

PROTOTYPE SYSTEM PENDETEKSI SPESIES IKAN MENGGUNAKAN VIOLA-JONES FEATURE EXTRACTION DAN BOOSTING BERBASIS DECISION TREE

Nehemiah Austen Pison- 1313619021

Dibawah bimbingan:
Muhammad Eka Suryana, M.Kom
Med Irzal, M.Kom



Revisi Pada SPS

- Kesalahan menyamakan deteksi objek dengan klasifikasi objek

Perbaikan: Penulisan ulang Bab 2.1 Pengertian Deteksi Objek menjadi 2.1 Pengertian Klasifikasi Objek

- Alur penjelasan yang tidak sesuai dengan flowchart, sehingga sulit dimengerti

Perbaikan: Merobak ulang isi dari Bab 2 agar lebih sesuai dengan flowchart Metode yang diikuti

- Tidak menjelaskan proses validasi setelah training

Perbaikan: Menambahkan Sub-bab 3.4 Skenario Eksperimen dan Validasi















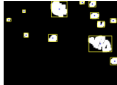
Tujuan Penelitian

- Menerus penelitian dari Hafizhun (2023), FISH MOVEMENT TRACKING MENGGUNAKAN METODE GAUSSIAN MIXTURE MODELS (GMM) DAN KALMAN FILTER.

Kelemahan Hafizhun (2023):

Metode belum dapat membedakan objek ikan dengan objek bergerak lainnya, dan memerlukan metode yang lebih kompleks untuk membedakan objek ikan dengan objek lainnya.

- Output segmentasi Kalman Filter dari Hafizhun (2023) bisa digunakan sebagai input klasifikasi ikan menggunakan Viola-Jones Feature Extraction, untuk melakukan penghitungan ikan

| Frame | Citra Asli | Hasil GMM | Groundtruth | Indeks | Frame | Groundtruth | Skala | Hasil |
|-------|---|---|---|--------|-------|---|-------|---|
| 705 |  |  |  | gt_124 | 705 |  | ×2 |  |
| 1173 |  |  |  | | | | |  |
| 1191 |  |  |  | | | | |  |

Proses deteksi objek bergerak menggunakan GMM, Hafizhun (2023)



Hasil tracking Hafizhun (2023) yang masih Banyak salah anotasi

Rumusan Masalah

Bagaimana caranya mengklasifikasi ikan menggunakan metode Viola-Jones Feature Extraction dan Boosting Berbasis Decision Tree?

Batasan Masalah

- *Klasifikasi ikan menggunakan Viola-Jones Feature Extraction dan Boosting Berbasis Decision Tree.*
- *Klasifikasi harus bisa melakukan klasifikasi tiga kelas genus ikan, Abudedefduf, Amphiprion, dan Chaetodon.*
- *Klasifikasi dilakukan dengan gambar tampak samping ikan saja dengan ikan menghadap ke kiri. Gambar berukuran 350 x 200 piksel, dan dengan latar belakang sudah dihilangkan.*

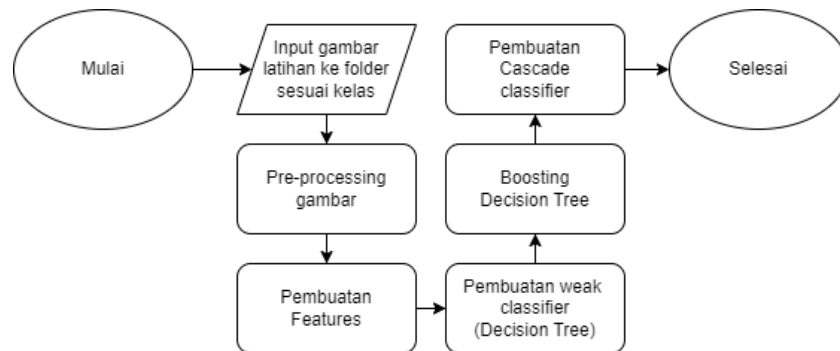
Tujuan Penelitian

Membuat program yang mampu mengklasifikasi ikan dengan menggunakan Viola-Jones Feature Extraction dan Boosting Berbasis Decision Tree.

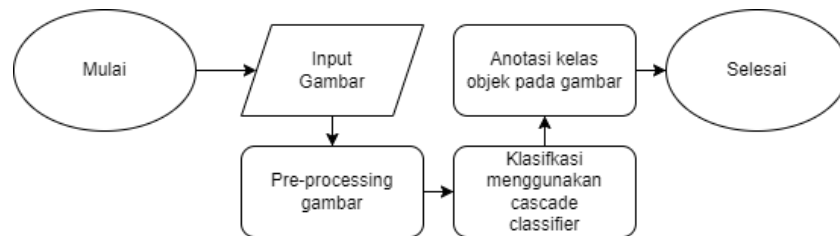


Hasil dan Pembahasan

Flowchart Pelatihan



Flowchart Pemakaian



Input Gambar + Pre-processing

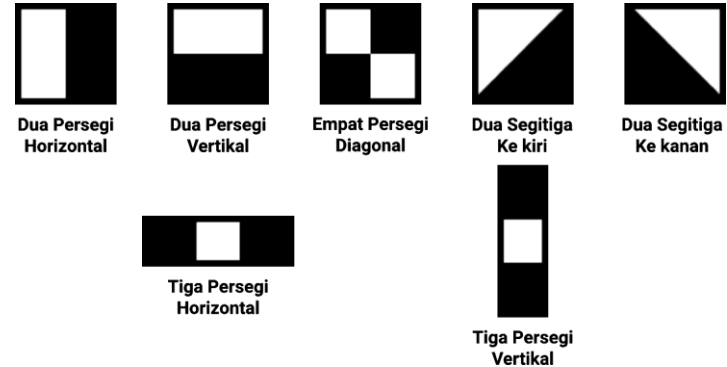
1. Gambar di-input kedalam folder sesuai kelasnya.
2. Ada 4 folder: abudefduf, amphiprion, chaetodon, dan negative_examples
3. Gambar di-load menggunakan fungsi Load_Images()
4. Gambar akan disesuaikan bila belum 350 x 200 piksel. Lalu diubah menjadi Greyscale



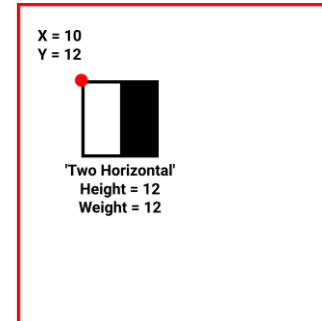
Semua gambar untuk Latihan dengan total 80 Gambar

Generate Features

1. Generate semua kemungkinan feature yang ada pada sebuah jendela berukuran 50x50 dengan fungsi generate_features()
2. Fitur-fitur yang dapat dibuat harus dipreset dalam bentuk paling kecilnya
3. Ada 7 Jenis fitur: Dua, Persegi Horizontal, Dua Persegi Vertikal, Empat Persegi Diagonal, Dua Segitiga Diagonal Menghadap Kiri, Dua Segitiga Diagonal Menghadap Kanan, Tiga Persegi Horizontal, Dan Tiga Persegi Vertikal



Ketujuh Preset Features

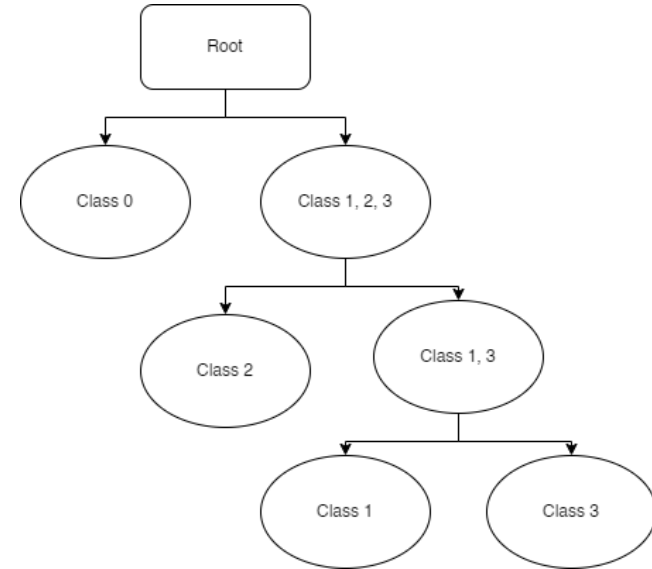


Gambaran Fitur: 'Two Horizontal', $X = 12$, $Y = 10$, Width = 12, Height = 12 pada window berukuran 50 x 50



Pembuatan Decision Tree

1. Semua gambar pertama akan dibaca dengan menggunakan semua Features. Hasil disimpan kedalam .csv untuk direferensi nanti. Ada 3 .csv yang dibuat untuk 3 classifier jendela berbeda
2. Bagi data dari pembacaan Features menjadi 3 kelompok: Train set, Test set, Validation set dengan perbandingan 1:1:1
3. Mulai pelatihan Decision Tree dengan fungsi `build_all_tree()` dengan menggunakan Train set sebagai referensi. Simpan semua Decision Tree ke file Pickle untuk nanti direferensi. Fungsi dipanggil 3x untuk ketiga jendela



Contoh gambaran sebuah
Decision Tree



Boosting

1. Semua Decision Tree akan diberikan berat bobot voting menggunakan metode Boosting. Dipanggil dengan fungsi `training_strong_clasiffier()`
2. Latih menggunakan Test set
3. Setiap satu iterasi pelatihan sudah selesai, seluruh decision tree akan mem-voting hasil prediksi dari set Validasi. Bila ditemukan saat iterasi tersebut akurasi menurun, maka bobot voting pada iterasi sebelumnya akan dijadikan bobot voting final.

```
(2751,)\n[('Right Triangular', 26, 24, 4, 4) ('Two Horizontal', 0, 0, 4, 4)\n ('Four Diagonal', 38, 4, 4, 16) ('Four Diagonal', 34, 39, 8, 8)\n ('Four Diagonal', 28, 29, 4, 8)]\n(2.0442409573736935, 1.9944920229321372, 1.0221204789753242, 0.49684825131159804, 0.16992632243171213)\n<class 'numpy.ndarray'>\n0.7142857142857143
```

Jumlah Decision Tree, 5 fitur pertama, 5 bobot voting pertama dan akurasi dari strong classifier untuk jendela pertama (kiri)

```
(4521,)\n[('Four Diagonal', 4, 40, 8, 4) ('Two Horizontal', 0, 0, 4, 4)\n ('Four Diagonal', 37, 36, 4, 12) ('Two Horizontal', 35, 43, 4, 4)\n ('Four Diagonal', 5, 45, 12, 4)]\n(2.0442409573736935, 1.9944920229321372, 1.0221204789753242, 0.5110602395151306, 0.24264270193875404)\n<class 'numpy.ndarray'>\n0.7142857142857143
```

Jumlah Decision Tree, 5 fitur pertama, 5 bobot voting pertama dan akurasi dari strong classifier untuk jendela kedua (tengah)

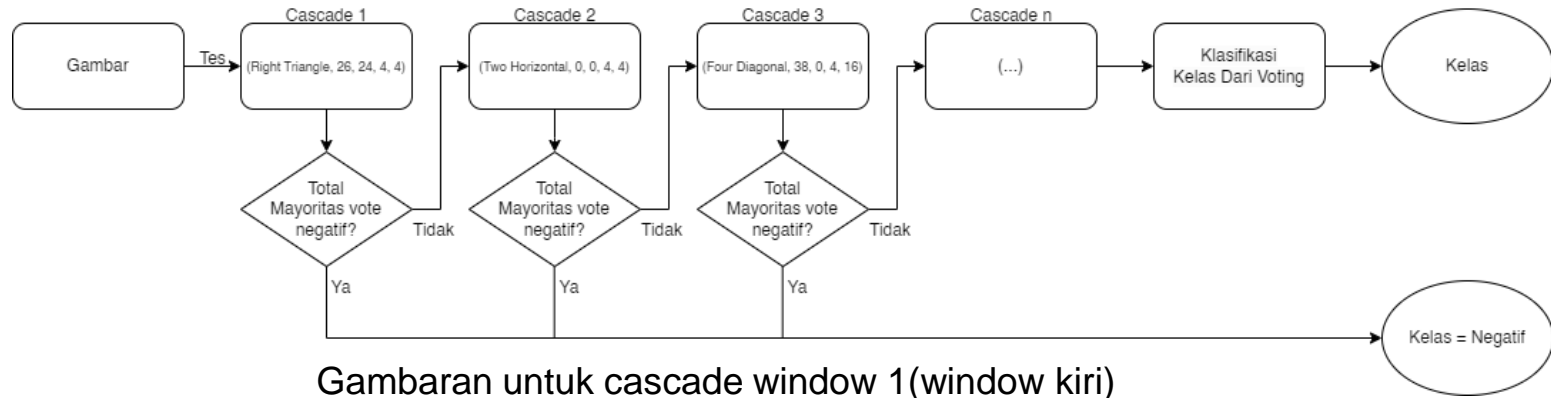
```
(6749,)\n[('Two Horizontal', 0, 0, 4, 4) ('Four Diagonal', 2, 40, 24, 4)\n ('Four Diagonal', 13, 15, 12, 8) ('Left Triangular', 1, 31, 4, 4)\n ('Four Diagonal', 8, 33, 28, 4)]\n(2.394118234656421, 1.6479184328021648, 1.3557734365574106, 0.6660687247632446, 0.342596137519398)\n<class 'numpy.ndarray'>\n0.7142857142857143
```

Jumlah Decision Tree, 5 fitur pertama, 5 bobot voting pertama dan akurasi dari strong classifier untuk jendela ketiga (kanan)



Training Cascade

1. Decision Tree yang telah dicari bobotnya lalu akan dibuatkan Cascade
2. Cascade dibuat dengan pertama membuat sebuah kelas object Cascade, dan menjalankan fungsi fill_cascade(). Tiap cascade berisi beberapa Decision Tree, Bobot voting, dan fitur.
3. Cascade disimpan dalam bentuk pickle untuk nanti digunakan



Gambaran untuk cascade window 1 (window kiri)



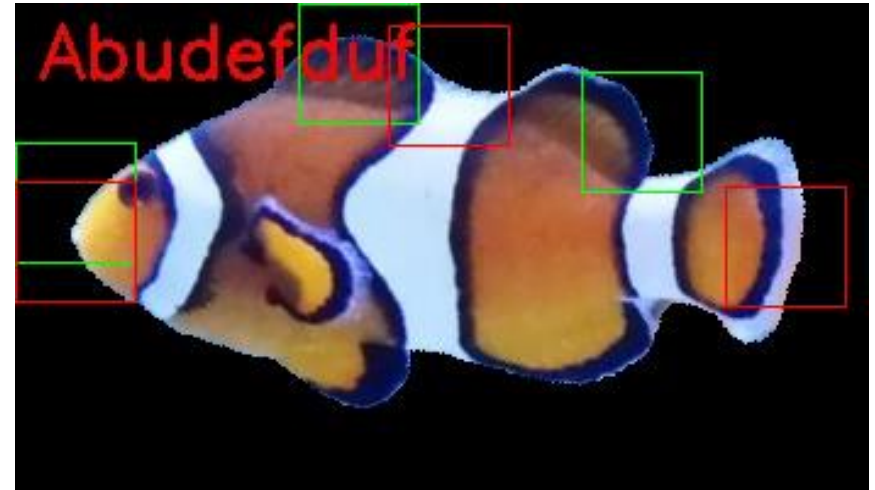
Validasi dan Testing

- Validasi dilakukan menggunakan seluruh dokumen prediction.py
- Masukkan semua gambar yang akan di klasifikasi kedalam folder khusus Bernama classification_target. Siapkan juga folder khusus untuk gambar hasil klasifikasi dengan nama classification_results



Analisa Hasil

- Walaupun akurasi awal pada saat Boosting menjanjikan, aplikasi Cascade pada sliding window sangat tidak akurat dengan 63 dari 75 gambar salah diklasifikasi. Akurasi klasifikasi akhir adalah 16%.
- Hal ini disebabkan karena Cascade mengembalikan hasil klasifikasi positif pada suatu lokasi yang tidak seharusnya. Semisalnya pada klasifikasi gambar Amphiprion23 klasifikasi menghasilkan kelas Abudefduf. Hal ini dikarenakan sliding-window mendeteksi pada lokasi window yang salah.
- Ini juga menunjukan kalau ada bias dari Cascade. Hal ini disebabkan oleh decision tree pada awal cascade yang memiliki bobot voting tertinggi, memiliki bias ke kelas tertentu dan jumlah bobot voting sisa decision tree lain tidak berhasil mengalahkan voting decision tree awal
- Dari Analisa ini dapat disimpulkan kalau metode ini tidak dapat digunakan untuk melakukan klasifikasi secara akurat tanpa penambahan atau pengantian metode untuk menyelesaikan problem klasifikasi akhir



Amphiprion23, Hasil klasifikasi = [1, 2, 1]

Kotak hijau menggambarkan lokasi dimana cascade mengklasifikasi dengan salah. Kotak Merah adalah estimasi harapan lokasi klasifikasi yang sebenarnya, karena merepakan offset pada saat fase pelatihan.



Kesimpulan dan Saran



Kesimpulan

1. Metode *Viola-Jones Feature Extraction dan Boosting* Berbasis *Decision Tree* berhasil dibuat
2. Metode *Viola-Jones Feature Extraction dan Boosting* Berbasis *Decision Tree* tidak berhasil melakukan klasifikasi secara akurat. Akurasi yang didapat dari validasi akhir hanya sebesar 16% saja.

Saran

1. Mencari solusi untuk salah klasifikasi oleh sliding-window
2. Perubahan rumus perhitungan bobot voting untuk menurunkan bias pada saat klasifikasi
3. Memodifikasi metode agar dapat mengklasifikasi gambar dengan ukuran dinamis
4. Pengimplementasian multi-processing untuk mempercepat proses training

Kekurangan Penelitian

1. Metode belum diuji menggunakan data lapangan yang sesungguhnya.
Kekurangan ini akan dipenuhi pada saat revisi

Terima Kasih !



Pertanyaan:

1. *Temp*