BAB 2
TINJAUAN PUSTAKA

2.1. Lalu Lintas

Lalu lintas diartikan sebagai pergerakan dari seseorang atau kendaraan di dalam ruang lingkup lalu lintas jalan. Sedangkan yang dimaksud dengan ruang lingkup lalu lintas jalan adalah semua sarana dan prasarana yang diperuntukan untuk gerak serta perpindahan dari pengguna jalan tersebut. Sarana dan prasarana ini disebut dengan jalan dan fasilitas pendukung.

Terdapat beberapa komponen di dalam lalu lintas yang selalu berhubungan satu dengan lainnya, yaitu:

a. Manusia

   Manusia disini berperan sebagai pengguna jalan. Peran manusia sebagai pengguna jalan adalah sebagai pejalan kaki ataupun sebagai pengendara.

b. Kendaraan

   Kendaraan merupakan sebuah alat transportasi yang digunakan untuk mengangkut/mengantar barang maupun manusia ke tujuan yang diinginkan.

c. Jalan

   Jalan adalah sebuah lintasan yang dibuat dan direncanakan untuk dapat dilalui oleh pengguna jalan baik sebagai pengendara, maupun sebagai pejalan kaki. Umumnya, jalan untuk pejalan kaki dipisahkan dari jalan untuk kendaraan. Dimana jalan yang dikhususkan untuk pejalan kaki biasa disebut dengan trotoar.

2.1.1. Rambu Lalu Lintas

Rambu Lalu lintas merupakan sebuah perlengkapan dijalan yang memiliki bentuk tertentu, mengandung makna tertentu, serta memiliki lambang, huruf, angka, kalimat dan/atau perpaduan diantaranya. Rambu lalu lintas memiliki fungsi untuk memberikan larangan, peringatan, atau

petunjuk untuk pengguna jalan [4]. Rambu lalu lintas diatur dalam Peraturan Menteri Perhubungan Nomor 13 Tahun 2014.

2.1.2. Rambu Larangan

Rambu larangan digunakan untuk menyatakan perbuatan/aktivitas yang tidak boleh dilakukan oleh pengguna jalan. Rambu larangan memiliki ciri berwarna dasar merah dengan lambang berwarna hitam serta aksen merah yang menunjukan larangan [5]. Terdapat beberapa jenis rambu larangan, yaitu:

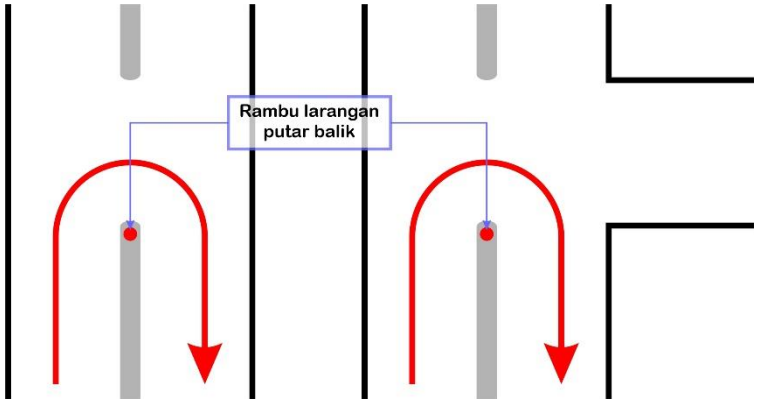
a. Larangan berjalan terus
b. Larangan masuk
c. Larangan parkir dan berhenti
d. Larangan pergerakan lalu lintas tertentu
e. Larangan membunyikan isyarat suara
f. Larangan dengan kata-kata

2.1.3. Rambu Larangan Putar Balik

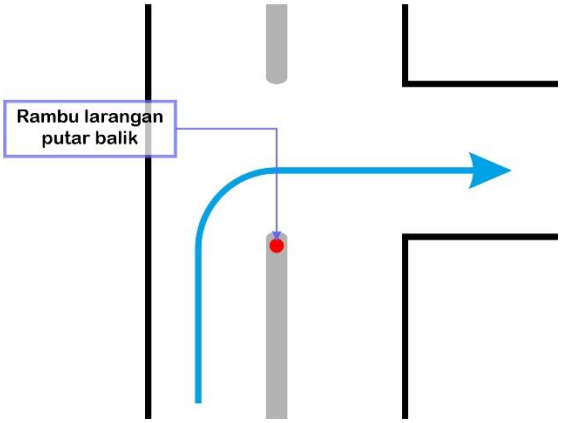
Rambu larangan putar balik termasuk kedalam jenis rambu larangan pada pergerakan lalu lintas tertentu. Biasanya rambu ini sering ditempatkan pada persimpangan, dikarenakan banyak pengendara yang memanfaatkan jalan yang tidak dibatasi antar jalurnya untuk memutar balik kendaraannya menujuh arah yang berlawanan.

2.1.4. Pelanggaran Pada Rambu Larangan Putar Balik

Pelanggaran pada rambu larangan putar balik terjadi apabila pengendara atau pengemudi memutar arah laju kendaraannya ke jalur berlawanan yang dimana terdapat rambu larangan putar balik di pembatas jalan tersebut. Untuk lebih jelasnya, dapat dilihat pada ilustrasi gambar berikut.



Gambar 2.1 Anak panah merah adalah pergerakan kendaraan yang melanggar rambu larangan putar balik



Gambar 2.2 Anak parah biru mudah adalah pergerakan kendaraan yang tidak melanggar

2.2. Citra Digital

Citra digital adalah sebuah gambaran dari sebuah objek. Citra digital adalah citra yang dihasilkan dari peralatan digital sehingga dapat dipresentasikan serta diproses dalam komputer [6]. Citra digital merupakan sebuah *array* (larik) yang berisikan nilai-nilai serta dipresentasikan oleh bit tertentu. Sebuah citra dapat didefinisikan dengan fungsi *f(x,y)* yang memiliki dimensi M baris serta N kolom dengan x dan y adalah koordinat spasial [7].

2.3. Pengolahan Citra

Pengolahan citra merupakan sebuah proses yang dilakukan untuk menghasilkan citra yang lebih baik dari citra aslinya, dimana dalam proses pengolahannya dilakukan dalam komputer. Pengolahan citra sangat dibutuhkan. Hal itu dikarenakan meskipun suatu citra mengandung informasi yang cukup banyak, namun sering kali sebuah citra mengalami penurunan baik kualitas maupun kuantitas dari informasi tersebut yang diakibatkan oleh gambar yang kurang tajam, terdapat banyak *noise* (derau), hingga gambar yang tingkat kontras nya sangat rendah [7].

2.4. Model Warna Pada Citra

Warna dapat diartikan secara objektif sebagai cahaya yang dipancarkan dan dapat diperkirakan oleh panjang gelombang. Sedangkan secara subjektif, warna dapat diartikan sebagai bagian dari indra penglihatan [8]. Warna merupakan elemen dasar dalam membentuk sebuah citra. Pada citra digital, warna dipresentasikan pada setiap pikselnya.

2.4.1. Citra RGB

Citra RGB merupakan citra yang setiap pixelnya memiliki 3 (tiga) nilai warna, yaitu warna merah *(red)*, warna hijau *(green)*, serta warna biru *(blue)*. Sehingga setiap pixel dari citra RGB merupakan perpaduan dari ketiga warna tersebut [6]. Setiap nilai warna masing-masing memiliki nilai 8 bit, sehingga pada citra RGB akan memiliki sekitar 16 juta variasi warna. Citra dengan warna RGB biasa disebut juga dengan *true color*.

2.4.2. Citra Keabuan

Cita keabuan merupakan citra yang hanya memiliki 1 (satu) nilai pada setiap pikselnya. Sehingga dapat dikatakan bahwa pada citra keabuan, nilai warna merah sama dengan hijau dan sama dengan biru. Warna yang dimiliki pada citra keabuan adalah warna putih sebagai nilai maksimalnya, dan warna hitam sebagai nilai minimalnya. Citra keabuan memiliki kedalaman warna 8 bit.

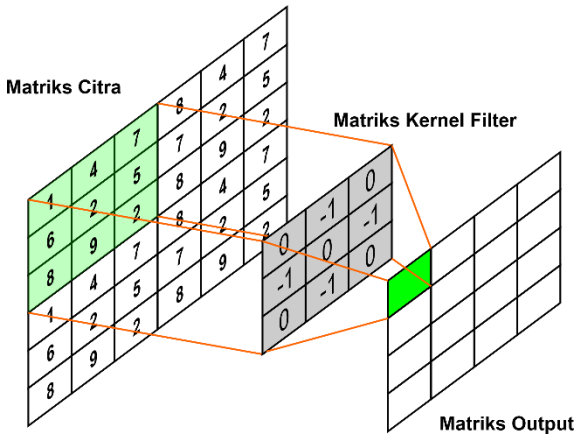
Untuk merubah citra RGB menjadi citra keabuan, dapat menggunakan rumus di bawah ini :

$$s = 0.299 * R + 0.587 * G + 0.114 * B \qquad (2.1)$$

2.5. Konvolusi

Konvolusi merupakan sebuah tahapan dimana citra diolah dengan memanfaatkan *external filter* untuk menghasilkan sebuah citra yang baru. Seperti halnya citra, filter memiliki dimensi tertentu. Filter yang digunakan pada tahap konvolusi memiliki nilai dimana nilai inilah yang akan menjadi sebuah parameter yang nantinya digunakan dalam tahap *learning*.

Pada proses konvolusi, setiap nilai pada matriks citra akan dilakukan operasi *dot* dengan matriks *filter*. Kemudian proses tersebut akan terus berulang hingga posisi *filter* ada di akhir matriks citra (pojok kanan bawah). Pergeseran filter pada tahap konvolusi disebut dengan *stride*. Hasil akhir dari proses konvolusi disebut dengan *feature map*.



Gambar 2.3 Proses Konvolusi

Terdapat beberapa hal yang perlu diperhatikan pada tahap konvolusi yang dimana harus ditentukan dalam layer konvolusi, yaitu:

a. Ukuran dari suatu *filter* harus mengikuti ketebalan dari citra *input*. Jika citra input adalah citra keabuan, maka ketebalan dari *filter* adalah 1 (satu).

b. Ukuran dari *filter* umumnya memiliki dimensi ganjil. Hal ini dikarenakan *filter* dengan dimensi ganjil akan memberikan representasi yang lebih baik dibandingkan dengan *filter* yang berdimensi genap. Serta untuk memudahkan dalam pengambilan nilai yang dihasilkan dari perkalian *dot* antara citra dengan *filter*. Kemudian filter yang digunakan berukuran sama dalam sebuah *convolutional layer*.
c. Jumlah dari filter yang digunakan adalah kelipatan 2 (dua).
d. *Zero padding* memiliki nilai yang menyesuaikan sehingga nilai dari dari *output* yang dihasilkan akan tetap sama dengan ukuran *input* spasial. *Zero padding* sendiri adalah sebuah parameter yang memiliki nilai nol yang ditambahkan pada setiap sisi dari citra *input*. Penambahan *zero padding* sebelum proses konvolusi dilakukan bertujuan agar seluruh piksel di citra i*nput* akan terkena proses konvolusi.

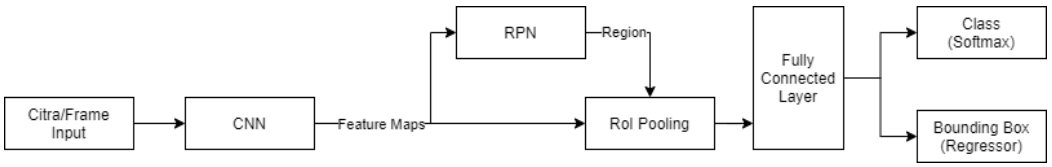
2.6. Faster R-CNN

Faster R-CNN adalah salah satu metode *deep learning* yang masih merupakan bagian dari *neural network*. *Faster R-CNN* merupakan penyempurnaan dari metode *Fast R-CNN* yang telah ditingkatkan kecepatan pemrosesannya menjadi 5 *frame* per detik. Hal itu dikarenakan pada *Faster R-CNN* sudah tidak menggunakan metode eksternal untuk menghasilkan sebuah *region proposer*, melainkan menggunakan *feature map* sehingga mempercepat dalam proses identifikasi sebuah objek [3].

Arsitektur dari algoritma *Faster R-CNN* adalah :

a. CNN (Convolutional Neural Network)
b. RPN (Region Proposal Network)
c. Fast R-CNN (detector)

Sehingga arsitektur dari algoritma Faster R-CNN dapat dilihat pada Gambar 2.4 ***Arsitektur Faster R-CNN***



Gambar 2.4 Arsitektur Faster R-CNN

CNN

Convolutional Neural Network atau yang lebih sering disebut dengan *CNN* merupakan salah satu jenis dari *neural network* yang digunakan pada *image recognition*. *CNN* digunakan pada *image recognition* untuk mengenali objek pada sebuah citra. *CNN* didasari oleh penelitian yang dilakukan oleh D. Hubel dan T. Wiesel yang menggambarkan sel sederhana dan sel kompleks pada sistem penglihatan manusia. Dimana sel sederhana dan kompleks tersebut mereka usulkan dalam pengenalan pola pada citra. Hasil penelitian mereka mengatakan bahwa sel sederhana akan menggambarkan tepi dari suatu objek. Sedangkan sel yang kompleks sama-sama mengenali tepi, namun letak dari tepi tersebut ketika berpindah tempat, sel kompleks masih dapat merespons dan mengenali tepi tersebut. Kedua hal inilah yang mendasari penelitian mereka [9].

CNN sendiri terdiri dari beberapa bagian atau yang lebih dikenal dengan CNN *Layer* yaitu *Convolutional Layer, Pooling Layer,* dan *Fully Connected Layer*. Pada Faster R-CNN, tahap CNN dilakukan hanya sampai *feature maps* dihasilkan. Yaitu hanya sampai tahap *feature learning* saja.

1. Convolutional Layer

Setelah dilakukan *preprocessing*, maka citra akan memasuki tahap *convolutional layer*. *Convolutional layer* dapat diilustrasikan seperti sistem saraf neuron pada manusia yang memiliki dimensi panjang dan tinggi yang dibentuk oleh beberapa filter. Terdapat 3 *hyperparameter* pada *convolutional layer* [10], *yaitu :*

a. Depth

Depth merupakan banyaknya suatu filter pada *convolutional layer*. Dimana setiap filter akan mencari fitur yang berbeda pada objek yang dikenal.

Jumlah filter yang digunakan harus lebih dari satu, dan pada umumnya berjumlah ganjil.

b. *Stride*

Stride merupakan parameter untuk menentukan setiap berapa piksel sekali filter akan bergeser. Dimana semakin kecil nilai dari *stride*, maka semakin detail pula ciri yang dikenali dari suatu objek, namun akan memakan waktu yang jauh lebih lama.

c. Zero-padding

Zero-padding merupakan sebuah parameter bernilai 0 yang disisipkan pada setiap tepi piksel citra input. *Zero-padding* berperan untuk menjaga dimensi antara citra input dan citra output tetap sama, sehingga akan meminimalisir terjadinya informasi yang terbuang.

Rasio dari filter yang digunakan pada proses konvolusi selalu sama, yaitu 1:1. Operasi konvolusi yang dilakukan dengan cara perkalian antara matriks filter dan matriks citra dapat didefinisikan sebagai berikut :

$$f(x,y) = f1.(1,1) + f2.(2,1) + f3.(3.1) + \cdots + fn.(x,y) \qquad (2.2)$$

2. Feature Maps

Feature maps didefinisikan secara singkat sebagai output dari *pooling layer*. *Feature Map* yang dihasilkan tidak dapat langsung digunakan, hal tersebut dikarenakan harus dilakukan *flatten* atau *reshape* ukuran dari *feature map* tersebut menjadi 1 (satu) buah kolom vektor agar dapat digunakan sebagai input untuk *Fully connected layer*.

Ukuran dari feature maps yang dihasilkan dapat dihitung dengan menggunakan persamaan berikut [10]:

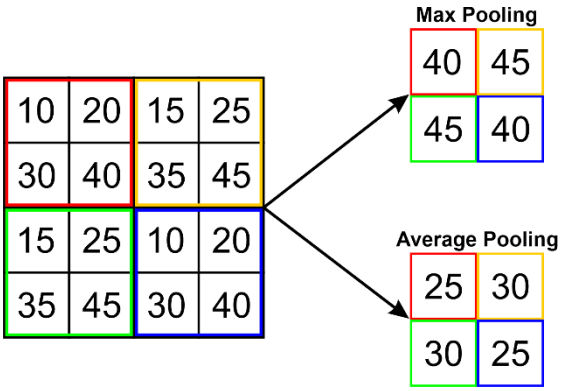
$$featuresize = \frac{inputsize - filtersize + 2pad}{str} + 1 \qquad (2.3)$$

pad = *zero padding*

str = *stride*

3. *Pooling Layer*

Pooling layer merupakan proses untuk memperkecil dimensi dari *feature maps* sehingga dapat mengurangi kemungkinan terjadinya *overfitting*. Pooling layer biasanya ditempatkan secara teratur setelah beberapa proses konvolusi secara berturut-turut. Sehingga dapat dikatakan bahwa dengan menentukan setiap berapa layer sekali pooling layer diterapkan serta berapa banyak jumlah dari pooling layer akan sangat menguntungkan kinerja sistem. Bentuk dari pooling layer yang sangat umum adalah 2x2 dengan *stride* 2, sehingga dapat mengurangi ukuran dari layer sebesar 75%. [11]



Gambar 2.5 *Pooling Layer*

Sesuai Namanya, *max pooling* akan mengambil nilai terbesar dari matriks citra, sedangkan *average pooling* akan mengambil nilai rata-rata dari hasil layer konvolusi. Namun, umumnya operasi yang sering digunakan adalah *max pooling*.

Terdapat dua buah *hyperparameter* pada *pooling layer*. Yaitu ukuran dari *pooling window* dan *stride*. Sehingga ukuran dari *feature maps* setelah melalui tahap *pooling layer* dapat dihitung menggunakan rumus berikut [10]:

$$featuresize = \frac{inputsize - windowsize}{str} + 1 \qquad (2.4)$$

inputsize = ukuran dari *feature maps* sebelumnya

windowsize = ukuran jendela *pooling layer*

str = *stride*

Kemudian, piksel hasil perkalian antara matriks *feature* maps sebelumnya dengan *pooling* window dapat dihitung menggunakan rumus

$$P_i = \max((1,1),(1,2),\dots,(x,y)) \qquad (2.5)$$

Pada CNN, *Convolutional Layer* dan *Pooling layer* termasuk pada tahap *feature learning* atau *feature extraction layer* yang akan menghasilkan *feature map*.

4. Fully Connected Layer

Fully Connected layer merupakan cara tercepat untuk mempelajari kombinasi fitur-fitur yang didapatkan dari *feature extraction layer*. *Fully Connected layer* berupa sebuah lapisan dimana seluruh neuron yang didapat dari lapisan sebelumnya saling terhubung satu sama lainnya. Oleh karena itu, sebelum dapat diproses, hasil dari lapisan sebelumnya harus diubah menjadi sebuah kolom vektor melalui proses *flatten*.

Fully Connected Layer termasuk kedalam *multilayer perceptron*, dimana terdapat beberapa lapisan lainnya, yaitu *hidden layer, activation layer, output layer,* serta, *loss function* [11]. Jumlah dari *hidden layer* yang terdapat dari *Fully Connected Layer* adalah sebanyak $^2/_3$ dari jumlah *input layer* ditambah dengan *output layer*.

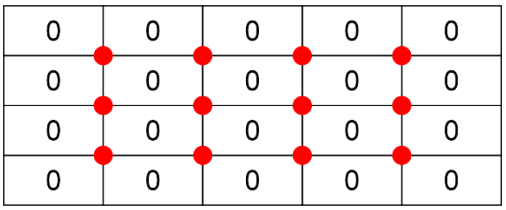
2.6.2. *ReLU Activation*

Aktivasi ReLu merupakan sebuah lapisan pada CNN yang melakukan *thresholding* pada sebuah pixel citra. *Thresholding* yang dilakukan yaitu dengan memberi nilai 0 pada seluruh citra yang bernilai negatif.

$$f(x) = \max(0, x) \tag{2.6}$$

2.6.3. RPN

Region Proposal Network (RPN) memiliki *classifier* dan *regressor*, sehingga *output* yang dihasilkan oleh *region proposal network* adalah sekelompok *box area* yang dimana akan diperiksa oleh *classifier* dan *box regressor* untuk menentukan apakah terdapat objek di dalamnya. Kemudian akan memprediksi apakah pada setiap *anchor* terdapat sebuah objek atau tidak. *Anchor* merupakan sebuah area dengan 3 rasio yang berbeda serta 3 skala yang berbeda. Yaitu rasio 1:1, 2:1, serta 1:2, dan skala 128, 256, dan 512. dimana setiap ujung piksel selain bagian paling luar dari matriks adalah sebagai pusatnya. Sehingga setiap *anchor* memiliki 9 buah *proposal*.



Gambar 2.6 Letak setiap *anchor*

RPN pada *faster R-CNN* menggantikan fungsi *selective search* yang terdapat pada *fast R-CNN*. Pada *output* pertama untuk menentukan *region proposal*, *RPN* tidak akan mementingkan kelas dari suatu objek, melainkan hanya mencari mana yang terlihat seperti objek pada suatu citra. Kemudian *output* yang kedua adalah memberikan *bounding box regressor* yang bertujuan untuk menyesuaikan posisi dari *anchor* agar lebih sesuai dengan objek yang akan diprediksi.

Banyaknya *anchor* pada sebuah *feature map* dapat dihitung dengan menggunakan persamaan berikut:

$$anchor = Fw * Fh * 9 \tag{2.7}$$

Fw = Panjang *feature map*

Fh = Lebar *feature map*

Struktur dari RPN sendiri adalah sebuah *sliding window* dengan kernel konvolusi yang hanya berukuran 3x3 yang kemudian akan menghasilkan sebuah *feature map* yang berisi *cls layer* dan *reg layer*. *Cls layer* merupakan sebuah matriks yang dimensinya sesuai dengan probabilitas antara *foreground* dan *background* dari setiap k *region* pada setiap piksel di *feature maps*. Sedangkan *reg layer* adalah matriks yang sesuai dengan posisi dari setiap bounding box yang menyerupai dengan setiap hasil dari *anchor box*. [12]

Untuk menentukan apakah dalam suatu *anchor* terdapat sebuah objek atau tidak, dapat menggunakan persamaan *Intersect over Union* dibawah ini.

$$\text{IoU} = \frac{A \cup Gt}{A \cap Gt} \begin{cases} > 0{,}7 = objek \\ < 0{,}3 = bukan\ objek \end{cases} \qquad (2.8)$$

Sedangkan untuk menghitung besaran *error* yang terjadi dapat menggunakan fungsi dibawah ini

$$L(p_i, t_i) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \qquad (2.9)$$

2.7. Anotasi

Anotasi merupakan tahap dimana setiap objek pada data latih yang hendak dipelajari oleh sistem akan diberikan label dan *bounding box* oleh user. *Bounding box* akan menghasilkan koordinat piksel, serta Panjang dan lebar dari objek yang ditandai. Anotasi termasuk kedalam tahap ekstraksi fitur, dimana ekstraksi fitur merupakan sebuah tahap untung mencari ciri khas dari suatu objek sehingga sistem mudah untuk mengenali objek tersebut [13].

2.8. Transfer Learning

Transfer learning adalah sebuah metode yang memanfaatkan *pre-trained model* yang sudah ada terhadap suatu dataset. Sehingga dapat mempercepat proses *training* tanpa mengurangi tingkat akurasi dari model tersebut. Pengguna dapat memperbaharui parameter dari *pre-trained model* tersebut, sehingga dapat disesuaikan dengan dataset yang akan digunakan. [14]

2.9. Sistem Berbasis Aturan

Sistem berbasis Aturan *(Rule Based System)* merupakan suatu metode untuk menginterpretasikan sebuah informasi dengan mengimplementasikan aplikasi dari kecerdasan buatan. Dimana sistem berbasis aturan terdiri dari dua bagian, yaitu [15]:

a. *Antecedent* yang menyatakan premis. Dimana pernyataan tersebut diawali dengan IF.

b. *Consequent*, yang dimana menyatakan Tindakan tertentu yang diterapkan apabila premis bernilai benar.

2.10. *Confusion Matrix*

Confusion matrix digunakan untuk mencatat kinerja dari hasil klasifikasi, dimana permasalahan yang diselesaikan dalam rupa klasifikasi biner dua kelas. Pada *confusion matrix* dikenal beberapa istilah, yaitu [16] :

a. *True positive (TP)* adalah jumlah klasifikasi data yang hasil klasifikasinya benar dan memenuhi kondisi yang diharapkan.

b. *True negative (TN)* adalah jumlah klasifikasi data yang hasil klasifikasinya salah dan memenuhi kondisi yang diharapkan.

c. *False positive (FP)* adalah jumlah klasifikasi data yang hasil klasifikasinya benar namun tidak memenuhi kondisi yang diharapkan.

d. *False negative (FN)* adalah jumlah klasifikasi data yang hasil klasifikasinya salah namun tidak memenuhi kondisi yang diharapkan.

Kemudian, hasil akurasi dari sistem yang diuji dapat dihitung menggunakan persamaan berikut

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.10)$$

2.11. Python

Python merupakan sebuah bahasa pemrograman tingkat tinggi interpretatif, multiguna, serta dinamis. *Python* mendukung berbagai paradigma pemrograman, seperti pemrograman berorientasi objek, pemrograman fungsional, dan pemrograman imperative. *Python* bersifat dinamis karena *python* dilengkapi dengan sistem manajemen memori yang otomatis. [17]

Python dikategorikan sebagai Bahasa pemrograman tingkat tinggi dikarenakan *python* mudah untuk dipelajari bahkan oleh seorang pemula yang sebelumnya belum memahami Bahasa pemrograman manapun. Bahasa ini dikembangkan oleh Guido van Rossum pada tahun 1990 di *Stichting Mathematisch Centrum* (CWI) Amsterdam sebagai bahasa pemrograman ABC yang telah diperbaharui. Kemudian, Guido melanjutkan pengembangan *python* pada *Corporation for National* Research (CNRI) di Virgina, sebelum akhirnya berpindah ke BeOpen.com hingga tahun 200, dan akhirnya hingga saat ini *python* dikembangkan oleh *Python Software Foundation*. [18]

2.12. Keras

Keras merupakan API *neural network* yang ditulis menggunakan Bahasa pemrograman *python*. Keras berjalan diatas *platform* TensorFlow. Keras bersifat *high-level*, sehingga mampu memecahkan masalah pembelajaran mesin hingga *modern deep learning*. Beberapa kemampuan utama dari Keras adalah [19]:

Mampu mengeksekusi *source code* menggunakan CPU, GPU, hingga TPU dengan lancer.

Memiliki API yang *user-friendly* sehingga pemula sekalipun mudah menggunakannya.

Mampu melakukan *scaling* komputasi ke banyak perangkat.

Mampu mengekspor program ke perangkat luar (seperti server, browser, mobile).