BAB III
METODE PENELITIAN

3.1 ALAT YANG DIGUNAKAN

Pada penelitian ini menggunakan alat yang terdiri dari perangkat keras (*hardware*), dan perangkat lunak (*software*) serta dataset. Adapun rincian dari alat yang digunakan diantaranya sebagai berikut.

3.1.1 PERANGKAT LUNAK *(SOFTWARE)*

Software yang akan digunakan dalam simulasi dan analisis sinyal EEG imajinasi pergerakan lima jari tangan manusia adalah *spyder* atau *google colab*. *Software* ini sering digunakan dalam penelitian pada bidang pengembangan sistem, desain sistem, pembuatan model, *machine learning* dan lain-lain. Bahasa pemrograman yang digunakan yaitu *python* 3.8 dengan menggunakan *library*

3.1.2 PERANGKAT KERAS *(HARDWARE)*

Perangkat keras yang digunakan untuk penelitian ini yaitu sebuah laptop. Dengan spesifikasi sebagai berikut :

1. Windows 10 Home (64 -bit)
2. Intel® Core ™ i3-6006U CPU 2.0 GHz
3. RAM 4GB
4. VGA NVIDIA GeForce 920MX
5. HDD 1TB

3.1.3 DATA SET SINYAL EEG

Pada proses perekaman *Electroenchepalography* untuk imajinasi pergerakan lima jari dengan menggunakan EEG-1200 JE-921A EEG (Nihon, Kohden, Jepang). Pada perekaman sinyal EEG terdapat 22 *elektrode* yang akan ditempelkan dikepala diantaranya yaitu Fp1, Fp2, F3, F4, C3, C4, P3, P4, O1, O2, A1, A2, F7, F8, T3, T4, T5, T6, Fz, Cz, Pz, X5. Dari 22 *elektrode* yang ditempelkan dikulit kepala, ada 3 *elektrode* yang tidak bisa dipakai pada tahap klasifikasi karena *electrode* A1 dan A2 merupakan referensi kemudian *electrode* X5 merupakan *sinkronisasi*. Penulisan ini menggunakan data dengan label 5F (*5 Fingers*) agar dapat membayangkan

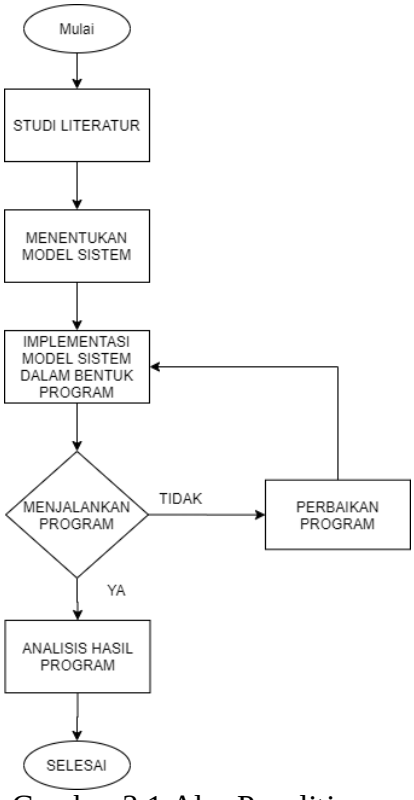
pergerakan lima jari yaitu : ibu jari, jari telunjuk, jari tengah, jari manis, dan jari kelingking. Sinyal EEG yang diperoleh pada perekaman ini ditandai dan disimpan menggunakan sinyal marker untuk menandakan imajinasi pergerakan pada jari tangan [4].

Tabel 3.1 Nilai marker

Marker	Keterangan
1	Ibu Jari
2	Jari Telunjuk
3	Jari Tengah
4	Jari Manis
5	Jari Kelingking

3.2 DIAGRAM ALUR PENELITIAN

Diagram alur penelitian ini menampilkan gambaran tahapan yang akan dilakukan pada penelitian ini. Adapun diagram alur penelitian ini yang disajikan dalam bentuk gambar sebagai berikut.



Gambar 3.1 Alur Penelitian

3.2.1 STUDI LITERATUR

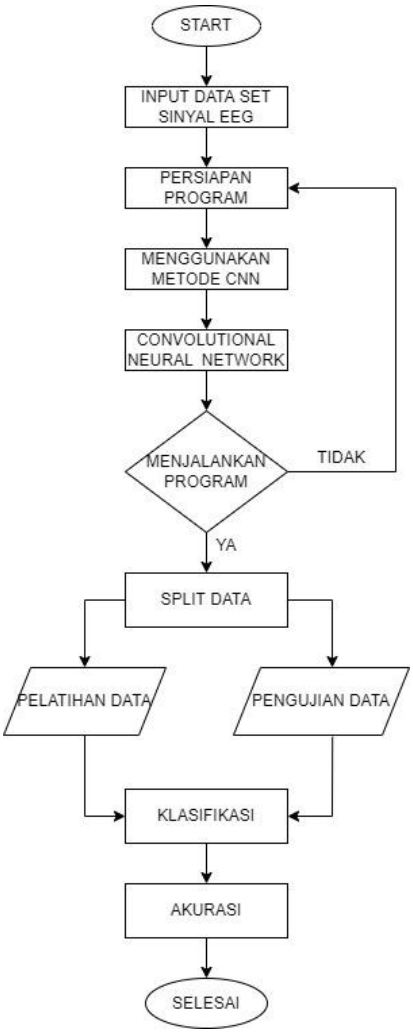
Studi literature bertujuan untuk mencari sumber referensi yang terkait dengan penelitian ini. SUmber yang digunakan diantaranya yaitu buku, jurnal, situs stack overflow dan GitHub.

3.2.2 MENENTUKAN MODEL SISTEM

Pada penelitian ini akan membahas tentang klasifikasi imajinasi pergerakan lima jari tangan dengan menggunakan metode deep learning yaitu *convolutional neural network* (CNN) 2D yang dapat melakukan pembelajaran mandiri agar dapat melakukan klasifikasi imajinasi pada lima jari tangan. Ada dua tahapan pada metode ini yang akan dilakukan, yang pertama yaitu *feature learning*, pada tahapan ini memiliki dua lapisan utama yaitu *convolution layer* dan *pooling layer*. Setelah melewati lapisan pertama, terdapat fungsi aktivasi yang akan digunakan pada penelitian ini yaitu dengan menggunakan fungsi aktivasi *ReLu*. Tahapan kedua yaitu c*lassification*, tahapan ini memiliki 1 *layer fully connected* dan *softmax* sebagai fungsi aktivasi yang akan digunakan. Selanjutnya peneliti meriset kode program dari penelitian sebelumnya dan merubah kode program sesuai dengan kebutuhan penelitian. Selanjutnya program dijalankan menggunakan *google colab*, jika kode program tidak jalan dan mengalami kesalahan, maka akan diriset Kembali kode program yang bermasalah, tetapi jika program berjalan maka akan dilanjutkan ketahap selanjutnya.

3.2.3 IMPLEMENTASI MODEL SISTEM

Implementasi model sistem ini meliputi data set sinyal EEG yang berupa perekaman sinyal EEG imajinasi pergerakan lima jari tangan manusia, lalu menggunakan metode *convolutional neural network* dua dimensi untuk mengklasifikasi sinyal EEG. Selanjutnya program dijalankan dengan membagi dua data menjadi pelatihan data dan pengujian data yang nantinya akan menampilkan akurasi dari model sistem tersebut.



Gambar 3 2 Implementasi Model Sistem

3.2.3.1 INPUT DATA SET SINYAL EEG

Data set ini menggunakan hasil perekaman sinyal EEG imajinasi pergerakan lima jari tangan dengan sampel yang berbeda. Hal ini menimbulkan data set hasil perekaman tidak dapat langsung digunakan oleh masukkan *two dimensional convolutional neurak network* karena setiap data memiliki lebar set yang berbeda, maka dari itu proses segmentasi sinyal EEG dibutuhkan untuk dapat memiliki lebar sampel yang sama. Proses segmentasi sinyal EEG menggunakan lebar sampel 200Hz. Data yang digunakan pada proses pengujian ini sebanyak 3800 sinyal yang diperoleh dari empat subjek diantaranya data kelas 1,3,4,5. Berikut merupakan kode program pada proses segmentasi sinyal EEG.

```
data1_numpy = data1.to_numpy()
a = 0
data_test_1 = np.zeros((959,200,19))
daftar_kelas_1 = np.zeros((959,1))
for x in range(len(kelas1)):
    if kelas1.iloc[x,0]==1 or kelas1.iloc[x,0]==2 or kelas1.iloc[
x,0]==3 or kelas1.iloc[x,0]==4 or kelas1.iloc[x,0]==5:
        daftar_kelas_1[a:a+1,0] = kelas1.iloc[x,0]
        batas_bawah = kelas1.iloc[x,1]
        data_test_1[a:a+1,:,:]= data1_numpy[batas_bawah+20:batas_
bawah+220,2:21
        a = a + 1
data3_numpy = data3.to_numpy()
a = 0
data_test_3 = np.zeros((959,200,19))
daftar_kelas_3 = np.zeros((959,1))
for x in range(len(kelas3)):
    if kelas3.iloc[x,0]==1 or kelas3.iloc[x,0]==2 or kelas3.iloc[
x,0]==3 or kelas3.iloc[x,0]==4 or kelas3.iloc[x,0]==5:
        daftar_kelas_3[a:a+1,0] = kelas3.iloc[x,0]
        batas_bawah = kelas3.iloc[x,1]
        data_test_3[a:a+1,:,:]= data3_numpy[batas_bawah+20:batas_
bawah+220,2:21]
        a = a + 1
data4_numpy = data4.to_numpy()
a = 0
data_test_4 = np.zeros((958,200,19))
daftar_kelas_4 = np.zeros((958,1))
for x in range(len(kelas4)):
    if kelas4.iloc[x,0]==1 or kelas4.iloc[x,0]==2 or kelas4.iloc[
x,0]==3 or kelas4.iloc[x,0]==4 or kelas4.iloc[x,0]==5:
        daftar_kelas_4[a:a+1,0] = kelas4.iloc[x,0]
        batas_bawah = kelas4.iloc[x,1]
        data_test_4[a:a+1,:,:]= data4_numpy[batas_bawah+20:batas_
bawah+220,2:21]
        a = a + 1
data5_numpy = data5.to_numpy()
a = 0
data_test_5 = np.zeros((957,200,19))
daftar_kelas_5 = np.zeros((957,1))
for x in range(len(kelas5)):
    if kelas5.iloc[x,0]==1 or kelas5.iloc[x,0]==2 or kelas5.iloc[
x,0]==3 or kelas5.iloc[x,0]==4 or kelas5.iloc[x,0]==5:
```

```
        daftar_kelas_5[a:a+1,0] = kelas5.iloc[x,0]
        batas_bawah = kelas5.iloc[x,1]
        data_test_5[a:a+1,:,:]= data5_numpy[batas_bawah+20:batas_
bawah+220,2:21]
        a = a + 1
data = np.concatenate([data_test_1[0:950,:,:],
data_test_3[0:950,:,:],
data_test_4[0:950,:,:],
data_test_5[0:950,:,:]])
daftar_kelas = np.concatenate([daftar_kelas_1[0:950,0:1],
daftar_kelas_3[0:950,0:1],
daftar_kelas_4[0:950,0:1],
daftar_kelas_5[0:950,0:1]])
kelas_full_min1 = daftar_kelas - 1
kelas_full = to_categorical(kelas_full_min1)
```

3.2.3.2 METODE CONVOLUTIONAL NEURAL NETWORK (CNN)

Penelitian ini menggunakan metodi *two-dimensional convolutional neural network.* Akan dilakukan beberapa tahapan dalam metode ini agar mendapatkan hasil dari inputan yang diberikan. Penelitian ini melakukan evaluasi *two-dimensional convolutional neural network* dengan memvariasikan *kernel* dan *pooling* layer pada lapisan konvolusi.

Tabel 3.2 Konfigurasi *hyper parameter* 2D-CNN

Hyper Parameter	*Konfigurasi*		
Jumlah *Kernel*	16	32	64
Panjang *Kernel*	3x3		
Pooling	*Maximum pooling*	*Average pooling*	
Dense	64		
Fungsi Aktivasi	ReLu		
Neuron Output Layer	5		
Optimasi MPL	Adam		

Perbedaan setiap kernel pada pada proses konvolusi dilakukan agar dapat mengetahui pengaruh terhadap nilai akurasi klasifikasi inajinasi pergerakan jari

tangan. Berikut kode program yang digunakan pada model *two-dimensional convolutional neural network*.

```
    model = Sequential()
    model.add(Conv2D(16,(3,3),activation='relu',input_shape=(200,
19,1)))
    model.add(MaxPooling2D(2,2))
    model.add(Conv2D(16,(3,3),activation='relu'))
    model.add(MaxPooling2D(2,2))
    model.add(Conv2D(16,(3,3),activation='relu'))
    model.add(Dropout(0.2))
    model.add(Flatten())
    model.add(Dropout(0.2))
    model.add(Dense(64, activation='relu'))
    model.add(Dense(n_outputs, activation='softmax'))
    model.compile(loss='categorical_crossentropy', optimizer='ada
m', metrics=['accuracy'])
```

3.2.3.3 TAHAPAN PELATIHAN DAN PENGUJIAN SISTEM

Pada proses pelatihan dan pengujian sistem *two-dimensional convolutional neural network* pada klasifikasi sinyal EEG imajiasi pergerakan lima jari tangan menggunakan *cross validation*. Untuk nilai K pada proses *Kfold cross validation* sama dengan 10 yang kedepannya akan dibagi menjadi data latih dan daya uji. 9 bagian akan digunakan untuk data latih dan 1 bagian akan digunakan untuk data uji.

Saat proses pelatihan *two-dimensional convolutional neural network* akan menimbulkan kesalahan pada klasifikasi yang akan dilihatkan pada grafik *training loss*. Setelah melakukan proses pelatihan data, selanjutnya melakukan proses pengujian data yang akan ditampilkan pada *confusion matrix*. Berikut kode program yang digunakan pada proses pelatihan dan pengujian data.

```
seed = 7
np.random.seed(seed)
kfold = StratifiedKFold(n_splits=10, shuffle=True, random_state=s
eed)
cvscores = []
class_names = ['Ibu Jari', 'Jari Telunjuk', 'Jari Tengah', 'Jari
Manis', 'Jari Kelingking']
#
for train, test in kfold.split(data, daftar_kelas):
    verbose, epochs, batch_size = 1, 100, 50
```

```
    n_timesteps, n_features, n_outputs = data.shape[1], data.shap
e[2], kelas_full.shape[1]
    history = model.fit(data[train,:,:], kelas_full[train,:], epo
chs=epochs, batch_size=batch_size, verbose=verbose, validation_sp
lit=0.2)
    loss = history.history['loss']
    val_loss = history.history['val_loss']
    accuracy = history.history['accuracy']
    acc = history.history['val_accuracy']
    epochs = range(1, len(loss) + 1)
    plt.figure()
    plt.plot(epochs, loss, 'y', label='Training Loss')
    plt.plot(epochs, accuracy, 'bo', label='Accuracy')
    plt.plot(epochs, val_loss, 'g', label='Validation Loss')
    plt.plot(epochs, acc, 'r', label='Validation Accuracy')
    plt.title('Training Loss & Accuracy')
    plt.xlabel('Epoch')
    plt.ylabel('Loss & Acc')
    plt.legend()
    plt.show()
    y_pred=np.argmax(model.predict(data[test,:,:]), axis=-1)
    cm=confusion_matrix(kelas_full_min1[test],y_pred)
    print(cm)
    fig, ax = plot_confusion_matrix(conf_mat=cm,
                                    colorbar=False,
                                    show_absolute=True,
                                    show_normed=False,
                                    class_names=class_names)
    plt.show()
    scores = model.evaluate(data[test,:,:], kelas_full[test,:], b
atch_size=batch_size, verbose=verbose)
    print("%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
    cvscores.append(scores[1] * 100)
```

3.3 TINGKAT AKURASI

Pada proses ini data hasil pengujian akan menampilkan tingkat akurasi disetiap *Kfold* dari 1 sampai 10 yang akan diperlihatkan pada *confusion matrix*. Penelitian ini menggunakan tiga tahapan proses konvolusi dengan tujuan mendapatkan tingkat akurasi terbaik. Pada penelitian ini akan memperlihatkan rata-rata akurasi dari setiap *kernel*. Berikut kode program untuk menampilkan ratairata akurasi.

```
    scores = model.evaluate(data[test,:,:], kelas_full[test,:], b
atch_size=batch_size, verbose=verbose)
    print("%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
    cvscores.append(scores[1] * 100)

print("%.2f%% (+/-
 %.2f%%)" % (np.mean(cvscores), np.std(cvscores)))
end_time = time.perf_counter()
print (end_time, "seconds - All CNN Processing time")
```

Pada penelitian ini akan menampilkan hasil data prediksi setiap kelas pada *confusion matrix multiclass*. Berikut tabel *confusion matrix* pada lima kelas jari tangan.

Tabel 3.3 *Confusion matrix* untuk lima kelas jari tangan

		Prediksi klasifikasi				
Kelas sebenarnya	Kelas	Ibu Jari	Jari Telunjuk	Jari Tengah	Jari Manis	Jari Kelingking
	Ibu Jari	T_{I_I}	F_{TL_I}	F_{TH_I}	F_{M_I}	F_{K_I}
	Jari Telunjuk	F_{I_TL}	T_{TL_TL}	F_{TH_TL}	F_{M_TL}	F_{K_TL}
	Jari Tengah	F_{I_TH}	F_{TL_TH}	T_{TH_TH}	F_{M_TH}	F_{K_TH}
	Jari Manis	F_{I_M}	F_{TL_M}	F_{TH_M}	T_{M_M}	F_{K_M}
	Jari Kelingking	F_{I_K}	F_{TL_K}	F_{TH_K}	F_{M_K}	T_{K_K}

Keterangan:

T_{I_I} : kondisi dimana hasil prediksi sesuai dengan kelas sebenarnya.

F_{TL_I} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{TH_I} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{M_I} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{K_I} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{I_TL} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

T_{TL_TL} : kondisi dimana hasil prediksi sesuai dengan kelas sebenarnya.

F_{TH_TL} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{M_TL} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{K_TL} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{I_TH} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{TL_TH} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

T_{TH_TH} : kondisi dimana hasil prediksi sesuai dengan kelas sebenarnya.

F_{M_TH} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{K_TH} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{I_M} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{TL_M} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{TH_M} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

T_{M_M} : kondisi dimana hasil prediksi sesuai dengan kelas sebenarnya.

F_{K_M} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{I_K} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{TL_K} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{TH_K} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

F_{M_K} : kondisi dimana hasil prediksi tidak sesuai dengan kelas sebenarnya.

T_{K_K} : kondisi dimana hasil prediksi sesuai dengan kelas sebenarnya.