BAB III
METODOLOGI PENELITIAN

Bab ini akan memberikan gambaran umum tentang metode penelitian dalam pembangunan sistem klasifikasi penyakit paru-paru dari citra x-ray dada menggunakan metode convolutional neural network yang terdiri dari alat dan bahan, perancangan sistem, dan pengujian sistem.

3.1 Alat dan Bahan

3.1.1 Alat

Alat yang digunakan dalam penelitian ini adalah:

1. Laptop
2. Jupyter Notebook
3. Python 3.7.3

3.1.2 Bahan

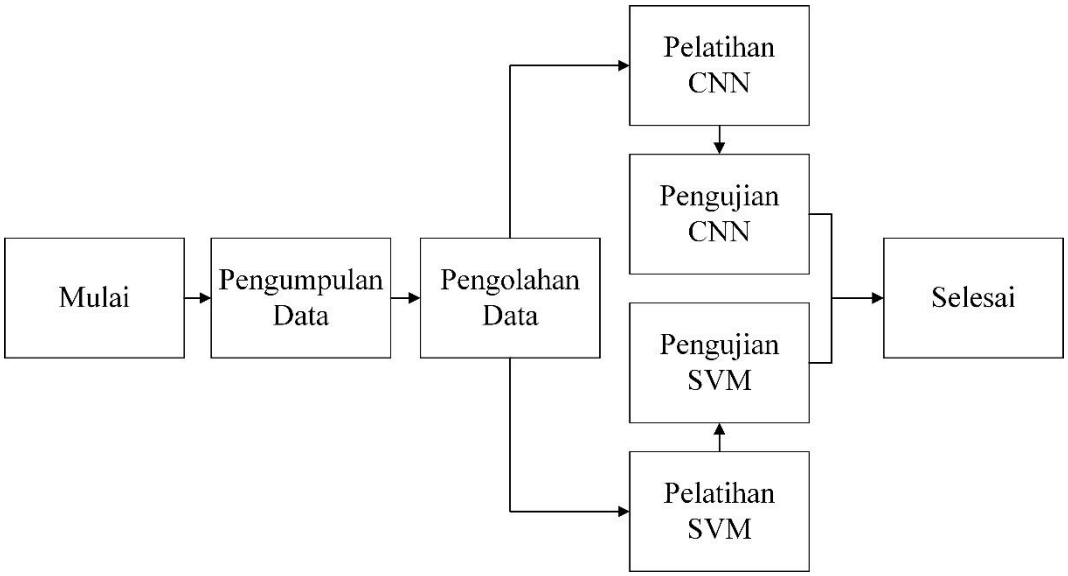
Bahan yang digunakan dalam penelitian ini adalah data hasil rontgen dada. Dalam penelitian ini jenis data yang digunakan adalah data primer. Data diperoleh dari sebuah website *opensource* yang menyediakan banyak dataset yang dapat diunduh secara gratis, link website:

1. https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database
2. https://www.kaggle.com/datasets/tawsifurrahman/tuberculosis-tb-chest-xray-dataset

3.2 Tahapan Penelitian

Tahapan dalam melakukan penelitian dimulai dari pengumpulan data, pengolahan data awal, pelatihan model dengan menggunakan CNN dan SVM, dan melakukan pengujian model. Pada pengumpulan data, kegiatan yang dilakukan yaitu mengumpulkan data dari website Kaggle dan akuisisi data citra rontgen dada. Tahap pengolahan data awal atau preprocessing untuk mempersiapkan data citra menjadi citra masukkan dalam CNN dan SVM. Tahap pelatihan model untuk proses pembelajaran dan mendapatkan model terbaik dengan mencari parameter CNN dan

SVM yang dapat menghasilkan akurasi terbaik. Tahap terkahir yaitu tahap pengujian untuk melakukan evaluasi dari model terbaik yang dihasilkan dari tahap penelitian.



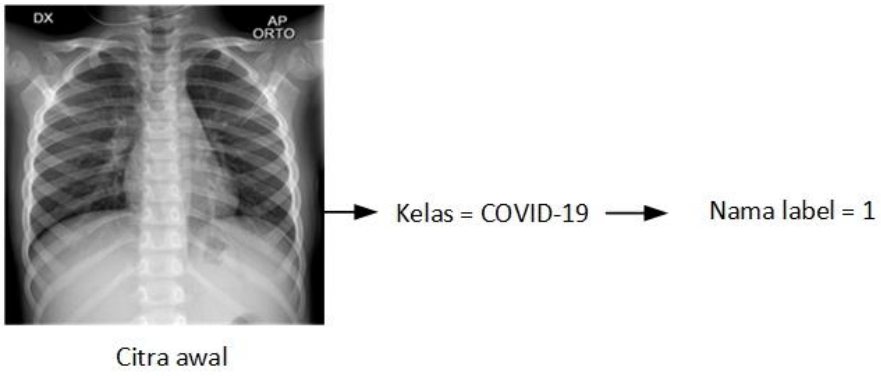
Gambar 3.1 Tahapan Penelitian

3.2.1 Pengumpulan Data

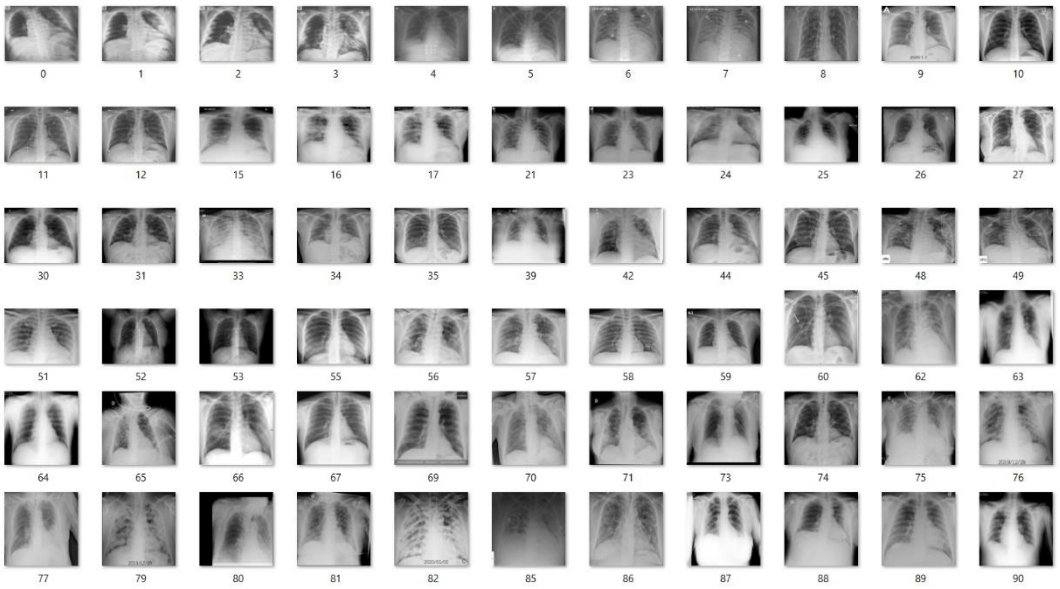
Dataset citra rontgen dada dikumpulkan dengan cara mengunduh pada website: https://www.kaggle.com/datasets/tawsifurrahman/tuberculosis-tb-chest-xray-dataset. Tahap pengambilan citra rontgen dada dilakukan dengan mengunduh dataset dari website. Pada gambar 3.2 menunjukkan contoh dataset citra rontgen dada.

Gambar 3.2 Contoh Citra Rontgen Dada Penyakit COVID-19

Kemudian masing-masing citra rontgen dada diberi label sesuai dengan penyakit. Setelah masing-masing citra diberi label data dibagi menjadi 4 jenis yaitu COVID-19, Pneumonia, Tuberculosis, dan Normal. Alur pelabelan data citra rontgen dada ditunjuukan pada Gambar 3.3 dan hasil palabelan data citra rontgen dada ditunjukkan pada Gambar 3.4.



Gambar 3.3 Pelabelan Data Citra Rontgen Dada



Gambar 3.4 Hasil palabelan data citra rontgen

Dari hasil pelabelan maka total data yang digunakan dalam penelitian ini dipaparkan pada Tabel 3.1.

Tabel 3.1 Total Data Citra Rontgen

No	Data	Total Data
1	Covid-19	3616
2	Normal	9573
3	Tuberculosis	700
4	Pneumonia	1995

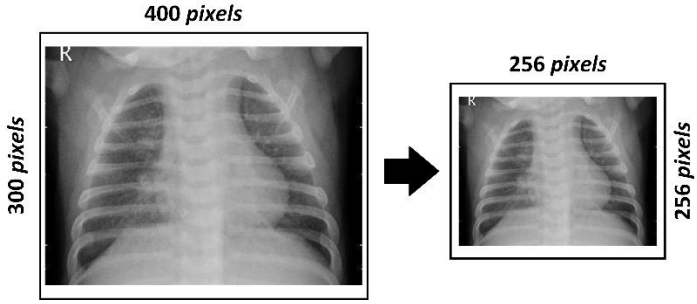
3.2.2 Preprocessing

Preprocessing adalah proses yang dilakukan terhadap data, bertujuan agar data siap dijadikan input proses pelatihan. Pada bagian pelatihan dan pengujian, *preprocessing* dilakukan dengan 2 tahap yaitu *resize*, dan augmentasi data. Pada tahap Pada penelitian ini *preprocressing* yang akan dilakukan adalah sebagai berikut:

1. *Resize*

   Resize merupakan tahap untuk menyamakan citra dari setiap data yang dimiliki, karena data yang dimiliki masih beraneka ragam ukuran. Pada penelitian ini data citra akan diresize menjadi 256x256 pixel, ukuran tersebut dipilih karena dianggap cukup kecil meminimalisir penggunaan memori tetapi

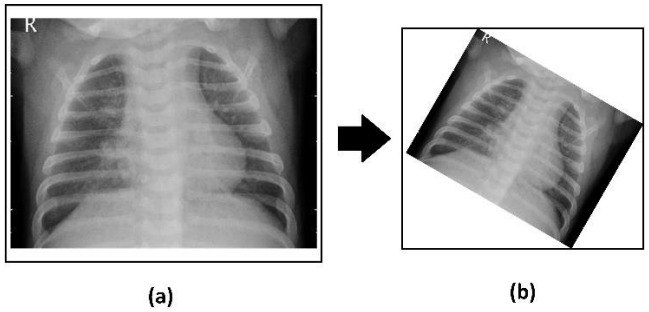
tidak menghilangkan informasi atau feature yang terdapat pada citra. Contoh resize ditunjukkan pada Gambar 3.5.



Gambar 3.5. *Resize* Data Rontgen Dada

2. Augmentasi Data

Preprocessing selanjutnya yaitu augmentasi data hal ini dilakukan karena data yang didapatkan pada saat proses pengumpulan data sangat terbatas untuk setiap kelas, agar klasifikasi mendapatkan hasil yang baik dibutuhkan data yang banyak. Proses augmentasi data meliputi *rescale, shear range, zoom range, rotation range, horizontal flip,*dan *fill mode*. Contoh data *augmentation* ditunjukkan pada Gambar 3.6.



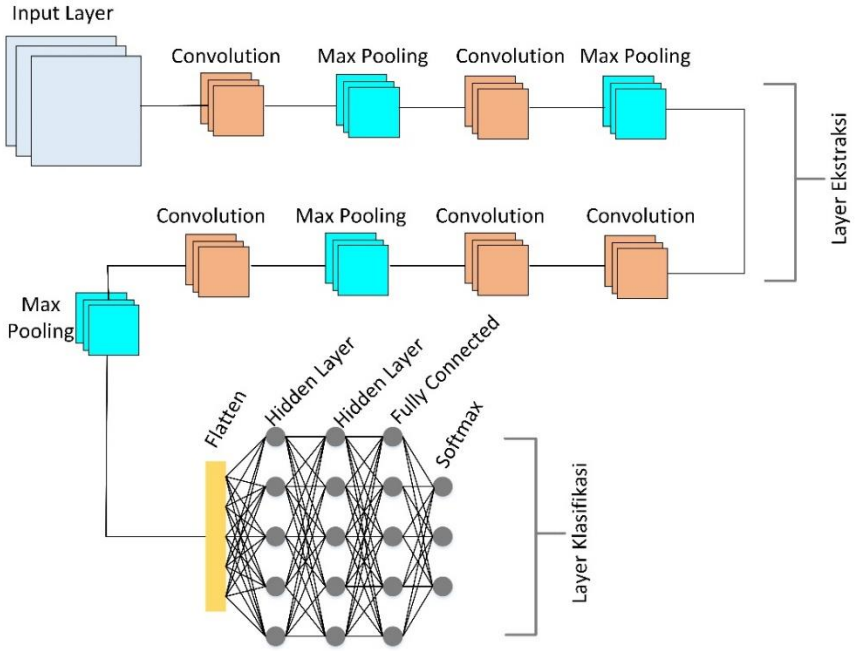
Gambar 3.6 Augmentasi data Rontgen Dada

3.2.3 Perancangan Model

Pada penelitian ini, setelah data terkumpul maka selanjutnya dilakukan proses perancangan model untuk melakukan proses pembelajaran terhadap data *training* dari setiap kelas, model yang dirancang sangat berpengaruh pada hasil klasifikasi.

1. *Convolutional Neural Network*

Pada model CNN sangat banyak arsitektur yang dapat dibuat, setiap arsitektur dengan data tertentu harus melewati *tuning* untuk mendapatkan model yang optimal. Arsitektur CNN pada penelitian ini menggunakan layer ekstraksi arsitektur AlexNet [45]. Rancangan arsitektur model CNN ditunjukkan Gambar 3.7.



Gambar 3.7 Rancangan arsitektur CNN

Pada arsitektur CNN, terdapat 2 bagian yaitu *layer* ekstraksi dan *layer klasifikasi*. Pada *layer* ekstraksi terdapat *input layer*, 5 *convolution layer*, 4 *max pooling layer*. Pada *layer* klasifikasi terdapat *flatten layer*, 2 *hidden layer*, dan *fully connected layer* dengan aktivasi *softmax*. Tabel 3.2 merupakan *detail* perancangan model arsitektur CNN yang digunakan pada penelitian ini.

Tabel 3.2 Rancangan model arsitektur CNN

Layers	CNN	Output
Input Layer	256x256x3	256x256x3
Convolution Layer	11x11x128	123x123x128
Pooling Layer	3x3 *max pooling*	61x61x128
Convolution Layer	5x5x128	61x61x128
Pooling Layer	3x3 *max pooling*	30x30x128

Convolution Layer	3x3x256	30x30x256
Convolution Layer	3x3x384	30x30x384
Pooling Layer	3x3 *max pooling*	14x141x384
Convolution Layer	3x3x256	14x14x256
Pooling Layer	3x3 *max pooling*	6x6x256
Flatten Layer	-	9216
Hidden Layer	1024, ReLU	1024
Hidden Layer	512, ReLU	512
Output Layer	*num class, softmax*	5

a. *Layer* Ekstraksi

Pada layer ekstraksi dilakukan proses ekstraksi feature dari data, proses ekstraksi pada CNN dilakukan dengan operasi matematis yang direpresentasikan sebagai layer, yaitu convolution layer, pooling layer, dan fungsi aktivasi ReLU. Pada arsitektur ini penentuan jumlah dari setiap layer tidak memiliki aturan yang tetap, sehingga penyusunan layer pada arsitektur ini disesuaikan dengan data.

Pada penelitian ini, perancangan arsitektur CNN dapat dilihat pada Gambar 3.7 dan detail perancangan dapat dilihat pada Tabel 3.2. Layer input menggunakan matriks berukuran 256x256x3 ukuran tersebut digunakan berdasarkan ukuran data citra pelatihan, yang telah melalui proses preprocessing, angka 3 yang dimaksud merupakan 3 channel pada matriks data citra yaitu Red, Green, Blue (RGB).

Layer ekstraksi pertama yaitu *convolution layer* dengan ukuran 11x11 dengan jumlah kernel 128. Matriks *input* akan melalui proses perkalian dengan matriks kernel, ilustrasi dapat dilihat pada Gambar 2.3. Proses lapisan ini akan diaktivasi dengan fungsi ReLU, dimana fungsi tersebut merubah semua nilai negative pada matriks menjadi 0, ilustrasi dapat dilihat pada Gambar 2.6.

Pada penelitian ini terdapat beberapa convolution layer yang digunakan. Proses dan ukuran *convolution* pertama sudah dijelaskan sebelulmnya hanya berbeda ukuran dan jumlah kernel nya. *Convolution layer* kedua dengan ukuran 5x5 dengan jumlah kernel 128, *convolution* ketiga dengan ukuran 3x3 dengan jumlah kernel 256, *convolution* keempat dengan ukuran

3x3 dengan jumlah kernel 384, *convolution* kelima dengan ukuran 3x3 dengan jumlah kernel 256.

Layer ekstraksi kedua yaitu *pooling layer*, pada penelitian ini menggunakan *max pooling layer*. *Max pooling* memiliki operasi sederhana untuk mengambil nilai maksimal dari *range* tertentu yang ditentukan dari kernel dan mengurangi dimensi *feature map*. Pada penelitian ini menggunakan empay layer *max pooling* dimana memiliki ukuran berbeda. Pada layer pertama sampai keempat *max pooling* memiliki ukuran 3x3, ilustrasi *max pooling* dapat dilihat pada Gambar 2.5.

Layer ekstraksi ketiga yaitu *flatten layer* yang bertujuan membentuk ulang fitur menjadi sebuah *vector* agar dapat digunakan sebagai input pada *fully connected layer*, setelah melakukan proses pada *convolution layer* dan *pooling layer* menghasilkan *vector* berukuran 1x9216 yang dijadikan sebagai input untuk layer klasifikasi.

b. *Layer* Klasifikasi

Pada layer klasifikasi ini menggunakan *fully connected layer* yang berbasis *backpropagation*. Vektor dari layer klasifikasi akan menjadi input untuk melatih bobot pada layer klasifikasi. Pada penelitian ini layer klasifikasi terdiri dari 2 hidden layer dengan jumlah node 1024 dan 512 dengan activation ReLU.

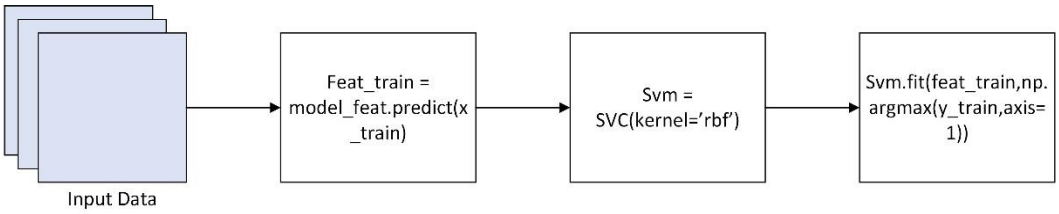
Vektor input akan melalui proses *forwardpropagation* yang dilakukan dengan mengkalikan matriks input dengan matriks bobot yang menghubungkan dengan *hidden layer*, hasil dari perkalian tersebut akan diaktivasi dengan fungsi ReLU, outputnya akan dikalikan dengan matriks bobot yang menghubungkan *hidden layer* dengan layer output dan fungsi aktivasi dengan softmax.

Selanjutnya adalah proses *backpropagation*, yang bertujuan untuk menyesuaikan kembali tiap *weight* berdasarkan *error* yang didapat pada proses *forwardpropagation*. Proses *forwardpropagation* dan *backpropagation* akan dilakukan secara *iterative* sehingga mendapatkan

bobot yang dianggap optimal, menghasilkan *accuracy* yang tinggi dan *error* yang kecil.

2. *Support Vector Machine*

   Metode setelah CNN adalah penggunaan SVM untuk klasifikasi data citra rontgen dada. Rancangan SVM ditunjukkan pada Gambar 3.8.



Gambar 3.8 Rancangan SVM

Pada Gambar 3.8 menunjukkan rancangan dari SVM yang akan digunakan. Input data untuk program SVM berasal dari data citra rontgen dada. Langkah selanjutnya adalah dilakukan prediksi terhadap data x_train. Inisialisasi kernel SVM yang akan digunakan untuk pengenalan karakternya, dan selanjutnya dilakukan *fitting* terhadap model SVM yang telah dibuat.

3.2.4 *Tuning Model*

Pada proses *tuning* bertujuan agar mendapatkan model yang lebih optimal, model awal akan melalui proses *tuning* untuk menyesuaikan model terhadap data yang ada. Pada rancangan model CNN dan SVM yang akan detuning sebagai berikut.

1. *Batch Normalization*

   Penambahan *batch normalization* pada proses *tuning* yang bertujuan membuat jaringan saraf lebih cepat dan lebih stabil dalam proses pelatihan.

2. *Hidden Layer*

   Pada proses tuning model tidak selalu menambahkan *hidden layer*, karena model awal itu sudah ada *hidden layer*, tetapi jika hasil pada saat pelatihan belum memuaskan maka *hidden layer* ditambahkan pada layer klasifikasi. Jadi *hidden layer* disini bersifat opsional.

3. *Node Hidden Layer*

   Meningkatkan jumlah *node* pada *hidden layer* pada saat tuning memungkinkan untuk mengurangi kesalahan pada saat pelatihan, tetapi juga mengurangi jumlah generalisasi. Pada penelitian ini jumlah node yang akan digunakan sebesar 1024 dan 512.

4. *Dropout*

   Pada proses *tuning dropout* sangat penting dilakukan, karena mencegah terjadi *overfitting* dalam jaringan saraf tiruan. Penelitian ini menggunakan *dropout* sebesar 0.2 dan 0.2.

5. *Loss function*

   Loss function adalah fungsi yang menghitung jarak antara *output* alogritma saat ini dan *output* yang diharapkan. Fungsi ini merupakan metode untuk mengevaluasi bagaimana algoritma kita memodelkan data. Penelitian ini menggunakan 'categorial_crossentropy' sebagai *loss function*.

6. *Learning function*

   Learning rate adalah *hyper parameter* yang mengontrol seberapa banyak kita menyesuaikan bobot jaringan kita dengan memperhatikan kerugian pada gradient.

7. *Decay*

   Fungsi *decay* adalah dimana setiap setelah pembaruan bobot dikalikat dengan *factor* yang sedikit yaitu kurang dari 1, dengan tujuan mencegah bobot tumbuh terlalu besar dan dapat dilihat sebagai penuran *gradient*.

8. *Momentum*

   Fungsi *momentum* adalah mempercepat penurunan *gradient* ke arah yang relevan dan meredam osilasi.

9. *Nesterov*

   Fungsi *nesterov* yaitu menghitung rata-rata *momentum* yang bergerak menurun ke dalam *gradient*, secara *default* nilai *nesterov* adalah *false*.

10. *Kernel*

    Fungsi utama kernel adalah mengambil ruang *input* berdimensi rendah dan mengubahnya menjadi ruang berdimensi lebih tinggi. Kernel sangat berguna

dalam masalah non-linear. Pada penelitian ini kernel yang digunakan adalah 'rbf'.

11. C (*Regularization*)

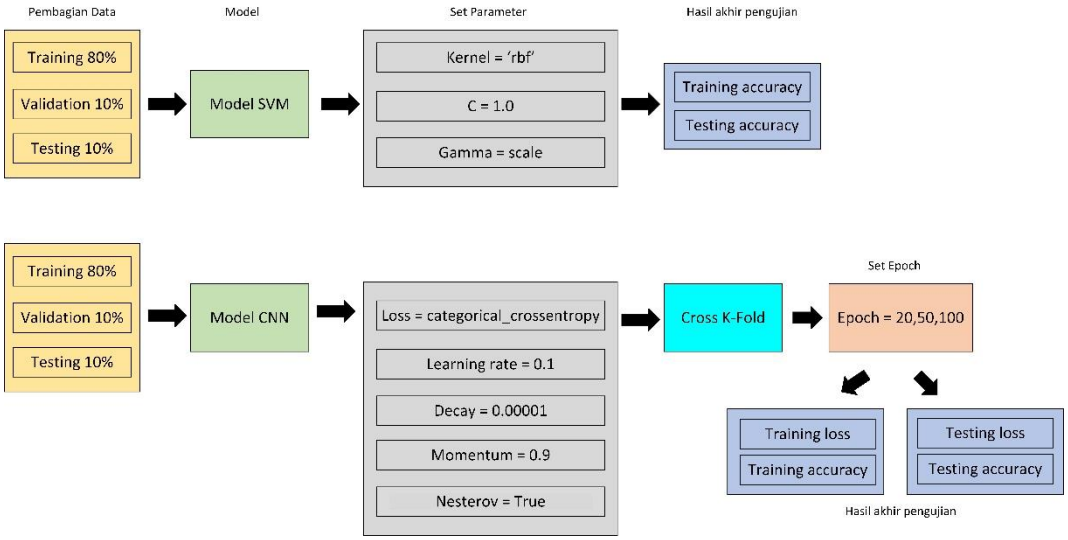
    C adalah parameter penalty yang mewakili kesalahan klasifikasi. Kesalahan klasifikasi memberi tahu pengoptimalan SVM berapa banyak kesalahan yang dapat ditanggung. Kita dapat mengontrol *trade-off* antara batas keputusan dan istilah kesalahan klasifikasi. Pada penelitain ini nilai C adalah 1.0.

12. *Gamma*

    Gamma bertujuan menentukan seberapa jauh pengaruh perhitungan garis pemisah yang dapat diterima. Pada penelitian ini *gamma* yang digunakan adalah '*scale*'.

3.2.5 Perancangan Pengujian

Setelah proses perancangan model selesai, maka didapatkan 2 model yaitu CNN dan SVM. Selanjutnya akan dilakukan beberapa skenario pengujian. Rancangan pengujian ditunjukkan pada Gambar 3.9.



Gambar 3.9 Rancangan pengujian

Pada perancangan pengujian langkah pertama adalah pembagian data. Data yang digunakan untuk model CNN dan SVM dibagai menjadi 3 dengan persentase

yang berbeda, yaitu *training* 80%, *validation* 10%, dan *testing* 10%. Langkah kedua adalah model CNN dan SVM yang telah dibuat dilakukan proses pembelajaran.

Langkah ketiga yaitu mengatur beberapa parameter. Pada model SVM parameter yang diatur yaitu *Kernel, C, Gamma* dan untuk model CNN parameter yang akan diatur yaitu *Loss, Learning rate, Decay, Momentum,* dan *Nesterov*. Pada model CNN juga K-*fold* dan *epoch* akan diatur,untuk K-*fold* bernilai 5 *fold*. Jika kita mengatur *epoch* 20 namun hasilnya akhir sesuai dengan harapan maka pengujian berhasil, namun jika tidak berhasil maka *epoch* ditingkatkan menjadi 50 atau 100. Setelah semuanya sudah diatur maka hasil akhir pada model SVM adalah *training accuracy, testing accuracy* dan pada model CNN adalah *training loss, training accuracy*, *testing loss,* dan *testing accuracy*.

3.2.6 Evaluasi *Accuracy, Precision, Recall, dan F1-score*

Proses terakhir adalah evaluasi untuk mengetahui kinerja dari model yang telah dibuat. Pada tahap ini dilakukan dengan melakukan klasifikasi terhadap semua data *testing* yaitu memiliki 3 kelas. Kemudian akan dilakukan proses perhitungan *accuracy, precision, recall,* dan f1-*score* dengan memanfaatkan *confusion matrix*.

1. *Accuracy*

$$Akurasi = \sum_{i=1}^{l} \frac{\frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}}{l} * 100\%$$

2. *Precision*

$$Presisi_{class} = \frac{TP_{class}}{TP_{class} + FP_{class}} * 100\%$$

3. *Recall*

$$Recall_{class} = \frac{TP_{class}}{TP_{class} + FN_{class}} * 100\%$$

4. F1-*score*

$$\text{F1-}score_{class} = 2 * \frac{Recall_{class} * Presisi_{class}}{Recall_{class} + Presisi_{class}}$$