DETEKSI API BERBASIS SENSOR VISUAL MENGGUNAKAN METODE SUPPORT VECTOR MACHINES

Penyusun Tugas Akhir :

Hamdi Ahmadi Muzakkiy

(5112 100 091)

Dosen Pembimbing :

Prof. Ir. Handayani Tjandrasa, M.Sc., Ph.D.

Dr. Eng. Chastine Fatichah, S.Kom, M.Kom

PENDAHULUAN

RANCANGAN & IMPLEMENTASI

SKENARIO UJI COBA

KESIMPULAN & SARAN

LATAR BELAKANG

- Sensor api biasa yang sering digunakan lambat dalam mendeteksi api, menunggu partikel menyentuh sensor
- Sensor biasa sulit untuk mendeteksi api diluar ruangan
- Pemanfaatan kamera CCTV pada gedung-gedung, sehingga tidak perlu memasang alat pendeteksi api

RUMUSAN MASALAH

- Bagaimana melakukan deteksi gerak setiap frame ?
- Bagaimana melakukan deteksi warna api setiap piksel ?
- Bagaimana menghilangkan noise setiap region ?
- Bagaimana meakukan verifikasi piksel api ?

BATASAN MASALAH

- Implementasi dilakukan dengan bahasa pemrograman Python
- Jumlah piksel objek api yang dideteksi lebih besar dari 1% dari luas piksel frame
- Data yang digunakan adalah data video dengan panjang video 6-16 detik
- Data video memiliki ukuran 240 x 320 piksel dengan *channel* R,G,B

BATASAN MASALAH

- Warna api yang didefinisikan adalah *range* warna kuning hingga merah
- Pergerakan dari kamera tidak terlalu besar
- Pantulan objek api termasuk kedalam objek api
- Data video berasal dari KMU *Fire & Smoke Database*, video *open source*, MIVIA *fire dataset*, dan video rekaman

TUJUAN

- Merancang dan membangun perangkat lunak deteksi api menggunakan data video secara *real time*

PENDAHULUAN

RANCANGAN & IMPLEMENTASI

SKENARIO UJI COBA

KESIMPULAN & SARAN

DIAGRAM ALIR PROSES UTAMA

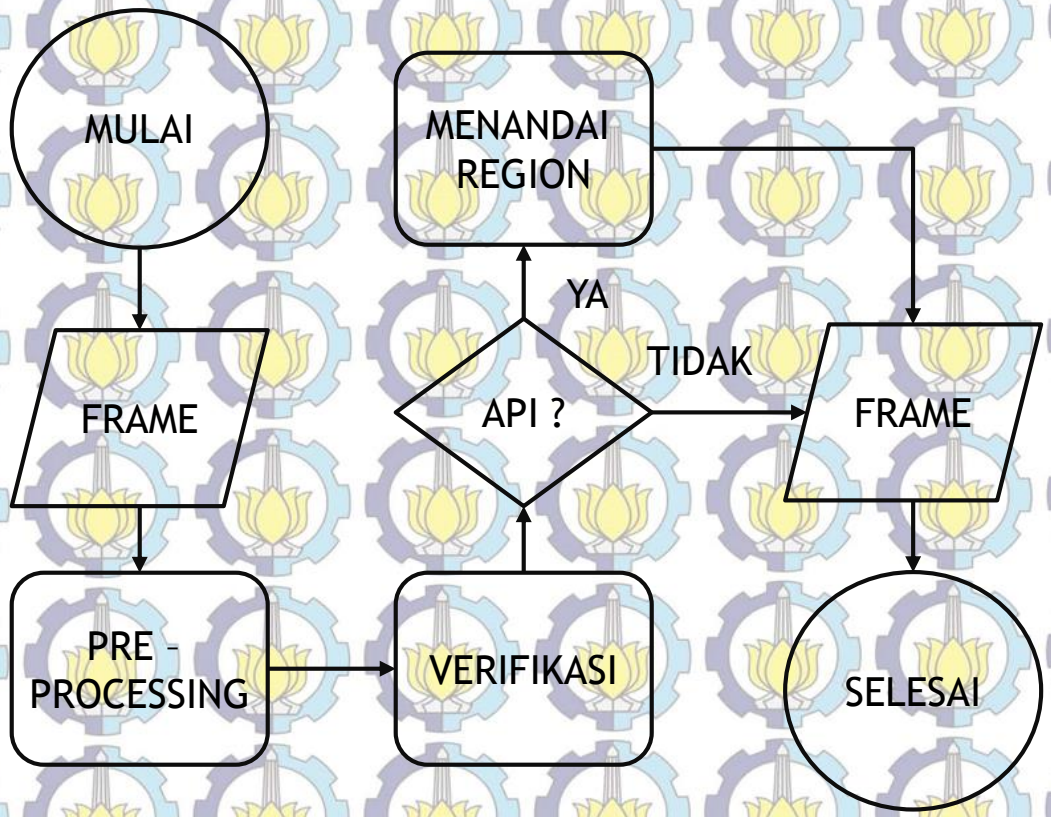


DIAGRAM ALIR PROSES UTAMA

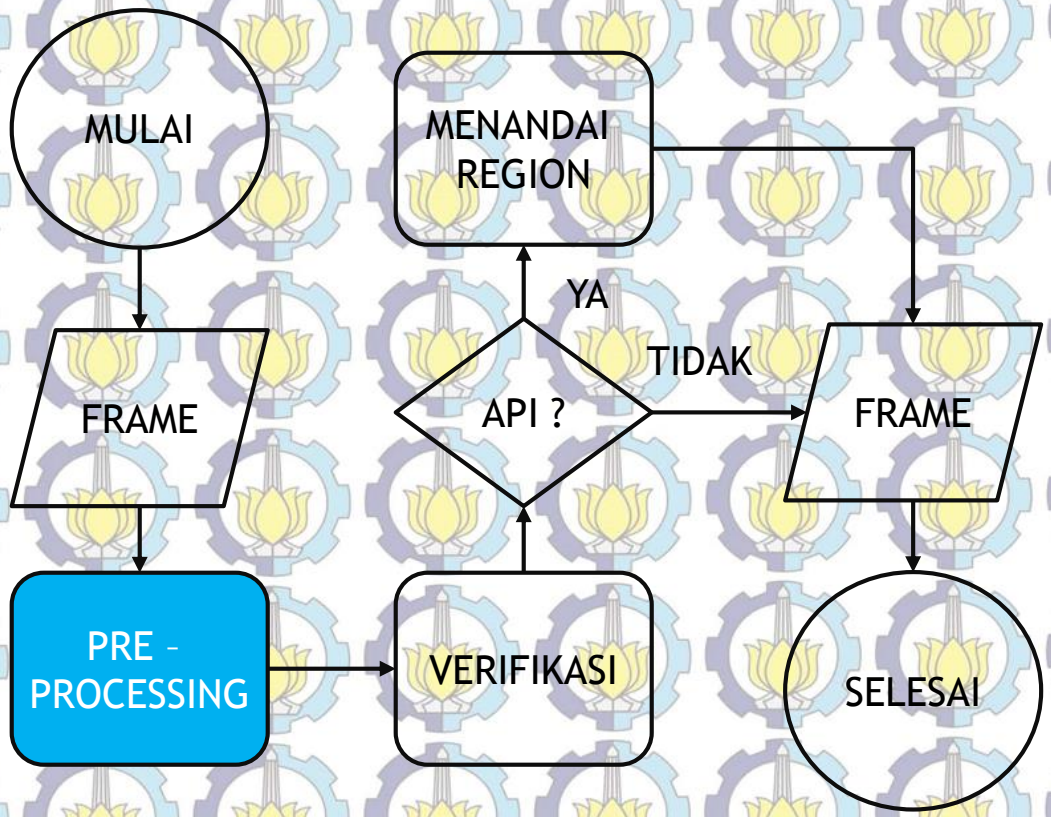
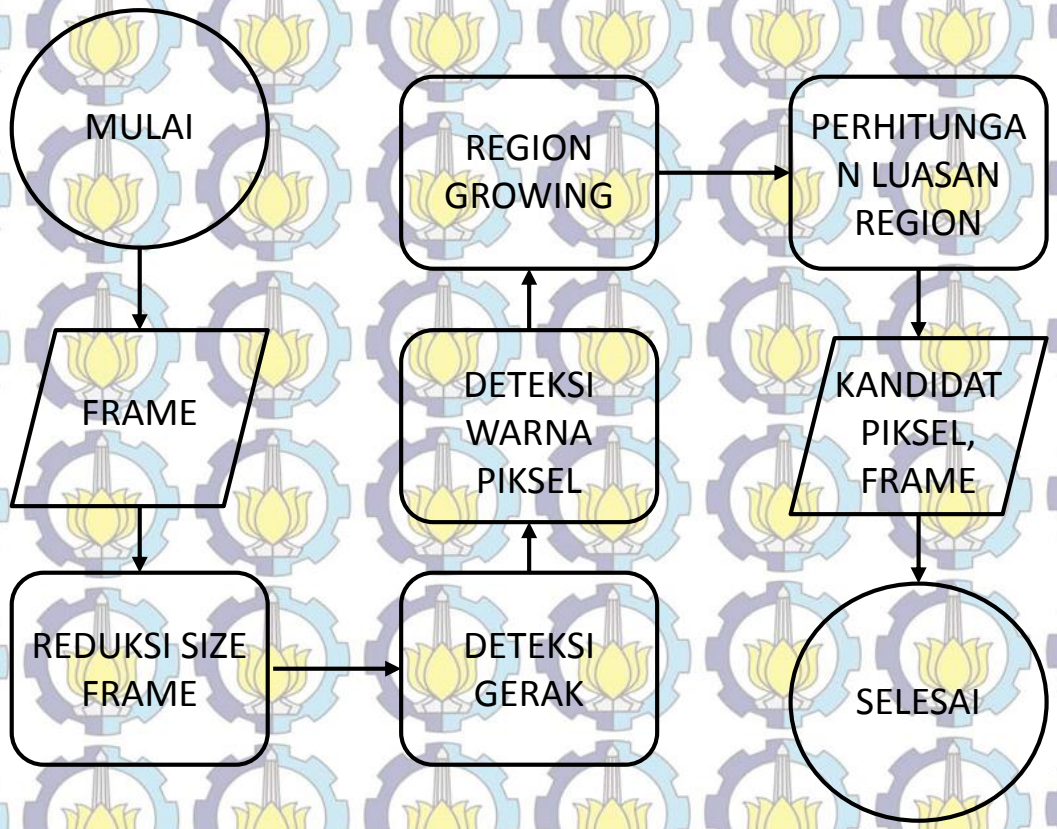


DIAGRAM ALIR PREPROCESSING



REDUKSI SIZE FRAME

- Melakukan reduksi *size frame* yang sedang diproses
- Menggunakan *gaussian pyramind*

$$g_1(i,j) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n) \, . \, g_0\,(2i+m, 2j+n)$$

i

REDUKSI SIZE FRAME

DETEKSI GERAK

- Menggunakan *Gaussian Mixture Model*
- Setiap piksel mempunyai K model (jumlah K berkisar antara 3-5)
- Setiap piksel mempunyai K nilai w, μ, σ dimana :
  - w adalah bobot
  - μ adalah rata-rata/*mean*
  - σ adalah standar deviasi

i

DETEKSI GERAK

- Setiap piksel akan dibandingkan dengan *background model*
- *Background model* didapatkan menggunakan persamaan berikut :

$$B = argmin(\sum_{j=1}^{b} w_j > T)$$

DETEKSI GERAK



Frame N | Frame N+1 | Frame N+2



DETEKSI WARNA PIKSEL

- Menggunakan probabilitas distribusi Gaussian
- Setiap nilai R,G,B piksel akan dihitung nilai probabilitas nya. Nilai probabilitas suatu piksel dihitung menggunakan persamaan

$$p = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

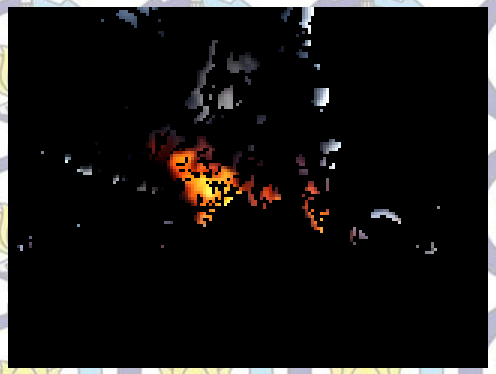
i

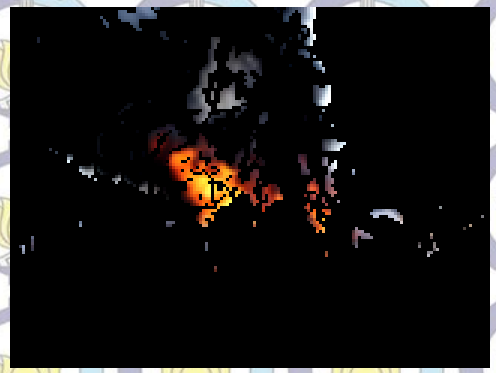
DETEKSI WARNA PIKSEL

- Setelah mendapatkan nilai probabilitas R,G,B dilakukan perhitungan probabilitas piksel menggunakan persamaan berikut.

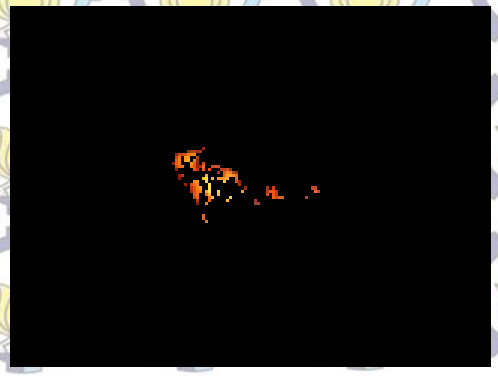
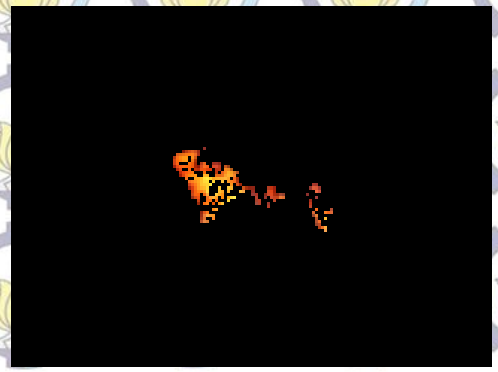
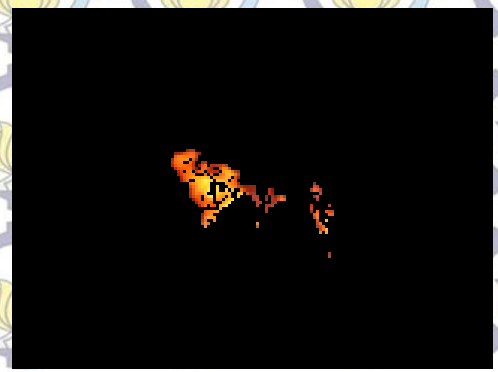
$$p(I(x,y)) = \prod_{i \in \{R,G,B\}} p_i(I_i(x,y))$$

DETEKSI WARNA PIKSEL





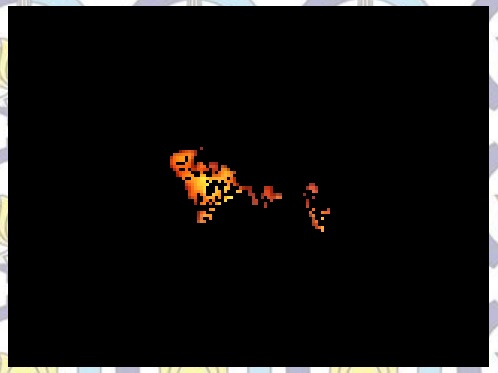
Frame N | Frame N+1 | Frame N+2

REGION GROWING

- Digunakan untuk mendapatkan *region api*
- Inisialisasi *seed* didapatkan dari kandidat piksel api yang dihasilkan pada tahap deteksi warna piksel
- Tingkat homogen piksel dilihat dari probabilitas warna piksel tetangga

REGION GROWING


Frame N | Frame N+1 | Frame N+2



PERHITUNGAN LUASAN REGION

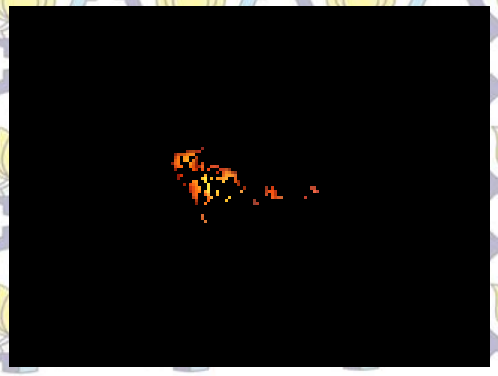
- Digunakan untuk menghilangkan noise
- Menggunakan luasan yang didapatkan pada proses region growing
- Jika luasan region > 1% total piksel frame diangga api

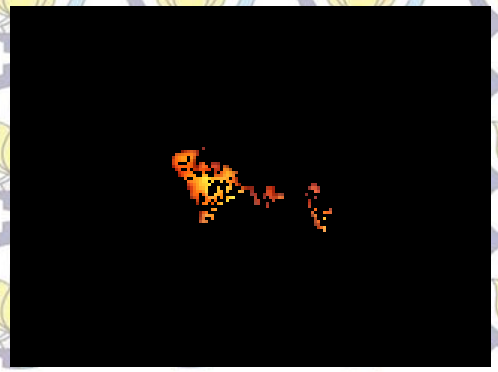
PERHITUNGAN LUASAN REGION

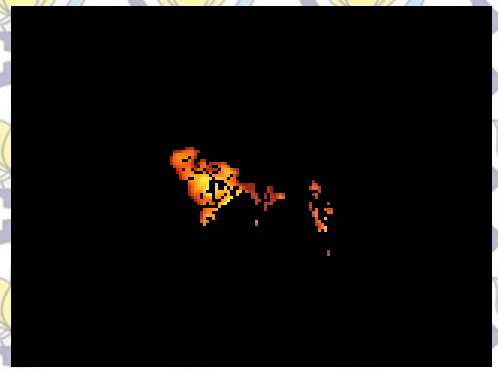
Frame N

Frame N+1

Frame N+2







FRAME MASUKAN

MULAI

SELESAI

DETEKSI GERAK

DETEKSI WARNA PIKSEL

REGION GROWING

DIAGRAM ALIR PROSES UTAMA

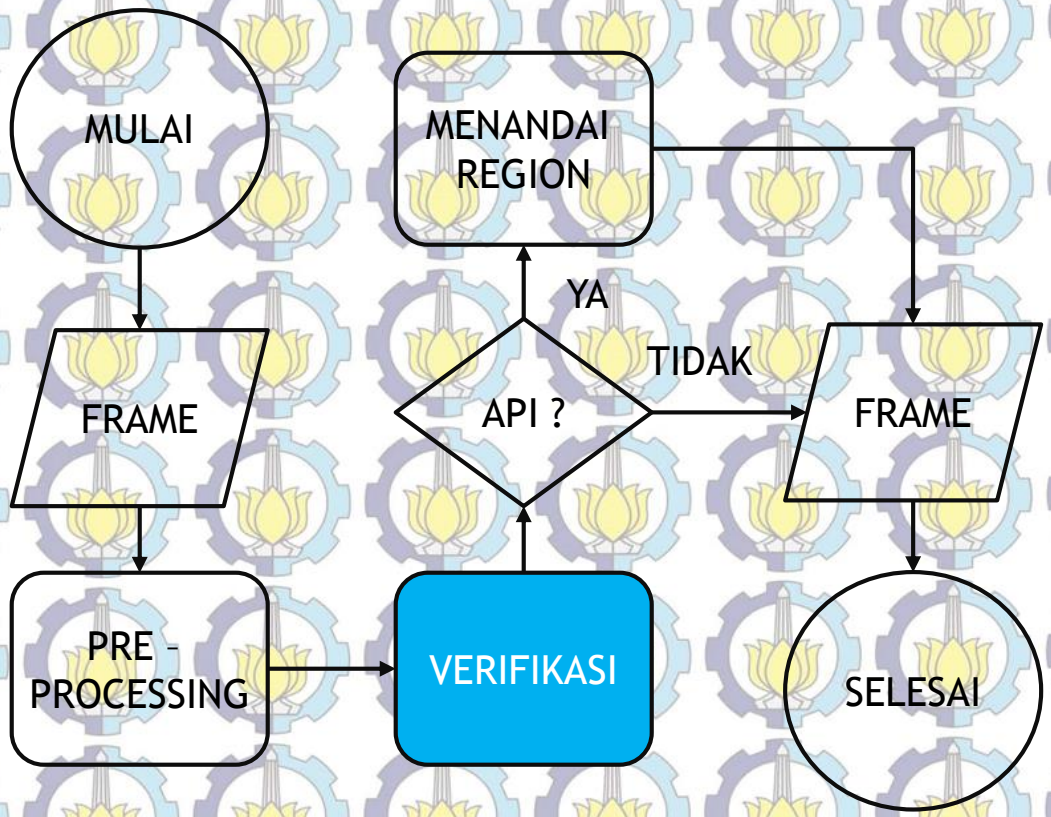
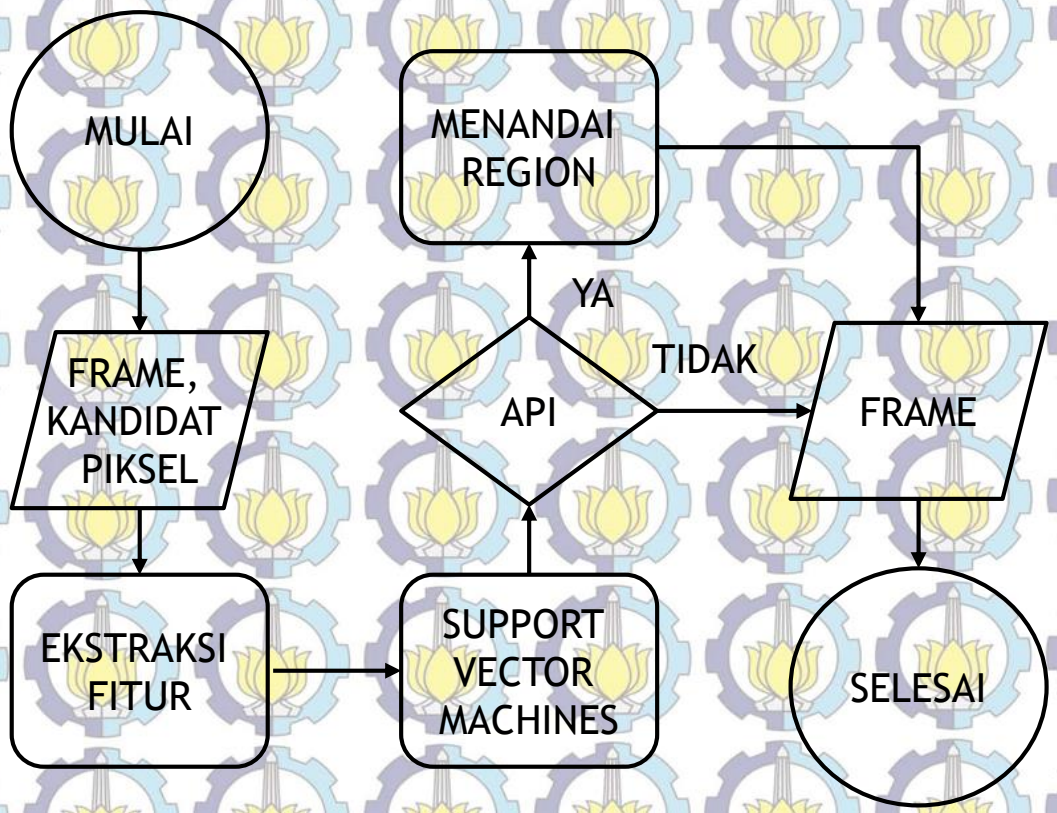
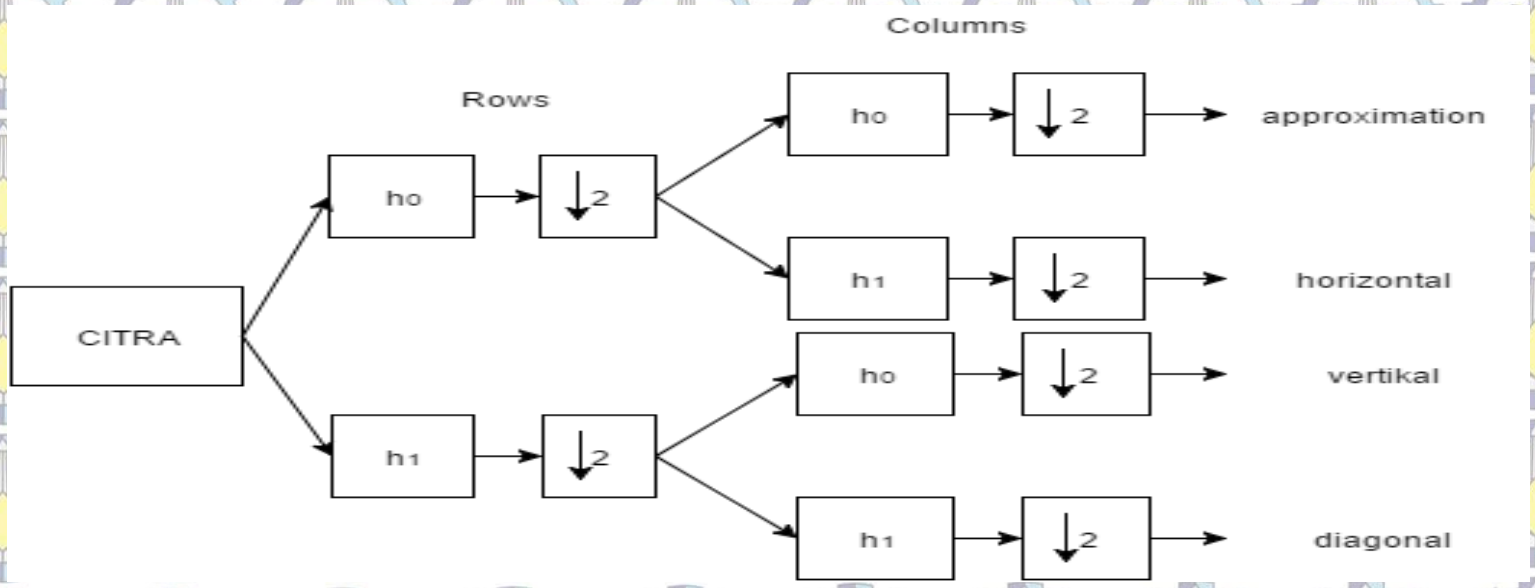


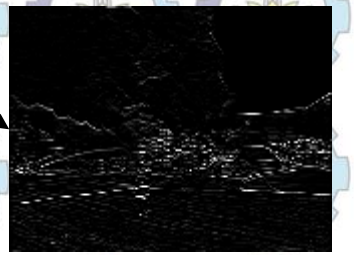
DIAGRAM ALIR VERIFIKASI

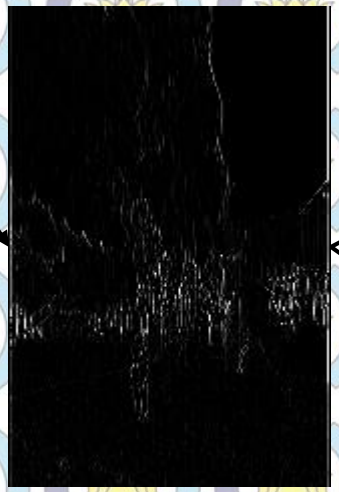


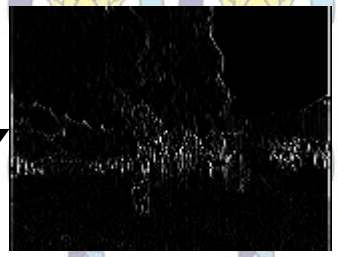
EKSTRAKSI FITUR

- Menggunakan wavelet daubachies 4









EKSTRAKSI FITUR

- Citra yang dilakukan wavelet sebanyak 10 buah, yaitu frame yang sedang diproses dan 9 citra sebelumnya secara berututan
- List piksel yang lolos tahap preprocessing akan dihitung nilai wavelet koordinat piksel tersebut menggunakan persamaan.

$$M_n(x,y) = |LH_n(x,y)|^2 + |HL_n(x,y)|^2 + |HH_n(x,y)|^2$$

- Nilai M_n didapat dari ekstraksi fitur sebanyak sepuluh buah

EKSTRAKSI FITUR

- Nilai fitur dilakukan normalisasi menggunakan normalisasi min max.
- Nilai min & max didapatkan dari nilai M_n min dan M_n max setiap frame

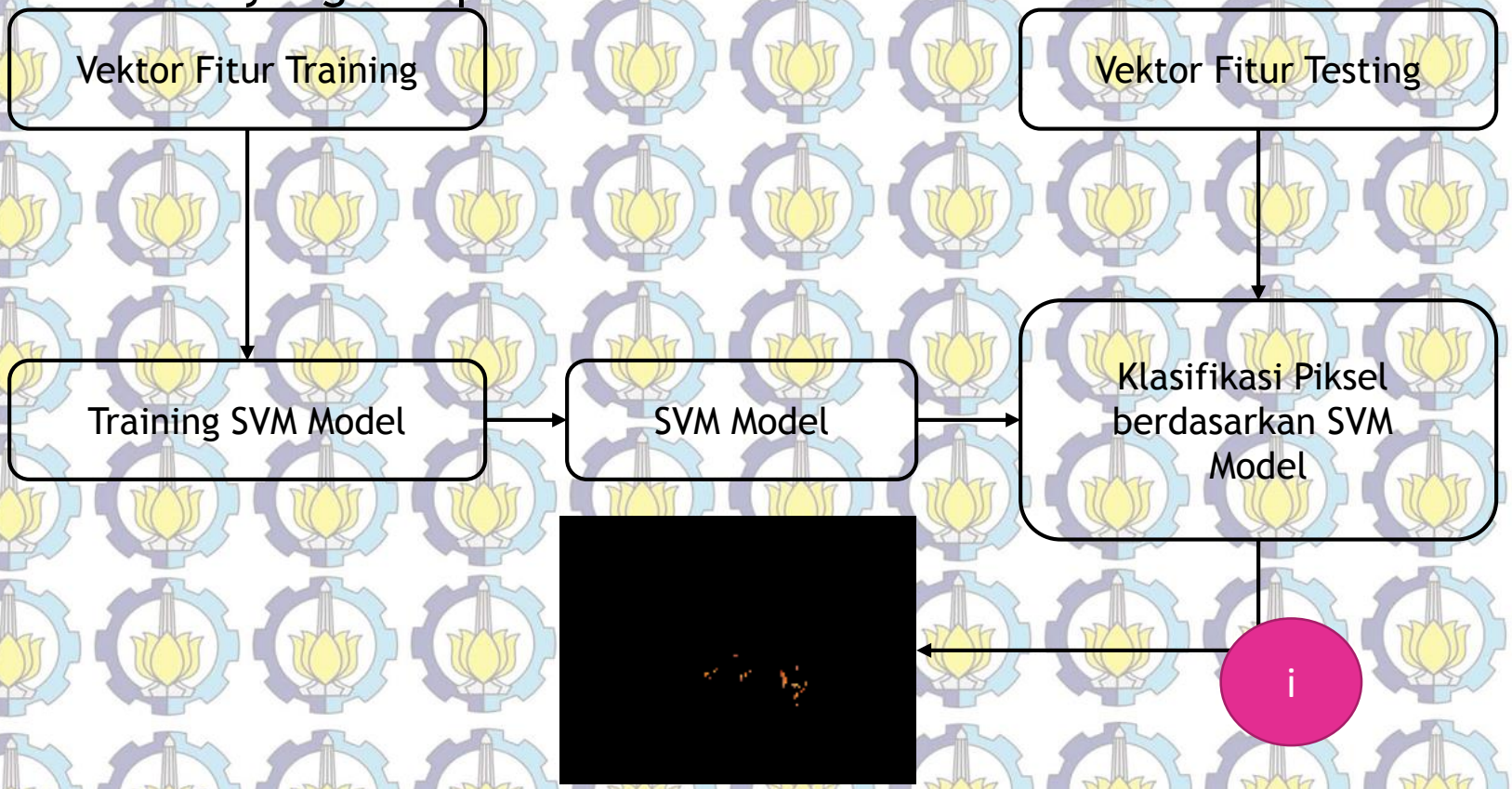
$$Y_{new} = \frac{(Y - Y_{min})}{Y_{max} - Y_{min}}$$

- Nilai fitur disorting secara ascending

i

SUPPORT VECTOR MACHINES

- Klasifikasi untuk melakukan verifikasi terhadap piksel api dari data fitur yang didapatkan

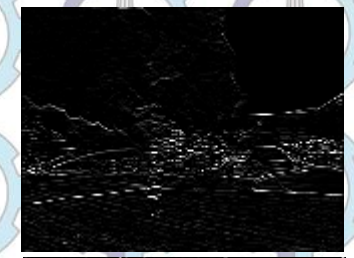


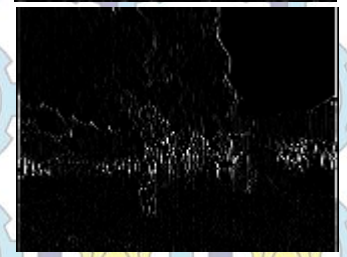
MENANDAI REGION

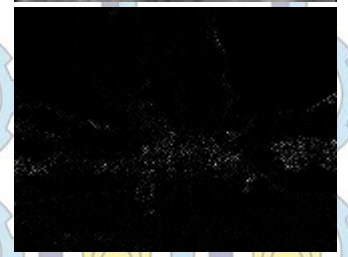
- Piksel-piksel yang lolos verifikasi akan ditandari sebagai piksel api.
- Digunakan titik ekstrim piksel api, yaitu :
  - X_{min}
  - Y_{min}
  - X_{max}
  - Y_{max}

MULAI

DATA MASUKAN







Ekstraksi Fitur

SVM

SELESAI

PENDAHULUAN

RANCANGAN & IMPLEMENTASI

SKENARIO UJI COBA

KESIMPULAN & SARAN

DATA UJI






38 Video Kejadian
34 Video Api
33 Video Bukan Api

DATA TRAINING

- 520 Data Fitur Api, 6 Video Api
- 932 Data Fitur Bukan Api, 4 Video Bukan Api

PARAMETER KEBENARAN

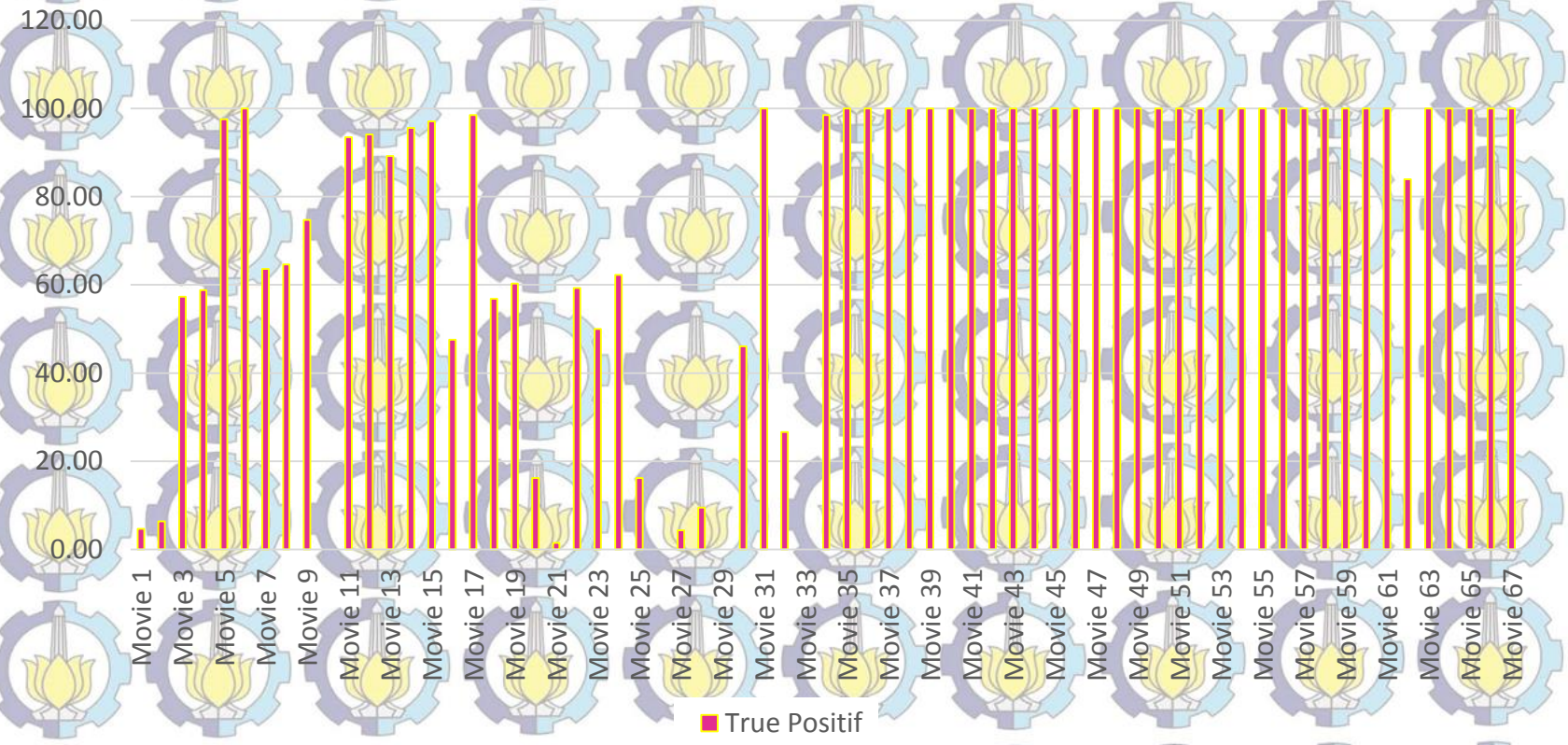
- True Positif
- False Positif
- Missing Rate

SKENARIO 1

- Variasi *Threshold* warna piksel api
  - 10^{-7}
  - 10^{-8}
  - 5×10^{-9}
  - 10^{-9}
- $C = 5$
- Kernel = RBF

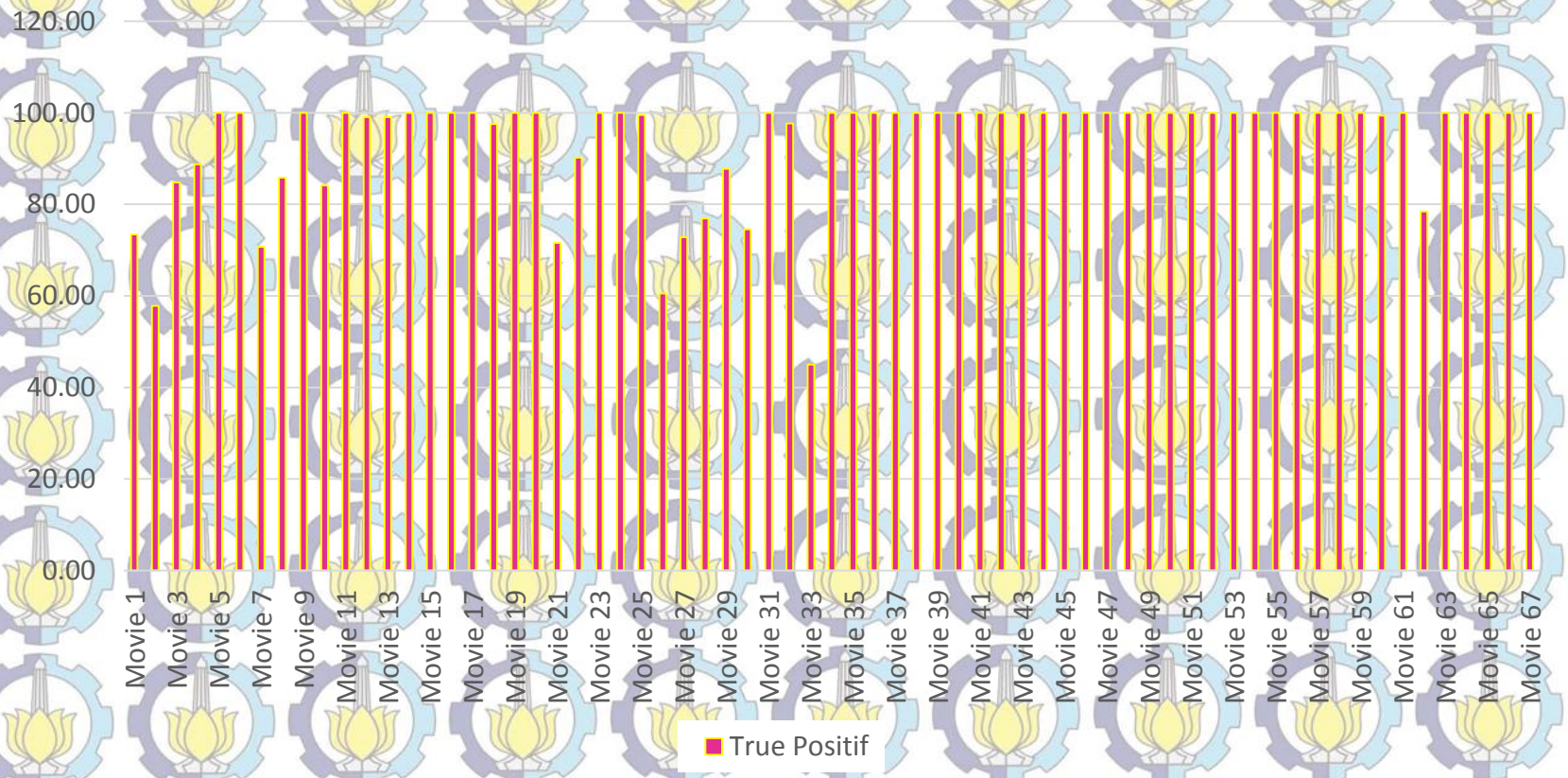
SKENARIO 1

Threshold $= 10^{-7}$



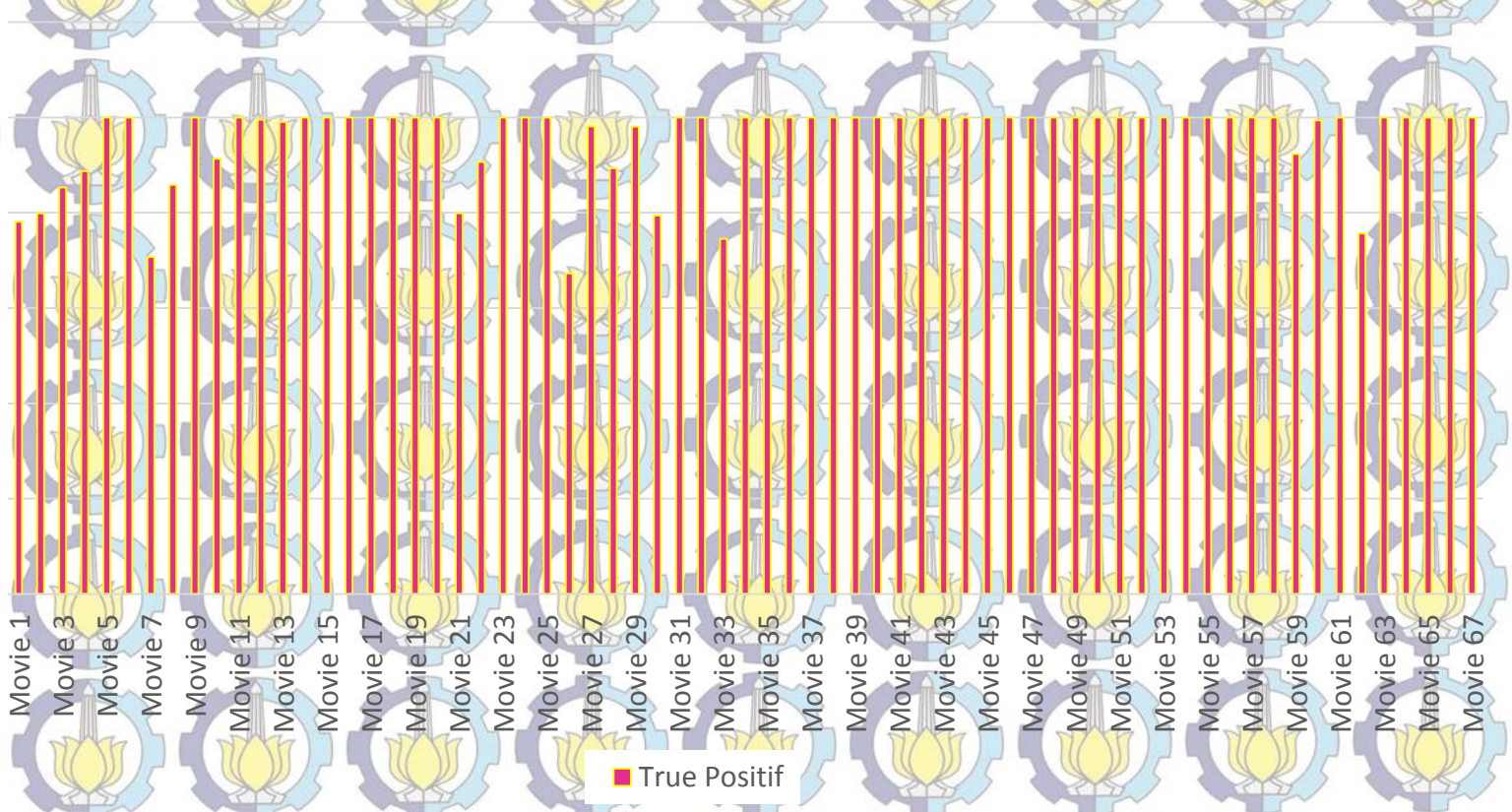
SKENARIO 1

Threshold = 10^{-8}



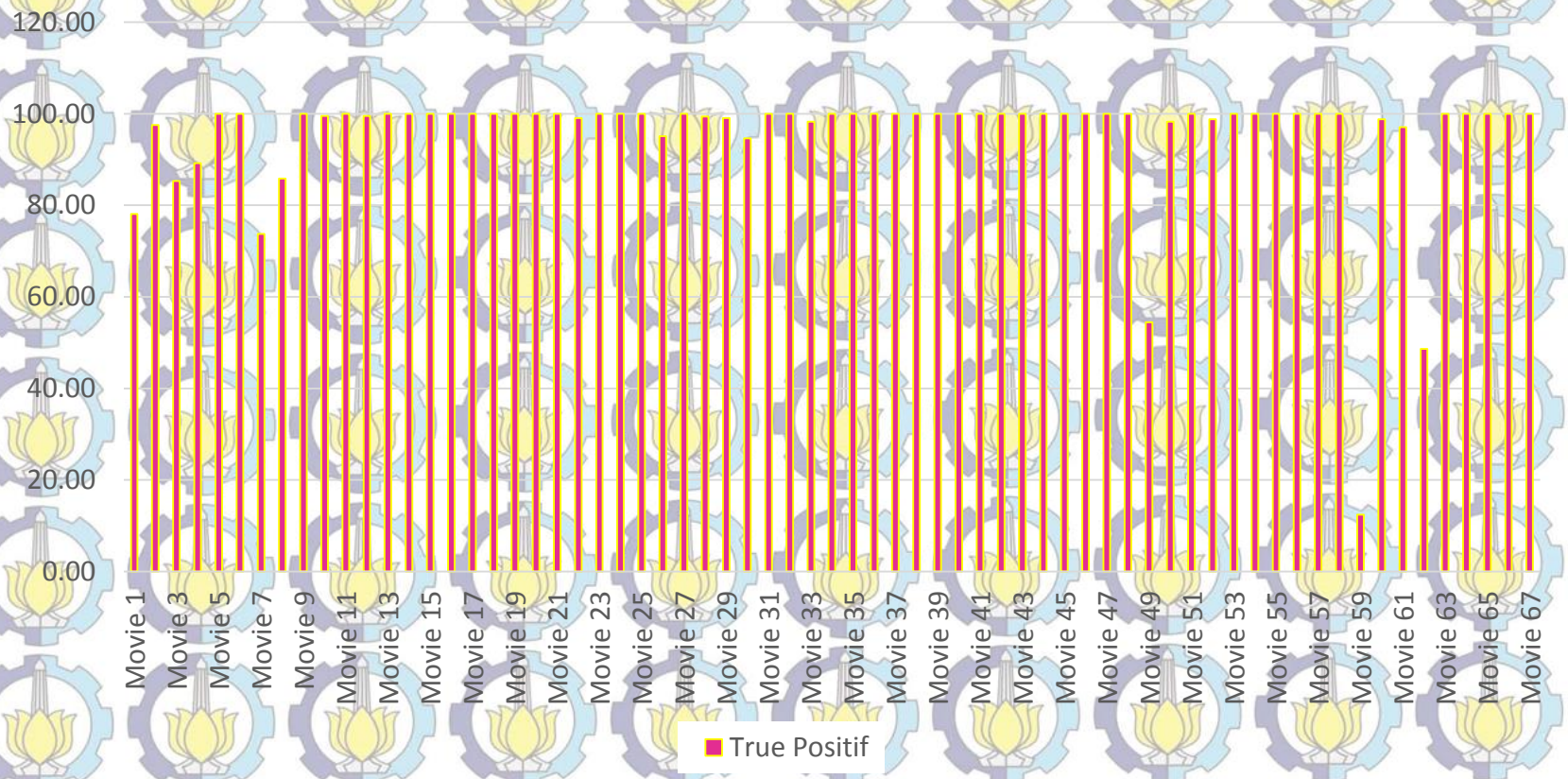
SKENARIO 1

Threshold = 5×10^{-9}



SKENARIO 1

Threshold = 10^{-9}



- Akurasi terbaik yang didapatkan ketika *threshold* = 5×10^{-9}

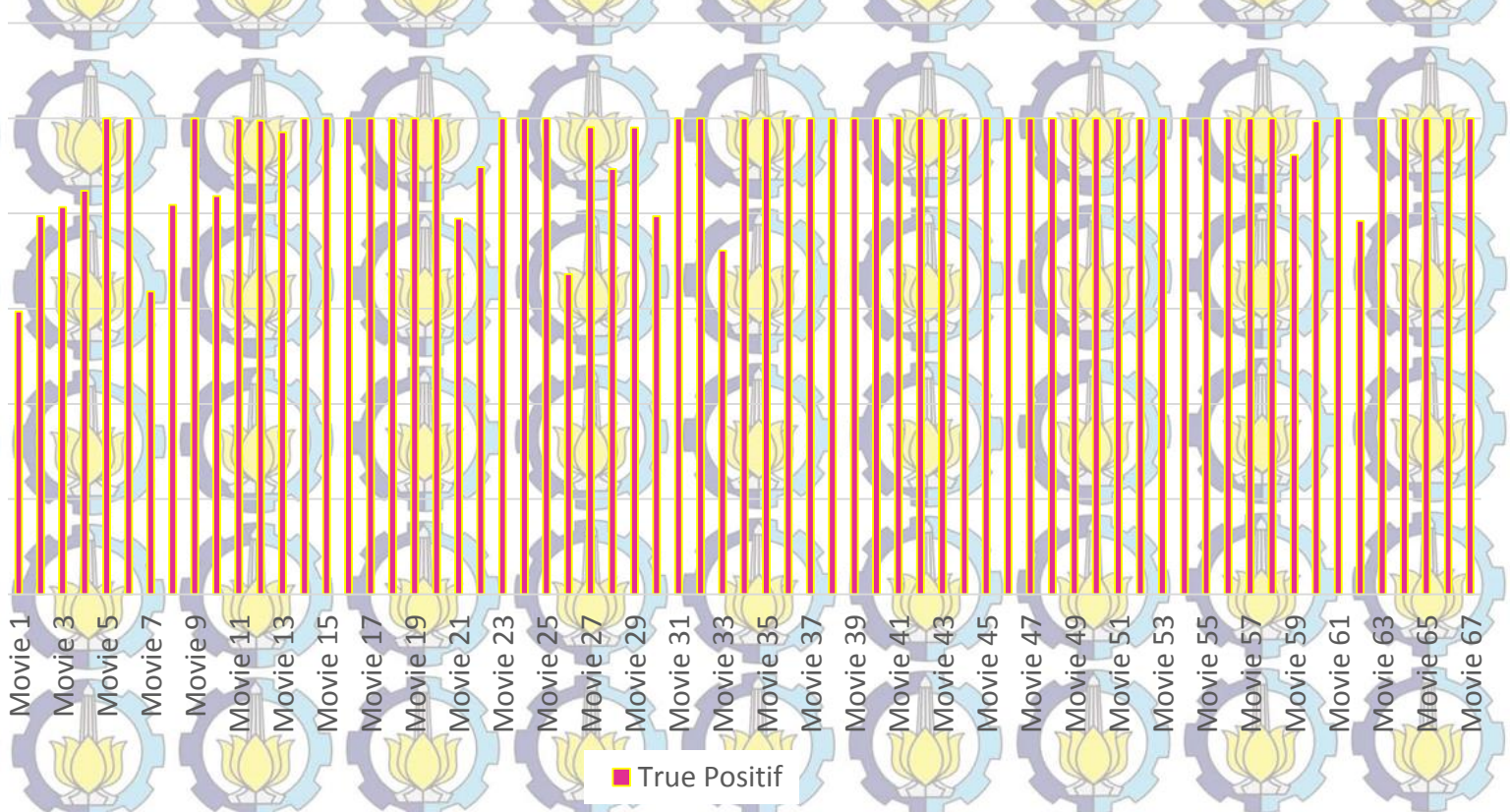
Threshold	True Positif (%)	False Positif (%)	Missing Rate (%)
10^{-7}	75.15	0.24	24.61
10^{-8}	93.96	0.33	5.71
5×10^{-9}	95.87	0.49	3.64
10^{-9}	95.56	2.86	1.58

SKENARIO 2

- Variasi C pada klasifikasi
  - 1
  - 3.5
  - 5
  - 7
- *Threshold* = 5×10^{-9}
- Kernel = RBF

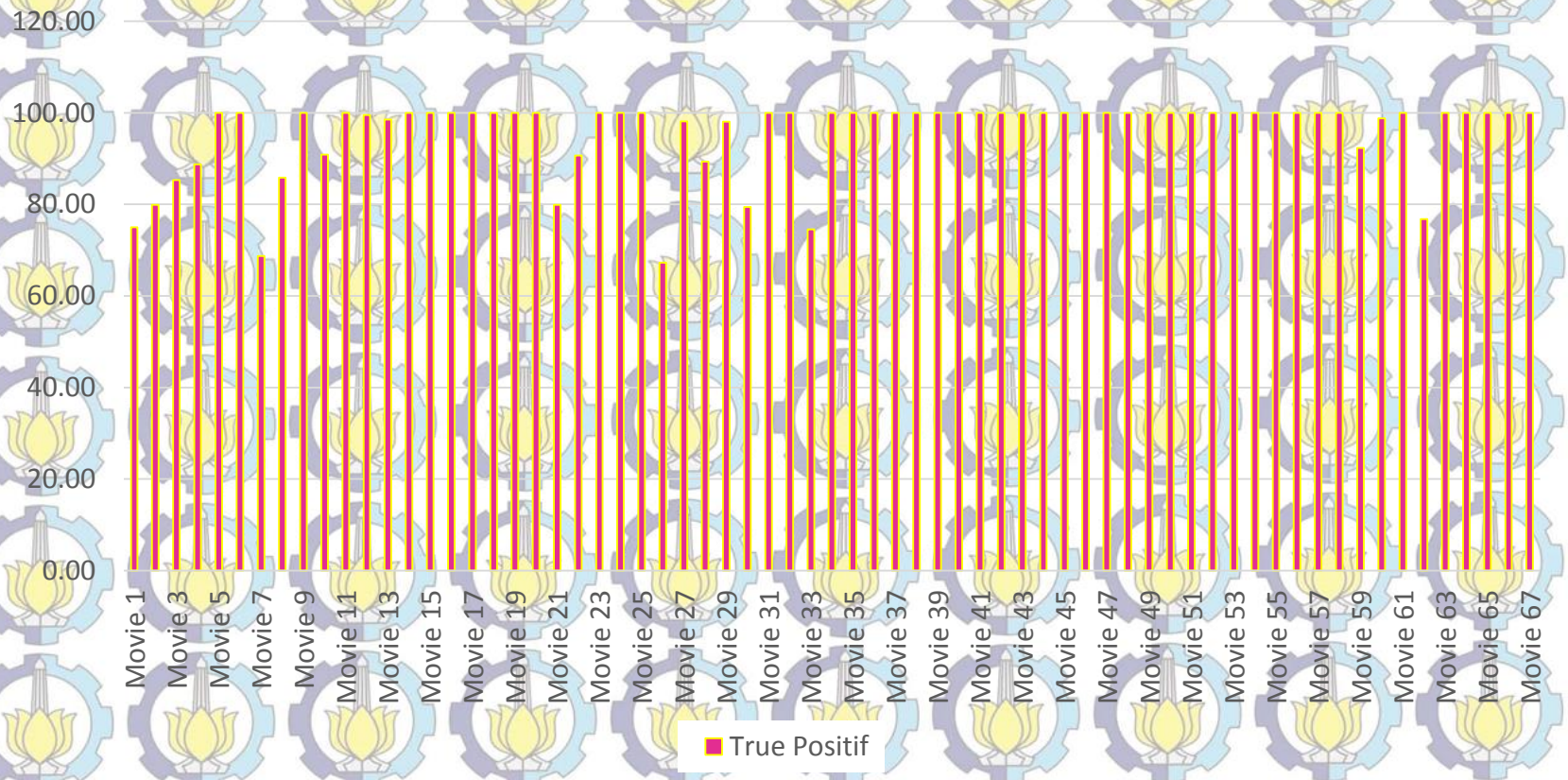
SKENARIO 2

$C = 1$



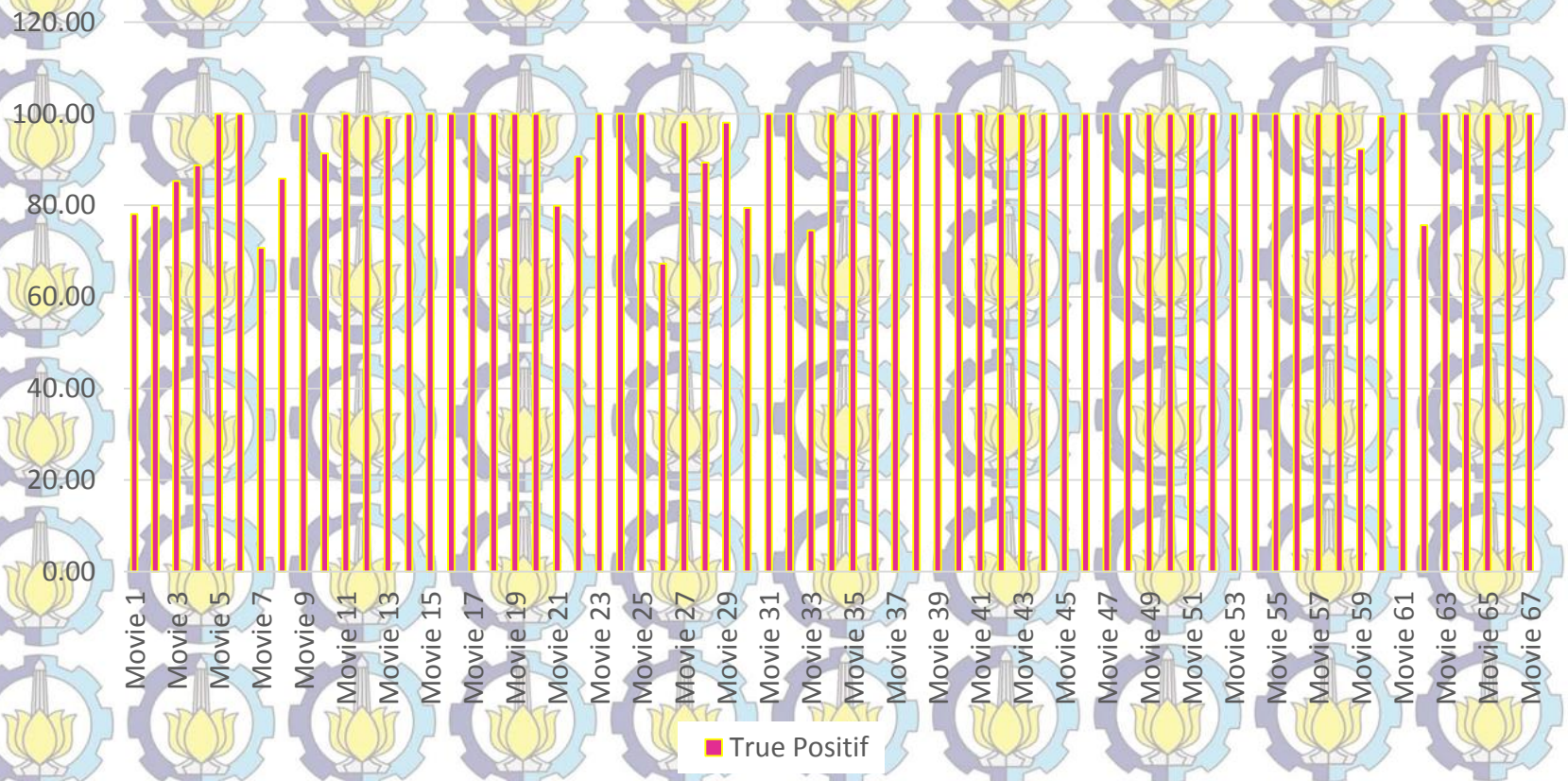
SKENARIO 2

$C = 3.5$



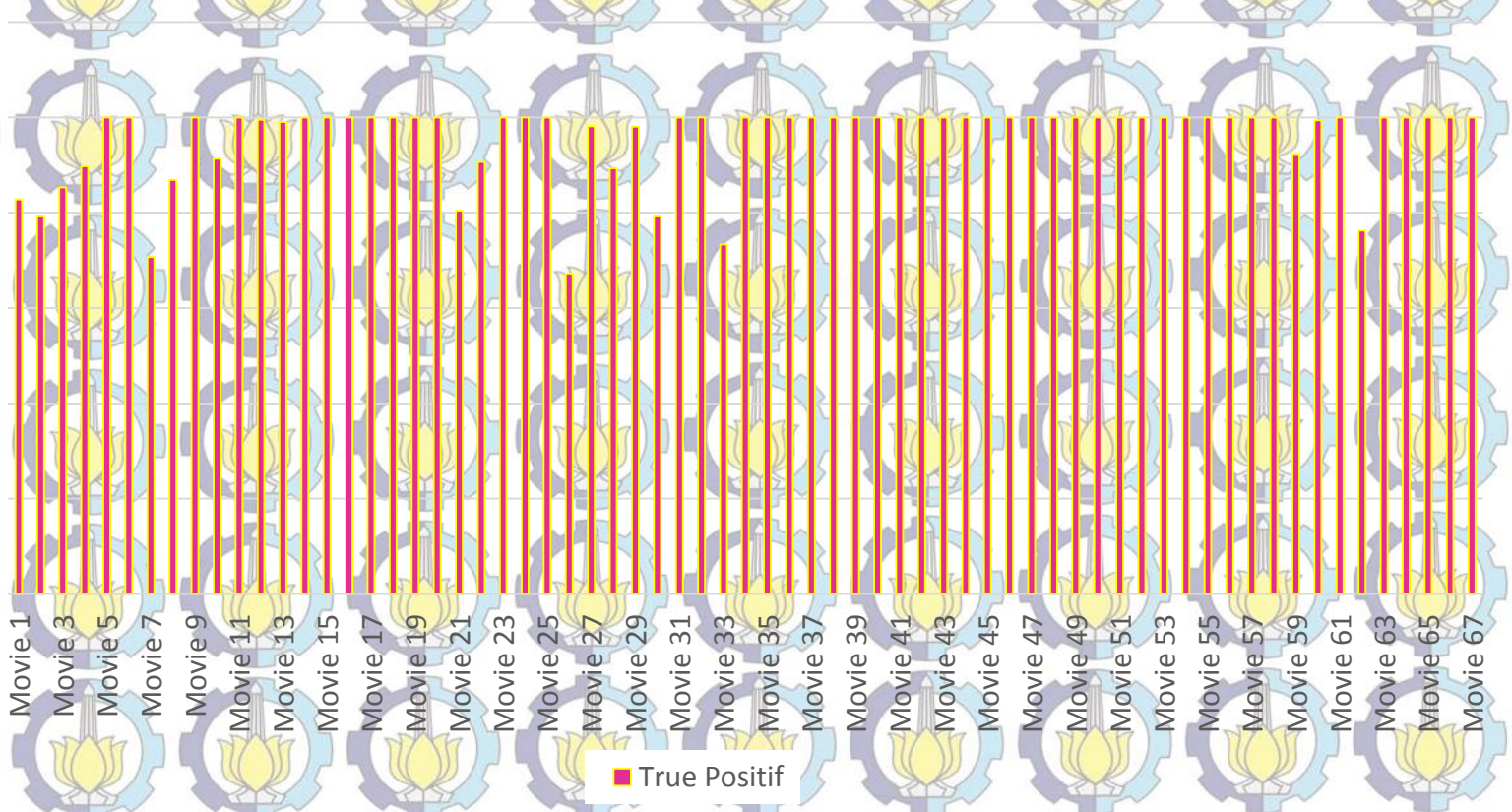
SKENARIO 2

$C = 5$



SKENARIO 2

$C = 7$



SKENARIO 2

- Akurasi terbaik yang didapatkan ketika C = 7

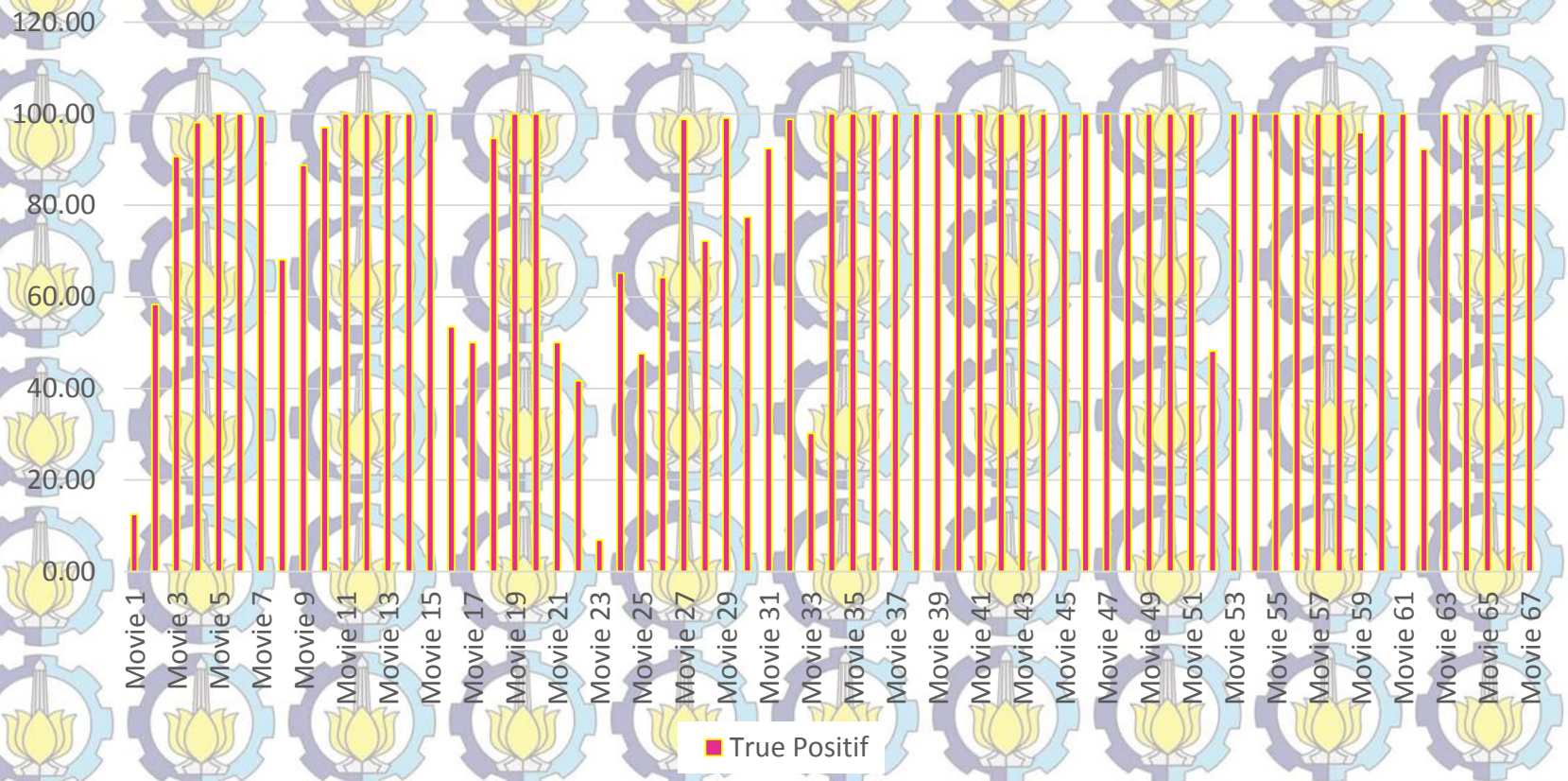
C	True Positif (%)	False Positif (%)	Missing Rate (%)
1	95.13	0.45	4.42
3.5	95.79	0.48	3.74
5	95.87	0.49	3.64
7	95.96	0.48	3.56

SKENARIO 3

- Variasi Kernel pada klasifikasi
  - *Polynomial 2*
  - *Polynomial 3*
  - RBF
- *Threshold* = 5×10^{-9}
- $C = 5$

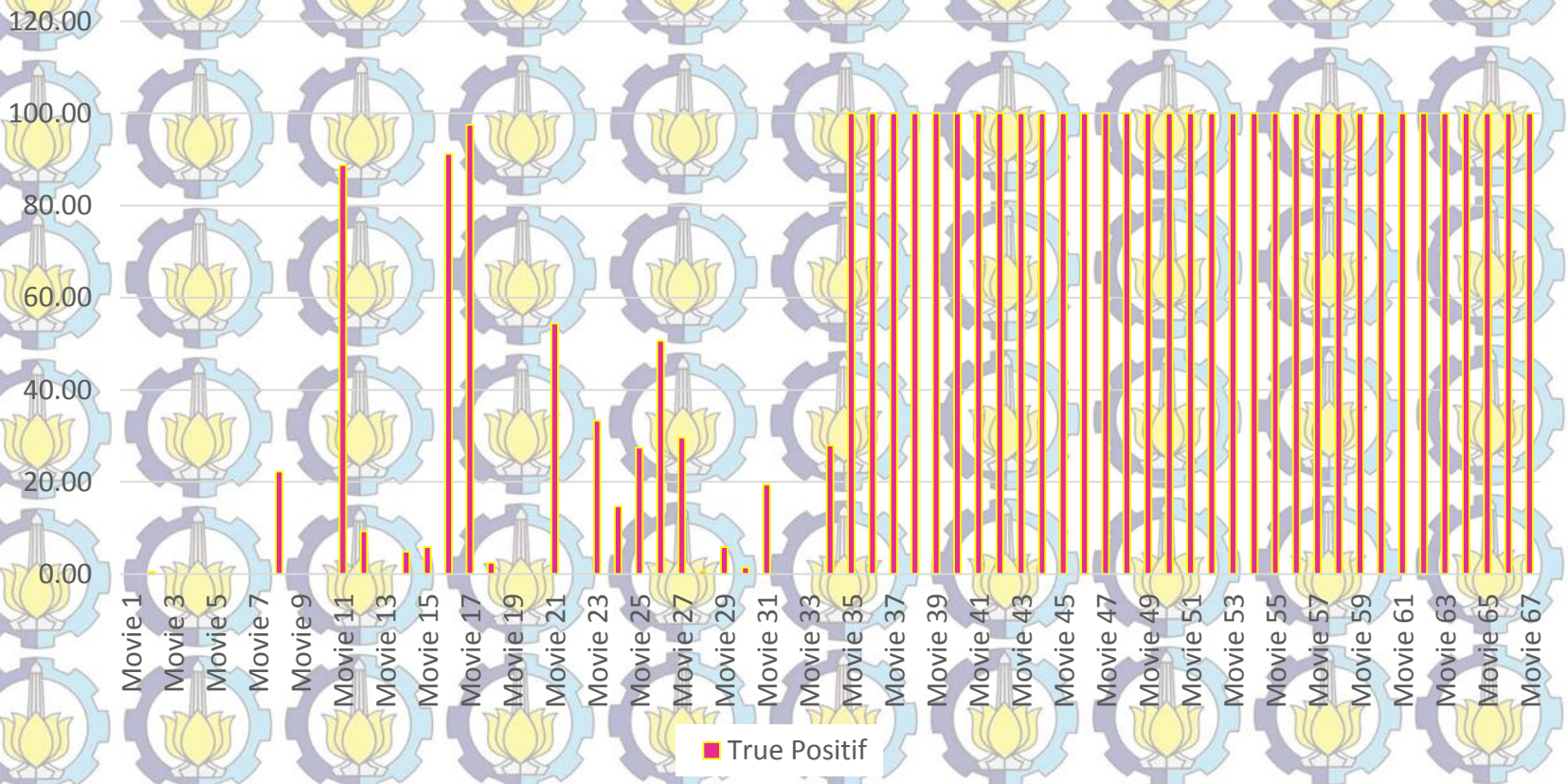
SKENARIO 3

Kernel = *Polynomial 2*



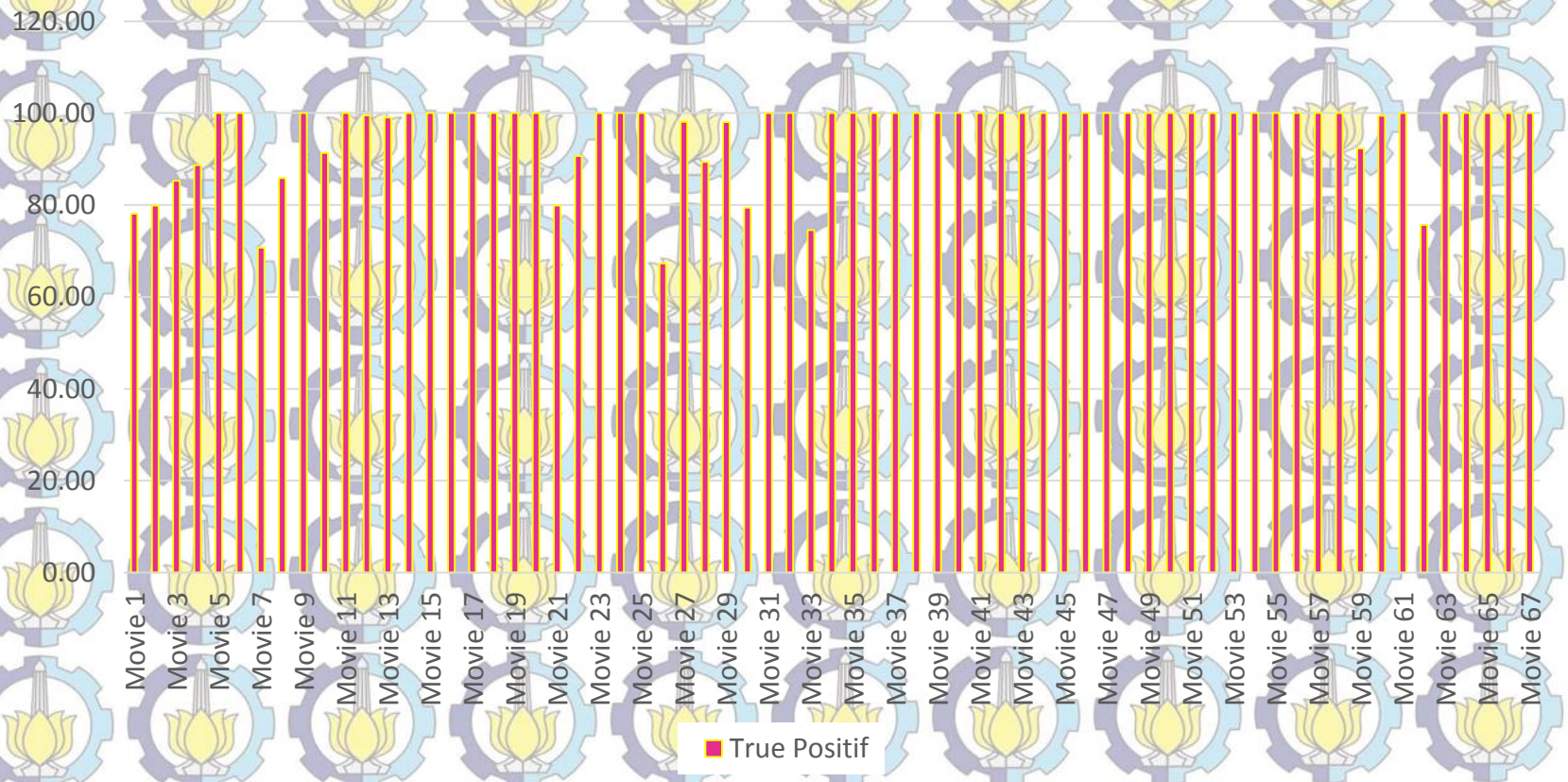
SKENARIO 3

Kernel = *Polynomial 3*



SKENARIO 3

Kernel = RBF



SKENARIO 3

- Akurasi terbaik yang didapatkan ketika Kernel yang digunakan adalah RBF

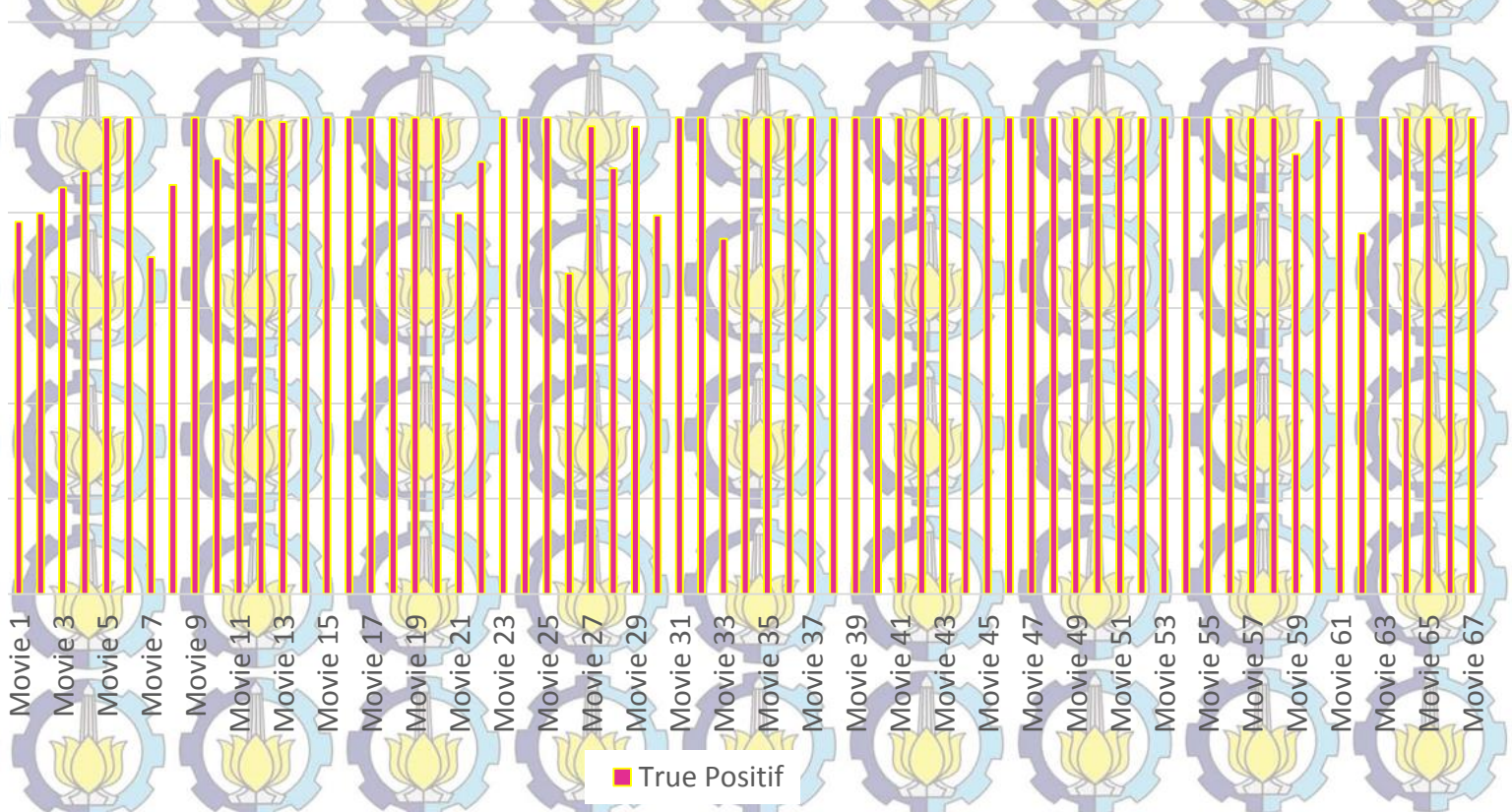
Kernel	True Positif (%)	False Positif (%)	Missing Rate (%)
Polynomial 2	87.93	0.95	11.12
Polynomial 3	58.03	0.00	41.97
RBF	95.87	0.49	3.64

SKENARIO 4

- Variasi Konstanta *Region*
  - 1%
  - 5%
  - 10%
- Kernel = RBF
- *Threshold* = 5×10^{-9}
- $C = 5$

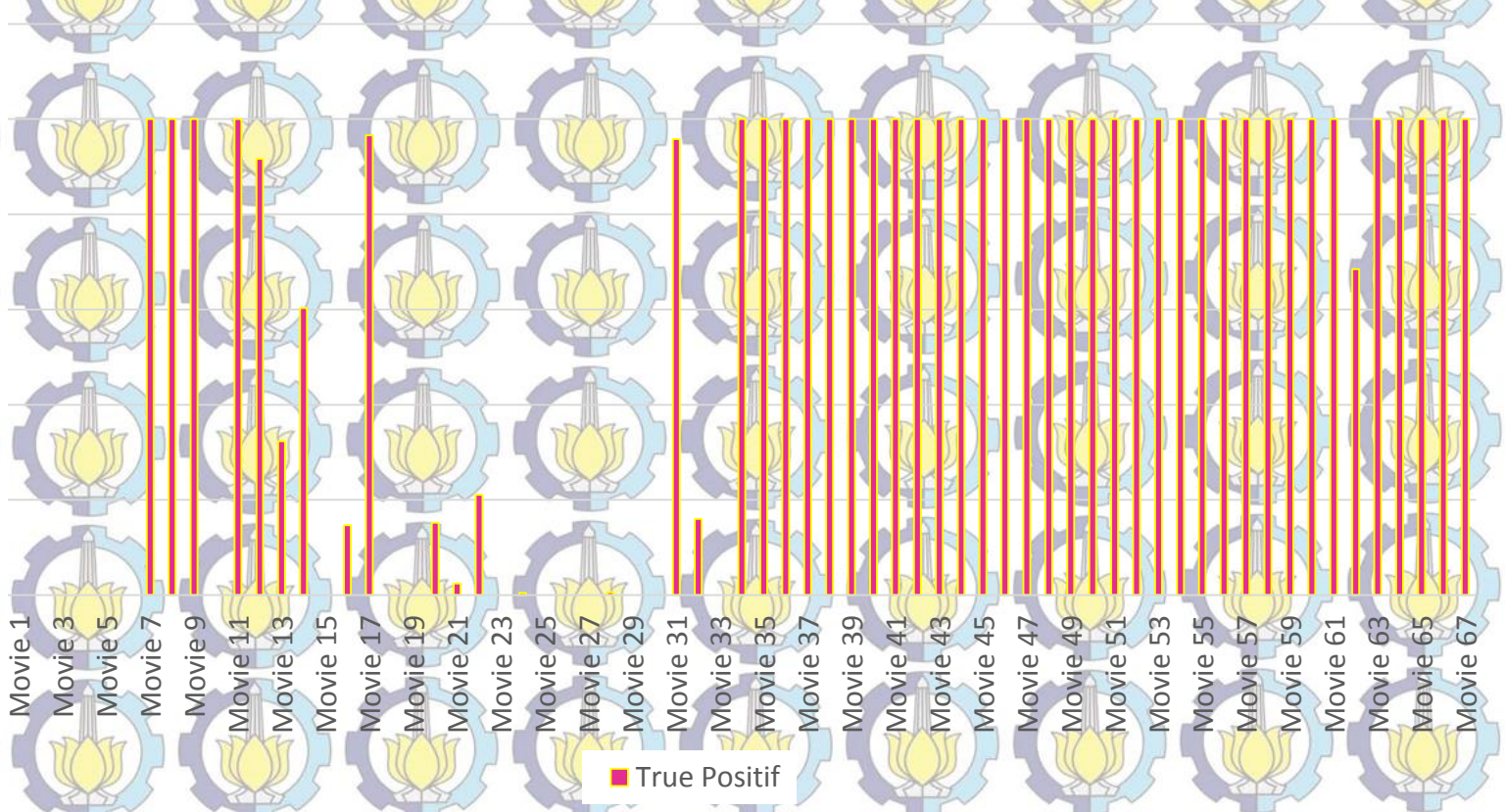
SKENARIO 4

Konstanta = 1%



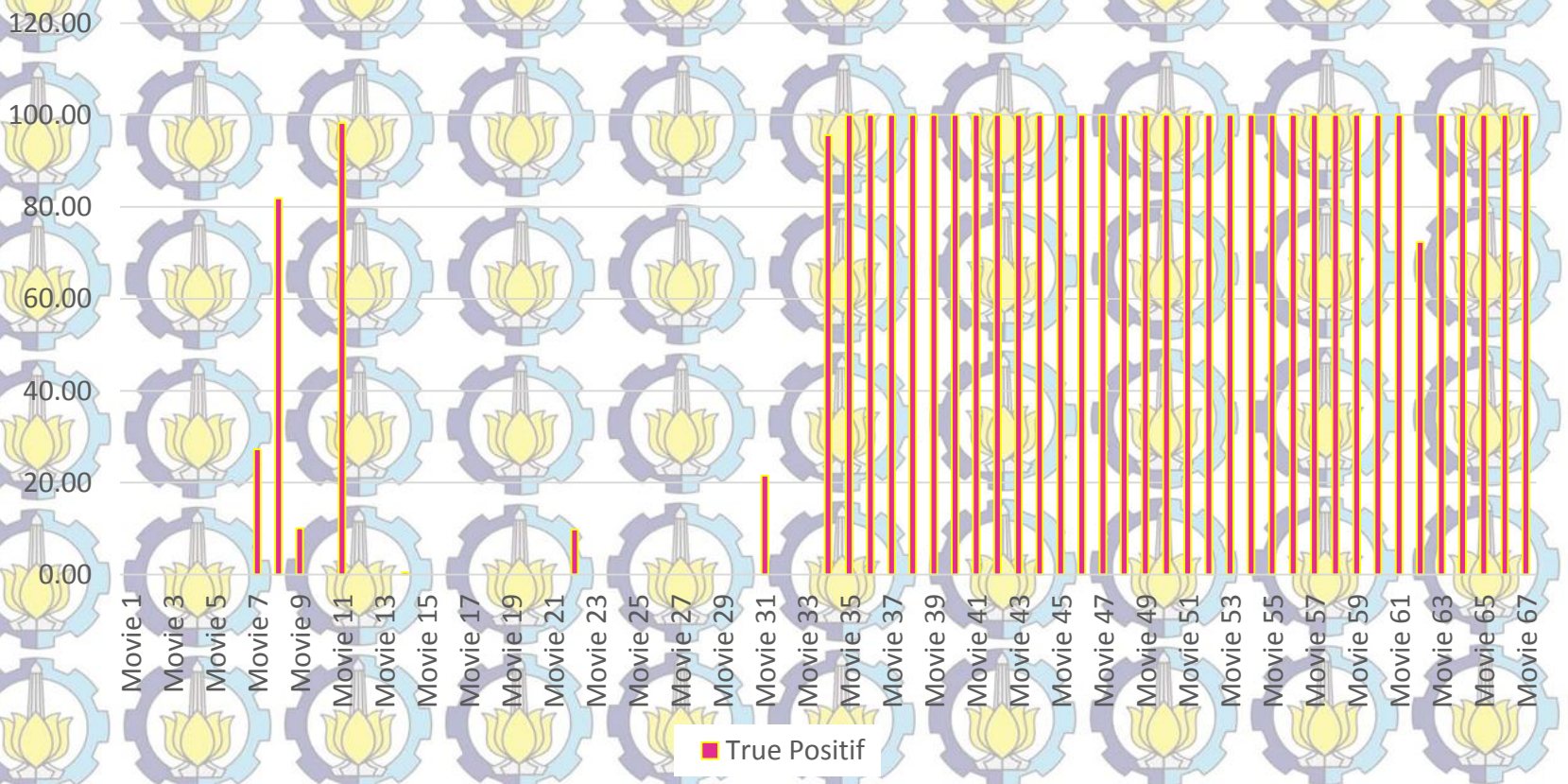
SKENARIO 4

Konstanta = 5%



SKENARIO 4

Konstanta = 10%



SKENARIO 4

- Akurasi terbaik yang didapatkan ketika Konstanta yang digunakan sebesar 1%

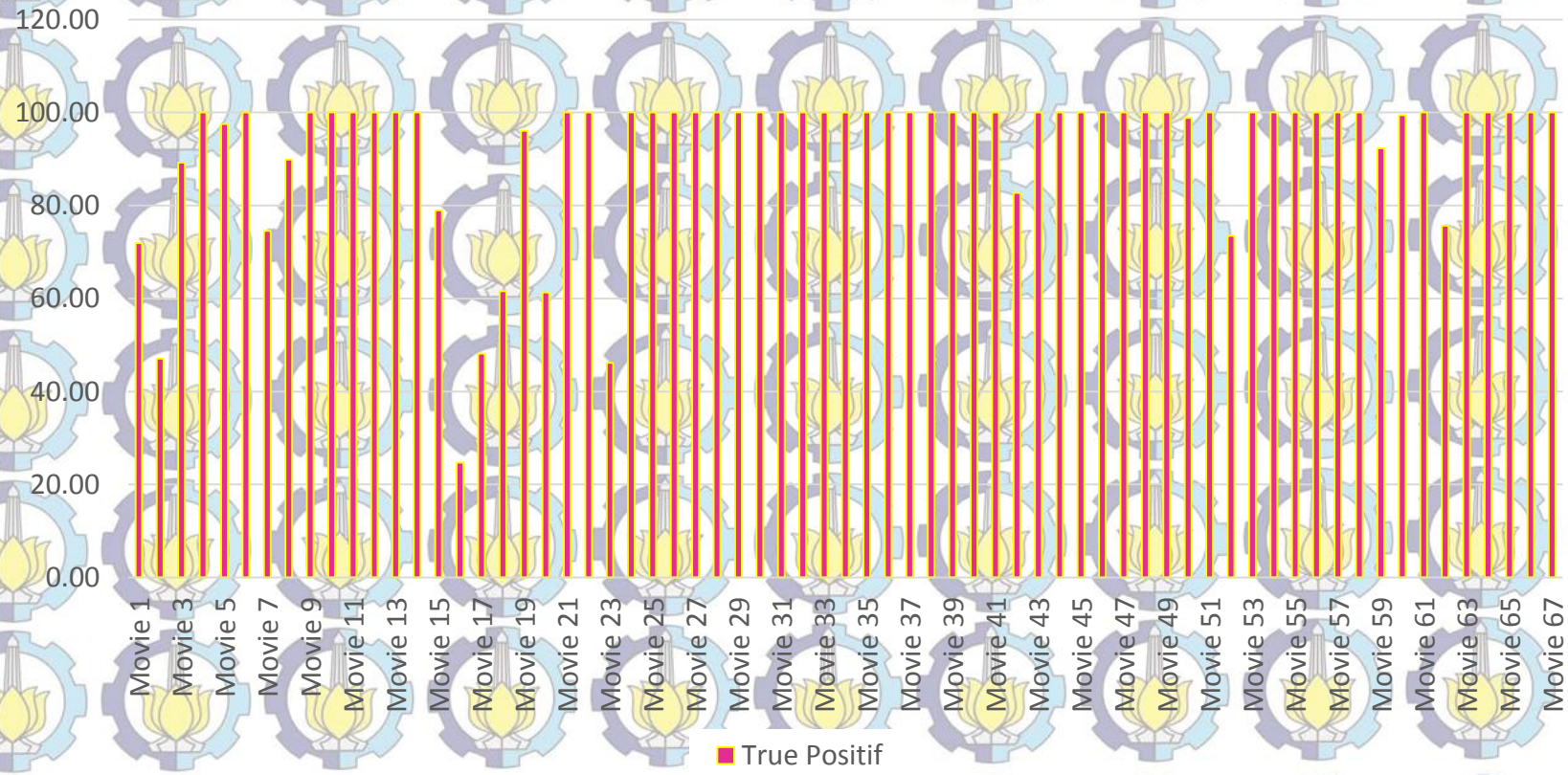
Konstanta	True Positif (%)	False Positif (%)	Missing Rate (%)
1%	95.87	0.49	3.64
5%	62.92	0.47	36.61
10%	53.99	0.41	45.60

SKENARIO 5

- Variasi Konstanta *size frame*
  - 240 x 320
  - 120 x 160
  - 60 x 80
- Kernel = RBF
- *Threshold* = 5×10^{-9}
- $C = 5$

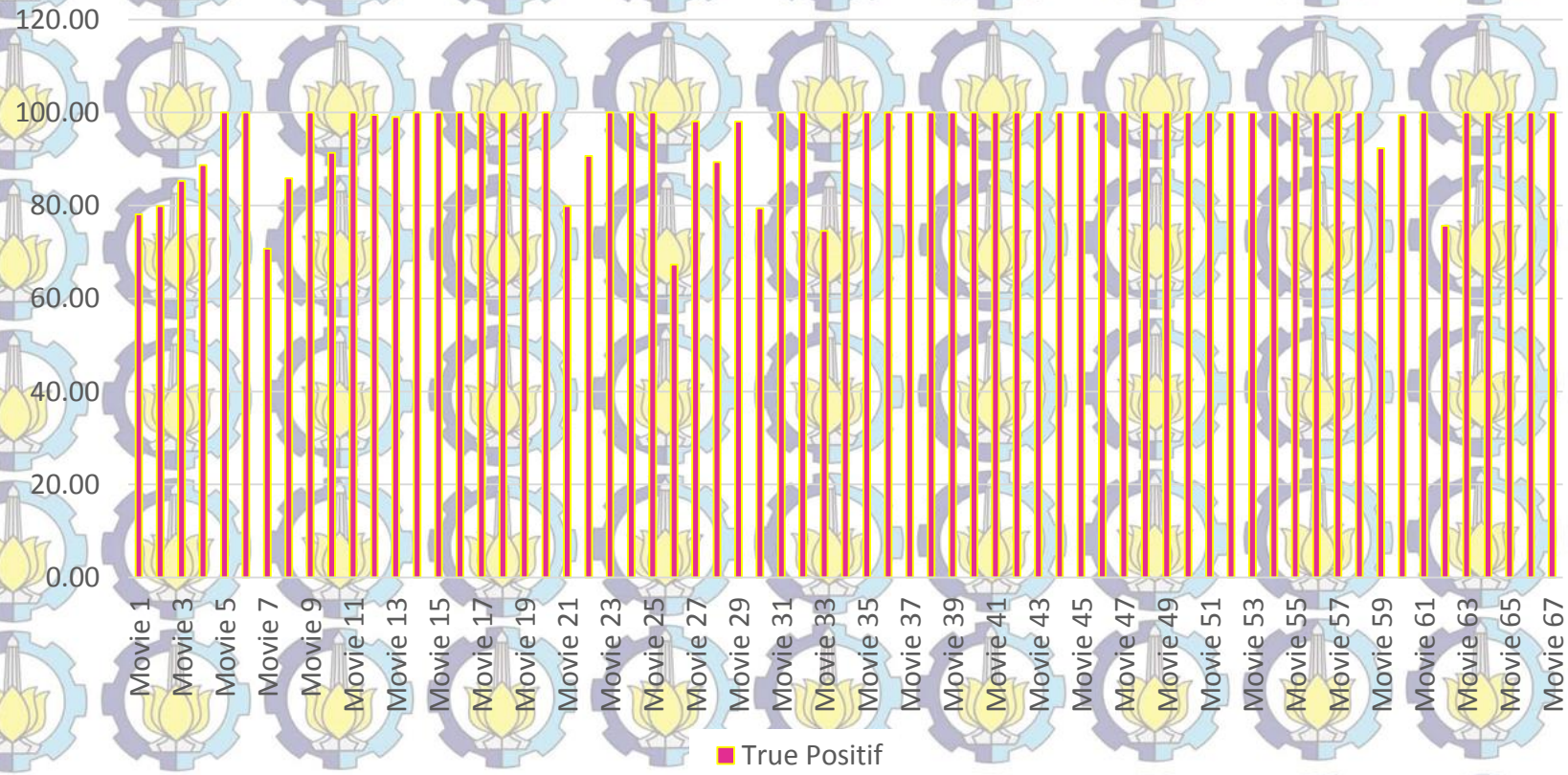
SKENARIO 5

Size 240 x 320



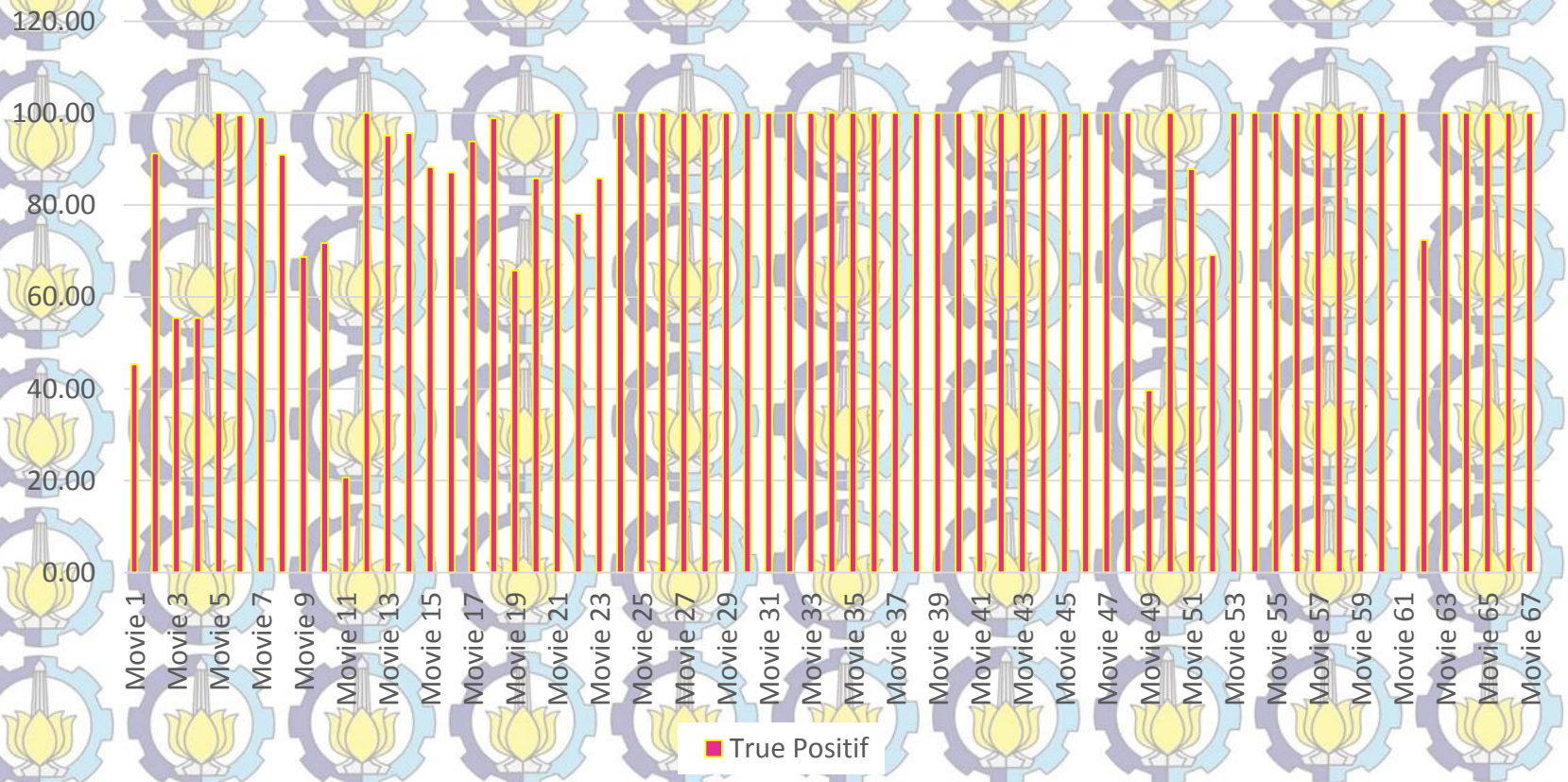
SKENARIO 5

Size 120 x 160



SKENARIO 5

Size 60 x 80



SKENARIO 5

- Akurasi terbaik yang didapatkan ketika Size Frame yang digunakan adalah 120 x 160

Size Frame	True Positif (%)	False Positif (%)	Missing Rate (%)	Execution Time (s)
240 x 320	91.96	0.79	7.25	67.38
120 x 160	95.87	0.49	3.64	26.57
60 x 80	90.67	1.82	7.51	7.30

PENDAHULUAN

RANCANGAN & IMPLEMENTASI

SKENARIO UJI COBA

KESIMPULAN & SARAN

KESIMPULAN

- Reduksi *size frame* mempercepat proses deteksi.
- Metode deteksi api menggunakan gaussian mixture model mendeteksi piksel-piksel yang bergerak dengan baik.
- Deteksi warna menyaring piksel-piksel yang tidak masuk kedalam range warna api dapat menyaring piksel-piksel yang tidak termasuk piksel api dengan baik.
- Metode perhitungan luasan region dapat menghilangkan noise dengan baik.

KESIMPULAN

- Penggunaan kernel pada klasifikasi mempengaruhi hasil dari verifikasi piksel.
- Hasil terbaik pada uji coba adalah menggunakan nilai threshold = 5 x 10-9 dan nilai C = 7. Menghasilkan nilai true positif sebesar 95.96, false positif sebesar 0.48 dan missing rate sebesar 3.56.

SARAN

- Analisa fitur pada pross verifikasi perlu dilakukan analisa lebih lanjut.

Institut Teknologi Sepuluh Nopember

TERIMA KASIH