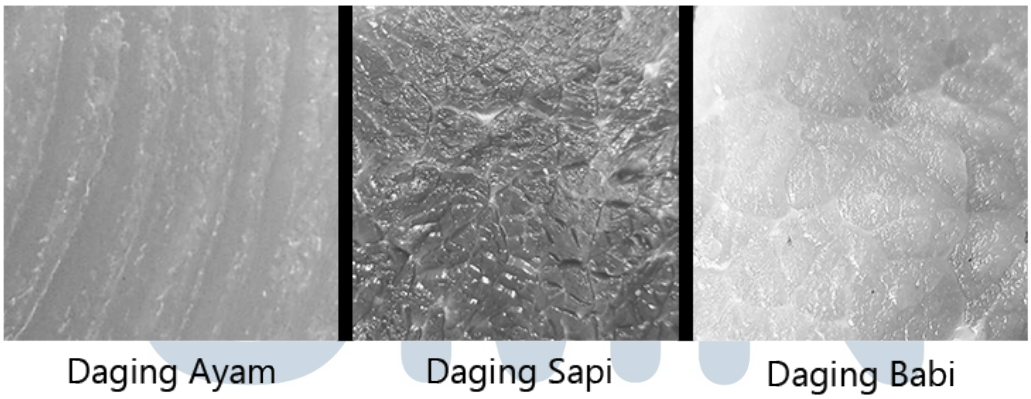
BAB 2
LANDASAN TEORI

Berikut adalah telaah literatur teori yang dapat mendukung penelitian yang akan dilakukan.

2.1 Daging

Perbedaan antara berbagai jenis daging dapat dikenali berdasarkan warna serta tekstur dagingnya. Daging sapi umumnya memiliki daging yang berwarna merah dengan serat yang halus dan sedikit lemak yang berwarna kekuningan [16]. Di sisi lain, daging babi cenderung memiliki warna yang lebih pucat dengan serat yang lebih halus daripada daging sapi, dan lemaknya yang tebal berwarna putih [17]. Selain itu, daging ayam memiliki warna yang keputih-putihan dengan serat daging yang halus dan panjang, tanpa lemak yang terlihat pada dagingnya, sementara lemak ayam biasanya terletak di bawah kulit dan berwarna kekuning-kuningan [18].



Gambar 2.1. Contoh gambar daging

2.2 Ekstraksi Fitur

Ekstraksi fitur adalah langkah untuk mengambil fitur atau karakteristik dari sebuah objek yang berfungsi sebagai penanda unik dari objek tersebut, membedakannya dari objek lainnya [19]. Ekstraksi fitur dilakukan untuk mendapatkan nilai ciri dari sebuah citra [11]. Ekstraksi fitur bisa dibagi menjadi tiga kategori yaitu, *low level*, *middle level*, dan *high level*. Fitur *low level* fokus

pada elemen visual seperti warna dan tekstur, *middle level* berpusat pada wilayah citra yang diidentifikasi melalui segmentasi, sementara fitur *high level* menekankan pada informasi semantik dalam citra [16].

2.3 Discrete Cosine Transform

Discrete Cosine Transform atau DCT pertama kali ditemukan oleh Nasir Ahmed pada tahun 1972, yang dapat digunakan dalam bidang pemrosesan digital untuk tujuan pengenalan pola dan penyaringan sinyal [20]. DCT adalah proses matematika yang mengubah sinyal dari ruang spasial ke ruang frekuensi [21]. DCT membagi gambar ke dalam komponen frekuensi yang berbeda, menghilangkan frekuensi yang kurang penting dengan proses kuantisasi [21]. DCT beroperasi dengan memecah gambar menjadi blok-blok berukuran 8x8 piksel. Proses DCT berlangsung dari kiri ke kanan dan dari atas ke bawah pada setiap piksel. Setiap blok kemudian dikompresi melalui kuantisasi. Blok-blok terkompresi yang menyusun gambar ini disimpan dengan menggunakan ruang yang sangat efisien [22]. DCT dapat dinyatakan pada pernyataan 1 [23].

$$F(u,v) = \left(\frac{2}{N}\right)^{\frac{1}{2}} \left(\frac{2}{M}\right)^{\frac{1}{2}} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \Lambda(i) \cdot \Lambda(j) \cdot \cos\left[\frac{\pi \cdot u}{2 \cdot N}(2i+1)\right] \times \cos\left[\frac{\pi \cdot v}{2 \cdot M}(2j+1)\right] \cdot f(i,j) \tag{2.1}$$

F(u, v) = nilai DCT indeks ke-(u,v)
f(i, j) = nilai *pixel* pada indeks ke-(i,j)
M, N = ukuran baris dan kolom matriks
$\Lambda(\varepsilon) = 1$ jika $(\varepsilon) > 0$
$\Lambda(\varepsilon) = \frac{1}{\sqrt{2}}$ jika $(\varepsilon) = 0$

2.4 Gray Level Co-occurrence Matrix

Gray Level Co-occurrence Matrix atau GLCM adalah metode bekerja dengan mengekstraksi fitur sudut dan karakteristik *pixel* untuk memperoleh informasi tentang kecerahan piksel pada posisi tertentu [24]. Biasanya, GLCM dipakai untuk mengidentifikasi tekstur dalam gambar. Output dari perhitungan GLCM adalah matriks yang menggambarkan relasi spasial antara *pixel* dalam

citra [23]. Proses metode GLCM dimulai dengan membentuk matriks awal GLCM dari pasangan dua piksel yang berjajar pada arah 0°, 45°, 90°, dan 135°. Kemudian, dibuat matriks simetris dengan menjumlahkan matriks awal GLCM dengan transposnya. Selanjutnya, matriks GLCM dinormalisasi dengan membagi setiap elemen matriks oleh jumlah pasangan piksel. Terakhir, fitur dari GLCM digunakan untuk ekstraksi ciri [15].

Dalam GLCM terdapat beberapa fitur yang dapat dihitung yang disebut fitur haralick. Fitur haralick yang digunakan untuk menganalisis tekstur meliputi, *Contrast*, *Corellation*, *Energy*, dan *Homogeneity* [25] [26].

1. Contrast

$$\sum_{i=0}^{G-1}\sum_{j=0}^{G-1}(i-j)^2 P(i,j) \tag{2.2}$$

2. Energy

$$\sum_{i=0}^{G-1}\sum_{j=0}^{G-1}(P(i,j))^2 \tag{2.3}$$

3. Homogeneity

$$\sum_{i=0}^{G-1}\sum_{j=0}^{G-1}\frac{P(i,j)}{1+|i-j|} \tag{2.4}$$

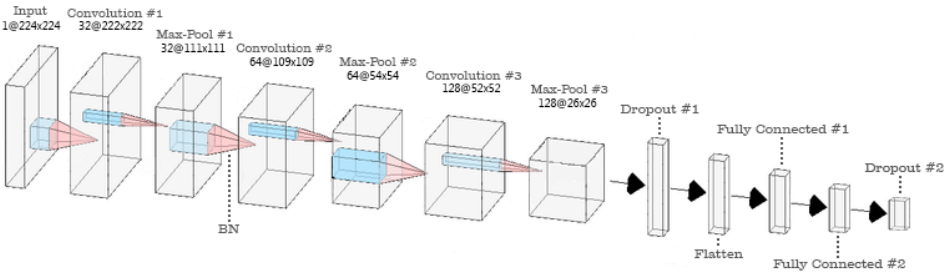
4. Correlation

$$\sum_{i=0}^{G-1}\sum_{j=0}^{G-1}\frac{(i-\mu_x)(j-\mu_y)P(i,j)}{\sigma_x\sigma_y} \tag{2.5}$$

Keterangan:
G: Ukuran citra
i: *Array* indeks ke i
j: *Array* indeks ke j
P(i,j): Nilai *pixel* pada indeks ke-(i,j)

2.5 Convolutional Neural Network

CNN, algoritma klasifikasi yang diadopsi secara luas dalam bidang DL, terdiri dari arsitek lapisan perceptron yang terhubung secara berurutan [27]. Lapisan yang menyaring data gambar input dengan rantai lapisan yang terhubung dan memprediksi output dengan mengubahnya menjadi representasi yang bermakna. Biasanya, arsitektur CNN memiliki tiga jenis lapisan utama; *convolutional layer*, *pooling layer* dan *fully connected layer* [28].



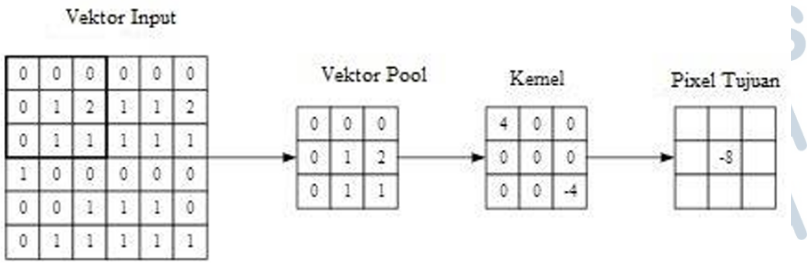
Gambar 2.2. Arsitektur CNN

2.5.1 Convolutional Layer

Convolution layer adalah lapisan pertama yang mengekstraksi ciri dari citra yang dimasukkan [29]. *Convolutional layer* memiliki sebuah filter *convolutional* bernamakan kernel yang berfungsi untuk mendapatkan sebuah *feature map* dengan cara memecah gambar ke bagian yang lebih kecil [30]. *Convolutional layer* dapat ditunjukan dengan rumus berikut [31].

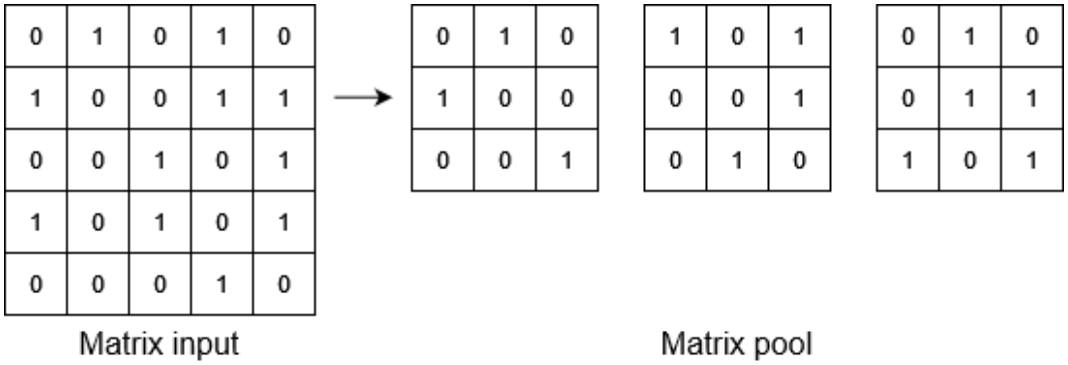
$$n_{(w,h)} = \left[\frac{n_{in} + 2p - k}{s}\right] + 1 \tag{2.6}$$

Keterangan:
$n_{(w,h)}$ = Hasil *input* ukuran citra
k = Ukuran *kernel* yang digunakan
s = Ukuran *stride*
p = Ukuran *padding*
n_{in}= Nilai ukuran citra *input*



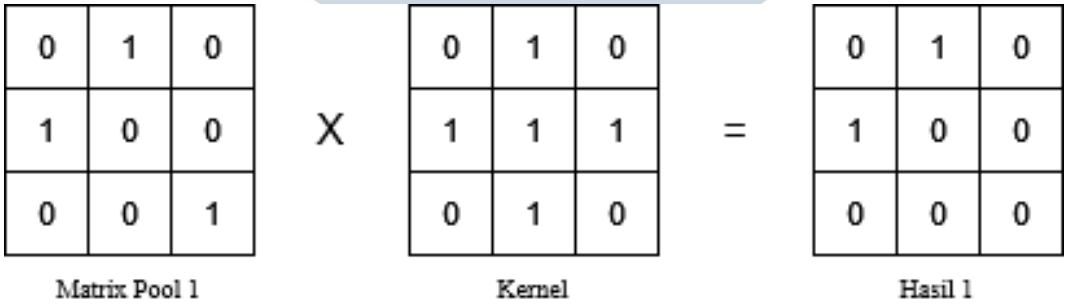
Gambar 2.3. *Convolution layer*
Sumber: [6]

Berikut adalah contoh sederhana dari proses konvolusi CNN.



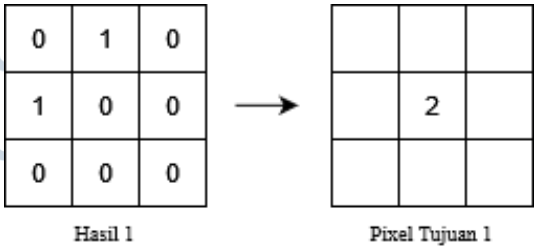
Gambar 2.4. Pembagian *matrix pool*

Pertama-tama, *matrix input* akan dibagi sebesar 3x3 *pixel*, sehingga akan menghasilkan *matrix pool*. Pembagian *matrix pool* akan dilakukan dari kiri atas citra sampai kanan bawah citra. Pada contoh di atas, akan menghasilkan 9 *matrix pool* dengan besar 3x3 *pixel*.



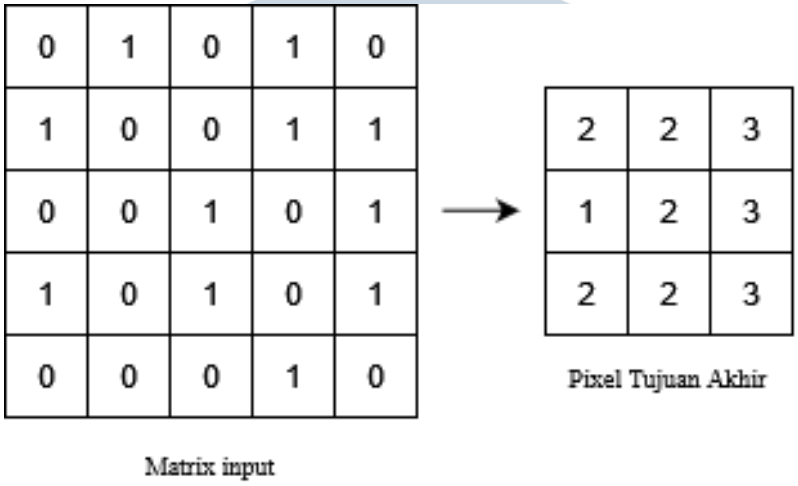
Gambar 2.5. Perkalian *matrix pool* terhadap kernel

Setiap *matrix pool* kemudian akan dikalikan dengan kernel, seperti pada contoh diatas. Hasil perkalian *matrix pool* dengan kernel ini akan menghasilkan 9 *matrix pool* yang baru.



Gambar 2.6. *Pixel* tujuan dari *matrix pool* baru

Setelah memperoleh hasil *matrix pool* yang baru, setiap angka dalam *matrix pool* tersebut akan dijumlahkan, menghasilkan satu nilai yang diinginkan.



Gambar 2.7. *Pixel* tujuan dari *matrix input*

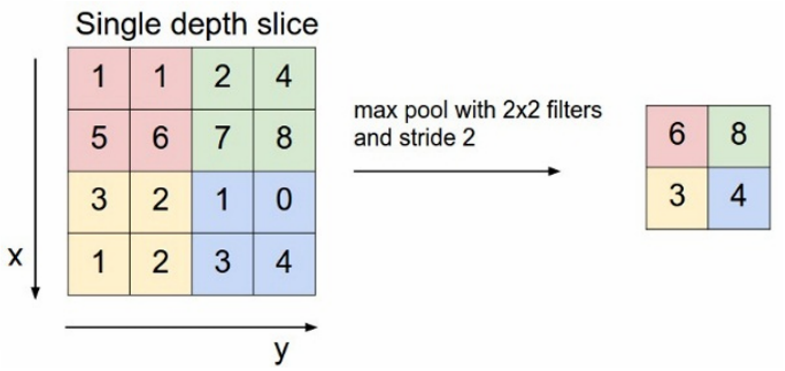
Setelah mendapatkan satu nilai *pixel* dari setiap *matrix pool*, nilai-nilai tersebut akan dikombinasikan menjadi sebuah *matrix* baru.

2.5.2 Pooling Layer

Pooling layer adalah struktur yang memanfaatkan fungsi dengan *feature map* sebagai inputnya, dan kemudian memprosesnya melalui beragam operasi statistik yang berdasarkan nilai piksel terdekat [8]. Terdapat dua tipe *pooling* yang sering dipakai, yakni *max pooling* yang mengambil nilai tertinggi dan *average pooling* yang mengambil nilai rata-rata [6] [31].

$$n_{(w,h)} = \frac{\left(n_{(w,h)-1} - f\right)}{s} + 1 \tag{2.7}$$

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

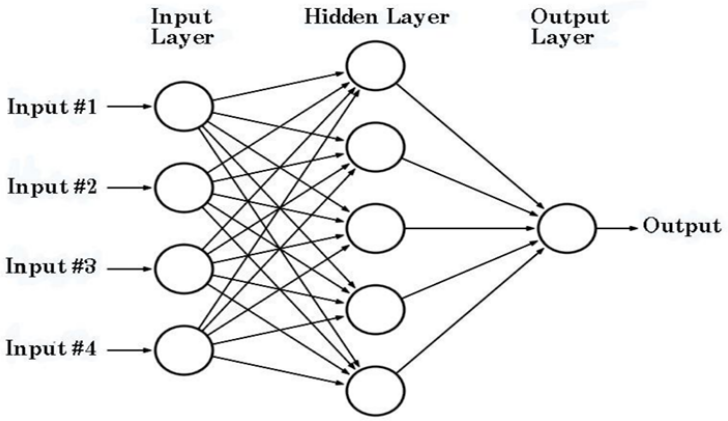


Gambar 2.8. *Max pooling*

Sumber: [6]

2.5.3 Fully Connected Layer

Fully connected layer adalah tahap di mana semua data akan dikonversi menjadi satu dimensi. Proses ini disebut *flattening*, yaitu mengubah dimensi data menjadi satu [32]. *Fully-Conected Layer* merupakan suatu struktur di mana semua *neuron* aktif dari lapisan sebelumnya terkoneksi dengan semua *neuron* pada lapisan berikutnya [33].



Gambar 2.9. *Fully connected layer*

Sumber: [34]