KLASIFIKASI IKAN MENGGUNAKAN VIOLA-JONES OBJECT DETECTION FRAMEWORK

Proposal Skripsi – Nehemiah Austen Pison

PENDAHULUAN

Rumusan Masalah

Dari uraian permasalahan diatas, perumusan masalah dalam penelitian ini adalah “Bagaimana cara mengklasifikasi ikan menggunakan metode *Viola-Jones Object Detection Framework*?”.

Batasan Masalah

1. Klasifikasi ikan menggunakan *Viola-Jones Object Detection Framework* yang didapat dari penelitian Viola-Jones (Viola et al, 2004)
2. Program didesan untuk mengklasifikasi ikan saja
3. Klasifikasi dilakukan dengan gambar tampak samping ikan saja
4. Klasifikasi harus bisa melakukan deteksi kelas ikan lebih dari satu (*multi-class classification)*

Tujuan Penelitian

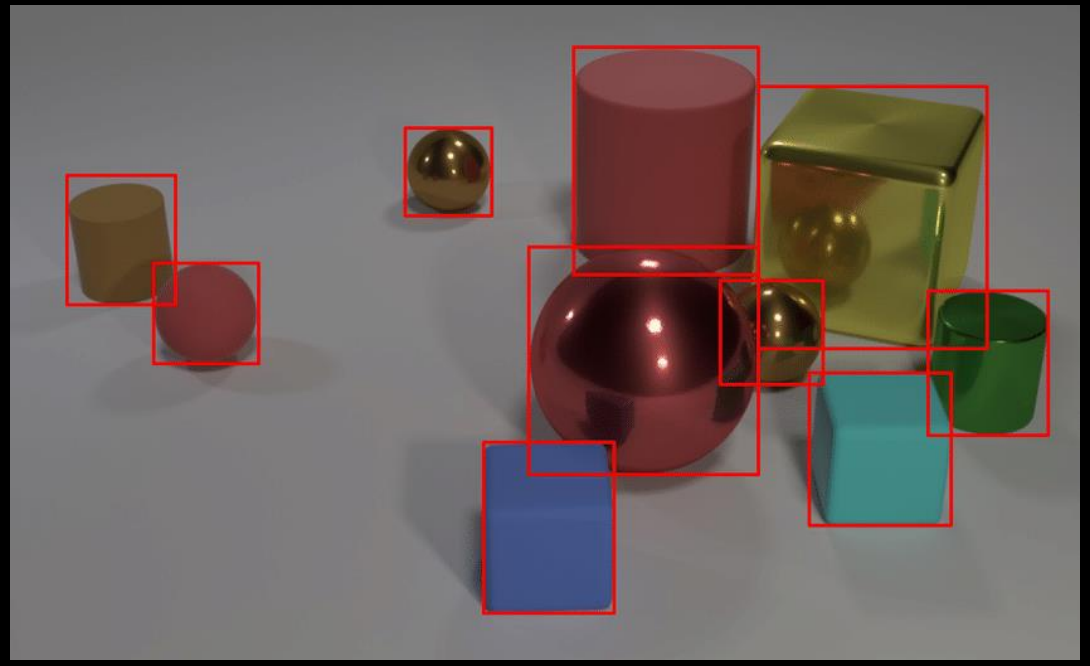
- Menciptakan metode yang mampu mempermudah klasifikasi ikan dengan menggunakan *Viola-Jones Object Detection Framework.*

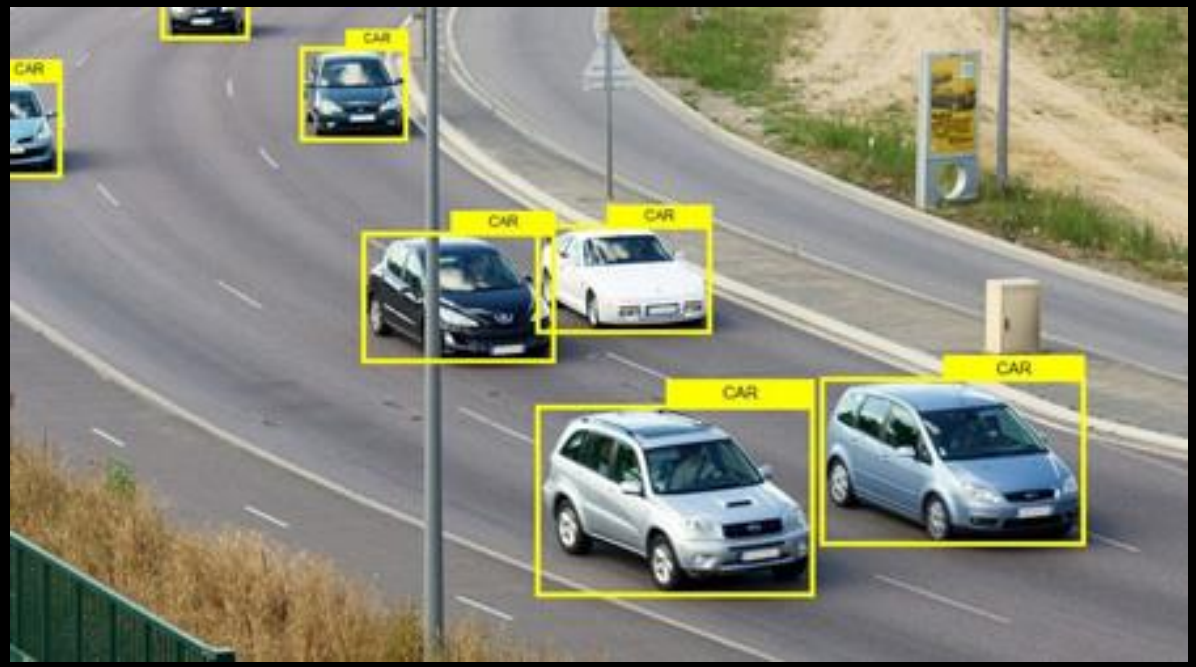
Manfaat Penelitian

- Memperoleh gelar sarjana dalam bidang Ilmu Komputer
- serta menambahkan pengalaman dalam pembuatan sebuah program komputer dengan aplikasi dunia nyata.
- Menambah pengetahuan penulis tentang deteksi objek, metode *Adaboost* dan *Harr-Like Features*.

Pengertian Deteksi Objek

Deteksi Objek berhubungan dengan pendeteksian objek dari satu atau lebih *class* didalam sebuah gambar. Pada dasarnya tujuan utama dari deteksi objek adalah untuk mendeteksi objek dari suatu *class* yang sudah diketahui seperti manusia, mobil, maupun ikan. Umumnya hanya ada sedikit objek yang ada didalam suatu gambar, namun ada banyak sekali lokasi, posisi maupun ukuran dari suatu objek yang telihat dalam sebuah gambar (Amit et al. 2020).

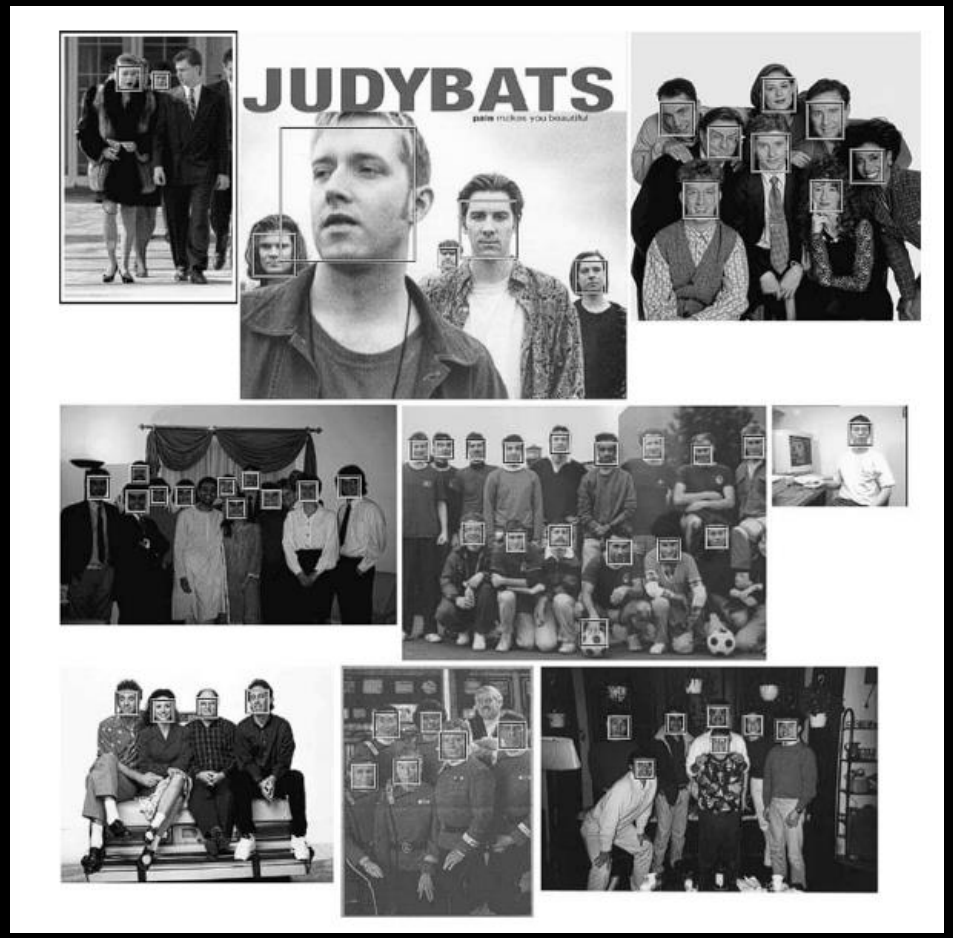




Viola Jones Object Detection Framework

Paul Viola dan Michael, J, Jones mempublikasikan sebuah makalah ilmiah dengan judul “Robust Real-Time Face Detection”. Makalah tersebut mendeskripsikan sebuah *framework* pendeteksian wajah yang dapat memproses gambar secara cepat dengan tingkat akurasi yang tinggi. Ada tiga kontribusi penting dari makalah tersebut, yaitu:

- Integral Image
- Adaboost Classifier
- Dan Attentional Cascade



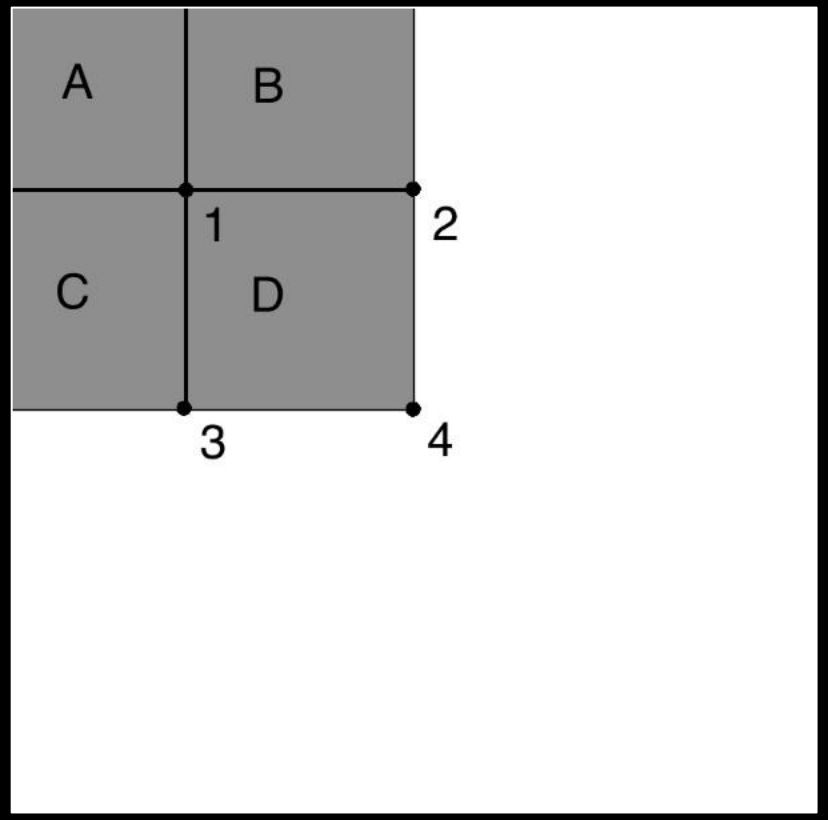
Integral Image

Fitur-fitur dari sebuah kotak dapat dikomputasi secara cepat menggunakan representasi tidak langsung dari gambar, hal ini diberi nama *integral image*. *Integral image* pada lokasi *x*, *y* dapat dihitung dengan rumus:

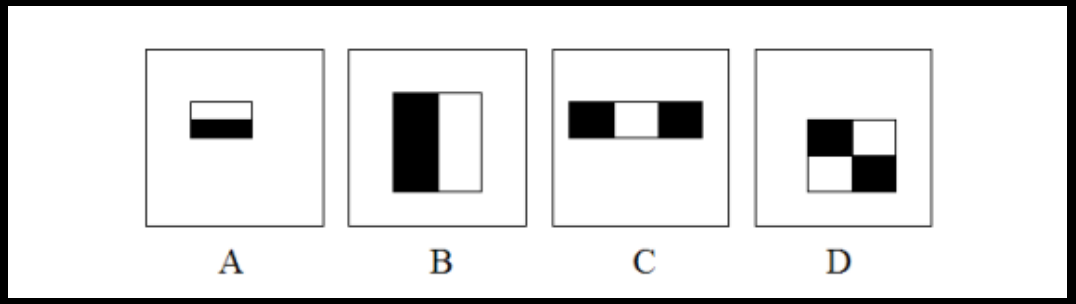
$$ii(x, y) = s(x, y) + ii(x - 1, y) + ii(x, y - 1) - ii(x - 1, y - 1)$$

Menggunakan *integral image*, semua jumlah nilai fitur pada sebuah persegi dapat dihitung dengan 4 referensi *array.* Perbedaan diantara kedua jumlah nilai persegi persegi dapat dihitung dengan delapan referensi. Karena persegi dua fitur yang digunakan dalam Metode Viola-Jones bersebelahan, mereka dapat dikomputasi dengan enam referensi *array*, delapan untuk persegi tiga fitur , dan sembilan untuk persegi empat fitur.

Bila dibandingkan, perhitungan manual mengharuskan kita menghitung sampai 100x untuk mendapat jumlah intensitas cahaya pada sebuah area 10x10 piksel, belum lagi proses ini harus diulang terus-menerus dengan ukuran dan lokasi yang berbeda.



Jumlah dari intensitas cahaya pada persegi D dapat dihitung dengan 4 referensi Array. Nilai dari 1 adalah jumlah intensitas cahaya pada persegi A, Nilai dari 2 adalah persegi A + B, nilai dari 3 adalah A + C dan nilai 4 adalah A+B+C+D.



Harr-Like features yang digunakan dalam framework Viola-Jones. Untuk perhitungan fitur menggunakan Integral image, fitur A dan B memerlukan 6x array lookup, fitur C memerlukan 8x Array lookup, dan fitur D memerlukan 9x array lookup.

Boosting

Boosting adalah sebuah metode umum yang digunakna untuk meningkatan kemampuan algoritma pembelajaran apapun. Dalam teori, *boosting* dapat digunakan untuk mengurangi error dari algoritma pembelajaran yang "lemah" secara signifikan, yang hanya perlu sedikit lebih baik dari menebak secara acak dengan menggabungkan algoritma-algoritma lemah tersebut menjadi sebuah *Strong Classifier*.

- Dalam framework deteksi objeknya, Viola-Jones menggunakan metode Boosting Adaboost M.1 buatan Yoav Freund (Freund et al, 1996).
- Algoritma pembelajaran yang digunakan adalah *Decision Tree* berbentuk *Stump* yang hanya memiliki dua daun yaitu true dan false.
- Framework Viola-Jones hanya bisa melakukan klasifikasi biner, atau hanya mengecek keberadaan suatu objek didalam gambar.

Algorithm AdaBoost.M1

Input: sequence of m examples $\langle (x_1, y_1), \ldots, (x_m, y_m) \rangle$ with labels $y_i \in Y = \{1, \ldots, k\}$
weak learning algorithm **WeakLearn**
integer T specifying number of iterations

Initialize $D_1(i) = 1/m$ for all i.

Do for $t = 1, 2, \ldots, T$

1. Call **WeakLearn**, providing it with the distribution D_t.
2. Get back a hypothesis $h_t : X \to Y$.
3. Calculate the error of h_t: $\epsilon_t = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$. If $\epsilon_t > 1/2$, then set $T = t - 1$ and abort loop.
4. Set $\beta_t = \epsilon_t / (1 - \epsilon_t)$.
5. Update distribution D_t: $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \beta_t & \text{if } h_t(x_i) = y_i \\ 1 & \text{otherwise} \end{cases}$
   where Z_t is a normalization constant (chosen so that D_{t+1} will be a distribution).

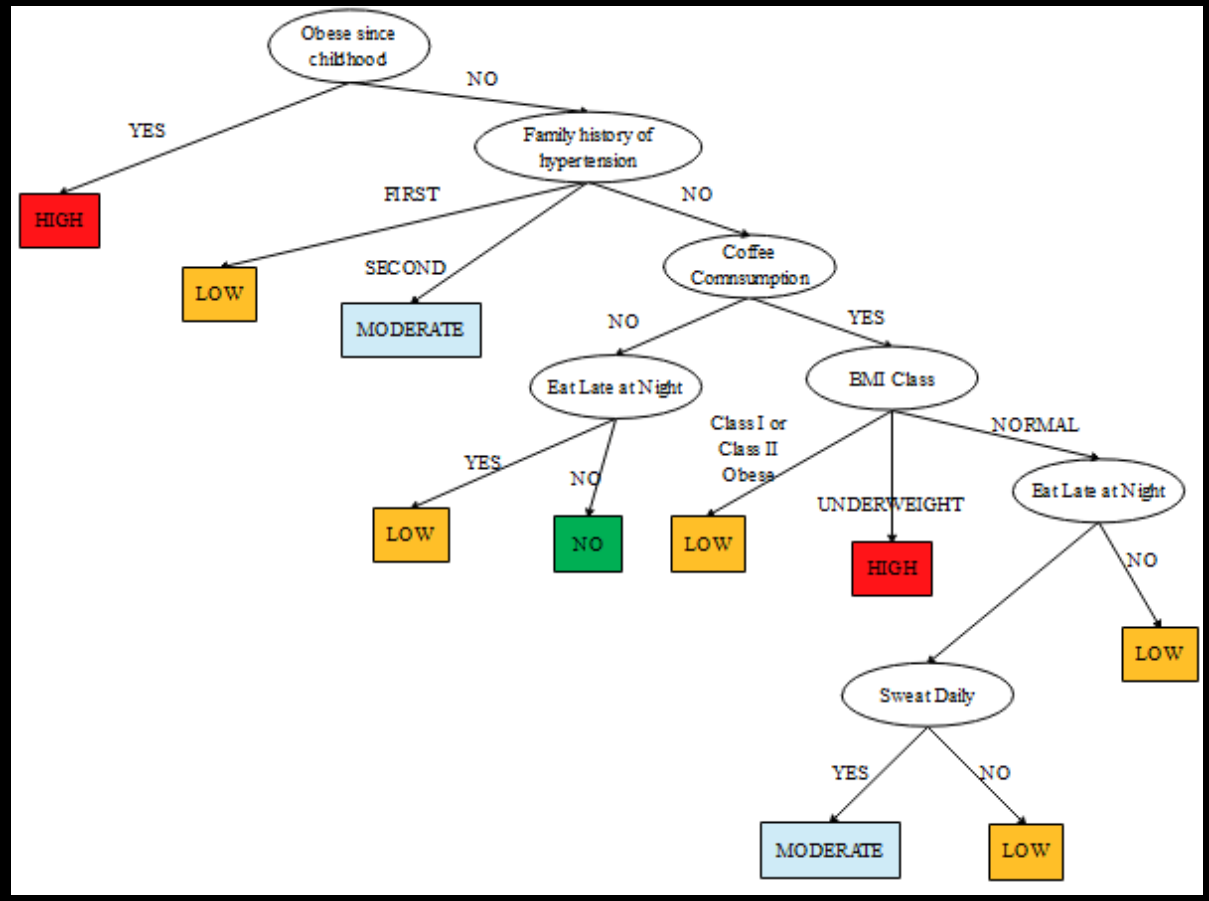
Output the final hypothesis: $h_{fin}(x) = \arg\max_{y \in Y} \sum_{t:h_t(x)=y} \log \frac{1}{\beta_t}$.

Pseudo-Code Adaboost M.1

C4.5 Decision Tree

Decision Tree menghasilkan sebuah classifier didalam bentuk sebuah decision tree, dalam sebuah struktur yang berbentuk:

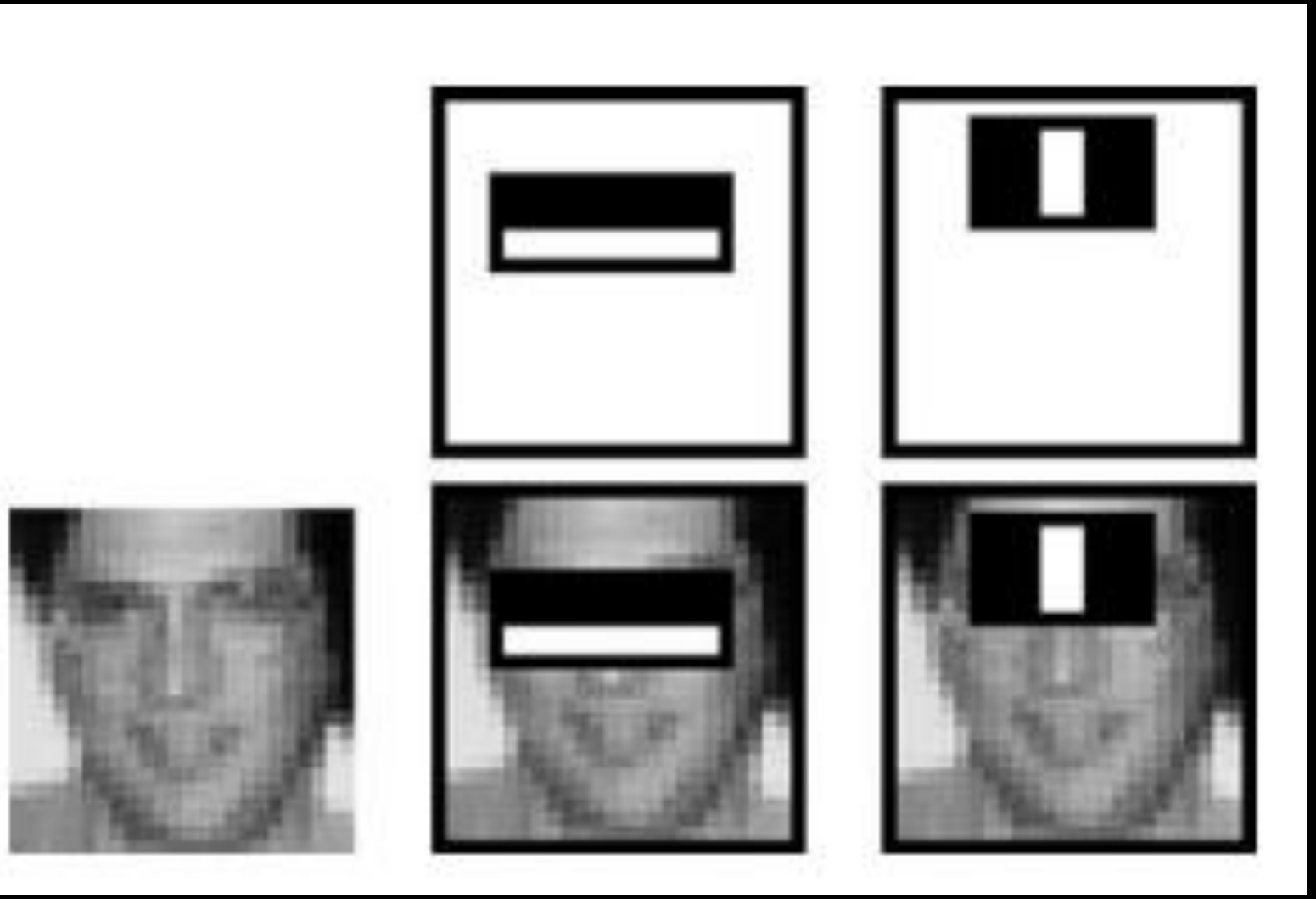
- Sebuah daun, mengindikasi sebuah kelas, atau
- Sebuah decision node yang menspesifikasi sebagian tes untuk dikerjakan atas sebuah nilai atribut, dengan satu branch dan subtree untuk setiap hasil dari tes.



Haar Like Features

Fungsi utama dari menggunakan fitur daripada nilai *raw pixel* sebagai input ke sebuah algoritma pembelajaran adalah untuk mengurangi/menambahkan variabel *in-class/out-class* dibandingkan dengan menggunakan data masukan mentah, yang membuat klasifikasi lebih mudah. Fitur biasanya meng-*encode* pengetahuan tentang domain yang sulit dipelajari dari set data masuk mentah yang terbatas.

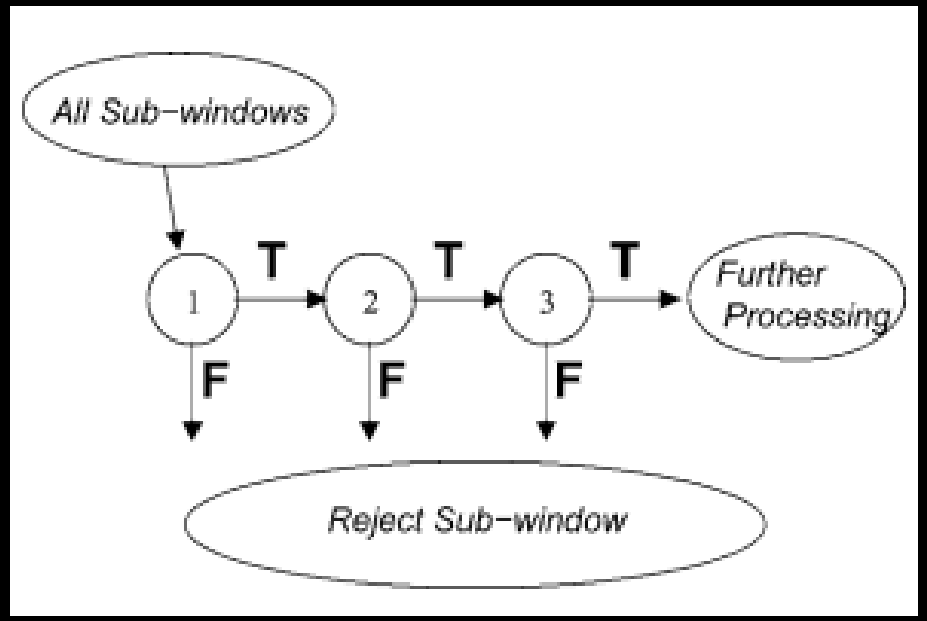
Sebuah fitur adalah perbandingan jumlah nilai piksel dari dua buah persegi atau lebih, yang biasanya digambarkan sebagai persegi putih dan hitam. Dengan metode ini, keberadaan sebuah fitur yang kritis seperti perbedaan objek dan latar belakang dapat dideteksi.



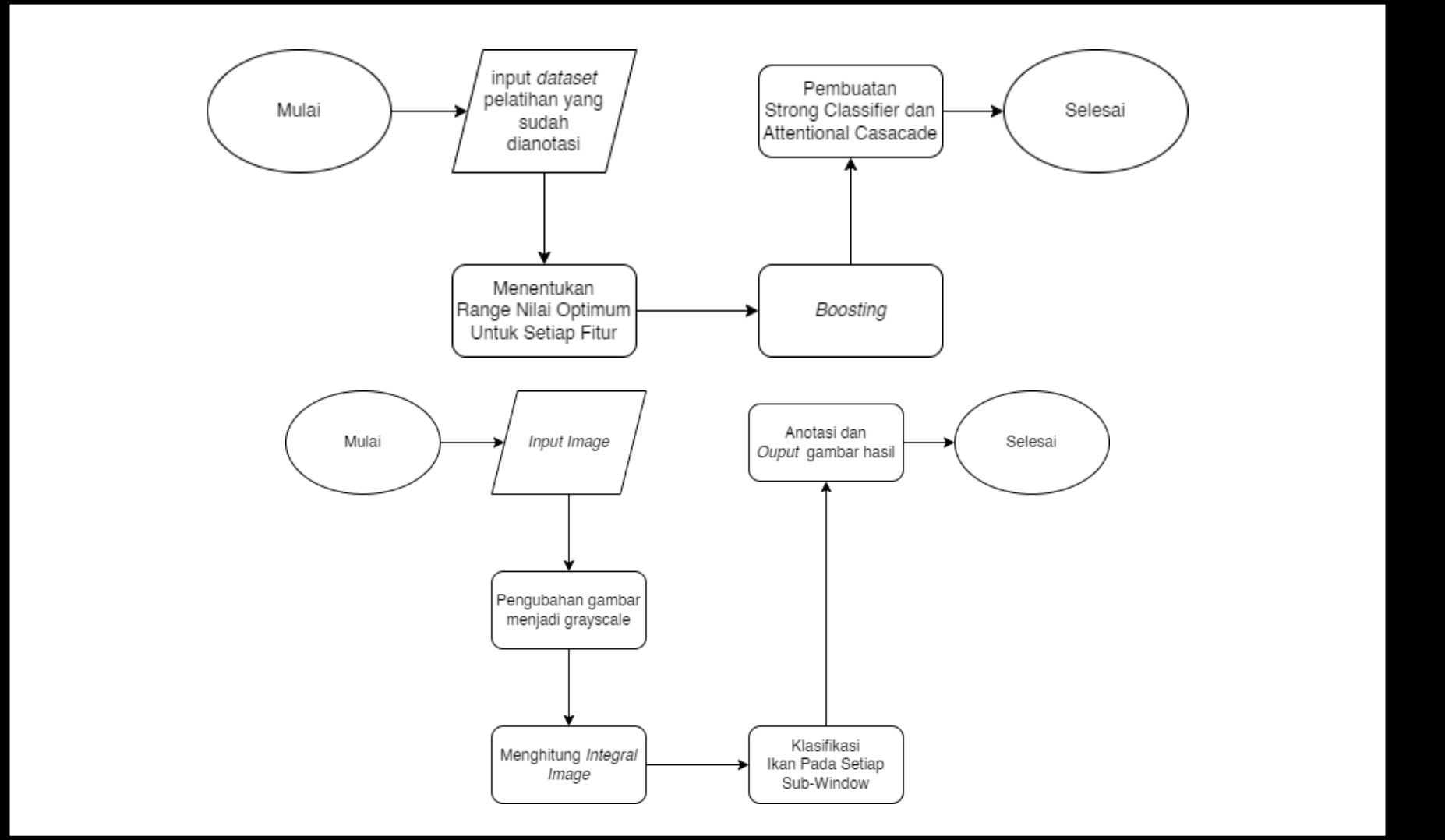
Dua fitur yang dipilih oleh Adaboost dalam framework Viola-Jones. Fitur pertama menghitung perbedaan intensitas cahay pada area mata dengan pipi bagian atas. Fitur ini menggunakan fakta kalua area mata umumnya lebih gelap dari bagian pipi. Fitur kedua menghitung intensitas di area mata dengan bagian hidung atas.

Attentional Cascade

Attentional Cascade adalah sebuah *Cascade* dari banyak *classifier* yang dibuat untuk meningkatkan performa deteksi dengan secara radikal mengurangi waktu komputasi. Intinya *classifier* kecil yang telah di *boosting* dapat dibuat lebih kecil dan efisien, yang dapat menolak mayoritas *sub-window* negatif dan mendeteksi sebagian besar dari *sub-window* positif. *Classifier* yang lebih sederhana digunakan untuk menolak mayoritas *sub-window* sebelum *classifier* yang lebih kompleks dipanggil untuk menurunkan tingkat *false positives.*

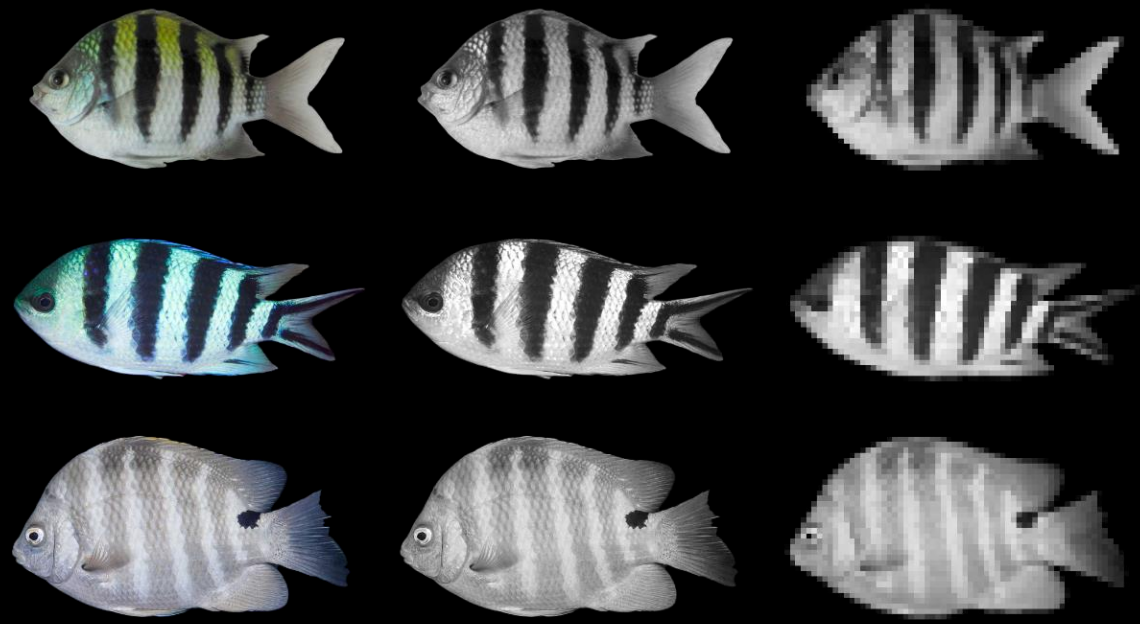


DESAIN MODEL



Training Strong Classifier

Pada Training ada tiga atau lebih tahapan yang perlu dijalankan untuk menghasilkan sebuah *strong classifer* pendeteksian objek, yaitu penginputan dataset pelatihan yang sudah dianotasi, penentuan nilai ambang optimum atau bobot untuk setiap fitur, dan pemilihan fitur oleh Adaboost untuk membuat *strong classifier*.



Contoh dataset training,
Gambar AsliAbudefduf - Grayscale – dan 72 x 41 pixel Grayscale

Menentukan Range Nilai Optimum Untuk Setiap Fitur

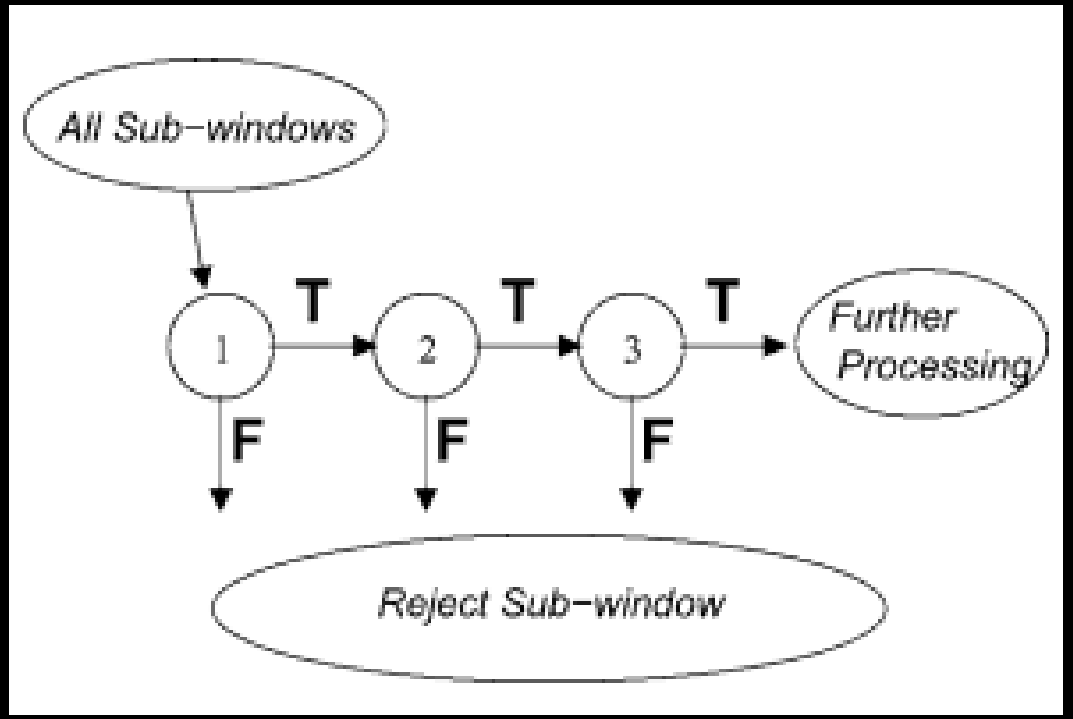
- Algoritma akan melatih *weak learner* untuk mencari nilai ambang atau *threshold* optimalnya untuk setiap kelas. Nilai ambang ini adalah hasil perhitungan sebuah *Haar-Like Features.*
- Pencarian ini dilakukan dengan menjalankan *weaklearner* dengan *dataset* pelatihan menggunakan sebuah rentang nilai untuk mencari nilai yang memberikan tingkat kesalahan klasifikasi terendah. Rentang nilai yang dites adalah nilai diantara -1 sampai 1 dengan nilai inkremental 1/255 untuk fitur 2 persegi, dan -2 sampai 2 dengan nilai inkremental yang sama untuk fitur 4 persegi.
- Setiap *weaklearner* akan berupa sebuah *decision tree* dengan beberapa kelas sebagai daun, salah satu daun pasti adalah kelas “Tidak ada”.

Boosting

- Setelah menentukan nilai ambang terbaik untuk setiap fitur, Adaboost akan memberikan bobot voting untuk setiap *weak learner* dengan melakukan *boosting* dimulai dari *weak learner* terbaik.
- Semua contoh pada dataset tahap ini awalnya akan dianggap memiliki bobot nilai yang sama dalam penilaian akurasi *weak learner*. *Adaboost* akan mengingat gambar mana saja yang dengan salah diklasifikasi oleh *weak learner* yang baru dinilai lalu menaikan bobot nilai gambar latihan yang sulit diklasifikasi. Hal ini dengan tujuan agar *weak learner* berikutnya yang berhasil mengklasifikasi gambar "sulit" tersebut dapat mendapatkan nilai yang lebih tinggi
- Pada akhir dari setiap sesi *boosting*, yaitu ketika semua *weak learner* sudah dinilai dan sudah diurutkan kembali dari yang paling diskriminatif ke yang tidak, Adaboost akan melakukan *cross-validation* menggunakan contoh dari kelompok *dataset* validasi untuk menentukan apakah boosting akan diulang atau dihentikan karena sudah mencapai hasil paling optimal

Pembuatan Strong Classifier dan Attentional Casacade

Setelah tahap *boosting* berakhir, Adaboost akan membuat *Final Strong Classifier* dan *Attentional Cascade* dengan memilih *weak classifier* terbaik dan membuang *weak classifier* yang kurang berguna dengan melihat bobot individu setiap *weak classfier*. Bila Adaboost merasa sebuah *weaklearner* kurang diskriminatif maupun kurang berguna dalam mengklasifikasi maka fitur tersebut tidak akan digunakan dalam pembuatan *Strong Classifier*.

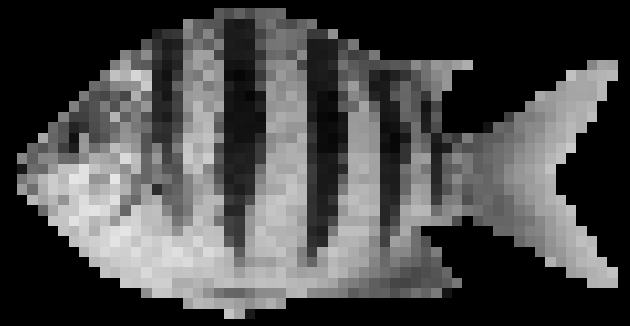


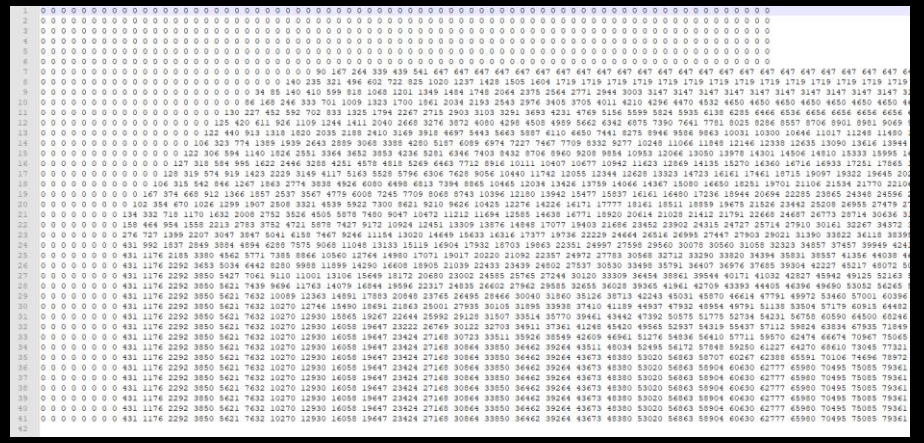
Pendeteksian

Tahapan ini adalah penggunakaan *classifier* yang sebenarnya. Gambar yang akan dipakai akan melalui beberapa langkah dalam tahap ini, yaitu: Pre-processing dalam bentuk grayscaling, Penghitungan Integral Image, dan deteksi yang sesungguhnya menggunakan *Strong Classifier* yang telah dibuat dengan metode *Sliding Window*.

Input Gambar, Grayscaling dan Penghitungan Integral Image

- Gambar awalnya akan melakui proses *pre-processing* dan dirubah kedalam warna *grayscale* untuk mempermudah penghitungan dengan bantuan *library OpenCV*
- Setelah tu sebuah matriks sebesar resolusi gambar akan dibuat untuk penghitungan *Integral Image* yang nantinya akan mempercepat proses penghitungan fitur





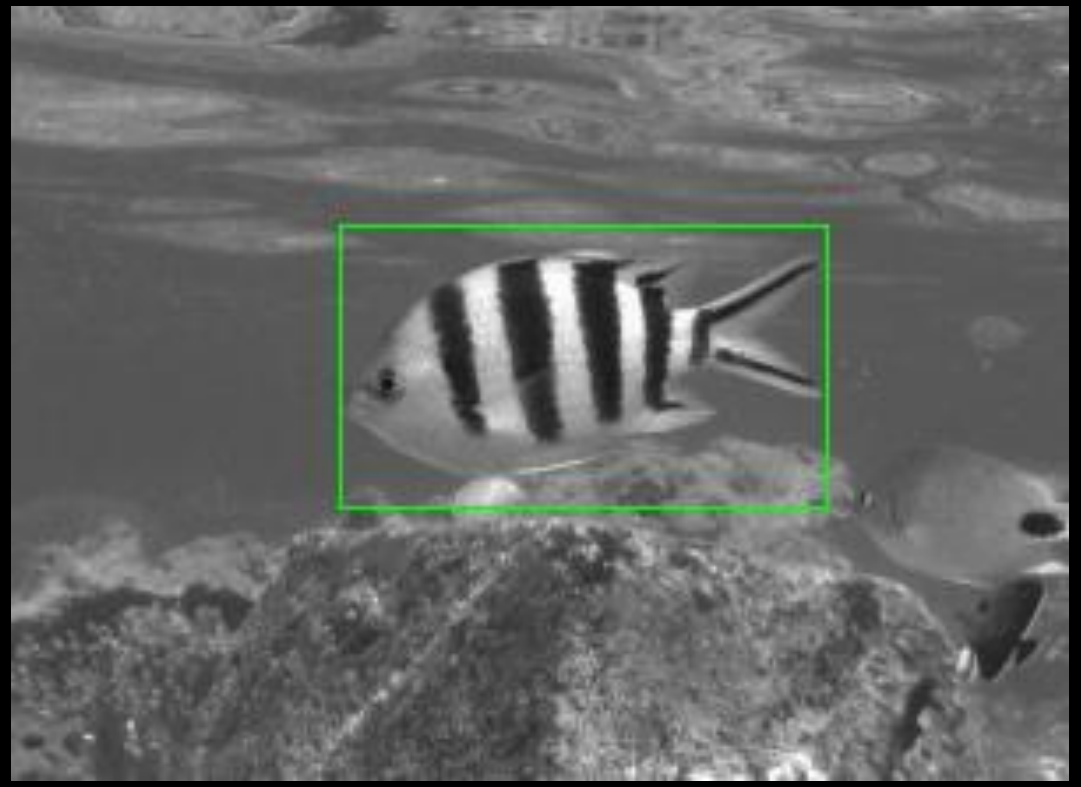
Sebuah gambar dan *Integral Image* miliknya.

Klasifikasi Ikan Pada Setiap Sub-Window

Sebuah *Sub-Window* sebesar 72x41 piksel akan dibuat untuk proses klasifikasi. Ukuran minimum dari *Sub-Window* mengikuti ukuran gambar pada proses pelatihan. Pendeteksian dilakukan dengan manggunakan *Strong Classifier* yang telah dibuat. *Sub-window* akan terus bergerak ke kanan sampai *Sub-window* tidak dapat dibuat lagi. Bila demikian proses akan mulai lagi dari kiri dengan *Sub-Window* diturunkan satu piksel kebawah. Proses ini dinamakan *Sliding-Window.*

- Bila *Strong Classifier* sudah selesai mendeteksi menggunakan *Sub-Window* dengan ukuran 72x41 piksel, maka *Strong Classifier* akan diperbesar dengan faktor 1.25 menjadi 90x51 piksel dan proses dimulai lagi dari kiri atas
- Setelah semua *Sub-Window* dicek, semua *Sub-Window* yang memiliki objek ikan yang dicari akan dianotasi dengan menggambarkan *bounding box* pada gambar hasil. Untuk kelas ikan apa yang terdeksi akan ditentukan dari hasil voting seluruh weaklearn, bobot votingnya adalah nilai yang sudah ditentukan pada fase boosting.

Ilustrasi ukuran Sub-Window 72x41 pada gambar 300x220



Gambaran bounding box pada ikan yang berhasil diklasasifikasi

Terima Kasih