BAB II
DASAR TEORI

2.1 Citra Digital

Citra digital adalah salah satu format gambar yang paling praktis digunakan dalam hal mentransfer data dan pengoperasian gambar. Walaupun citra menampilkan banyak informasi, tetapi citra juga bisa terjadi penurunan nilai ketajamannya (intensitas), contohnya dari artefak atau *noise*. Tingkat intensitas citra bisa terlalu kontras atau buram. Penurunan nilai ketajaman citra dapat dialami saat mengambil atau menyimpan gambar (gambar digital). Hal ini yang membuat berkurangnya jumlah informasi yang akan diperoleh pada citra. Agar informasi yang dihasilkan oleh citra memuat banyak informasi, maka diperlukan suatu cara peningkatan gambar digital (Saselah *et al.*, 2013).

Gambar digital terdiri atas piksel-piksel yang berbentuk matriks dua dimensi. Persamaan (2.1) menunjukkan citra digital yang berisikan baris (M), kolom (N), dan nilai intensitas (f). Titik (x, y) pada persamaan (2.1) memiliki makna berupa posisi piksel pada matriks sedangkan nilai $f(x, y)$ memiliki arti nilai intensitas yang berisikan keterangan posisi baris dan kolom di masing-masing koordinat x dan y (Wardana *et al.*, 2021).

$$f(x,y) = \begin{bmatrix} f(1,1) & f(1,2) & \cdots & f(1,N) \\ f(2,1) & f(2,2) & \cdots & f(2,N) \\ \cdots & \cdots & \ddots & \vdots \\ f(M,1) & f(x,y) & & f(M,N) \end{bmatrix} \quad (2.1)$$

Citra digital dibagi menjadi beberapa jenis berdasarkan nilai pikselnya, yaitu *grayscale*, RGB, dan biner. Citra RGB ialah gambar digital yang mempunyai intensitas nilainya terdiri atas 3 kanal warna, yaitu *red*, *green* dan *blue*. Citra *grayscale* adalah potret digital yang tersusun atas satu nilai saja pada tiap pikselnya. Artinya, angka R=G=B. Warna yang dimiliki citra *grayscale* ialah warna hitam, keabu-abuan dan putih. Nilai *grayscale* dimulai dari angka 0-256. Citra biner merupakan citra digital yang terdiri atas piksel yang berwarna monokrom (hitam atau putih).

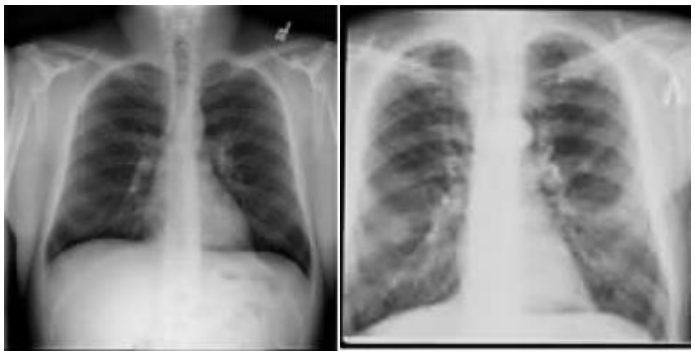
2.1.1 Citra Rontgen

Citra rontgen adalah citra yang disebabkan oleh sinar-X yang melewati objek dan mengenai film, mengubah susunan kristal perak *halide* menjadi butiran perak hitam (Irmayani, 2019). Gambaran hasil citra rontgen dapat dilihat pada Gambar 2.1 yang menunjukkan citra rontgen paru-paru dalam kondisi normal atau sehat.

Gambar 2.1 Citra rontgen normal (Narin *et al.*, 2021)

2.2 Tuberkulosis

Paru-paru adalah organ pernapasan pada manusia yang berperan untuk melakukan respirasi atau tempat bertukarnya oksigen dan karbon dioksida (Rahmadewi, 2016). Paru-paru dapat terganggu fungsinya jika tubuh terjangkit penyakit pernapasan, yaitu tuberkulosis dan Corona Virus Disease 2019 (COVID-19). Tuberkulosis ialah penyakit yang dapat terjangkit oleh manusia akibat dari infeksi *mycobacterium tuberculosis complex*. Perbedaan hasil rontgen antara paru-paru manusia normal dan paru-paru yang terinfeksi tuberkulosis dapat dilihat pada Gambar 2.2.



Gambar 2.2 Paru-paru normal dan tuberkulosis (Narin *et al.*, 2021, Moch Diar *et al.*, 2022)

Tuberkulosis merupakan salah satu penyakit yang dapat menyebabkan kematian pada manusia. Umumnya, tuberkulosis menyerang organ paru-paru. Tetapi, tuberkulosis juga dapat menyerang darah, tulang, dan otak. Menurut WHO dalam *Global Tuberculosis Report* (GTR), distribusi kasus tuberkulosis di tahun 2016 menunjukkan bahwa Indonesia menjadi salah satu negara di Asia Tenggara yang memiliki jumlah penduduk terjangkit tuberkulosis cukup tinggi sebesar 45%. GTR juga menunjukkan bahwa angka kematian akibat tuberkulosis dominan terjadi di negara berkembang dan negara miskin (Agussationo *et al.*, 2018).

2.3 Corona Virus Disease-19 (COVID-19)

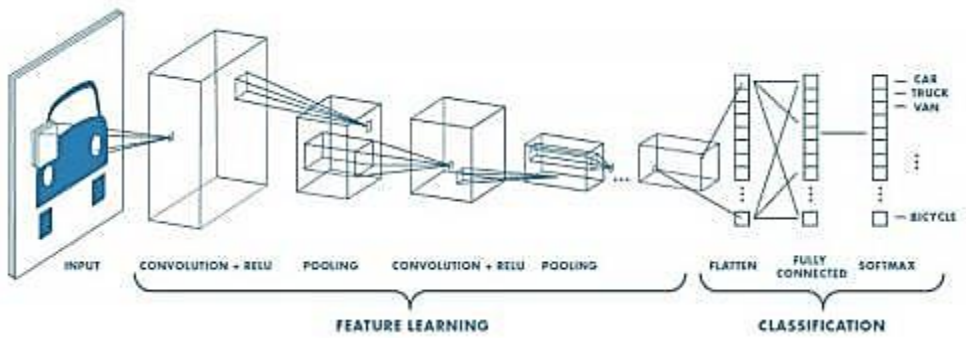
COVID-19 merupakan penyakit yang dapat menyerang organ pernapasan paru-paru. COVID-19 adalah penyakit yang diakibatkan oleh virus yang bernama corona. Gejala penyakit ini terlihat seperti flu biasa, sindrom pernapasan akut parah (SARS), dan sindrom pernapasan Timur Tengah (MERS). COVID-19 ditemukan pertama kali pada tahun 2019 di Wuhan China (Boldog *et al.*, 2020). COVID-19 ditetapkan menjadi pandemi global oleh *World Health Organization* (WHO) pada akhir desember 2019 (Utami *et al.*, 2020). Artinya, COVID-19 telah tersebar hampir di seluruh negara dan sudah menjadi ancaman dunia. Gambaran seseorang terpapar COVID-19 ataupun tidak terdapat pada Gambar 2.3.

Gambar 2.3 Paru-paru normal dan COVID-19 (Narin *et al.*, 2021)

2.4 *Convolutional Neural Network* (CNN)

CNN merupakan sistem pengembangan dari *Neural Network* (NN) yang memiliki sejumlah lapisan perceptron yang bisa difungsikan pada identifikasi atau

klasifikasi objek dua dimensi (Tobias *et al.*, 2016). Adapun proses CNN dalam mengenali objek dapat dilihat pada Gambar 2.4.



Gambar 2.4 Pemodelan CNN (Faizin *et al.*, 2022)

Gambar 2.4 menunjukkan proses CNN yang diawali dengan mengatur nilai piksel dari citra masukan. Langkah kedua, akan dilakukan *convolutional layer* berupa mengatur nilai kernel yang akan digunakan, lalu mengaktifkan fungsi aktivasi berupa *Rectified Linear Unit* (ReLU). Langkah ketiga, melakukan *pooling layer*. Langkah pertama hingga ketiga diulangi hingga mendapatkan hasil fitur yang layak sehingga dapat melangkah ke tahap *fully connected layer*. Kemudian, mengaktifkan fungsi *softmax* dengan tujuan menghasilkan nilai akurasi pada masing-masing kelas. Pembagian kelas pada data tergantung dari nilai akurasi. Pemodelan CNN secara umum terdiri atas 2 bagian, yaitu *feature learning* dan klasifikasi (Fattah, 2021).

2.4.1 *Feature Learning*

Feature learning terdiri atas lapisan konvolusi, *pooling* dan ReLU.

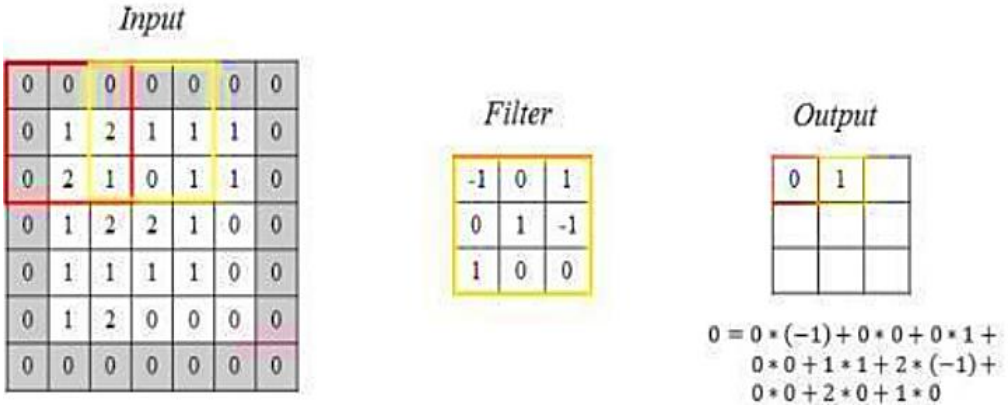
a) Lapisan konvolusi

Lapisan konvolusi digunakan untuk mengubah fitur pada citra dengan melaksanakan proses konvolusi antara matriks *filter* dengan citra masukan. Tahap ini akan menghasilkan fitur dengan level lebih baik dari sebelumnya (*edge*, *curve* dan fitur warna) apabila menggunakan banyaknya matriks *filter* pada tiap tingkat lapisan yang berbeda (Hariyani *et al.*, 2020).

Filter merupakan matriks yang sudah dibagi-bagi sehingga memiliki bentuk lebih kecil dari citra masukan, dengan *filter* mempunyai ukuran m×m piksel. Banyaknya *filter* dan ukurannya akan disesuaikan dengan pemodelan yang

digunakan. *Filter* akan melakukan perkalian titik antara matriks masukan dengan matriks K *filter*, sehingga memperoleh keluaran berupa *feature map* atau komponen baru. Proses pengoperasian antara matriks masukan dan K *filter* akan bergeser satu per satu sesuai dengan nilai *stride* (Fattah, 2021). *Filter* perlu digunakan dalam tahapan konvolusi karena berisikan nilai bobot yang dapat memperbaiki kualitas gambar dan mendeteksi bagian-bagian gambar. Perhitungan pada tahapan konvolusi dapat dihitung dengan persamaan (2.2) dan Gambaran proses pemfilteran dalam CNN dapat dilihat pada Gambar 2.5 (Fattah, 2021). $C_{h,h}$ merupakan hasil konvolusi berdasarkan posisi baris dan kolom pada nilai *feature map*. W_x merupakan nilai bobot pada *filter* dan a_x ialah nilai piksel masukan.

$$C_{h,h} = f\left(\sum_{x=1}^{n} a_x \times w_x\right) \tag{2.2}$$



Gambar 2.5 Proses pemfilteran dengan CNN (Faizin *et al.*, 2022)

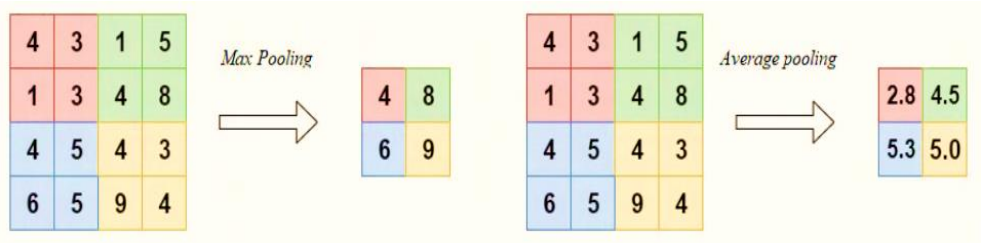
Stride (S) adalah proses bergesernya pengoperasian *filter* dengan citra masukan secara satu per satu. *Stride* akan bergeser baik secara vertikal maupun horizontal sesuai dengan nilai yang telah ditentukan (Fattah, 2021). Parameter lainnya yang ada pada lapisan konvolusi ialah *padding*.

Padding (P) adalah penambahan jumlah piksel di setiap sisi matriks masukan. Jumlah piksel yang dimaksudkan dalam *padding* adalah berupa nilai 0 yang diposisikan di tiap sisi tepi matriks masukan. *Padding* dilakukan agar menghasilkan dimensi yang sama antara dimensi masukan dari *feature map* dengan dimensi keluaran hasil konvolusi (Fattah, 2021). Jumlah *padding* pada lapisan konvolusi dapat disesuaikan dengan ukuran matriks pada citra masukan. Proses selanjutnya

ialah akan melewati lapisan *pooling* dan akan diaktivasi dengan sebuah fungsi aktivasi berupa ReLU (*Rectified Linear Unit*).

b) Lapisan *Pooling*

Pooling difungsikan agar mengurangi parameter dan komputasi yang tidak dapat memperkirakan hasil akurasi secara tepat. Fungsi lainnya dari *pooling* ialah bisa digunakan untuk mencegah keadaan *overfitting* pada pemodelan data latih yang memiliki nilai prediksi sangat tepat tetapi tidak dapat mengidentifikasi data selain data latih itu sendiri. Ilustrasi lapisan *pooling* terdapat pada Gambar 2.6. *Pooling* dibagi menjadi dua jenis, yaitu *average pooling* dan *max pooling*. *Average pooling* adalah jenis *pooling* dengan menggunakan nilai rata-rata. *Max pooling* adalah jenis *pooling* dengan menggunakan nilai maksimum (Hariyani *et al.*, 2020).



Gambar 2.6 Ilustrasi *pooling layer* (Fattah, 2021)

c) ReLU

Hasil konvolusi pada CNN dibutuhkan suatu fungsi aktivasi dan umumnya menggunakan ReLU (*Rectified Linear Unit*). Fungsi aktivasi berfungsi agar dapat mengubah piksel bernilai negatif pada citra dengan menerapkan persamaan (2.3). Nilai negatif piksel hasil konvolusi dilambangkan dengan x. Jika nilai piksel positif, maka nilai tersebut tidak perlu diubah. Jika nilai piksel negatif, maka nilai tersebut perlu diubah menjadi 0 (nol) (Fattah, 2021).

$$f(x) = \max(0, x) \tag{2.3}$$

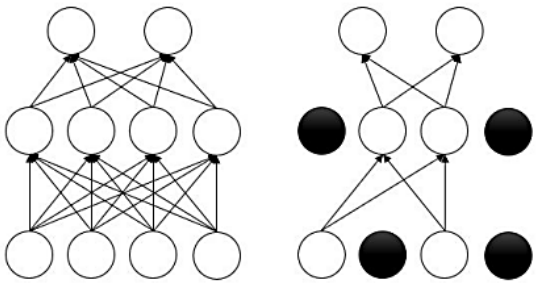
2.4.2 Klasifikasi

Klasifikasi terdiri atas lapisan *dropout*, *fully connected layer*, *softmax*, *loss function.*

a) Lapisan *Dropout*

Dropout juga merupakan lapisan yang berfungsi agar tidak terjadinya keadaan *overfitting* dan mempercepat proses pengenalan data. Selain itu, *dropout*

dilakukan untuk menghapus *neuron* yang tersembunyi ataupun yang tampak pada lapisan CNN. Penghapusan *neuron* tersebut hanya dilakukan untuk sementara dan dipilih secara *random*. Nilai probabilitas neuron ialah bernilai antara 0 dan 1. Penentuan nilai probabilitas ditentukan dari jumlah percobaan yang dilakukan. *Dropout* jika diterapkan pada lapisan CNN, maka akan mempengaruhi hasil pelatihan data dan mencegah terjadinya *overfitting* (Fattah, 2021). Tahapan *classification* juga dapat dilakukan dengan *dropout* ataupun tidak, seperti yang tertera pada Gambar 2.7.

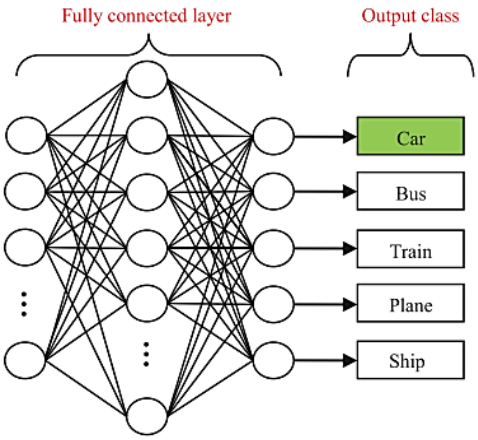


Gambar 2.7 Ilustrasi menggunakan *dropout* (kiri) dan tanpa *dropout* (kanan) (Fattah, 2021)

b) *Fully Connected Layer*

Fully connected layer adalah gabungan beberapa lapisan yang ada di CNN. *Fully connected layer* berisikan sejumlah lapisan *neuron* yang terdiri atas lapisan konvolusi, ReLU, dan lapisan *pooling* yang digabung menjadi satu untuk tahap klasifikasi. Penggabungan beberapa lapisan dapat dilakukan dengan bantuan *neural network* yang berbentuk *single layer perceptron* maupun *multi layer perceptron* dengan di dalamnya terdiri dari lapisan tersembunyi, fungsi aktivasi, dan lapisan keluaran. Perhitungan pada lapisan *fully connected* dapat dilakukan dengan menerapkan persamaan (2.4). *Fully connected layer* dilambangkan dengan z_n. W dan b merupakan nilai bobot dan nilai bias. Sedangkan, x merupakan nilai-nilai piksel yang dihasilkan pada proses *feature learning*. Gambar 2.8 menunjukkan lapisan *fully connected* dengan tujuan untuk mengubah dimensi agar data dapat terklasifikasi.

$$z_n = b_j + f \sum W_{i,j} x_i \tag{2.4}$$



Gambar 2.8 Ilustrasi *fully connected layer* (Rawat, W dan Wang, 2017)

c) *Softmax*

Hasil yang didapatkan dari *softmax (output)* adalah pembagian kategori di tiap kelas data. *Softmax* merupakan suatu fungsi *neural network* yang berada di lapisan terakhir dengan tujuan menghasilkan nilai keluaran klasifikasi. *Softmax* mengganti nilai yang sebenarnya pada K-dimensi vektor s menjadi bernilai vektor yang sama dengan nilai berkisar antara 0 hingga 1 dengan cara menerapkan persamaan (2.5) (Fattah, 2021):

$$s_j = \frac{e^{z_j}}{\sum_{n=1}^{K} e^{z_n}} \tag{2.5}$$

dengan s_j ialah nilai probabilitas dari z_j untuk kategori kelas K yang berbeda-beda.

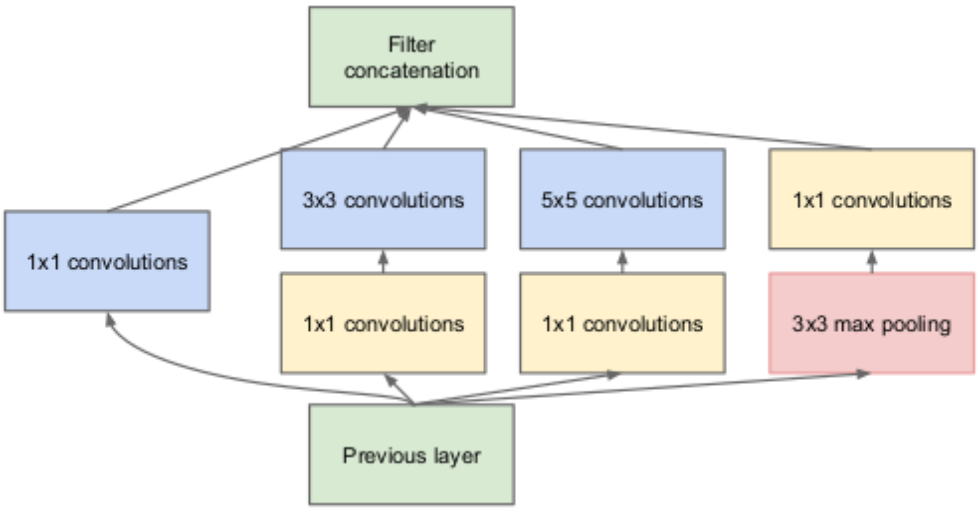
d) *Loss Function*

Loss function ditargetkan untuk mengukur tingkat keberhasilan pengenalan data dengan menghitung hasil keluaran nilai prediksi dengan nilai asli data. *Loss function* dapat dihitung melalui metode *cross entropy*. *Cross entropy* berfungsi untuk menghitung nilai simpangan hasil prediksi sj dan nilai asli yj dengan menggunakan persamaan (2.6) (Fattah, 2021):

$$E(s, y) = -\sum_{y_j} y_j \log s_j \tag{2.6}$$

2.5 GoogLeNet

GoogLeNet adalah salah satu jenis CNN yang diperkenalkan oleh Google di tahun 2014. GoogLeNet merupakan arsitektur yang memenangkan kompetisi ImageNet Large Scale Visual Recognition Challenge (ILSVRC) pada tahun 2014 (Szegedy *et al.*, 2015). Lapisan *input* berukuran 224×224×3 pada GoogLeNet. GoogLeNet memiliki keunggulan berupa *Inception Modules* yang dapat dilihat pada Gambar 2.9 yang menunjukkan bahwa GoogLeNet terdiri atas *convolution* kecil yang dibentuk untuk mengecilkan banyaknya parameter tanpa mengurangi proses kerja jaringan (Jahandad *et al.*, 2019). Kinerja *Inception Modules*, yaitu melakukan proses konvolusi sebanyak 6 lapis pada lapisan *output* sebelumnya, kemudian dilakukan ReLU sebanyak 6 lapis dan dilakukan satu lapis *pooling layer*. Ukuran lapisan konvolusi yang berukuran 1×1 terdiri dari 4 lapisan, lapisan konvolusi berukuran 3×3 terdiri atas 1 lapis, lapisan konvolusi berukuran 5×5 terdiri dari 1 lapis, dan *pooling layer* memiliki ukuran 3×3, serta penggunaan ReLU akan dilakukan setelah proses konvolusi selesai (Jahandad *et al.*, 2019). Jaringan GoogLeNet dirancang untuk efisiensi komputasi dan mempertimbangkan kepraktisan untuk memungkinkan inferensi dilakukan pada perangkat tunggal, termasuk perangkat dengan sumber daya komputasi terbatas, terutama perangkat dengan memori rendah. Arsitektur *inception* GoogLeNet dapat dilihat pada Gambar 2.10 (Szegedy *et al.*, 2015).



Gambar 2.9 *Inception block* (Jahandad *et al.*, 2019)

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Gambar 2.10 Arsitektur *inception* (Szegedy *et al.*, 2015)

2.6 *Transfer Learning*

Transfer learning merupakan teknik pembelajaran yang memanfaatkan pemodelan yang sudah ada tanpa perlu membuat pemodelan dari tahap awal. Konsep *transfer learning* ialah memindahkan fitur yang dilatih sebelumnya dan memodifikasi bagian ekstraksi fitur sesuai dengan data baru yang ingin dilatih. Keuntungan dari *transfer learning*, yaitu tidak perlu membuat model dari tahap *feature learning* hingga tahapan klasifikasi. Selain itu, hasil *fine tuning* yang dilakukan oleh *transfer learning* lebih optimal daripada penggunaan pemodelan yang dibuat dari tahap awal karena perlu memberikan nilai awal bobot *network*. Jumlah data yang digunakan pada *transfer learning* bisa lebih kecil dari pelatihan sebelumnya. Hal tersebut dikarenakan *transfer learning* telah mempelajari banyak data sehingga tidak masalah jika data latih terbaru memiliki jumlah lebih sedikit dari pelatihan sebelumnya (Wonohadidjojo, 2021).

2.7 *Hyperparameter*

Hyperparameter adalah jenis parameter yang sifatnya manual. Artinya, penggunaan berbagai parameter yang digunakan dalam pemodelan diatur sesuai dengan data dan model yang digunakan. *Hyperparameter* merupakan parameter yang berfungsi untuk meningkatkan proses performa pelatihan sehingga perlu dilakukan untuk menghasilkan perkiraan yang tepat (Aristyanto dan Kurniawan,

2021). *Hyperparameter* yang digunakan pada penelitian ini terdiri atas *batch size*, optimizer, *learning rate*, dan *epoch*.

2.7.1 *Batch size*

Batch size berfungsi untuk membagi jumlah dataset menjadi beberapa bagian sehingga memudahkan sistem yang dibangun dalam mengenali data pada tahap pelatihan data (Fattah, 2021).

2.7.2 Optimizer

Optimizer adalah salah satu bentuk *hyperparameter* yang berfungsi untuk membantu pemodelan mengatasi nilai *loss* dalam mengenali data pelatihan (Aristyanto dan Kurniawan, 2021). Penggunaan optimizer pada penelitian ini ialah menggunakan jenis *Adaptive Moment Estimation* (Adam). Adam merupakan gabungan dari dua optimizer, yaitu *Root Mean Square Propagation* (RMSProp) dan *Adaptive Gradient* (AdaGrad) sehingga penggunaannya lebih efektif di dalam komputasi. Adam digunakan pada data latih yang cara kerjanya mirip dengan *learning rate adaptive*, yaitu memakai nilai prediksi gradien momentum pertama dan kedua untuk menentukan nilai *learning rate* pada pemodelan *network* (Vani dan Rao, 2019).

2.7.3 *Learning Rate*

Learning rate merupakan bagian dari *hyperparameter* yang menentukan tingkat keberhasilan model CNN dalam mengenali data secara cepat dan efisien. Besar kecilnya nilai *learning rate* akan mempengaruhi hasil akurasi pelatihan data nantinya. Nilai *learning rate* yang terlalu besar akan mengakibatkan pemodelan CNN terlalu cepat mempelajari data sehingga hasil pelatihan data menjadi tidak maksimal. Jika memberikan nilai *learning rate* yang terlalu kecil, maka akan menyebabkan pemodelan CNN terlalu lama menganalisis data pelatihan. Sehingga, membutuhkan nilai *learning rate* yang tepat agar hasilnya juga tepat sasaran (Wardana *et al.*, 2021).

Hasil akurasi akan semakin optimal jika dibantu dengan *learning rate scheduler*. Umumnya, *learning rate scheduler* dikenal dengan SGDR *scheduler*. Parameter tersebut berfungsi untuk mengubah nilai akurasi secara bertahap

sehingga hasil akhir pemodelan dapat keluar dari zona *plateau* (Pradiptatmaka *et al.*, 2021).

2.7.4. *Epoch*

Epoch adalah parameter pengulangan yang dilakukan saat proses pelatihan data berdasarkan berapa banyak pengulangan yang di-*input*. Nilai *epoch* dapat bervariasi baik bernilai kecil maupun besar. Jumlah pengulangan yang dilakukan tergantung dari waktu pemodelan mempelajari data (Aristyanto dan Kurniawan, 2021).

2.8 *Confusion Matrix*

Confusion matriks berfungsi sebagai tahap mempermudah studi pembelajaran dalam menarik kesimpulan pada data klasifikasi. Selain itu, *confusion matrix* bertujuan untuk mendapatkan hasil akurasi dari data klasifikasi (Fattah, 2021).

2.8.1 *Recall*

Sensitivitas atau *recall* merupakan kinerja sistem data dalam mengenali hasil data yang benar-benar positif. Rumus *recall* dapat dihitung dengan persamaan (2.7) (Wonohadidjojo, 2021):

$$Recall = \frac{TP}{TP + FN} \tag{2.7}$$

2.8.2 *Precision*

Presisi atau *precision* adalah perbandingan jumlah data benar-benar positif berdasarkan data yang diperkirakan benar positif (bisa positif atau negatif). Rumus presisi dapat dihitung dengan persamaan (2.8) (Wonohadidjojo, 2021):

$$Precision = \frac{TP}{TP + FP} \tag{2.8}$$

2.8.3 Akurasi

Akurasi (*accuracy*) merupakan angka yang menunjukkan besarnya jumlah keberhasilan pada identifikasi data. Rumus akurasi dapat dihitung dengan persamaan (2.9) (Fattah, 2021):

$$Akurasi = \frac{\sum TP + \sum TN}{\sum TP + \sum FP + \sum TN + \sum FN} \tag{2.9}$$

2.8.4 *F1-Score*

F1-Score adalah pembagian nilai rata-rata yang dihitung dari presisi dan sensitivitas. Rumus *f1-score* dapat dihitung dengan persamaan (2.10) (Wonohadidjojo, 2021):

$$f1 - score = 2 \times \frac{precision \times recall}{precision + recall} \quad (2.10)$$

True positive (TP) adalah data positif benar yang juga terprediksi dengan benar. *True negative* (TN) ialah data negatif benar yang juga terprediksi dengan benar. *False positive* (FP) merupakan data positif benar yang terprediksi salah. *False negative* (FN), yaitu data negatif yang salah tetapi terprediksi dengan benar.