
Perancangan Aplikasi Realtime Berbasis Desktop dengan Sensor IMU pada Klasifikasi Gerakan Semaphore Menggunakan Metode CNN

Tobias Nagata Budimartono^{1*}, Romy Budhi Widodo¹, dan Paulus Lucky Tirma Irawan²

¹Pusat Studi Human-Machine Interaction, Teknik Informatika, Universitas Ma Chung

²Pusat Studi Artificial Intelligence on Digital Images and Technopreneurship, Teknik Informatika, Universitas Ma Chung, Jalan Villa Puncak Tidar Blok N-1, Malang, Indonesia, 65151

Correspondence: 311810030@student.machung.ac.id

Received: 23 07 22 – Revised: 01 08 22 - Accepted: 04 08 22 - Published: 09 09 22

Abstrak. Semaphore merupakan komunikasi nonverbal yang menggunakan bendera sebagai sarana penyampaiannya. Informasi disampaikan melalui formasi bendera yang terdiri dari huruf, angka, dan simbol. Tujuan dari penelitian ini adalah untuk membuat aplikasi berbasis desktop yang mampu melakukan klasifikasi terhadap 27 gerakan semaphore yang dipilih yaitu seluruh huruf dan 1 karakter spasi atau istirahat. Aplikasi yang dirancang bersifat realtime sehingga data yang berasal dari sensor IMU (Inertial Measurement Unit) akan langsung diklasifikasi menggunakan model deep learning dengan algoritma CNN. Dataset yang digunakan untuk membuat model klasifikasi diambil dari 6 subjek yang dibagi menjadi 2 kelompok, kelompok pertama akan memperagakan secara berurutan mulai dari space/rest hingga Z sedangkan kelompok kedua akan memperagakan dengan urutan terbalik. Adapun pengambilan data setiap subjek dilakukan sebanyak 3 repetisi sehingga mendapatkan total data sebesar 92,200 data. Pembentukan model dilakukan dengan menggunakan 2 dan 3 sensor yang terletak di pergelangan tangan kanan, pergelangan tangan kiri, dan bagian punggung. Masing-masing sensor dibentuk dengan 3 tipe model yang berbeda yaitu simple, medium, dan complex. Setiap tipe model memiliki 3 proporsi pembagian data train dan data test yaitu 90% train 10% test, 80% train 20% test, dan 70% train 30% test. Berdasarkan model yang telah dibentuk pada penelitian ini akan menggunakan model dengan 2 sensor pada tipe model complex dengan proporsi 70% data train dan 30% data test yang mendapatkan nilai akurasi sebesar 76,65%. Pengujian dilakukan pada 2 subjek yang berbeda subjek pertama melakukan pengujian dengan urutan ascending sedangkan subjek kedua dengan urutan descending. Pengujian realtime menghasilkan nilai rata-rata akurasi sebesar 83,33%.

Kata kunci: *convolutional neural network, inertial measurement unit, realtime, semaphore*

Citation Format: Budimartono, T.N., Widodo, R.B., & Irawan, P.L.T. (2022). Perancangan Aplikasi Realtime Berbasis Desktop dengan Sensor IMU pada Klasifikasi Gerakan Semaphore Menggunakan Metode CNN. *Prosiding Seminar Nasional Universitas Ma Chung*, 75-88.

PENDAHULUAN

Secara garis besar manusia memiliki dua cara dalam menyampaikan pesan yaitu secara verbal dan non-verbal. Semaphore merupakan salah satu contoh komunikasi non-verbal yang dimana dalam pengaplikasiannya menggunakan dua buah bendera sebagai alat untuk membentuk formasi. Setiap formasi memiliki arti yang berbeda, arti dapat berupa huruf, angka, ataupun simbol. Formasi gerakan semaphore disusun dengan memperhatikan setiap patokan pada sekitar tubuh dengan titik pusat berada di tengah dada. Adapun semaphore saat ini sering kali digunakan pada kegiatan pembelajaran kepramukaan. Di Indonesia kegiatan kepramukaan berlangsung pada tingkatan sekolah dasar hingga sekolah menengah keatas yang dimana pada setiap pembelajarannya didampingi oleh guru pramuka. Akan tetapi kegiatan pembelajaran tersebut terbilang cukup tidak efektif dikarenakan seorang guru harus mengawasi beberapa siswa. Sehingga kesalahan dalam melakukan formasi gerakan semaphore lebih sering terjadi dan tidak diketahui oleh guru. Sehingga sangat diperlukan media pembelajaran lain dalam mempelajari gerakan semaphore dengan baik. Media berupa aplikasi untuk klasifikasi gerakan semaphore sangat disarankan sehingga setiap siswa dapat mempelajari gerakan semaphore yang benar secara mandiri.

Adapun beberapa penelitian yang dilakukan dalam kegiatan semaphore. Penelitian pertama merupakan penelitian yang dilakukan oleh Muhammad Fuad dan Eka Prasetya pada tahun 2015. Pada penelitian tersebut dilakukan pembentukan klasifikasi terhadap gerakan semaphore. Penelitian tersebut menggunakan sensor Kinect sebagai alat untuk mengambil data gerakan semaphore. Data yang didapat merupakan citra skeleton dan pada penelitian tersebut akan diambil tiga titik yaitu pergelangan tangan kiri, titik tengah bahu, serta pergelangan tangan kanan. Ketiga titik tersebut akan dihitung dengan perhitungan matematis untuk mengukur besaran sudut antara ketiga titik tersebut. Kemudian penelitian dilanjutkan dengan menguji sistem terhadap 3 subjek dan menghasilkan kesimpulan bahwa setiap masing-masing huruf dapat dikenali dalam waktu rata-rata 5,10 detik (Fuad & Prasetya, 2015).

Selanjutnya terdapat penelitian serupa yang dilakukan oleh Ratna Aisuwarya, Nadia Alfitri, dan Herry Wahyudi pada tahun 2017. Penelitian tersebut juga mengembangkan sistem klasifikasi gerakan semaphore dengan menggunakan sensor Kinect. Hal yang membedakan dengan penelitian di atas adalah dalam hal metode klasifikasinya. Pada penelitian ini metode yang digunakan adalah dengan memperhatikan kedelapan titik

semaphore. Penetapan titik tersebut akan berfungsi sebagai informasi gerakan semaphore. Dengan memberikan nilai toleransi sebesar 40% dari jari-jari lingkaran penelitian ini mendapatkan akurasi terbaik sebesar 70,19% (Aisuwarya *et al.*, 2017).

Adapun penelitian lain untuk mengklasifikasikan gerakan semaphore yang dilakukan oleh Tobias Nagata Budimartono pada tahun 2021. Penelitian tersebut juga melakukan klasifikasi gerakan semaphore dengan menggunakan sensor IMU. Dengan penggunaan sensor IMU diharapkan sistem klasifikasi yang dibuat tidak terpengaruh oleh faktor cahaya dan jarak. Sistem yang dibentuk adalah menggunakan metode *fuzzy* mamdani yang dimana mendapatkan hasil akurasi sebesar 67,50% setelah di uji pada 5 subjek untuk memperagakan 8 karakter semaphore (Budimartono, 2021). Penggunaan sensor IMU bertujuan untuk mengatasi keterbatasan sensor Kinect yaitu yang terkendala faktor jarak dan cahaya. Sensor IMU yang digunakan adalah seri MTw Awinda dari Xsens yang memiliki 5 sensor didalamnya yaitu *gyroscope*, *accelerometer*, *magnetometer*, *thermometer*, dan *barometer* (Paulich *et al.*, n.d.).

Penggunaan CNN pada klasifikasi gerakan manusia dengan menggunakan sensor IMU juga telah dilakukan sebelumnya. Seperti pada penelitian yang dilakukan oleh Valarezo dan rekannya pada tahun 2017 mengklasifikasikan 5 gerakan sehari-hari manusia. Hasil yang diperoleh adalah penggunaan CNN mendapatkan akurasi 95,43% dan RNN mendapatkan akurasi 96,95% (Valarezo *et al.*, 2017). Sedangkan pada penelitian oleh Rahn dan rekannya pada tahun 2021 melakukan klasifikasi terhadap 6 gerakan manusia sehari-hari. Penelitian tersebut menghasilkan akurasi sebesar 97% (Rahn *et al.*, 2021). Berdasarkan penelitian tersebut dapat diketahui bahwa penggunaan CNN terhadap klasifikasi gerakan dengan sensor IMU dapat dilakukan, sehingga akan digunakan sebagai metode klasifikasi pada penelitian ini.

MASALAH

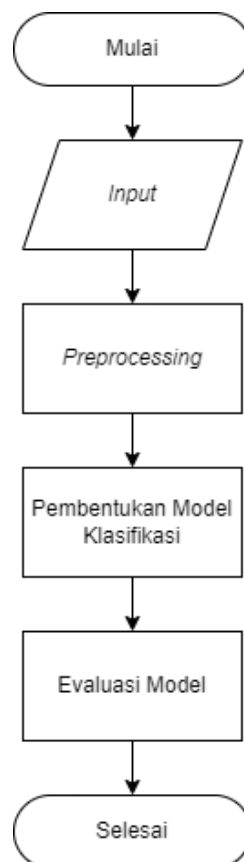
Berdasarkan latar belakang yang dipaparkan penelitian ini akan mengangkat mengenai masalah pembelajaran kepramukaan yang kurang efektif dengan menggunakan buku. Dengan memahami gerakan semaphore melalui buku sering menyebabkan kesalahan dalam praktiknya. Untuk mengurangi kesalahan-kesalahan tersebut memerlukan sebuah media interaktif berupa aplikasi yang dapat digunakan secara mandiri. Sehingga pembelajaran gerakan semaphore dapat lebih optimal tanpa perlu bantuan seorang guru.

METODE PELAKSANAAN

Pengumpulan Data

Pengumpulan data dilakukan pada 6 subjek yang kemudian akan dibagi menjadi 2 kelompok. Kelompok pertama terdiri dari subjek 1 sampai subjek 3 yang akan melakukan gerakan semaphore secara berurutan dari space/rest hingga Z, kelompok kedua yang terdiri dari subjek 4 sampai 6 akan melakukan gerakan semaphore dengan urutan terbalik yaitu Z hingga space/rest. Kemudian setiap subjek akan melakukan pengambilan data sebanyak 3 repetisi. Data yang diambil merupakan data yang didapat dari sensor IMU keluaran Xsens dengan tipe MTw Awinda. Sensor akan diletakkan pada 3 posisi yaitu pergelangan tangan kanan, pergelangan tangan kiri, dan tulang punggung. Setiap sensor akan menghasilkan nilai *roll*, *pitch*, dan *yaw* sehingga secara total akan mendapatkan 9 *feature* variabel dan 1 target variabel. Pengambilan akan dilakukan untuk setiap karakter, setiap karakter akan berlangsung selama 2 detik dengan menggunakan pengaturan *refresh rate* sebesar 100Hz yang berarti akan mendapatkan 200 data setiap detiknya.

Pembentukan Model Klasifikasi



Gambar 1. Alur Pembentukan Model Klasifikasi

Pembentukan aplikasi dimulai setelah mendapatkan dataset pada proses pengumpulan data. Selanjutnya dilakukan proses cleaning terhadap *missing value* dan *outlier value*. Setelah data telah bersih dan siap digunakan selanjutnya adalah menyiapkan model yang dirancang. Model akan dirancang terhadap 3 model yang dimana masing-masing model akan dibentuk pada 3 proporsi pembagian dataset yaitu 90% *train* 10% *test*, 80% *train* 20% *test*, dan 70% *train* 30% *test*. Pembagian tersebut akan dilakukan secara terpisah yaitu pada setiap karakteristiknya yang kemudian akan digabungkan kembali. Hal tersebut dilakukan agar pembagian dataset menjadi seimbang dan tidak terjadi *inbalance data*. Secara garis besar model akan memiliki 4 node input untuk 3 sensor dan 2 node untuk 2 sensor serta memiliki output layer dengan 27 node.

Hasil model yang didapat kemudian diuji dengan menggunakan nilai akurasi dan loss. Selain itu terdapat juga metrik pengukuran *confusion matrix* sebagai pendukung pengambilan keputusan model yang dipakai.

Pembentukan Aplikasi Realtime

Aplikasi dibentuk dengan berbasis desktop yang kemudian memiliki fitur utama yang direncanakan adalah sebagai berikut.

1. Aplikasi dapat terkoneksi dengan sensor IMU dan *station* untuk memperoleh data input.
2. Aplikasi dapat mengaktifkan kamera sehingga *user* dapat melihat gerakan yang sedang diperagakan.
3. Aplikasi dapat menampilkan derajat sudut *roll*, *pitch*, dan *yaw* pada masing-masing sensor di layar.
4. Aplikasi dapat menampilkan hasil alfabet yang diprediksi berdasarkan gerakan yang diperagakan pada layar.
5. Aplikasi dapat menyimpan data sensor dan data hasil klasifikasi.

Hasil aplikasi kemudian akan diuji terhadap 2 subjek yang dimana subjek pertama akan melakukan gerakan semaphore secara berurutan dari space/rest hingga Z, sedangkan subjek kedua akan melakukan dengan urutan terbalik. Sistem dapat dinyatakan berhasil menebak bila pada hitungan kurang dari 5 detik sistem dapat mengenali gerakan yang diperagakan oleh subjek. Rancangan hasil pengujian dapat dilihat pada tabel berikut dimana nilai TRUE berarti sistem berhasil mengklasifikasi gerakan dan sebaliknya nilai FALSE berarti sistem gagal mengklasifikasi gerakan.

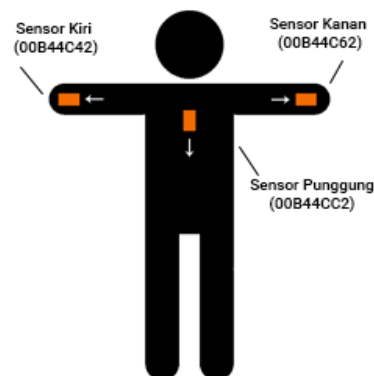
Tabel 1. Mode atau Tingkat Kesulitan

Kelas	Subjek 1	Subjek 2
Spasi	TRUE/FALSE	TRUE/FALSE
A	TRUE/FALSE	TRUE/FALSE
...
Y	TRUE/FALSE	TRUE/FALSE
Z	TRUE/FALSE	TRUE/FALSE
Jumlah Benar	0-27	0-27
Akurasi Subjek	n%	n%

HASIL DAN PEMBAHASAN

Pengambilan Data

Pengambilan data dilakukan terhadap 6 subjek dengan menempatkan 3 sensor IMU pada bagian tubuh subjek. Setiap subjek memiliki seri sensor yang setiap posisinya yaitu, seri 00B44C42 berada pada pergelangan tangan kiri, seri 00B44CC2 berada pada tulang punggung, dan seri 00B44C62 berada pada pergelangan tangan kanan. Peletakkan pada tubuh subjek dapat terlihat pada gambar 2.



Gambar 2. Letak Posisi Sensor

Sebelum sensor diletakkan pada tubuh subjek, sensor diletakkan terlebih dahulu pada meja dengan posisi sesuai arah yang akan dipasangkan pada tubuh subjek. Setelah itu sensor perlu untuk diterapkan *inclination reset* sehingga nilai *roll* dan *pitch* pada sensor menjadi ketitik nol sehingga membuat posisi sensor menjadi posisi default. Posisi sensor saat dilakukan *inclination reset* dapat terlihat seperti gambar dibawah ini.



Gambar 3. Letak Posisi Sensor Pada Proses Reset

Setelah sensor telah direset maka sensor siap diletakkan pada tubuh untuk pengambilan data. Posisi sensor pada tubuh subjek dapat dilihat pada gambar 4.



(a) Posisi sensor pada punggung



(b) Posisi sensor kedua tangan

Gambar 4. Letak Posisi Sensor Pada Tubuh Subjek

Kemudian pengambilan data dilakukan pada setiap subjek dengan dataset yang didapat adalah sebanyak 97.200 data. Seluruh dataset kemudian akan dibagi menjadi data *train test* dan validasi, untuk data *train test* akan menggunakan dataset pada repetisi 1 dan 2 sedangkan untuk data validasi akan menggunakan dataset repetisi ke 3. Contoh dataset yang diperoleh dapat dilihat pada gambar 5.

	roll_left	pitch_left	yaw_left	roll_back	pitch_back	yaw_back	roll_right	pitch_right	yaw_right	class
0	-10.505745	-72.127195	-6.652217	4.790394	-4.788344	-121.978950	8.437665	-76.235678	154.251618	space
1	-10.580209	-72.144434	-6.626842	4.790770	-4.787852	-121.973642	8.384441	-76.232125	154.272011	space
2	-10.628519	-72.165879	-6.612732	4.790489	-4.788128	-121.973122	8.321064	-76.226938	154.297352	space
3	-10.638134	-72.189794	-6.618461	4.788334	-4.788729	-121.979933	8.279996	-76.220683	154.304453	space
4	-10.645121	-72.220388	-6.622201	4.784529	-4.784001	-121.981655	8.231515	-76.214150	154.319472	space

Gambar 5. Contoh dataset yang diperoleh

Pengolahan Dataset

Tidak semua variabel pada dataset digunakan dalam pembentukan model melainkan hanya variabel yang memiliki informasi mengenai gerakan semaphore yang diambil. Pada penggunaan 3 sensor variabel yang digunakan adalah *pitch_left*, *roll_back*, *pitch_back*, dan

pitch_right sedangkan untuk 2 sensor hanya menggunakan *pitch_left* dan *pitch_right*. Setelah variabel dipilih selanjutnya adalah membersihkan dataset, pertama pada pengecekan *missing value* tidak ditemukan adanya indikasi *missing value* pada dataset ini. Pengecekan dapat dilihat pada gambar 6 dimana kotak hitam merupakan variabel yang digunakan sedangkan untuk mengecek *missing value* dapat dilihat pada non-null count dimana pada data *train test* memiliki jumlah penuh yaitu 64.800 dan untuk data validasi juga memiliki jumlah penuh yaitu sebesar 32.400 data.

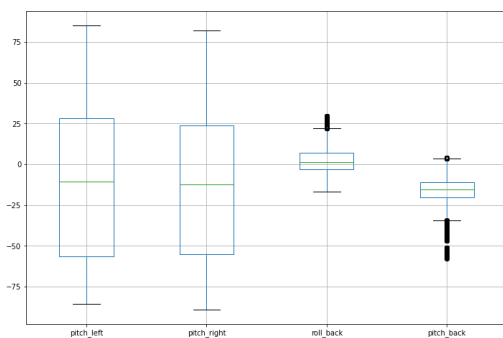
#	Column	Non-Null Count	Dtype	#	Column	Non-Null Count	Dtype
0	roll_left	64800 non-null	float64	0	roll_left	32400 non-null	float64
1	pitch_left	64800 non-null	float64	1	pitch_left	32400 non-null	float64
2	yaw_left	64800 non-null	float64	2	yaw_left	32400 non-null	float64
3	roll_back	64800 non-null	float64	3	roll_back	32400 non-null	float64
4	pitch_back	64800 non-null	float64	4	pitch_back	32400 non-null	float64
5	yaw_back	64800 non-null	float64	5	yaw_back	32400 non-null	float64
6	roll_right	64800 non-null	float64	6	roll_right	32400 non-null	float64
7	pitch_right	64800 non-null	float64	7	pitch_right	32400 non-null	float64
8	yaw_right	64800 non-null	float64	8	yaw_right	32400 non-null	float64
9	class	64800 non-null	object	9	class	32400 non-null	object

(a) Dataset *train* dan *test*

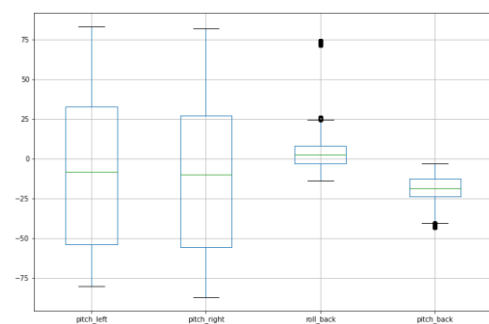
(b) Dataset validasi

Gambar 6. Pengecekan *missing value*

Pada pengecekan *outlier* terlihat ditemukannya nilai *outlier* yang dimana ditandakan oleh titik hitam pada gambar 7.



(a) Boxplot dataset *train* dan *test*

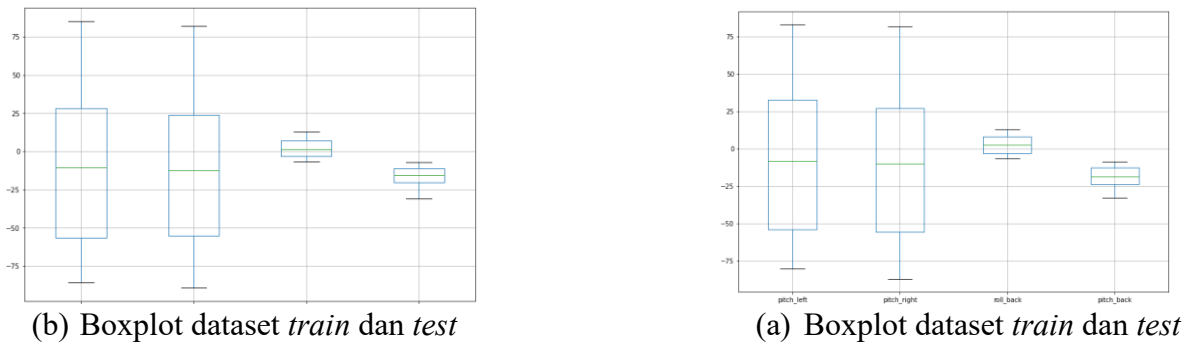


(b) Boxplot dataset validasi

Gambar 7. Boxplot dataset sebelum pengolahan outlier

Penanganan *outlier* tersebut diproses dengan menggunakan metode *winsorize* dimana metode tersebut akan membuat persentil baru terhadap keseluruhan data yang kemudian nilai persentil tersebut akan menjadi batas atas dan bawah. Setiap nilai yang melebihi batas-batas tersebut akan diubah menjadi nilai pada batas tersebut. Pada penelitian ini

menggunakan 10% dari data terendah dan tertinggi sebagai nilai batas. Hasil setelah diterapkan metode *winsorize* dapat dilihat pada gambar 8.



Gambar 8. Contoh dataset yang diperoleh

Pembentukan Model Klasifikasi

Model klasifikasi pada penelitian ini dibangun menggunakan algoritma *Convolutional Neural Network (CNN)*. Model dibentuk sebanyak 3 variasi yang diberi label sebagai model simple, model medium, dan model complex. Model simple terdiri dari 5 layer yang dimana susunan layer dapat dilihat pada gambar 9.

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 4, 16)	48
max_pooling1d (MaxPooling1D)	(None, 2, 16)	0
flatten (Flatten)	(None, 32)	0
dense (Dense)	(None, 64)	2112
dense_1 (Dense)	(None, 27)	1755

Gambar 9. Susunan layer CNN model simple

Setelah itu terdapat model medium yang terdiri dari 7 layer yang terdiri dari 1 layer *convolution* dan 1 layer *max pooling*. Susunan layer model medium dapat dilihat pada gambar 10.

Layer (type)	Output Shape	Param #
conv1d_1 (Conv1D)	(None, 4, 32)	96
dense_2 (Dense)	(None, 4, 64)	2112
dropout (Dropout)	(None, 4, 64)	0
max_pooling1d_1 (MaxPooling1D)	(None, 2, 64)	0
flatten_1 (Flatten)	(None, 128)	0
dense_3 (Dense)	(None, 128)	16512
dense_4 (Dense)	(None, 27)	3483

Gambar 10. Susunan layer CNN model medium

Untuk model complex terdapat 10 layer yang terdiri dari 1 layer *convolution*, 1 layer *max pooling*, dan 1 layer *dropout*. Susunan layer model complex dapat dilihat pada gambar 11.

Layer (type)	Output Shape	Param #
conv1d_2 (Conv1D)	(None, 4, 8)	24
dense_5 (Dense)	(None, 4, 16)	144
max_pooling1d_2 (MaxPooling 1D)	(None, 2, 16)	0
dense_6 (Dense)	(None, 2, 32)	544
conv1d_3 (Conv1D)	(None, 2, 64)	4160
dense_7 (Dense)	(None, 2, 64)	4160
dropout_1 (Dropout)	(None, 2, 64)	0
flatten_2 (Flatten)	(None, 128)	0
dense_8 (Dense)	(None, 256)	33024
dense_9 (Dense)	(None, 27)	6939

Gambar 11. Susunan layer CNN model simple

Berdasarkan ketiga model yang dibentuk selanjutnya diterapkan pada 2 sensor dan 3 sensor serta dibentuk pada 3 proporsi yang berbeda. Hasil pembentukan setiap model dapat dilihat pada tabel 2.

Tabel 2. Tabel *accuracy* dan *loss* pada seluruh model

Sensor	Proporsi	Kategori Model	Accuracy	Loss	Epoch
2 Sensor	90% train 10% test	Simple	0.957	0.107	64
		Medium	0.949	0.124	55
		Complex	0.952	0.126	41
	80% train 20% test	Simple	0.95	0.117	78
		Medium	0.936	0.154	35
		Complex	0.96	0.091	53
	70% train 30% test	Simple	0.939	0.15	71
		Medium	0.947	0.125	61
		Complex	0.945	0.138	40
3 Sensor	90% train 10% test	Simple	0.996	0.013	52
		Medium	0.995	0.012	46
		Complex	0.999	0.003	43
	80% train 20% test	Simple	0.995	0.014	63
		Medium	0.991	0.026	29
		Complex	0.997	0.006	32
	70% train 30% test	Simple	0.993	0.02	20
		Medium	0.995	0.02	52
		Complex	0.995	0.012	32

Pengujian Model Klasifikasi

Model yang telah dibentuk selanjutnya akan diuji terhadap data validasi yang telah disiapkan sebelumnya. Pengujian ini bertujuan untuk mengetahui apakah model yang telah dibentuk dapat digunakan secara general dengan menggunakan dataset dengan variasi yang berbeda. Hasil yang pengujian ini juga menjadi pemilihan untuk model yang akan digunakan pada aplikasi *realtime*. Hasil evaluasi model pada data validasi dapat terlihat pada tabel 3.

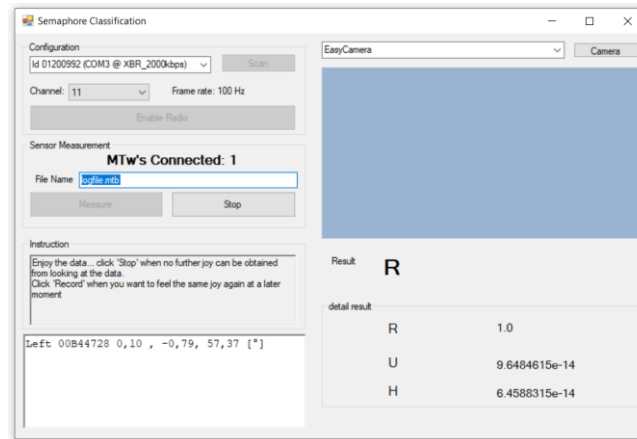
Tabel 3. Tabel *accuracy* dan *loss* pada pengujian data validasi

Sensor	Proporsi	Kategori Model	Prediksi Benar	Prediksi Salah	Akurasi
2 Sensor	90% train 10% test	Simple	24.253	8.147	74.85
		Medium	24.184	8.216	74.64
		Complex	23.454	8.946	72.39
	80% train 20% test	Simple	24.658	7.742	76.1
		Medium	24.056	8.344	74.25
		Complex	24.249	8.151	74.84
3 Sensor	70% train 30% test	Simple	25.052	7.348	77.32
		Medium	24.097	8.303	74.37
		Complex	24.835	7.565	76.65
	90% train 10% test	Simple	25.893	6.507	79.92
		Medium	25.844	6.556	79.77
		Complex	25.338	7.062	78.2
80% train 20% test	Simple	25.842	6.558	79.76	
	Medium	26.631	5.769	82.19	
	Complex	25.575	6.825	78.94	
70% train 30% test	Simple	26.437	5.963	81.6	
	Medium	26.302	6.098	81.18	
	Complex	25.936	6.464	80.05	

Berdasarkan hasil pengujian diketahui bahwa dengan menggunakan 2 sensor model terbaik didapatkan dengan proporsi 70% *train* 30% *test* pada model *complex* dengan akurasi sebesar 76.65%. Sedangkan dengan menggunakan 3 sensor model terbaik didapatkan dengan proporsi 80% *train* 20% *test* pada model *medium* dengan akurasi sebesar 82.19%.

Pembentukan Aplikasi Realtime

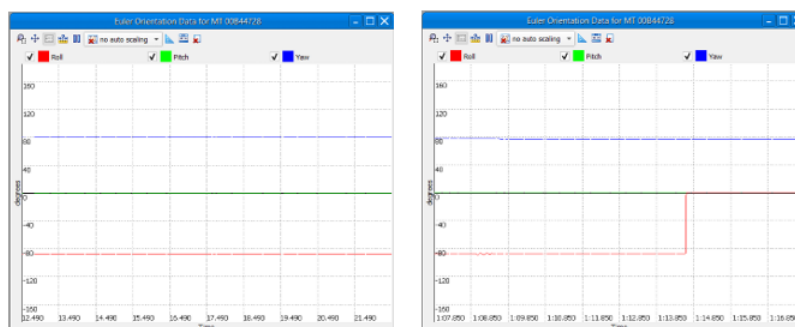
Aplikasi dibentuk dengan menggunakan bahasa C# dengan bentuk aplikasi desktop. Adapun tampilan aplikasi yang telah dibentuk dapat dilihat pada gambar 12.



Gambar 12. Tampilan UI Aplikasi

Pembentukan Aplikasi Realtime

Pengujian dilakukan dengan menggunakan 2 sensor saja dan tidak menggunakan 3 sensor. Hal tersebut dikarenakan API untuk *inclination reset* yang disediakan oleh Xsens untuk pengembangan aplikasi masih tidak dapat berjalan pada versi Beta 2019. Penggunaan 3 sensor tidak dapat digunakan karena praktiknya sangat memerlukan *inclination reset* pada sensor bagian punggung. Akan tetapi pada sensor di pergelangan tangan tidak memerlukan *inclination reset* hal tersebut dapat dilihat pada gambar 13 bahwa nilai *pitch* tidak berubah saat sebelum dan sesudah reset melainkan hanya *roll* saja yang berpengaruh adanya reset.



(a) Grafik sebelum di reset

(b) Grafik setelah di reset

Gambar 13. Grafik perbandingan nilai *roll*, *pitch*, dan *yaw* sebelum dan sesudah diterapkan *inclination reset*

Selanjutnya pengujian aplikasi realtime dilakukan pada 2 subjek dan menghasilkan hasil klasifikasi seperti pada tabel 4.

Tabel 4. Tabel Hasil Pengujian Aplikasi

Kelas	Subjek 1	Subjek 2
Space / Rest	TRUE	TRUE
A	TRUE	TRUE
B	TRUE	TRUE
C	TRUE	TRUE
D	TRUE	TRUE
E	TRUE	TRUE
F	TRUE	TRUE
G	TRUE	TRUE
H	TRUE	TRUE
I	TRUE	TRUE
J	TRUE	TRUE
K	FALSE	TRUE
L	FALSE	TRUE
M	TRUE	TRUE
N	FALSE	TRUE
O	TRUE	TRUE
P	TRUE	TRUE
Q	TRUE	FALSE
R	TRUE	TRUE
S	FALSE	TRUE
T	TRUE	TRUE
U	TRUE	TRUE
V	TRUE	TRUE
W	TRUE	FALSE
X	FALSE	FALSE
Y	TRUE	TRUE
Z	TRUE	FALSE
Jumlah Benar	22	23
<i>Accuracy</i>	81.48%	85.18%

KESIMPULAN

Pengembangan aplikasi *realtime* untuk klasifikasi gerakan semaphore dapat dilakukan dengan menggunakan sensor IMU. Hal tersebut dibuktikan dengan penggunaan 2 sensor pada proporsi 70% *train* 30% *test* dengan model complex didapat nilai akurasi sebesar 76.65% dan dengan penerapan aplikasi *realtime* mendapat pengaruh baik yaitu peningkatan akurasi menjadi 83.33%.

DAFTAR PUSTAKA

- Aisuwarya, R., Alfitri, N., & Wahyudi, H. (2017). Sistem penerjemah sandi semaphore menggunakan sensor kinect dengan pengenalan pola delapan titik. *Seminar Nasional Sains dan Teknologi, TE-005*, 1-6.
- Budimartono, T. N. (2021). *Implementasi sensor inertial measurement unit untuk klasifikasi gerakan semaphore menggunakan metode fuzzy mamdani*. Malang: Universitas Ma Chung.
- Fuad, M., & Prasetya, E. (2015). Pengenalan gestur semaphore menggunakan sensor kinect. *Seminar Nasional Aplikasi Teknologi Informasi, 11*, 266-270.
- Paulich, M., Schepers, M., Rudigkeit, N., & Bellusci, G. (2018). Xsens MTw Awinda: miniature wireless inertial-magnetic motion tracker for highly accurate 3d kinematic applications. 1-9.
- Rahn, V. X., Zhou, L., Klieme, E., & Arnrich, B. (2021). Optimal sensor placement for human activity recognition with a minimal smartphone–imu setup. *the 10th International Conference on Sensor Networks (SENSORNETS)*, 37-48.
- Valarezo, E., Rivera, P., Park, J. M., Gi, G., Kim, T. Y., Al-Antari, M. A., & Kim, T. S. (2017). Human activity recognition using a single wrist imu sensor via deep learning convolutional and recurrent neural nets. *Journal of ICT, Design, Engineering and Technological Science (JITDETS)*, 1(1), 1-5.

