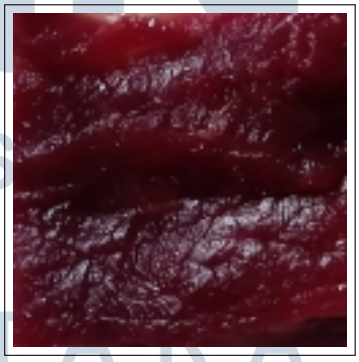
BAB 2
LANDASAN TEORI

2.1 Daging Segar dan Daging Thawed

Daging adalah makanan yang sangat bergizi dan lezat yang dihargai oleh konsumen di seluruh dunia. Daging memerlukan metode penyimpanan yang tepat untuk menjaga kualitasnya yang optimal hingga digunakan [1]. Kualitas daging segar menunjukkan kegunaannya bagi konsumen dan kelayakannya untuk dimasak. Sifat kualitas penting untuk daging segar adalah warna, *water holding capacity* (WHC), tekstur, dan jumlah lemak (lemak intramuskular / lemak *intermuscular* / lemak subkutan), sementara sifat penting untuk kualitas makanan dari daging yang dimasak adalah kelembutan, rasa, dan kelezatan. Secara umum, konsumen menilai warna sebagai sifat kualitas paling penting untuk daging segar, sementara kelembutan dinilai sebagai sifat kelezatan paling penting untuk daging yang dimasak diikuti oleh rasa dan kelezatan [13].

Pembekuan daging adalah praktik untuk memperpanjang umur simpannya. Meskipun produk berkualitas tinggi telah dipraktikkan selama ribuan tahun, sebagian besar perbaikan dalam teknologi pembekuan terjadi dalam satu abad terakhir. Secara umum, kualitas makanan beku sangat terkait dengan proses pembekuan dan pencairan. Pencairan merujuk pada proses pelelehan yang mengubah dari keadaan beku menjadi keadaan cair (mencair) atau menjadi bebas dari efek (seperti kekakuan, mati rasa, atau keras) dingin sebagai hasil dari paparan panas [14].

Gambar 2.1. Daging Thawed



Gambar 2.2. Daging Segar

2.2 Feature Extraction

Feature Extraction adalah proses utama dalam mengekstraksi informasi berharga dari gambar dan kemudian menggunakan informasi ini untuk menyelesaikan pekerjaan. Ini juga merupakan bagian dari proses reduksi dimensi. Karakteristik penting dari kumpulan data yang lebih besar adalah kumpulan data tersebut memiliki sejumlah besar variabel, sehingga memerlukan banyak sumber daya komputasi untuk memprosesnya. Ekstraksi fitur pada dasarnya adalah proses utama dalam mengekstraksi informasi berharga dari kumpulan data ini dan kemudian menggunakan informasi tersebut untuk menyelesaikan pekerjaan. [15]

Image processing adalah salah satu domain yang menerapkan proses ekstraksi fitur untuk memperkecil gambar masukan dan mengekstrak informasi berharga darinya, sehingga mengurangi tingginya kebutuhan penyimpanan dan daya komputasi sistem. Ada beberapa metode atau algoritma yang ditentukan untuk melakukan tugas ekstraksi fitur. [15]

Dalam ekstraksi fitur, fokusnya adalah menemukan set baru dari k dimensi, yang merupakan kombinasi dari d dimensi asli. Metode ekstraksi fitur yang paling dikenal dan paling umum digunakan adalah analisis komponen utama dan analisis diskriminan linear, teknik pembelajaran tanpa pengawasan dan dengan pengawasan. Analisis komponen utama sangat mirip dengan dua metode linear tanpa pengawasan lainnya, analisis faktor dan skala multidimensi. Ketika ada dua set variabel yang diamati, analisis korelasi kanonik dapat digunakan untuk menemukan fitur bersama, yang menjelaskan ketergantungan antara keduanya. [16]

2.3 Descrete Cosine Transform

Transformasi *Descrete Cosine Transform* (DCT) adalah transformasi yang terkenal dalam matematika, yang pada dasarnya mentransformasikan sinyal dari domain spasial ke domain frekuensi. Ini mendekomposisi sebuah sinyal menjadi serangkaian fungsi harmonik kosinus. Karena sifat decorrelation dan kompaksi energi yang lebih baik, DCT telah digunakan dalam banyak aplikasi seperti kompresi data dan pengenalan pola. Secara matematis, transformasi DCT 2D dapat didefinisikan sebagai: [17]

$$C(u,v) = \alpha(u)\alpha(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cos\left[\frac{\pi(2x+1)u}{2N}\right] \cos\left[\frac{\pi(2y+1)v}{2M}\right], \quad (2.1)$$

Dimana:

C(u, v) = nilai DCT indeks ke-(u,v)

f(x, y) = nilai *pixel* pada indeks ke-(x,y)

M = ukuran baris matriks

N = ukuran kolom matriks

$$\alpha(u), \alpha(v) = \begin{cases} \sqrt{\frac{1}{N}} & \text{for} \quad u, v = 0 \\ \sqrt{\frac{2}{N}} & \text{for} \quad u, v \neq 0. \end{cases}$$

2.4 Gray Level Co-occurrence Matrix

Gray Level Co-occurrence Matrix (GLCM) adalah sebuah fitur ekstraksi yang mengambil konsep dari *seon-order statistical features* yang merupakan pecahan dari *Haralick features* [18]. *Gray Level Co-Occurrence Matrix (GLCM)* telah terbukti menjadi metode statistik populer untuk mengekstraksi fitur tekstur dari gambar. Menurut *co-occurrence matrix*, Haralick mendefinisikan empat belas fitur tekstur yang diukur dari matriks probabilitas untuk mengekstraksi karakteristik statistik tekstur dari gambar penginderaan jauh [19].

Persamaan satu set 28 fitur tekstural yang dapat diekstraksi dari setiap matriks ketergantungan spasial nada abu-abu berikut mendefinisikan fitur-fitur ini [20]:

Notasi:

- $p(i, j)$: (i, j) entri ke-(i, j) dalam matriks ketergantungan spasial nada abu-abu yang dinormalisasi, $= P(i, j)/R$.
- $p_x(i)$: i entri ke-i dalam matriks probabilitas marjinal yang diperoleh dengan menjumlahkan baris dari $p(i, j)$, $= \sum_{j=1}^{N_g} P(i, j)$.
- N_g: Jumlah tingkat abu-abu yang berbeda dalam citra yang dikuantisasi.
- $$\sum_{i} \text{ and } \sum_{j} \sum_{i=1}^{N_g} \text{ and } \sum_{j=1}^{N_g}, \text{ respectively.} \tag{2.2}$$
- $$p_y(j) = \sum_{i=1}^{N_g} p(i, j). \tag{2.3}$$

- $$p_{x+y}(k) = \sum_{\substack{i=1 \\ i+j=k}}^{N_g} \sum_{j=1}^{N_g} p(i,j), \quad k = 2,3,\cdots,2N_g. \tag{2.4}$$

- $$p_{x-y}(k) = \sum_{\substack{i=1 \\ |i-j|=k}}^{N_g} \sum_{j=1}^{N_g} p(i,j), \quad k = 0,1,\cdots,N_g - 1. \tag{2.5}$$

Textural Features:

1) *Angular Second Moment*:

$$f_1 = \sum_i \sum_j \{p(i,j)\}^2 \tag{2.6}$$

2) *Contrast*:

$$f_2 = \sum_{n=0}^{N_g-1} n^2 \left\{ \sum_{\substack{i=1 \\ |i-j|=n}}^{N_g} \sum_{j=1}^{N_g} p(i,j) \right\}. \tag{2.7}$$

3) *Correlation*:

$$f_3 = \frac{\sum_i \sum_j (ij) p(i,j) - \mu_x \mu_y}{\sigma_x \sigma_y} \tag{2.8}$$

dimana μ_x, μ_y, σ_x, dan σ_y merupakan *mean* dan *standard deviation* dari p_x dan p_y.

4) *Sum of Squares: Variance*

$$f_4 = \sum_i \sum_j (i-\mu)^2 p(i,j). \tag{2.9}$$

5) *Inverse Difference Moment*:

$$f_5 = \sum_i \sum_j \frac{1}{1+(i-j)^2} p(i,j). \tag{2.10}$$

6) *Sum Average*:

$$f_6 = \sum_{i=2}^{2N_g} i p_{x+y}(i). \tag{2.11}$$

7) *Sum Variance*:

$$f_7 = \sum_{i=2}^{2N_g} (i - f_8)^2 p_{x+y}(i). \tag{2.12}$$

8) *Sum Entropy*: [2]

$$f_8 = -\sum_{i=2}^{2N_g} p_{x+y}(i) \log\left\{p_{x+y}(i)\right\}. \tag{2.13}$$

[2] Karena beberapa probabilitas mungkin bernilai nol, dan $\log(0)$ tidak terdefinisi, disarankan agar istilah $\log(p+\varepsilon)(\varepsilon$ adalah konstanta positif yang sangat kecil) digunakan sebagai pengganti $\log(p)$ dalam perhitungan entropi [20].

9) *Entropy*:

$$f_9 = -\sum_i \sum_j p(i,j) \log(p(i,j)). \tag{2.14}$$

10) *Difference Variance*:

$$f_{10} = \textit{variance} \text{ dari } p_{x-y}. \tag{2.15}$$

11) *Difference Entropy*:

$$f_{11} = -\sum_{i=0}^{N_{g-1}} p_{x-y}(i) \log\left\{p_{x-y}(i)\right\}. \tag{2.16}$$

12), 13) *Information Measures of Correlation*:

$$f_{12} = \frac{HXY - HXY1}{\max\{HX, HY\}} \tag{2.17}$$

$$f_{13} = (1 - \exp[-2.0(HXY2 - HXY)])^{1/2} \tag{2.18}$$

$$HXY = -\sum_i \sum_j p(i,j) \log(p(i,j)) \tag{2.19}$$

dimana HX and HY adalah entropi-entropi dari p_x dan p_y, dan

$$HXY1 = -\sum_i \sum_j p(i,j) \log \left\{ p_x(i) p_y(j) \right\}$$
$$HXY2 = -\sum_i \sum_j p_x(i) p_y(j) \log \left\{ p_x(i) p_y(j) \right\}. \tag{2.20}$$

14) *Maximal Correlation Coefficient*:

$$f_{14} = (\text{ Nilai eigen terbesar kedua dari } Q)^{1/2} \tag{2.21}$$

dimana

$$Q(i,j) = \sum_k \frac{p(i,k)p(j,k)}{p_x(i)p_y(k)}. \tag{2.22}$$

Terdapat beberapa fitur yang dapat diekstraksi dari matriks GLCM, yaitu sebagai berikut: [21]

1. *Homogeneity*, merupakan sebuah ukuran keseragaman teksur lokal pada sebuah citra. Apabila nilai dari *Homogenity* tinggi maka terdapat keseragaman yang tinggi [21]. *Homogenity* dapat dinyatakan pada persamaan (2.23).

$$\text{Homogeneity } = \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{\text{GLCM}(i,j)}{1+|i-j|} \tag{2.23}$$

2. *Dissimilarity*, merupakan sebuah nilai yang menyatakan ketidak miripan pada tekstur. Apabila tekstur acak maka nilai *dissimilarity* akan tinggi, dan apabila tekstur seragam maka nilai *dissimilarity* akan rendah [21]. *Dissimilarity* dapat dinyatakan pada persamaan (2.24).

$$\text{Dissimilarity } = \sum_{i,j=0}^{N-1} GLCM|i-j| \tag{2.24}$$

Dimana:

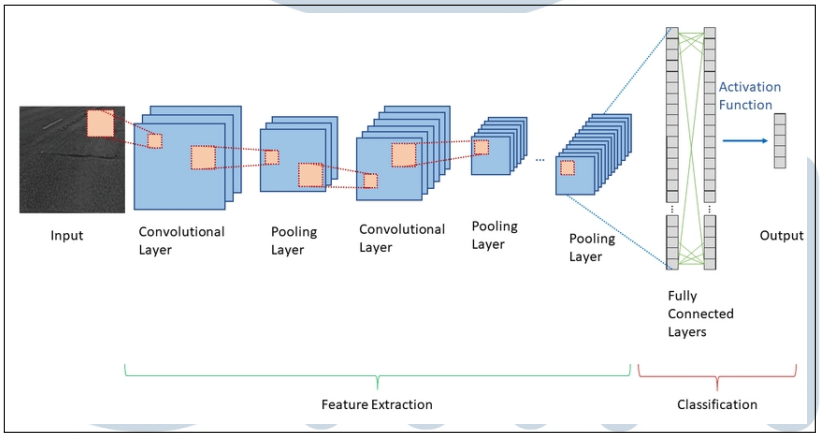
$$\text{GLCM}(i,j) = \sum_{(x,y)} P(x,y \mid dx,dy) \tag{2.25}$$

GLCM(i,j) = Persamaan dalam pembentukan matriks GLCM

$P(x, y \mid dx, dy)$ = Sebuah probabilitas untuk menemukan pasangan piksel *(x, y)* dengan intensitas abu *i* dan piksel *(x + dx,y + dy)* dengan intensitas abu *j* pada jarak *dx* dan *dy*

2.5 Convolutional Neural Network

Convolutional Neural Networks (CNN) adalah sistem kecerdasan buatan berbasis jaringan saraf multi-lapisan yang dapat mengidentifikasi, mengenali, dan mengklasifikasikan objek serta mendeteksi dan memisahkan objek dalam gambar. CNN atau ConvNet merupakan arsitektur *deep learning* yang populer dan diskriminatif yang dapat dipelajari langsung dari objek masukan tanpa memerlukan ekstraksi fitur manusia. Jaringan ini sering digunakan dalam pengenalan visual, analisis gambar medis, segmentasi gambar, NLP, dan berbagai aplikasi lainnya karena dirancang khusus untuk menangani berbagai bentuk 2D [22]. Model umum dari CNN terdiri dari empat komponen, yaitu (a) *convolution layer*, (b) *pooling layer*, (c) *activation function*, and (d) *fully connected layer* [23].



Gambar 2.3. Arsitektur CNN
Sumber: [24]

2.5.1 Convolution Layer

Convolutional layer adalah lapisan yang digunakan untuk melakukan operasi konvolusi pada *output layer* sebelumnya. *Layer* ini termasuk blok utama pada *Convolutional Neural Network* (CNN) yang didalamnya terdiri dari filter – filter yang di pelajari secara acak untuk melakukan operasi konvolusi yang bertujuan

sebagai ekstraksi fitur untuk mempelajari representasi fitur dari *input layer*. Tujuan dilakukannya operasi konvolusi pada data citra untuk mengekstraksi fitur dari input citra [25]. Berikut ini merupakan rumus yang digunakan pada *convolutional layer* [26]:

$$n_{(w,h)} = \left[\frac{n_{in} + 2p - k}{s}\right] + 1 \tag{2.26}$$

Dimana:

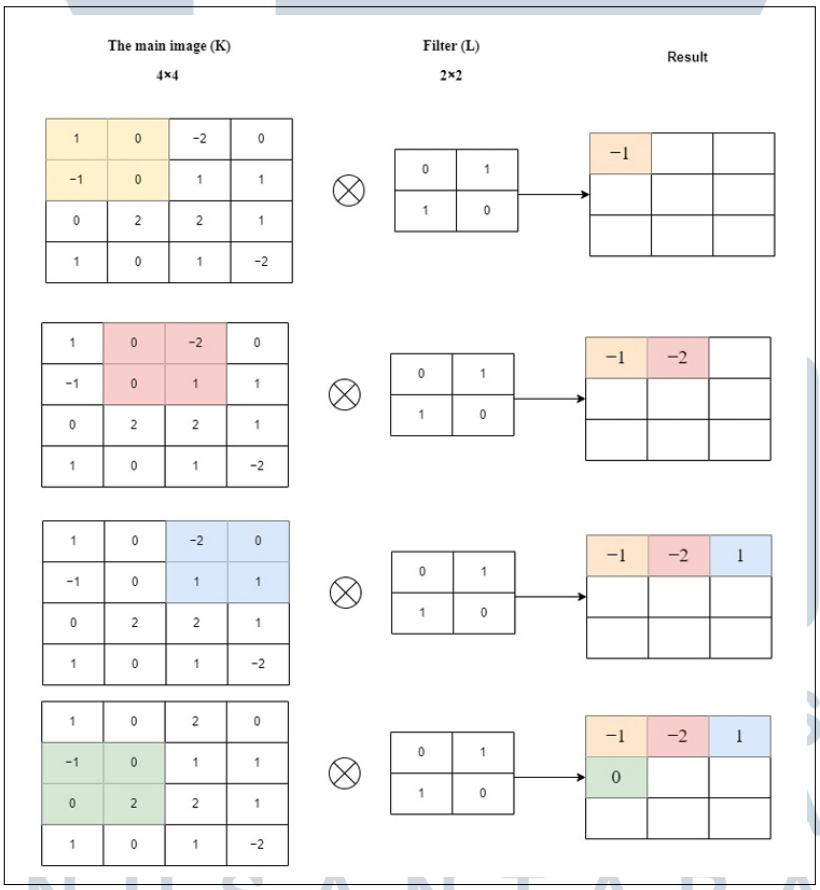
$n_{(w,h)}$ = Hasil *input* ukuran citra

k = Ukuran *kernel* yang digunakan

s = Ukuran *stride*

p = Ukuran *padding*

n_{in}= Nilai ukuran citra *input*



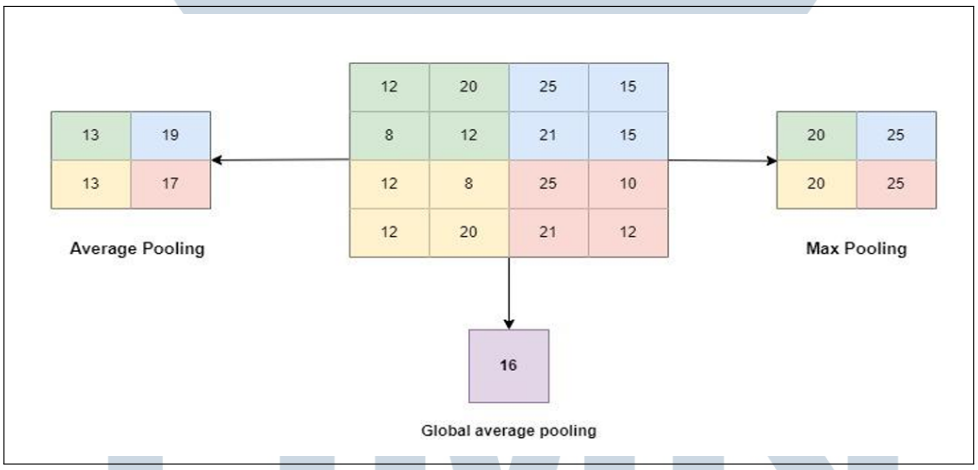
Gambar 2.4. Perhitungan Utama Dieksekusi pada Setiap Langkah Lapisan Konvolusional

Sumber: [22]

2.5.2 Pooling Layer

Pooling layer, yang juga dikenal sebagai *down-sampling layer*, digunakan untuk mengurangi dimensi peta fitur sambil mempertahankan data yang paling penting. Filter menerapkan operasi *pooling* pada data input dengan menggesernya di lapisan *pooling* (*max*, *min*, *avg*). Dalam literatur, *pooling* maksimum paling sering digunakan [27].

Bagian penting dari *pooling*, yang digunakan untuk mengurangi kompleksitas lapisan atas, adalah *down-sampling*. Dalam hal pemrosesan gambar, ini mungkin sebanding dengan mengurangi resolusi. Jumlah filter tidak terpengaruh oleh *pooling*. *Max-pooling* adalah salah satu metode *pooling* yang paling sering digunakan. Gambar dibagi menjadi *subregion* berbentuk persegi panjang, dan hanya nilai terbesar yang ditemukan di dalam setiap *subregion* yang dikembalikan. Salah satu ukuran *max-pooling* yang paling umum adalah 2x2 [27].



Gambar 2.5. Pooling Layer
Sumber: [22]

Seperti yang ditunjukkan pada Gambar 2.5, ketika *pooling* digunakan pada blok 2x2 di sudut kiri atas, perhatian dialihkan ke sudut kanan atas, dan dua langkah dipindahkan. Akibatnya, *stride* 2 digunakan untuk *pooling*. Dimungkinkan untuk menggunakan *stride* 1, yang tidak biasa, untuk mencegah *downsampling*. Perlu diingat bahwa *downsampling* tidak mempertahankan posisi data [27].

Pada berbagai tingkat *pooling*, berbagai teknik *pooling* dapat diterapkan. *Global average pooling* (GAP), *global max pooling*, *average pooling*, *min pooling*, dan *gated pooling* adalah beberapa metode tersebut. Gambar 2.5 menggambarkan masing-masing dari tiga teknik *pooling* ini [27]. Berikut ini merupakan rumus untuk

menghitung *max pooling layer* [26]:

$$n_{(w,h)} = \frac{\left(n_{(w,h)-1} - f\right)}{s} + 1 \tag{2.27}$$

Dimana:
$n_{(w,h)}$ = Hasil ukuran *height* dan *width*
$n_{(w,h)-1}$ = Ukuran *weight* dan *height* sebelumnya
s = Ukuran *stride*
f = Ukuran *kernel*
.

2.5.3 Activation Function

Fungsi aktivasi adalah fungsi *non linear* yang memungkinkan sebuah Jaringan Syaraf Tiruan (JST) untuk dapat mentransformasi data *input* menjadi dimensi yang lebih tinggi sehingga dapat dilakukan pemotongan *hyperlane* sederhana yang memungkinkan dilakukan klasifikasi [28]. Fungsi aktivasi juga harus memiliki kemampuan untuk diferensiasi, yang merupakan fitur yang sangat penting, karena memungkinkan penggunaan *back-propagation* kesalahan untuk melatih jaringan. Jenis-jenis fungsi aktivasi berikut ini paling sering digunakan dalam CNN dan jaringan saraf dalam lainnya [29].

- *Sigmoid*: Input dari fungsi aktivasi ini adalah bilangan real, sedangkan outputnya dibatasi antara nol dan satu. Kurva fungsi *sigmoid* berbentuk S dan dapat diwakili secara matematis oleh Persamaan 2.25 [29]:

$$f(x)_{\text{sigm}} = \frac{1}{1+e^{-x}} \tag{2.28}$$

- *Tanh*: Ini mirip dengan fungsi *sigmoid*, karena inputnya adalah bilangan real, tetapi outputnya dibatasi antara -1 dan 1. Representasi matematisnya ada dalam Persamaan 2.26 [29]:

$$f(x)_{\tanh} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.29}$$

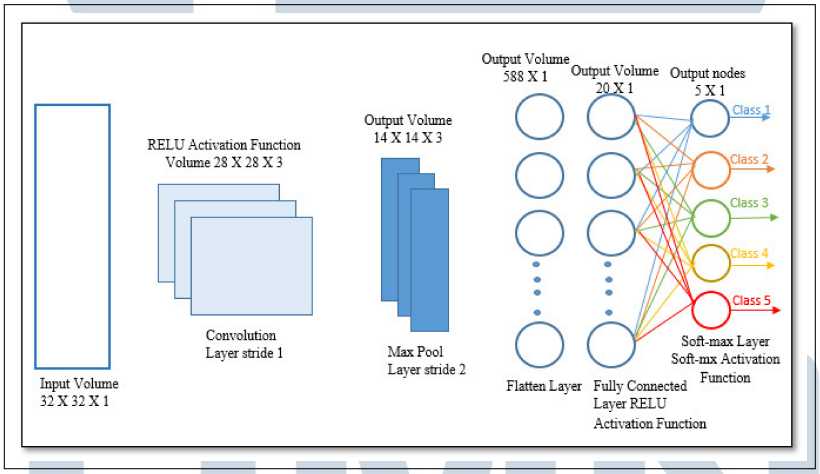
- *ReLU*: Fungsi yang paling sering digunakan dalam konteks CNN. Fungsi ini mengubah semua nilai *input* menjadi bilangan positif. Beban komputasi yang lebih rendah adalah manfaat utama *ReLU* dibandingkan yang lain.

Representasi matematisnya ada dalam Persamaan 2.27 [29]:

$$f(x)_{\text{ReLU}} = \max(0, x) \tag{2.30}$$

2.5.4 Fully Connected Layer

Fully connected layer adalah lapisan yang digunakan untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear. Untuk mendapatkan hasil keluaran dari *layer* ini tidak dibutuhkan operasi konvolusi, tetapi menggunakan komputasi perkalian matriks yang diikuti dengan *bias offset*. Dengan penggunaan operasi tersebut, setiap *neuron* memiliki koneksi penuh ke semua aktivasi dalam lapisan sebelumnya, sehingga *layer* ini disebut sebagai f*ully conected layer* [25].

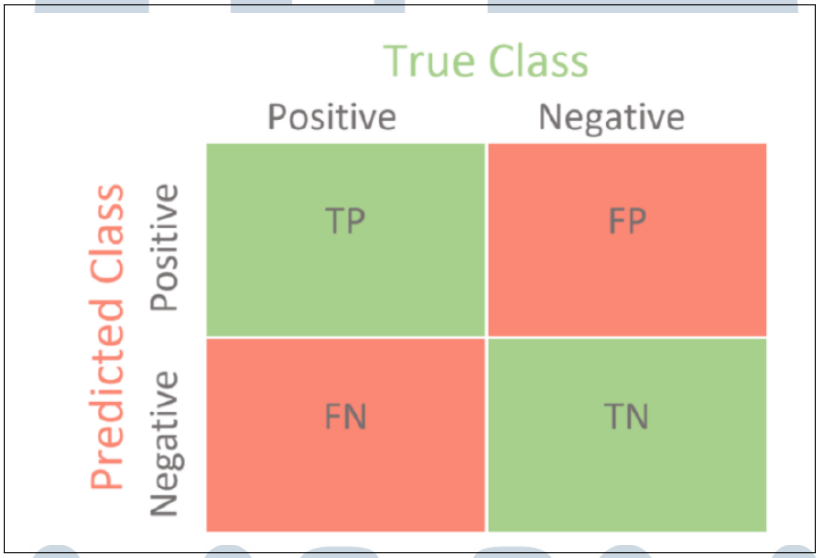


Gambar 2.6. Fully Connected Layer
Sumber: [30]

Fully connected layer tidak lain adalah jaringan saraf *feed-forward* seperti yang ditunjukkan pada Gambar 2.6. Lapisan *fully connected* ditemukan di lapisan paling bawah jaringan. Lapisan *fully connected* menerima *input* dari lapisan *output pooling* atau *convolutional* terakhir, yang diratakan sebelum dikirim sebagai *input*. Meratakan *output* melibatkan mengurai semua nilai dari *output* yang diperoleh setelah lapisan *pooling* atau *convolutional* terakhir menjadi sebuah *vektor* (matriks 3D). Menambahkan lapisan *fully connected* adalah teknik sederhana untuk mempelajari kombinasi *nonlinier* dari fitur tingkat tinggi yang diwakili oleh *output* lapisan *convolutional*. Di ruang tersebut, lapisan *fully connected* mempelajari fungsi yang mungkin *nonlinier* [30].

2.6 Confusion Matrix

Confusion matrix adalah alat untuk analisis prediktif dalam pembelajaran mesin. Untuk memeriksa kinerja model pembelajaran mesin berbasis klasifikasi, *confusion matrix* digunakan. *Confusion matrix* adalah matriks NxN yang digunakan untuk mengevaluasi kinerja model klasifikasi, di mana N adalah jumlah kelas target. Dengan memvisualisasikan *confusion matrix*, dapat ditentukan akurasi model dengan mengamati nilai diagonal untuk mengukur jumlah klasifikasi yang akurat [31].



Gambar 2.7. Confusion Matrix
Sumber: [31]

Confusion matrix berbentuk matriks persegi di mana kolom mewakili nilai aktual dan baris menggambarkan nilai prediksi model, dan sebaliknya [31].

- *True Positive* (TP): Jumlah sampel positif yang benar-benar diidentifikasi dengan benar oleh model.
- *False Positive* (FP): Jumlah sampel negatif yang salah diklasifikasikan sebagai positif oleh model.
- *False Negative* (FN): Jumlah sampel positif yang salah diklasifikasikan sebagai negatif oleh model.
- *True Negative* (TN): Jumlah sampel negatif yang benar-benar diidentifikasi dengan benar oleh model.