Support Vector Machine (SVM) merupakan algoritma yang digunakan untuk permasalahan klasifikasi dan regresi, SVM bekerja dengan cara mencari hyperplane terbaik dalam memisahkan data dalam kelas yang berbeda[4][10]. Dalam kasus dimana data dapat dipisahkan secara linear, *hyperplane* akan membentuk garis yang membagi data ke dalam dua kelas yang berbeda. Untuk data yang tidak bisa dipisahkan secara linear, SVM akan memetakan data ke dimensi yang lebih tinggi sehingga data dapat dipisahkan secara linear.

Konsep utama dari SVM adalah untuk menemukan *hyperplane* yang dapat memaksimalkan *margin*, *margin* yang dimaksudkan disini adalah jarak antara *hyperplane* dengan titik data terdekat pada masing-masing kelas. *Margin* bertindak sebagai *buffer zone* yang membantu menggeneralisasi model dengan mengurangi *overfitting*.

Dalam SVM, *Hyperplane* adalah batasan yang membantu memisahkan titik data kedalam kelas yang berbeda. Dalam ruang 2-D, *hyperplane* merupakan sebuah garis lurus yang memisahkan dua sisi dan dalam ruang n-D, hyperplane merupakan sebuah bidang (n-1)-dimensi yang memisahkan dua sisi. gambar 3 merupakan contoh *hyperplane* untuk kasus data yang dapat dipisahkan secara *linear*, dimana



Gambar 3 *Hyperplane* pada SVM

dapat dilihat terdapat tiga titik data pada batas margin, titik data ini disebut dengan *support vector*. Pada gambar 3, *hyperplane* membagi titik data menjadi 2 kelas. Berikut persamaan yang digunakan.

$$y = ax + b$$
$$ax + b - y = 0 \quad (3)$$

Biarkan vektor dan maka vektor pada hyperplane menjadi

$$W.X + b = 0 \quad (4)$$

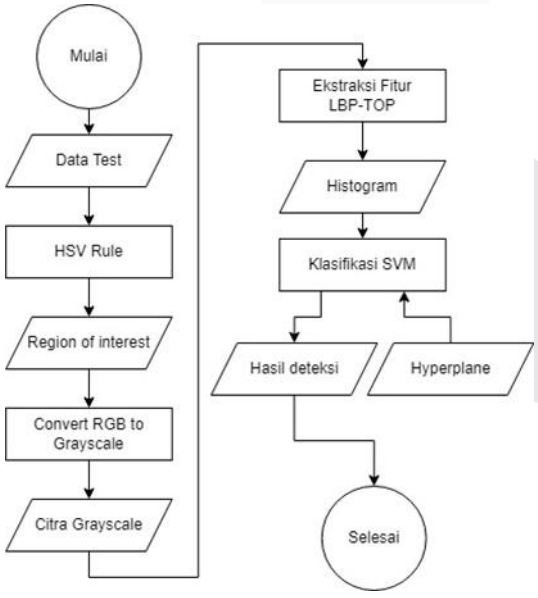
Ada kala dimana data tidak dapat dipisahkan secara linear. Untuk mengatasi keterbatasan ini dapat digunakan trik *Kernel*. Trik *kernel* merupakan teknik matematika yang digunakan dalam pembelajaran mesin untuk memungkinkan penggunaan algoritma *linear* dalam menyelesaikan permasalahan non-linear.

III. METODE

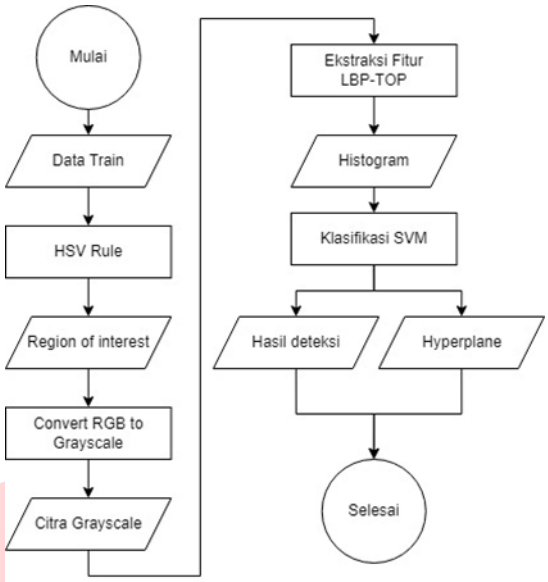
Sistem yang dibangun pada penelitian ini mampu mendeteksi area yang dianggap api dan mengklasifikasikan apakah termasuk ke dalam label kelas api atau bukan api, proses ini menerapkan metode pemrosesan citra (ruang warna HSV, LBP-TOP) dan pembelajaran mesin (SVM)

A. Desain Sistem

Tujuan dari penelitian ini adalah menghasilkan sistem yang dapat mendeteksi area api dan mengklasifikasi api dan objek bukan api. Sistem dimulai dengan menggunakan HSV untuk mendeteksi area api dan memisahkannya dengan latar, setelah itu diubah ke ruang warna *grayscale* dan akan dilakukan ekstraksi fitur untuk mendeskripsikan karakteristik tekstur api yang hasilnya merupakan nilai histogram yang kemudian akan digunakan untuk masukan SVM. Pada SVM terdapat dua proses yaitu *train* dan *test*. *Train* dilakukan terlebih dahulu, proses ini menghasilkan model prediksi yang setelahnya akan digunakan pada *test* yang akan menghasilkan nilai performansi sebagai acuan evaluasi model prediksi yang dimiliki. Seluruh proses ini dapat dilihat pada diagram alir pada gambar 4 dan gambar 5.



Gambar 6 *Training Flowchart*



Gambar 4 *Testing Flowchart*

B. *Dataset*

Penelitian ini menggunakan *dataset* dalam format video dengan data berjumlah total 26 dengan masing-masing label kelas berjumlah 10 video api dan 16 video bukan api dengan ukuran 150x150 dan berformat .avi untuk setiap videonya. Setiap video memiliki ukuran, *frame rate*, durasi dan tingkat pencahayaan yang beragam. Contoh *dataset* seperti pada gambar 6.

Gambar 5 *Dataset*

C. HSV

Daerah yang diduga api pada sebuah citra memiliki karakteristik spesifik dimana warnanya akan lebih terang dibandingkan dengan daerah sekitarnya, HSV merupakan ruang warna yang cocok digunakan karena HSV memisahkan antara informasi kecerahan dan informasi warna[1]. Data yang masuk masih dalam ruang warna RGB yang kemudian akan dikonversikan ke dalam ruang warna HSV. HSV terdiri dari tiga parameter yang merepresentasikan warna yaitu *hue*, *saturation*, dan *value*. Proses konversi dapat dilakukan dengan menggunakan formula pada persamaan 5, 6, dan 7. Setelah dilakukan konversi akan diterapkan *rule* HSV untuk memisahkan daerah yang diduga api dengan latar.

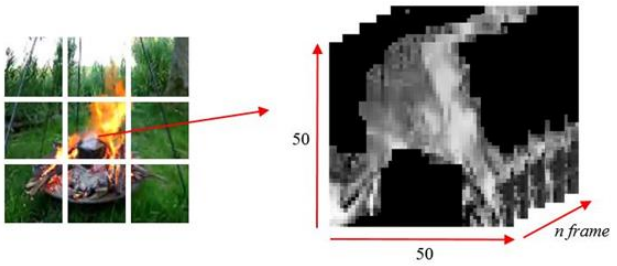
$$V = \max(R, G, B) \quad (5)$$

$$S = \begin{cases} \dfrac{V - \min(R,G,B)}{V}, & jika\ V \neq 0 \\ 0 \quad , & jika\ V = 0 \end{cases} \qquad (6)$$

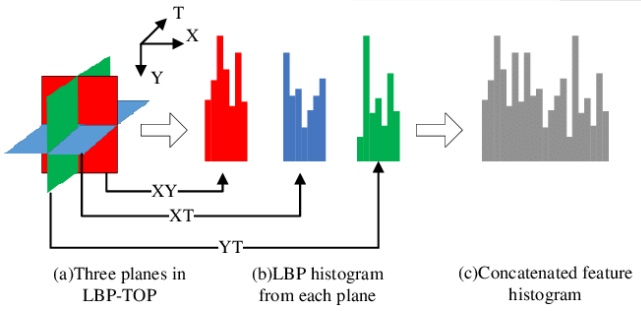
$$H = \begin{cases} \dfrac{60(G-B)}{V - \min(R,G,B)}, & jika\ V = R \\ \dfrac{120 + 60(B-R)}{V - \min(R,G,B)}, & jika\ V = G \\ \dfrac{240 + 60(R-G)}{V - \min(R,G,B)}, & jika\ V = B \\ 0 \quad , & jika\ R = G = B \end{cases} \qquad (7)$$

D. LBP-TOP

Feature extraction dilakukan untuk mengenali objek berdasarkan pola tertentu yang dimilikinya. Daerah api pada sebuah citra memiliki tektur tertentu yang membedakannya dengan objek yang berwarna api. Sistem pada penelitian ini menggunakan video sebagai *dataset* sehingga akan digunakan *Local Binary Pattern Three Orthogonal Planes* (LBP-TOP) yang dapat melakukan ekstraksi tekstur sebuah citra terhadap waktu. digabungkan. Video akan diproses menjadi blok-blok kecil atau kubus dengan ukuran 50x50x *n frame*, *frame* disini melambangkan berapa banyak *frame* kebelakang terhadap waktu atau informasi temporal yang ada pada satu blok kecil, prosesnya seperti pada gambar 7.



Gambar 8 Contoh pemrosesan video



Gambar 9 Contoh proses LBP-TOP

Kubus atau blok *array* 3D yang diproses hanya kubus yang setelah dihitung nilai *mean* dan *variance* harus lebih dari 10 untuk memastikan jika blok atau kubus *array* 3D bukan merupakan *array* kosong. Setelah dipastikan, data yang akan menjadi masukan pada LBP-TOP merupakan sebuah kubus

Gambar 7 Hasil LBP-TOP

atau blok *array* 3D yang sudah diterapkan rule HSV dan dikonversikan ke ruang warna *grayscale*, yang kemudian bidang XY,XT,YT dari kubus tersebut akan diekstraksi fiturnya dan histogramnya digabungkan, Contoh prosesnya seperti pada gambar 8 dan hasilnya pada gambar 9. Pada penelitian ini akan dicoba banyak frame dengan nilai [15, 30, 45]. Perbedaan nilai ini akan berpengaruh terhadap jumlah *dataset* yang akan dihasilkan dari proses ini yang nantinya akan digunakan pada proses klasifikasi. Adapun pengaruh perubahan nilai ini dapat dilihat pada tabel 1.

Tabel 1 Jumlah data

Frame	Banyak data	Kelas api	Kelas Non api
F=15	3690	2607	1083
F=30	1866	1312	554
F=45	1232	860	372

E. SVM Classifier

Classification dilakukan untuk mengklasifikasikan pasangan data dan label yang sudah ditentukan berdasarkan fitur histogram yang didapatkan pada proses ekstraksi fitur. Pada proses ini SVM akan dilatih dengan data yang didapat pada proses ekstraksi fitur sebagai acuan untuk mencari hyperplane terbaik yang dapat memisahkan antara data label kelas api dan bukan api. *Hyperplane* terbaik dipilih berdasarkan jarak maksimum *margin* dengan garis yang memisahkan antar kelas. Pada penelitian ini parameter yang akan disesuaikan untuk menghasilkan margin terbaik adalah jenis *kernel* dan nilai *cost* SVM. *Kernel* yang biasa digunakan untuk pengklasifikasian pada SVM adalah *radial basis function* (RBF), *linear*, *polynomial*, dan *sigmoid*. *Cost* merupakan nilai penalti yang diberikan kepada titik data yang berada didalam *margin*. Semakin rendah nilai *cost* maka semakin kecil penalti yang diberikan semakin mudah model SVM mengalami *overfitting*, berlaku juga sebaliknya dimana nilai *cost* yang tinggi akan memberikan penalti yang besar sehingga model SVM *underfitting*, ini akan mempengaruhi seberapa kaku model SVM yang dihasilkan.

F. Performance Evaluation

Confusion matrix merupakan tabel yang digunakan untuk mengevaluasi performa dari algoritma klasifikasi. Tabel ini memberikan ringkasan prediksi oleh *classifier* pada sekumpulan data dengan cara membandingkannya dengan

label data yang sebenarnya. *Confusion matrix*[3] yang biasa digunakan untuk masalah klasifikasi adalah seperti tabel 1, dimana nilai TP dan TN mengacu pada jumlah sampel yang kelasnya diperkirakan dengan benar. FP dan FN adalah perkiraan jumlah sampel yang kelasnya diperkirakan dengan salah[3]. Dalam penelitian ini TP merupakan jumlah sampel positif api yang kelasnya diperkirakan dengan benar, TN merupakan jumlah sampel negatif api yang kelasnya diperkirakan dengan benar, FP merupakan jumlah sampel positif api yang kelasnya diperkirakan dengan salah, dan FN merupakan jumlah sampel negatif api yang kelasnya diperkirakan dengan salah. Penelitian ini menggunakan nilai *Accuracy*, *Precision*, *Recall* dan *F1-score* sebagai acuan untuk evaluasi performa yang dihasilkan oleh system.

Tabel 2 Confusion Matrix

		Kelas yang diprediksi	
		Positive	*Negative*
Kelas Sebenarnya	*Positive*	*True Positive(TP)*	*False Positive(FP)*
	Negative	*False Negative(FN)*	*True Negative(TN)*

Nilai akurasi (*Accuracy*) sangat penting dalam mengevaluasi performa sistem. Akurasi mendefinisikan hubungan antara nilai hasil prediksi dengan nilai sesungguhnya yang ada pada *dataset*[3]. Persamaan 8 digunakan untuk menghitung nilai akurasi.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (8)$$

Presisi (*Precision*) atau *positive rate* merupakan ukuran banyaknya prediksi positif yang benar di prediksi[11]. Persamaan 9 digunakan untuk menghitung nilai presisi.

$$Precision = \frac{TP}{TP + FP} \quad (9)$$

Recall atau biasa juga disebut dengan *sensitivity* merupakan seberapa banyak prediksi positif yang benar diprediksi terhadap keseluruhan data yang seharusnya diprediksi positif[3]. Persamaan 10 digunakan untuk menghitung nilai recall.

$$Recall = \frac{TP}{TP + FN} \quad (10)$$

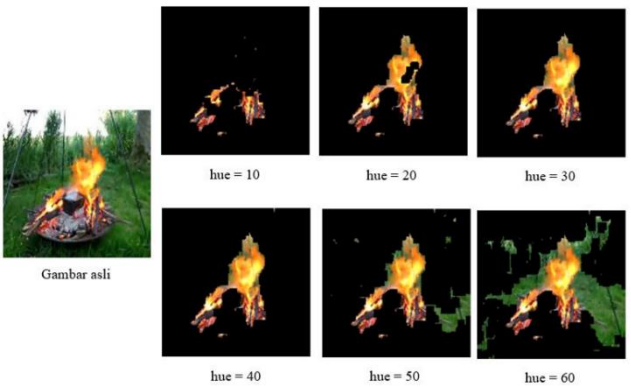
F1-score adalah nilai rata-rata yang dihitung berdasarkan nilai akurasi dan presisi yang sudah didapat[11]. Persamaan 11 digunakan untuk menghitung nilai *F1-score*.

$$F1 - score = 2 * \frac{(precision * recall)}{(precision + recall)} \quad (11)$$

IV. HASIL DAN PEMBAHASAN

A. Percobaan

Percobaan dilakukan untuk memilih *rule* HSV yang akan diterapkan pada proses segmentasi untuk memisahkan bagian yang diduga api dengan latar. Nilai HSV yang terdiri dari *hue*, *saturation*, dan *value*. Percobaan ini akan berfokus pada nilai *hue* yang merupakan representasi panjang gelombang dari warna, akan dicoba nilai antara 0-60 derajat dimana warna merah, oranye, dan kuning berada. Sedangkan untuk nilai *saturation* dan *value* akan konstan pada 100-255 derajat.



Gambar 10 Percobaan parameter hue

Dari percobaan terhadap nilai *hue* yang sudah dilakukan, dapat dilihat jika nilai *hue* 30 dan 40 memberikan hasil segmentasi yang optimal. Penelitian ini akan menggunakan *hue* dengan nilai 40 ketimbang *hue* dengan nilai 30, ini bertujuan untuk mengantisipasi adanya bagian api yang hilang karena segmentasi yang terlalu rapat. Penelitian ini akan menggunakan *rule* HSV seperti yang ditunjukkan pada persamaan 12.

$$ROI_{HSV} = \begin{cases} 1, & (0 < H(x,y) < 40)\ dan \\ & (100 < S(x,y) < 255)\ dan \\ & (100 < V(x,y) < 255) \\ 0, & yang\ lain \end{cases} \quad (12)$$

B. Skema Pengujian

Pengujian sistem dilakukan dengan mencoba beberapa nilai parameter yang dapat menghasilkan nilai akurasi, presisi, *recall*, dan *F1-score* yang memuaskan.

1. Pengujian parameter radius dan tetangga pada LBP-TOP serta pengaruh yang dihasilkan pada perubahan parameter tersebut.
2. Pengujian parameter *kernel* serta pengaruh yang dihasilkan.
3. Pengujian parameter *cost* serta pengaruh yang dihasilkan.
4. Pengujian banyak frame yang diambil pada ranah temporal untuk ekstraksi fitur pada LBP-TOP. Banyak frame yang diambil ini akan dinotasikan dengan simbol f (*frame*) pada analisis hasil pengujian.

5. Mengevaluasi performa sistem dengan parameter yang sudah didapat pada tahap 1-4 dengan mengujinya pada data *test*.

C. Analisis Hasil Pengujian

Hasil pengujian pada skenario pertama dapat dilihat pada Tabel 2 parameter radius dan tetangga dengan nilai yang berbeda-beda dicoba dengan menggunakan pengaturan bawaan SVM yaitu *kernel* RBF dan nilai *cost* 1 dengan jumlah *frame* yang diambil sebanyak 15 frame.

Tabel 3 Pengujian parameter radius dan tetangga

Radius	Tetangga	Akurasi	Presisi	Recall	F1-score
1	2	70.70%	85.34%	50.09%	41.60%
	4	84.99%	83.55%	78.85%	80.63%
	8	**91.98%**	**91.33%**	**89.01%**	**90.06%**
2	2	70.65%	35.33%	50.00%	41.40%
	4	83.06%	81.02%	76.30%	78.02%
	8	89.38%	87.15%	87.27%	87.21%
3	2	70.65%	35.33%	50.00%	41.40%
	4	81.49%	79.87%	73.00%	75.08%
	8	88.81%	86.69%	88.14%	86.41%

Dapat dilihat pada Tabel 3 terdapat trend dimana nilai radius yang kecil dengan jumlah tetangga yang banyak menghasilkan akurasi yang baik, ini berhubungan dengan cara kerja LBP yang pada kasus ini menghasilkan ekstraksi fitur yang lebih rinci sehingga meningkatkan hasil klasifikasi model SVM. Parameter terbaik dari percobaan ini akan digunakan untuk percobaan skenario kedua.

Tabel 4 Pengujian parameter kernel

Kernel	Akurasi	Presisi	Recall	F1-Score
RBF	91.98%	91.33%	89.01%	90.06%
Linear	**96.02%**	**95.66%**	**94.67%**	**95.15%**
Polynomial	92.20%	91.32%	89.59%	90.39%
Sigmoid	89.02%	90.40%	82.84%	85.52%

Tabel 4 menunjukkan bahwa *kernel linear* merupakan kernel yang paling optimal untuk permasalahan pada penelitian ini. *Kernel linear* cenderung memiliki performa yang baik saat jumlah fitur yang dimiliki sudah cukup besar [12]. Fitur yang besar ini jika dipetakan akan berada di dimensi yang tinggi sesuai dengan besarnya fitur. Pada dimensi yang tinggi data cenderung memiliki sifat dan dapat mudah dipisahkan secara *linear*. sehingga *kernel linear* akan andal untuk data yang dapat terpisah secara *linear*, ini menghasilkan nilai akurasi, presisi, *recall*, dan *f1-score* yang lebih tinggi jika dibandingkan dengan kernel SVM yang lain. Parameter terbaik yang didapat dari percobaan ini akan digunakan untuk percobaan skenario ketiga.

Tabel 5 Pengujian parameter cost

Cost	Akurasi	Presisi	Recall	F1-Score
C=1	96.02%	95.66%	94.67%	95.15%
C=2	96.21%	95.85%	94.94%	95.35%
C=3	96.15%	95.74%	94.93%	95.32%
C=4	96.21%	95.80%	94.99%	95.39%
C=5	**96.37%**	**95.98%**	**95.22%**	**95.59%**
C=6	96.37%	95.93%	95.27%	95.59%
C=7	96.37%	95.93%	95.27%	95.59%
C=8	96.34%	95.91%	95.22%	95.56%

Tabel 5 tidak ada perubahan signifikan yang dihasilkan setelah nilai *cost* yang digunakan lebih besar dari 5 pada penelitian ini. Dapat disimpulkan jika *margin* antara *support vector* dan *hyperplane* yang dihasilkan sudah optimal karena sudah sedikit data yang berada didalam *margin* tersebut. Selanjutnya parameter yang didapat dari tahap ini akan digunakan untuk percobaan skenario keempat.

Tabel 6 Pengujian parameter frame

Frame	Akurasi	Presisi	Recall	F1-Score
F=15	96.37%	95.98%	95.22%	95.59%
F=30	98.18%	97.91%	97.71%	97.81%
F=45	**98.62%**	**98.19%**	**98.55%**	**98.37%**

Tabel 6 menunjukkan bahwa semakin banyak *frame* pada ranah temporal maka semakin bagus hasil yang didapatkan ini dikarenakan tekstur yang didapat mengandung informasi gerakan dari api sehingga lebih memisahkan objek api dan objek bukan api tetapi memiliki warna seperti api, Parameter untuk mencoba banyak *frame* hanya pada rentang [15,45], ini dikarenakan *dataset* video yang digunakan pada penelitian ini memiliki *frame rate* dengan durasi yang berbeda-beda, jika nilai parameter *frame* yang digunakan melebihi dari rentang tersebut akan ada video yang tidak terproses karena tidak memenuhi syarat. Selanjutnya seluruh parameter yang sudah didapat akan digunakan pada skenario kelima.

Seluruh parameter yang didapat dari tahap pengujian 1-4, yaitu radius 1, tetangga 8, dan *frame* 45 pada LBP-TOP dengan model klasifikasi SVM yang menggunakan parameter *kernel linear* dengan nilai *cost* 5 akan diuji pada data *test* untuk mendapatkan nilai akurasi, presisi, *recall*, dan *f1-score* sebagai acuan performa sistem yang dapat dilihat pada tabel 7.

Tabel 7 Pengujian parameter 1-4 pada data test

Akurasi	Presisi	Recall	F1-Score
90.94%	89.48%	89.00%	89.23%

Terdapat *gap* sebesar 8% antara akurasi *train* dan *test*, ini kemungkinan besar dikarenakan model SVM merupakan model *machine learning* tradisional sehingga ada keterbatasan dalam mengatasi *dataset imbalance* sehingga membutuhkan metode tambahan untuk mengurangi *gap* yang terjadi.

V. KESIMPULAN

Dari penelitian yang dilakukan, sistem yang dibangun dapat mendeteksi dan mengklasifikasikan api dengan baik. Ruang warna HSV dengan rule yang diimplementasikan sudah mampu mendeteksi area api dan memisahkannya dengan latar, kombinasi ekstraksi fitur menggunakan LBP-TOP dengan parameter radius 1, neighbours 8, dan nilai *frame* 45 serta pembelajaran mesin menggunakan SVM dengan *kernel linear* merupakan parameter terbaik dalam penelitian ini.

Untuk penelitian selanjutnya diharapkan mampu mengatasi *data imbalance* dengan menerapkan *sampling* pada *dataset*, menambahkan class *weight*, ataupun mencoba menggunakan model pembelajaran lain.

REFERENSI

[1] Maedeh Jamali, Nader Karimi, Shadrokh Samavi, "Saliency Based Fire Detection Using Texture and Color Features". 2020 28th Iranian Conference on Electrical Engineering (ICEE). doi: 10.1109/ICEE50131.2020.926065

[2] Xiaopeng Hong, Yingyue Xu, Guoying Zhao, "LBP-TOP: A Tensor Unfolding Revisit". ACCV 2016: Computer Vision – ACCV 2016 Workshops pp 513–527. doi: 10.1007/978-3-319-54407-6_34

[3] Sandra Vieira, Walter Hugo Lopez Pinaya, Andrea Mechelli, "Chapter 2 - Main concepts in machine learning". Academic Press on Machine Learning Methods and Applications to Brain Disorders 2020, Pages 21-44. doi: https://doi.org/10.1016/B978-0-12-815739-8.00002-X

[4] Derek A. Pisner, David M. Schnyer, "Chapter 6 - Support vector machine". Academic Press on Machine Learning Methods and Applications to Brain Disorders 2020, Pages 101-121. doi: https://doi.org/10.1016/B978-0-12-815739-8.00006-7

[5] Chi Yuan, Zhixiang Liu, Youmin Zhang, "Fire Detection Using Infrared Images for UAV-based Forest Fire Surveillance". 2017 International Conference on Unmanned Aircraft Systems (ICUAS). doi: 10.1109/ICUAS.2017.7991306

[6] Nurul Shakira Bakri, Ramli Adnan, Abd Manan Samad, Fazlina Ahmat Ruslan, "A methodology for fire detection using colour pixel classification". 2018 IEEE 14th International Colloquium on Signal Processing & Its Applications (CSPA). doi: 10.1109/CSPA.2018.8368692

[7] Pasquale Foggia, Alessia Saggese, Mario Vento, "Real-Time Fire Detection for Video-Surveillance Applications Using a Combination of Experts Based on Color, Shape, and Motion". IEEE Transactions on Circuits and Systems for Video Technology (Volume: 25, Issue: 9, September 2015). doi: 10.1109/TCSVT.2015.2392531

[8] Gwangsu Kim, Junyeong Kim, SungHwan Kim, "Fire Detection Using Video Images and Temporal Variations". 2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC). doi: 10.1109/ICAIIC.2019.8669083

[9] Jinkyu Ryu, Dongkurl Kwak, "Flame Detection Using Appearance-Based Pre-Processing and Convolutional Neural Network". 2021 Applied Science, mdpi. doi: https://doi.org/10.3390/app11115138

[10] Esa Prakasa, "Texture Feature Extraction by Using Local Binary Pattern". 2016 Jurnal INKOM 9(2):45. doi: 10.14203/j.inkom.420

[11] K. Ting, "Confusion Matrix". 2010 Encyclopedia of Machine Learning and Data Mining. doi: 10.1007/978-1-4899-7687-1_50

[12] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, "A Practical Guide to Support Vector Classification". 2016 Technical Report, Department of Computer Science, National Taiwan University.