# BAB II
# LANDASAN TEORI

## 2.1 Tuberkulosis

Tuberkulosis adalah penyakit infeksi kronik yang disebabkan oleh kuman *Mycobacterium tuberculosis* dan dapat ditularkan melalui udara. Penyakit yang juga dikenal dengan sebutan ‘flek paru’ ini seringkali membuat miskonsepsi tersendiri di masyarakat, bahwa penyakit TBC hanya menyerang paru. Padahal, infeksi TB dapat menyerang berbagai organ, sehingga pada dunia medis dikenal juga istilah ‘TB ekstra paru’ yaitu ketika infeksi TB ditemukan pada organ selain paru seperti kulit, kelenjar getah bening, usus, liver, selaput otak, hingga sumsum tulang belakang.

Secara umum, gejala utama atau sistemik pada pasien TB meliputi demam yang tidak terlalu tinggi, penurunan berat badan, keringat pada malam hari, dan dapat disertai juga dengan nafsu makan yang menurun. Untuk TB pada paru sebagai organ yang paling sering terkena, ditemukan juga gejala berupa batuk berkepanjangan yang bisa juga sampai batuk darah, nyeri dada, serta sesak napas. Jika TB mengenai usus, ditemukan gejala pada saluran cerna seperti diare

berkepanjangan dan nyeri perut. Sedangkan, pada pasien dengan infeksi TB pada kelenjar getah bening dapat ditemukan benjolan di sekitar leher (Syam, 2019).

## 2.2 Grayscale Image

*Grayscale* merupakan gambar yang disetiap nilai pikselnya adalah sebuah sampel, yang merupakan informasi intensitas. *Grayscale* secara eksklusif terdiri dari nuansa abu-abu yang bervariasi nilai intensitasnya dari hitam yang memiliki intensitas terendah hingga putih sebagai intensitas tertinggi (Babu, dkk., 2016). Untuk menkonversikan gambar menjadi grayscale dapat digambarkan dengan rumus sebagai berikut (Babu, dkk, 2016):

$$Gray = 0.56 * G + 0.33 * R + 0.11 * B \quad \ldots(2.1)$$

$G$ merupakan nilai dari komponen warna hijau, $R$ merupakan nilai dari komponen warna hijau, $B$ merupakan nilai dari komponen warna biru, 0.56 merupakan bobot dar i warna hijau, 0.33 merupakan bobot dari warna merah, dan 0.11 merupakan bobot dari warna biru.

## 2.3 Gaussian Filter

Gaussian filter merupakan salah satu proses *image blurring* untuk mengurangi *noise* dari suatu gambar dengan metode *convulation.* Hasil dari metode ini merupakan angka dengan tipe data float. Untuk melakukan Gaussian

filter diperlukan array ($m,n$) (Shipitko dan Grigoryev, 2018). Berikut merupakan *mask* dari Gaussian filter (Shipitko dan Grigoryev , 2018):

$$K = \frac{1}{16}\begin{matrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{matrix} \quad \ldots(2.2)$$

## 2.4 Image Thresholding

*Image Thresholding* merupakan proses yang dilakukan pada suatu gambar yang sudah diubah menjadi *grayscale* untuk memisahkan suatu obyek dengan *background* gambar. Gambar yang dikonversikan merupakan gambar dengan setiap piksel yang bernilai dari 0 hingga 255. Hasil dari *image thresholding* merupakan gambar dalam bentuk bilangan biner. Salah satu metode yang paling sering digunakan merupakan metode *otsu thresholding* (Sankur dan Sezgin, 2004).

*Otsu thresholding* memperkecil bobot total dari varian kelas dan varian dari *pixel background* dan *foreground* gambar untuk mendaparkan *thresholding* yang optimal. Metode ini memberikan hasil yang cukup baik jika piksel di setiap kelas mirip satu dengan yang lainnya (Sankur dan Sezgin, 2004).

Menurut Yousefi (2015), secara matematis *otsu thresholding* dapat di gambarkan sebagai berikut :

$$P(i) = \frac{number\{(r,c)|image(r,c)=i\}}{(R,C)} \quad \ldots(2.3)$$

$$\sigma^2_W = w_b(t) * \sigma^2_b(t) + w_f(t) * \sigma^2_f(t) \quad \ldots(2.4)$$

$$W_b(t) = \sum_{i=1}^{t} P(i) \quad \ldots(2.5)$$

$$W_f(t) = \sum_{i=t+1}^{l} P(i) \quad \ldots(2.6)$$

$$\mu_b(t) = \frac{\sum_{i=1}^{t} i*P(i)}{W_b(t)} \quad \ldots(2.7)$$

$$\mu_f(t) = \frac{\sum_{t+1}^{l} u*P(i)}{W_f(t)} \quad \ldots(2.8)$$

$$\sigma^2_b(t) = \frac{\sum_{i=1}^{t}(i-\mu_b(t))^2*P(i)}{W_b(t)} \quad \ldots(2.9)$$

$$\sigma^2_f(t) = \frac{\sum_{i=t+1}^{l}(i-\mu_f(t))^2*P(i)}{W_f(t)} \quad \ldots(2.10)$$

Pada rumus (2.3), $r$ dan $c$ merupakan indeks untuk baris dan kolom dari sebuah gambar, sedangkan $R$ dan $C$ merupakan jumlah dari baris dan kolom gambar. Pada rumus (2.5), $W_b(t)$ merupakan bobot dari *T0*, $\sigma^2_b(t)$ merupakan varian dari kelas *T0*, $W_f(t)$ merupakan bobot dari *T1*, $\sigma^2_f(t)$ merupakan varian dari kelas T1, dan $\sigma^2_W$ merupakan penjumlah dari varian ke dua kelas. Pada rumus (2.7), $\mu_b(t)$ merupakan mean dari kelas T0 dan pada rumus (2.8), $\mu_f(t)$ merupakan mean dari kelas *T1*.

## 2.5 Pembelajaran Mesin

Pembelajaran mesin, menurut definisinya adalah bidang komputer sains yang berkembang dari mempelajari pengenalan pola dan teori pembelajaran komputasi dalam kecerdasan buatan. Pembelajaran mesin merupakan pembelajaran

dan pembuatan algoritma yang bisa belajar dari dataset dan dapat membuat prediksi dari dataset tersebut. Prosedur ini dilakukan dengan memberikan contoh data yang dimasukan untuk membuat prediksi yang berbasis *data-driven* daripada mengikuti instruksi program statis (Babu, dkk., 2016).

Menurut Simeone (2018), teknik mesin learning terbagi menjadi 3 kelas, yaitu:

### 2.5.1 Supervised Learning

Dalam *supervised learning*, pelatihan data yang dilakukan terdiri dari pasangan *input* serta *output* yang diberikan target yang ingin dicapai, tujuan dari *supervised learning* merupakan pembelajaran tentang pemetaan jarak antara *input* dan *output*

### 2.5.2 Unsupervised Learning

Dalam *unsupervised learning*, pelatihan data terdiri dari *input* yang tidak diberikan label serta output yang tidak diberikan target yang ingin dicapai. Tujuan dari *unsupervised learning* merupakan menemukan atribut dari suatu mekanisme yang menghasilkan data tersebut.

### 2.5.3 Reinforcement Learning

*Reinforcement learning* berada diantara *supervised learning* dan *unsupervised learning.* Tidak seperti unsupervised learning, ada pengawasan yang dilakukan terhadap data yang dilatih, tetapi pengawasan ini tidak bertujuan untuk mengontrol *output* yang dikeluarkan oleh pembelajaran ini. Pengawasan yang dilakukan lebih kepada lingkungan sekitar yang memberikan masukan untuk output, sehingga output yang dihasilkan akan dipengaruhi besar dari observasi yang dilakukan di lingkungan tersebut.

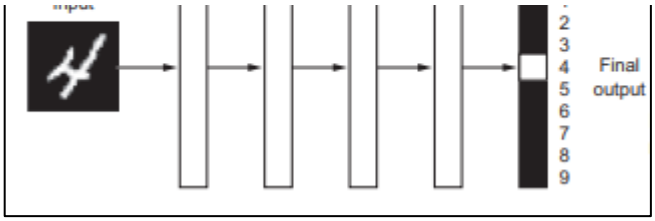
## 2.6 Deep Learning

*Deep learning* adalah subbidang khusus dari pembelajaran mesin yang menggunakan pandangan baru tentang representasi pembelajaran dari data yang menekankan pada pembelajaran data yang semakin bermakna dengan bertambahnya lapisan yang berturut-turut. *Deep* dari *deep learning* merupakan lapisan data representatif yang berturut-turut. Banyaknya lapisan yang ada pada suatu model data pada Deep learning disebut sebagai *depth*. *Deep learning* modern biasanya terdiri dari puluhan atau bahkan ratusan lapisan yang secara otomatis dan

langsung belajar dari paparan antara tiap lapisan tersebut terhadap data *training* (Chollet, 2018).

Pada *deep learning*, lapisan ini biasanya belajar dari model yang disebut *neural network.* Neural network ini terstruktur yang secara harafiah dapat dikatakan ditumpuk diatas satu sama lain. Istilah jaringan saraf adalah referensi ke neurobiologi, tetapi meskipun beberapa inti konsep dalam *deep learning* dikembangkan sebagian dengan inspirasi dari memahami otak, akan tetapi model *deep learning* bukanlah model dari otak. Tidak ada bukti bahwa otak manusia mengimplementasikan sesuatu seperti mekanisme pembelajaran yang digunakan dalam model *deep learning* modern karena deep learning adalah kerangka kerja matematika untuk mempelajari representasi dari suatu data (Chollet, 2018).

Berikut adalah gambaran beberapa lapisan *network* mengubah gambar menjadi angka

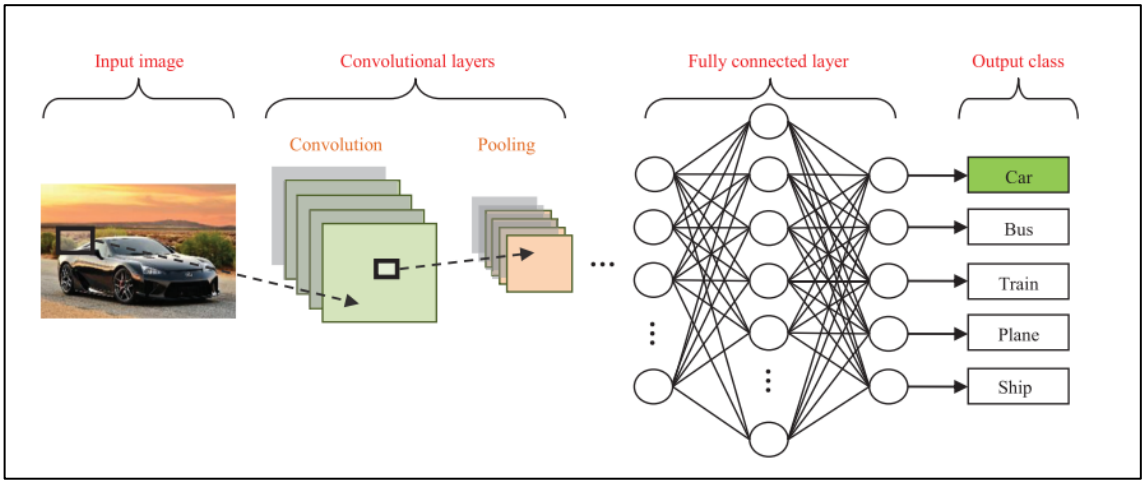



Gambar 2.1: Neural Network Untuk Klasifikasi Digit (Chollet, 2018)

## 2.7 Convolutional Neural Network

*Convolutional Neural Network* (CNN) merupakan operasi konvolusi

yang menggabungkan beberapa lapisan pemrosesan, menggunakan beberapa elemen yang beroperasi secara paralel dan terinspirasi oleh sistem saraf biologis (Hu, dkk., 2015). Pada CNN setiap neuron dipresentasikan dalam bentuk 2 dimensi, sehingga metode ini cocok untuk pemrosesan dengan input berupa citra (Maggiori, dkk., 2016). Arsitektur jaringan dengan menggunakan CNN ditunjukkan pada Gambar 1. Struktur CNN terdiri dari input, proses ekstraksi fitur, proses klasifikasi dan output. Proses ekstraksi dalam CNN terdiri dari beberapa lapisan tersembunyi atau *hidden layer*, yaitu lapisan konvolusi, fungsi aktifasi *Rectification Linear Unit* (ReLU), dan *pooling*. CNN bekerja secara hierarki, sehingga output pada lapisan konvolusi pertama digunakan sebagai input pada lapisan konvolusi selanjutnya. Pada proses klasifikasi terdiri dari *fully-connected layer* yang outputnya berupa hasil klasifikasi (Katole, dkk., 2015).




Gambar 2: Proses Klasifikasi Gambar CNN (Rawat, 2017).

### 2.7.1 Convolutional Layers

*Convolutional layers* berfungsi sebagai fitur *extractor*, dan dengan demikian mereka mempelajari fitur representasi dari input gambar yang telah di berikan. Neuron pada lapisan konvolusional disusun menjadi feature map. Setiap neuron dalam *feature map* memiliki bidang reseptif, yang terhubung ke neighborhood neuron pada lapisan sebelumnya melalui serangkaian bobot yang dapat dilatih, kadang-kadang disebut sebagai filter bank (LeCun, dkk., 2015). *Input* dililit dengan bobot yang dipelajari untuk menghitung *feature map* baru dan hasil yang berbelit-belit tersebut dikirim melalui fungsi aktivasi nonlinier. Semua neuron dalam *feature map* memiliki bobot yang dibatasi menjadi sama namun, *feature map* yang berbeda dalam lapisan konvolusional yang sama memiliki bobot yang berbeda sehingga beberapa fitur dapat diekstraksi di setiap lokasi (LeCun, dkk., 1998; LeCun, dkk., 2015). Untuk mencari *zero padding* dapat dilakukan dengan perhitungan matematis sebagai berikut:

$$Zero\ Padding\ = \frac{(K-1)}{2} \qquad \ldots(2.11)$$

*K* merupakan ukuran dari *filter* yang digunakan, sehingga output dari *convolutional neural network* memiliki dimensi yang sama. Untuk mencari dimensi output yang dihasilkan dapat digunakan perhitungan sebagai berikut:

$$O = \frac{(W-K+2P)}{S} + 1 \quad \ldots \quad (2.12)$$

$O$ merupakan panjang dari *output* matriks yang dihasilkan dari *convolutional layer*, $W$ merupakan panjang dari input matriks, $K$ merupakan ukuran filter, $P$ merupakan padding, dan $S$ merupakan *stride* yang merupakan pergerakan matriks *kernel* pada matriks *input.*

### 2.7.2 Pooling Layers.

Tujuan dari *pooling layers* adalah untuk mengurangi resolusi spasial dari *feature map* dan dengan demikian mencapai invarian spasial untuk dapat memasukan distorsi dan terjemahan (LeCun, dkk., 1989a, 1989b LeCun, dkk., 1998, 2015; Ranzato, dkk., 2007). Perhitungan untuk mendapatkan matrik dari max pooling dapat dilakukan dengan perhitungan matematis sebagai berikut:

$$fx, y(s) = max\, S_{ix+a, iy+b} \quad \ldots(2.13)$$

$x$ dan $y$ merupakan posisi matrik *pooling* saat ini, lalu $S$ merupakan *stride* yang dilakukan oleh *matrix pooling*, $i$ merupakan seberapa banyak *stride* yang akan di lakukan. $a$ dan $b$ merupakan angka yang berfungsi untuk menambahkan stride yang di lakukan pada $x$ dan $y$.

### 2.7.3 Fully Connected Layers

Lapisan fully connected layer merupakan kumpulan dari proses konvolusi (Hijazi, dkk., 2015). Lapisan ini mendapatkan input dari proses sebelumnya untuk menentukan fitur mana yang paling berkorelasi dengan kelas tertentu. Fungsi dari lapisan ini adalah untuk menyatukan semua node menjadi satu dimensi (Albewi & Mahmood, 2017).

### 2.7.4 Reactification Linear Unit Layer

*Rectification Linear Unit* (ReLU) merupakan operasi untuk mengenalkan nonlinearitas dan meningkatkan representasi dari model (Heaton, 2015). Nilai *output* dari neuron bisa dinyatakan sebagai 0 jika *input*-nya adalah negatif. Jika nilai input adalah positif, maka output dari neuron adalah nilai input aktivasi itu sendiri (Kim, dkk., 2016). Perhitungan untuk mengubah nilai *input* gambar yang negatif menjadi 0 dapat direpresentasikan dengan fungsi sebagai berikut:

$$f(x) = \max(0, x) \qquad ...(2.14)$$

$x$ merupakan data biner pixel suatu gambar yang telah di ubah jadi *matrix* jika nilai x merupakan nilai negatif maka nilai $x$ akan diganti dengan nilai 0 dan jika tidak negatif maka nilai x tidak akan di ganti dengan nilai 0.

### 2.7.5 Dropout

*Dropout* merupakan kemampuan untuk membuang suatu hasil layer sebelumnya secara acak dengan tujuan untuk menghindari terjadinya model yang terlalu teradaptasi pada *dataset* yang ada (*overfitting*) pada model yang di buat (Srivastava, dkk, 2014).

### 2.7.6 Flatten

*Flatten* merupakan pengubahan hasil matriks dari convolutional layer dan max pooling menjadi array 1 dimensi atau vektor yang terjadi di *fully connected layer* (Yamashita, dkk, 2015).

### 2.7.7 Dense

*Dense* merupakan *layer* yang berisi lebih dari satu *fully connected layer* yang *inputnya* tersambung dengan *outputnya* karena bobot yang dapat di pelajari(Yamashita, dkk, 2015).

### 2.7.8 Deep Convolutional Neural Network

DCNN merupakan convolutional neural network yang menggunakan lebih dari 2 hidden layer yang dapat meningkatkan akurasi dari sebuah prediksi terutama dalam pengklasifikasian gambar. (Albewi, 2017)

## 2.8 Perbedaan Neural Network Dengan CNN

Neural Network pada kecerdasan buatan merupakan sistem koneksionis yang kuat yang secara samar-samar terinspirasi oleh jaringan saraf biologis (Chen, dkk, 2019). Perbedaan yang signifikan antaran NN dan CNN merupakan pada CNN hanya *layer* terakhir dari sebuah model CNN yang sepenuhnya terhubung antara satu dengan yang lain, sedangkan pada NN setiap *neuron* terhubung sepenuhnya antara satu dengan yang lain dari awal hingga akhir (Gogul dan Kumar, 2017).

## 2.9 Confusion Matrix

Confusion Matrix terdiri dari TP, TN, FP, dan FN yang menggambarkan hasil pengklasifikasian yang dilakukan model yang dibuat. Secara keseluruhan confusion matrix dapat dilihat pada tabel 2.1:

Tabel 2.1: Confusion Matrix (Prasetyo, 2014).

| Data Aktual / Data Prediksi | *Positive* | *Negative* |
|---|---|---|
| *Positive* | *True Positive* | *False Positive* |
| *Negative* | *False Negative* | *True Negative* |

Sensitifitas berhubungan dengan kemampuan pengujian untuk mengidentifikasi hasil yang positif dari sejumlah data yang sebenarnya positif (Prasetyo, 2014). Persamaan yang digunakan untuk menghitungnya disajikan sebagai berikut:

$$Sensitifitas = \frac{TP}{TP+FN} \quad \text{…(2.15)}$$

Menurut Sokolova (2006), persamaan yang dapat digunakan untuk mencari *accuracy*, *precission*, dan *f1-score* adalah sebagai berikut:

$$Precision = \frac{TP}{TP+FP} \quad \text{...(2.16)}$$

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad \text{…(2.17)}$$

$$F1 - Score = 2 * \frac{precision*recall}{precision+recall} \quad \text{…(2.18)}$$

Menurut Prasetyo (2014), pengertian label data pada *confusion matrix* adalah sebagai berikut:

*TP* (*True Positive*) = data yang teridentifikasi secara benar.

*FP*(*False Positive)* = data yang teridentifikasi secara salah.

*TN*(*True Negative)* = data yang tertolak secara benar.

*FN(False Negative)* = data yang tertolak secara salah