Pengenalan Alfabet Bahasa Isyarat Tangan Secara Real-Time dengan Menggunakan Metode Convolutional Neural Network dan Recurrent Neural Network

Devina Yolanda, Kartika Gunadi, Endang Setyati
Program Studi Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) – 8417658
E-Mail: devinayolanda98@gmail.com, kgunadi@petra.ac.id, endang@stts.edu

ABSTRAK

Bahasa isyarat tangan merupakan salah satu alat komunikasi yang biasa digunakan oleh penyandang disabilitas atau difabel. Bahasa isyarat alfabet merupakan alat bantu dasar yang digunakan para pengajar untuk mengajar para penyandang tunarungu dan tunawicara untuk mengenal huruf alfabet dasar. Akan tetapi, banyak masyarakat yang kesusahan dalam berkomunikasi dengan kalangan tersebut dikarenakan kurangnya wawasan masyarakat tentang bahasa isyarat tangan. Penelitian pada bahasa isyarat tangan telah mengalami banyak kemajuan dalam memproses gambar statik tetapi masih mengalami kendala karena kesulitan dalam memproses gambar dinamik / video mengingat kebanyakan dari bahasa isyarat tangan direpresentasikan dengan gerakan tubuh, tangan, dan wajah.

Penelitian ini menggunakan metode *Convolutional Neural Network* (CNN) dan *Recurrent Neural Network* (RNN) dengan input berupa video. Metode CNN digunakan sebagai *feature extraction* di *spatial feature* sementara RNN bertugas untuk mengkolerasikan antar frame yang di ekstrak oleh CNN di *temporal feature*.

Hasil akhir yang ditampilkan berupa text alphabet yang merupakan hasil dari pengenalan alphabet bahasa isyarat tangan tersebut. Berdasarkan pengujian yang dilakukan, didapatkan nilai akurasi rata-rata sebesar 60.58% pada seluruh huruf sementara pengujian *real-time* mengalami kegagalan karena teknologi yang digunakan tidak bisa menopang arsitektur yang dibuat.

Kata Kunci: *Neural Network, Convolutional Neural Network, Recurrent Neural Network, Bahasa Isyarat Tangan*

ABSTRACT

Sign language is one of the communication tools commonly used by people with disabilities. The alphabet sign language is a basic tool used by teachers to teach people with hearing impairment and speech impairment to recognize basic alphabet letters. However, many people find it difficult to communicate with these groups because of a lack of community insight into hand sign language. Research on sign language has experienced much progress in processing static images but is still experiencing problems due to difficulties in processing dynamic images / video given that most of the sign language is represented by body, hand, and face movements.

This study uses Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) methods with video input. The CNN method will be used as a feature extraction in the spatial feature while the RNN is tasked to tolerate between frames extracted by CNN on the temporal feature.

The final result to be displayed is in the form of text alphabet which is the result of the recognition of the sign language alphabet. Based on the test carried out, obtained an average accuracy value of 60.58% for all letters while real-time testing has failed because the technology used cannot sustain the architecture created.

Keywords: *Neural Network, Convolutional Neural Network, Recurrent Neural Network, Sign Language*

1. PENDAHULUAN

Komunikasi merupakan salah satu hal penting dalam bersosialisasi. Banyak jenis media komunikasi yang dapat digunakan untuk berinteraksi dengan satu sama lain salah satunya adalah dengan menggunakan bahasa isyarat tangan. Salah satu kegunaan bahasa isyarat tangan adalah untuk membantu kalangan penyandang disabilitas atau yang biasa disebut difabel untuk berkomunikasi. Bahasa isyarat tangan juga digunakan oleh orang-orang yang memiliki kelainan perilaku seperti autisme dan *down syndrome*. Banyak masyarakat yang kesusahan dalam berkomunikasi dengan kalangan tersebut dikarenakan kurangnya wawasan masyarakat tentang bahasa isyarat tangan. Bahasa isyarat tangan yang umum digunakan adalah *American Sign Language* dan resmi diakui di Indonesia dengan sebutan SIBI (Sistem Isyarat Bahasa Indonesia) [1].

Pada dasarnya bahasa isyarat tangan menggunakan bahasa tubuh yang menggunakan gerakan tangan, wajah, dan gerakan tubuh untuk mempresentasikan sebuah kata tetapi bahasa isyarat yang berupa alfabet juga dapat digunakan untuk membantu proses komunikasi pada kata-kata yang tidak memiliki bahasa tubuh dalam bahasa isyarat. Bahasa isyarat alfabet merupakan alat bantu dasar yang digunakan para pengajar untuk mengajar para penyandang tunarungu dan tunawicara untuk mengenal huruf alfabet dasar.

Penelitian pada bahasa isyarat tangan telah mengalami banyak kemajuan dalam memproses gambar statik tetapi masih mengalami kendala karena kesulitan dalam memproses gambar dinamik / video. Video merupakan rangkaian gambar dan lebih susah untuk diklasifikasi karena mengandung *temporal* dan *spatial features* [5]. Beberapa peneliti telah mencoba untuk membuat penelitian dengan video tetapi dengan bantuan alat khusus seperti sensor sarung tangan [8] dan sensor *Kinect* [12].

Salah satu algoritma yang teruji cepat dan sangat baik dalam memproses gambar adalah *Convolutional Neural Network* (CNN). CNN sangat baik untuk memproses data spasial dan mengekstrak *unlabeled features*. Tetapi salah satu kelemahan CNN adalah CNN tidak sensitif terhadap rotasi atau gambar yang berbeda karena adanya *pooling layer* sehingga diklasifikasikan sebagai gambar yang sama [9]. Metode lain yang dikembangkan adalah *Recurrent Neural Network* (RNN). RNN cocok digunakan untuk memproses

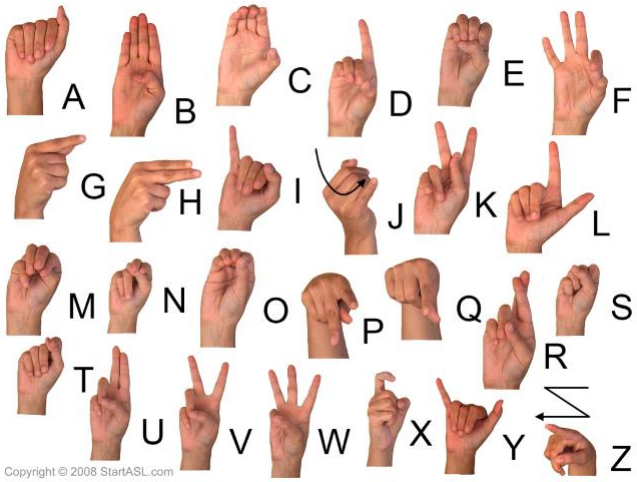
data yang bersifat *sequence* karena RNN dapat mengakses *hidden layer* dari data/frame sebelumnya. Metode yang diusulkan adalah menggunakan CNN sebagai *feature extraction* di *spatial feature* sementara RNN akan mengkolerasikan antar frame yang di ekstrak CNN di *temporal feature*.

2. DASAR TEORI

2.1 Bahasa Isyarat Tangan

Bahasa isyarat merupakan salah satu alat komunikasi yang sering digunakan khususnya bagi kalangan penyandang disabilitas atau yang biasa disebut difabel. Perbedaan mendasar antara bahasa isyarat dan bahasa lisan terletak pada modalitas atau sarana produksi dan persepsinya. Bahasa lisan diproduksi melalui alat ucap (oral) dan dipersepsi melalui alat pendengaran (auditoris), sementara bahasa isyarat diproduksi melalui gerakan tangan (gestur) dan dipersepsi melalui alat penglihatan (visual) [7].

Bahasa isyarat merupakan salah satu alat komunikasi yang menggunakan gerakan tangan, wajah, dan gerakan tubuh untuk mempresentasikan suatu kata maupun huruf. Selain itu, bahasa isyarat juga merupakan penanda identitas bagi penggunanya [7]. Pada kenyataannya, bahasa isyarat memiliki variasi dan jenis yang berbeda bergantung pada negara asal bahasa isyarat tersebut. Di Indonesia sendiri, terdapat 2 sistem isyarat yang digunakan yaitu SIBI (Sistem Isyarat Bahasa Indonesia) dan BISINDO (Bahasa Isyarat Indonesia). SIBI diadaptasi dari bahasa isyarat Amerika (*American Sign Language*) dan resmi diakui di Indonesia [1]. Contoh bahasa isyarat dapat dilihat pada Gambar 1.



Gambar 1. *American Sign Language*

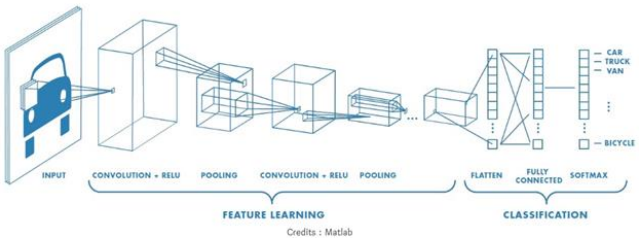
2.2 *Artificial Neural Network*

Artificial Neural Network (ANN) adalah suatu model yang memiliki pola seperti jaringan saraf otak manusia. Tiap neuron pada otak manusia saling berhubungan dan informasi mengalir dari setiap neuron tersebut. ANN menghasilkan model yang sulit dibaca oleh manusia karena memiliki *multilayer perception* dan sifatnya yang non-linear. ANN disusun dalam bentuk *layer* yang terdiri dari *input layer*, *hidden layer*, dan *output layer*.

2.2.1 Convolutional Neural Network

Convolutional Neural Network (CNN) memiliki beberapa lapisan layer yang mengekstrak informasi dari gambar dan menentukan klasifikasi dari gambar tersebut [4]. Dengan supervised traning pada jumlah gambar yang besar, CNN dapat mempelajari *pattern* yang kompleks pada gambar.

Secara umum, CNN merupakan gabungan layer dari *feature learning* dan *classification*. *Feautre learning* bertugas untuk mentranslasikan suatu input menjadi fitur berdasarkan ciri dari input tersebut. *Feature extraction* ini terdiri dari *convolutional layer*, *activation function*, dan *pooling layer*. Sementara *classification* bertugas untuk mengklasifikasi tiap neuron yang telah diekstrak sebelumnya. Terdiri dari *flatten*, *fully-connected layer*, dan *softmax* [10]. *Convolution layer* merupakan inti dari CNN. *Convolution layer* akan menghitung output dari neuron yang terhubung ke daerah lokal dalam input [10]. *Activation function* berfungsi untuk menentukan apakah neuron tersebut harus aktif atau tidak berdasarkan *weighted sum* dari input. *Pooling layer* adalah lapisan yang mengurangi dimensi dari *feature map* sehingga mempercepat komputasi karena parameter yang harus diupdate semakin sedikit dan mengatasi *overfitting* [10]. *Flatten* membentuk ulang fitur menjadi sebuah vektor agar bisa digunakan sebagai input dari *fully-connected layer*. *Fully-connected layer* akan menghitung skor kelas seperti *neural network* pada umumnya.



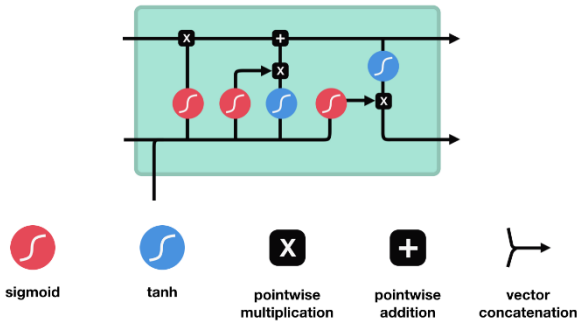
Gambar 2. Arsitektur CNN

2.2.2 Recurrent Neural Network

Recurrent Neural Network (RNN) adalah salah satu tipe *neural network* dimana *output* dari step sebelumnya akan dipakai sebagai *input* pada step selanjutnya. Pada *traditional neural networks*, semua *input* dan *output* independen terhadap satu sama lain tetapi ada saat ketika kata sebelumnya dibutuhkan untuk memprediksi kata berikutnya maka kata sebelumnya tersebut harus diingat/disimpan. RNN memberikan solusi dengan bantuan dari *Hidden Layer*. Fitur utama dan paling penting dari RNN adalah *hidden state*, yang mengingat beberapa informasi tentang suatu *sequence* [2]. RNN memiliki "*memory*" yang mengingat semua informasi tentang apa yang telah dihitung. RNN menggunakan parameter yang sama untuk setiap *input* karena RNN melakukan tugas yang sama pada semua *input* atau *hidden layer* untuk menghasilkan *output* yang mengurangi kompleksitas parameter, tidak seperti jaringan saraf lainnya [2].

Pada kenyataannya RNN tidak dapat mempelajari long-term dependencies (adanya jarak antara data sekarang dan data sebelumnya) karena RNN gagal mempelajari bagaimana cara menyambungkan input tersebut. Tetapi salah satu tipe RNN yaitu *Long Short Term Memory* (LSTM) dapat mengatasi masalah tersebut [13]. LSTM memiliki kemampuan untuk menghapus atau menambahkan informasi ke *cell state* yang diatur oleh struktur khusus yang disebut *gates*. *Gates* digunakan oleh LSTM untuk menentukan informasi mana saja yang ingin disimpan (opsional).

LSTM memiliki 3 *gates* yaitu *input gate*, *output gate*, dan *forget gate*. *Input gate* bertugas untuk memutuskan informasi mana yang akan diupdate. *Output gate* bertugas untuk mengeluarkan hasil yang telah di filter sebelumnya. *Forget gate* bertugas untuk memutuskan informasi apa yang akan dibuang dari *cell state* [13].



Gambar 3. Gates pada LSTM

2.2.3 Time Distributed Layer

Masalah yang sering di timbulkan oleh penggunaan *Convolutional Neural Network* (CNN) adalah CNN hanya bisa menerima satu input gambar dalam satu waktu saja untuk di klasifikasi sehingga apabila klasifikasi yang diinginkan berhubungan dengan *sequence/time series*, CNN akan membutuhkan waktu dan *layer* yang lebih banyak agar dapat menangkap *long-term dependencies* pada data tersebut [3].

Tetapi masalah tersebut dapat diselesaikan dengan memberikan sebuah layer di CNN yaitu *Time Distributed Layer*. *Time Distributed Layer* adalah layer yang biasanya digunakan pada *Recurrent Neural Network* (RNN) untuk menjaga *one-to-one relations* pada *input* dan *output*. Apabila menggunakan CNN saja tanpa bantuan dari *Time Distributed layer*, gambar-gambar yang diinputkan akan berjalan pada "*convolution flow*" yang sama sehingga gambar-gambar tersebut akan langsung tergabung pada *layer* pertama [6].

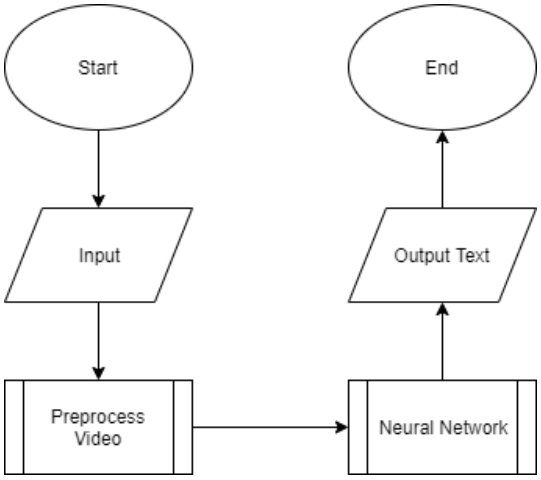
Sementara apabila data berbentuk *sequence/time series* maka hal tersebut sangat dihindari. Dengan menggunakan *Time Distributed layer*, masing-masing gambar akan berjalan pada "*convolution flow*" yang berbeda dengan satu sama lainnya sehingga tidak akan tercampur walaupun sebenarnya merupakan model yang sama, lalu pada akhirnya akan disatukan pada "*convolution block*".

2.3 *Real-Time*

Sebuah sistem *real-time* adalah dimana fungsi dari sistem bergantung pada hasil yang dihasilkan oleh sistem dan waktu [11]. Suatu sistem yang *real-time* akan memungkinan komputer untuk mengikuti sebuah proses yang berubah secara terus menerus. Pada sistem *real-time*, waktu merupakan hal yang dianggap paling penting tetapi bukan berarti apabila sistem tersebut tidak bisa berjalan dengan cepat maka sistem tersebut gagal. Sebuah sistem dapat dikatakan *real-time* jika sistem tersebut mampu memenuhi batasan respon yang dibutuhkan. Tetapi batasan waktu pada sistem tersebut harus tegas agar tidak terjadi penyimpangan.

3. DESAIN SISTEM

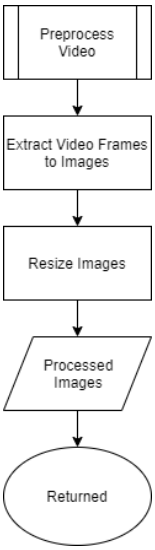
Sistem yang dibuat pada dasarnya adalah pengenalan alfabet dengan input awal berupa video kemudian video di konversi menjadi kumpulan gambar yang di proses di dalam sub-proses berikutnya yaitu *Preprocess Video* lalu selanjutnya hasil dari proses tersebut akan dilanjutkan kedalam sub-proses *neural network* yang terdiri dari *Convolution Neural Network* (CNN) dan *Recurrent Neural Network* (RNN). Hasil output yang dikeluarkan oleh *neural network* adalah berupa hasil klasifikasi dari pengenalan yang berupa angka hasil class dan akan di konversi menjadi text. Gambar dapat dilihat pada Gambar 5.



Gambar 5. Gambaran Besar Sistem

3.1 *Preprocess Video*

Sub-proses ini adalah proses awal untuk mempersiapkan dataset agar dapat diproses ke proses selanjutnya. Karena *Convolutional Neural Network* (CNN) tidak bisa menerima input berupa video sehingga video tersebut harus diubah menjadi bentuk gambar. Video adalah kumpulan dari beberapa gambar (frame) yang berurutan sehingga gambar-gambar tersebut harus dipisahkan. Video-video ini memiliki jumlah frame yang berbeda sehingga akan disamakan agar lebih mudah untuk di proses. Jumlah frame yang digunakan adalah 10 yang merupakan jumlah frame terkecil dari video-video tersebut. Proses ini dilakukan dengan mengambil frame pertama, frame terakhir, dan 8 frame secara acak. Setelah video di ekstrak, ukuran gambar akan dirubah (*resize*) menjadi ukuran 128x128 pixel. Gambar-gambar yang telah dihasilkan akan diteruskan ke proses selanjutnya yaitu *training neural network*. Alur prorocess video dapat dilihat pada Gambar 6.

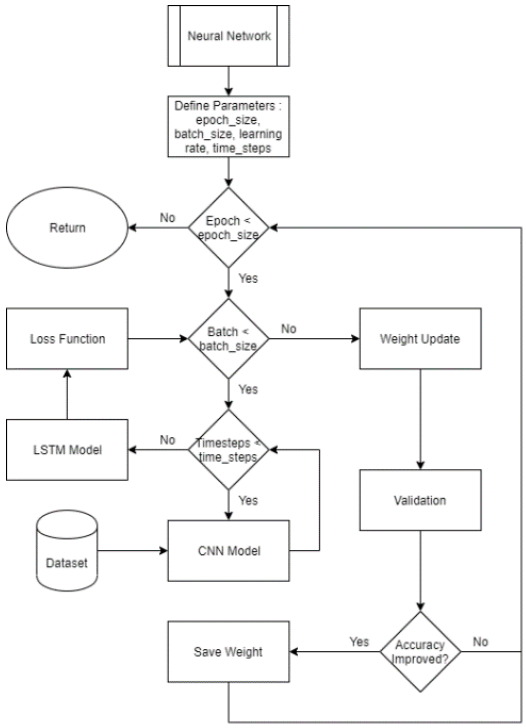


Gambar 6. Preprocessing Video

3.2 *Neural Network*

Training pada model *neural network* dilakukan dengan menggunakan gambar hasil preprocessing sebelumnya. *Training* dilakukan dengan menggunakan *weight* yang diinisialisasi secara *random* mengikuti *normal distribution*. Dataset *training* tidak hanya terdiri dari gambar saja melainkan terdapat label sebagai

tanda pengenal untuk setiap gambar tersebut. Label ini merupakan huruf A-Z sesuai dengan *class* hasil klasifikasi. Proses ini terdiri dari gabungan *Convolutional Neural Network* (CNN) dan *Recurrent Neural Network* (RNN), RNN yang digunakan adalah *Long Short-Term Memory* (LSTM). CNN digunakan sebagai *feature extraction* sementara LSTM bertugas untuk mengkolerasikan antar satu frame dengan yang lainnya. Rincian kerja CNN-LSTM dapat dilihat pada Gambar 7.



Gambar 7. *Flowchart* CNN-LSTM

4. ANALISA DAN PENGUJIAN

4.1 Dataset

Data dikumpulkan dengan dua cara, yaitu pertama diambil langsung dengan melakukan video model tangan yang bervariasi dan setiap tangan memperagakan 26 huruf alfabet. Hasil video tersebut memiliki rasio tinggi dan lebar sebesar 1:1, dengan resolusi 480 × 480 pixel. Kedua, dataset diperoleh dari hasil *browsing* pada internet. Data yang digunakan adalah video dengan format mp4. Ada beberapa video yang tidak memenuhi ruang lingkup seperti ukuran objek yang terlalu kecil, resolusi yang kurang bagus, objek telapak tangan terpotong, dan sebagainya sehingga menghasilkan sebanyak total 2582 video yang berhasil didapatkan dari hasil pengumpulan video sendiri dan pencarian video dari internet. Contoh dataset hasil pengambilan sendiri dapat dilihat pada Gambar 8.

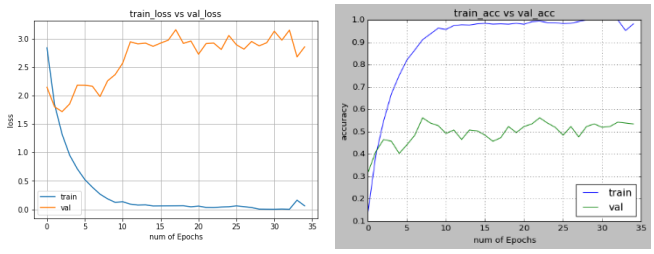
Gambar 8. Contoh Dataset Hasil Pengambilan Sendiri

4.2 Pengujian Untuk Mencari Model Terbaik

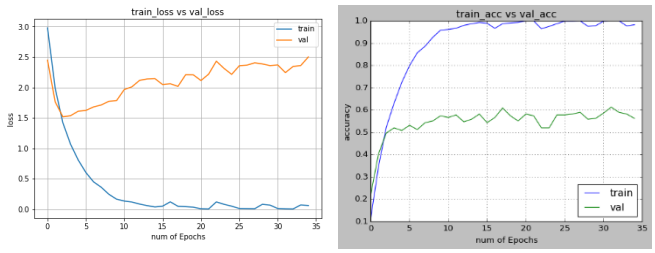
Pengujian didasarkan dengan tujuan untuk mencari model terbaik dengan menguji beberapa parameter. Pengujian dilakukan terhadap penambahan *layer* pada CNN, learning rate dan *number of steps* (epoch), dan *dropout layer* dan *batch normalization layer*.

4.2.1 Pengujian Terhadap Penambahan Conv Layer, Activation Layer ReLU, dan Max Pooling

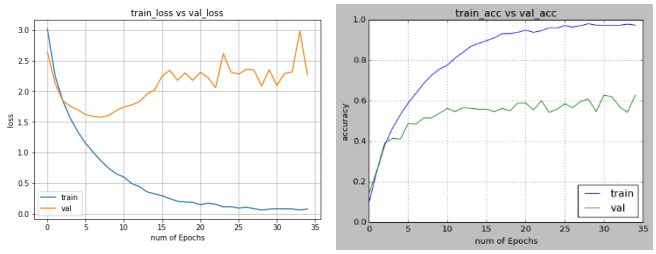
Pengujian ini dilakukan dengan menambahkan jumlah set layer. Satu set layer terdiri dari dua *convolutional layer*, *activation layer ReLU*, dan *max pooling*. Pengujian dilakukan dengan 3 model yang terdiri dari 3 set layer, 4 set layer, dan 5 set layer. Hasil yang diperoleh dari pengujian ini dapat dilihat pada Gambar 9, Gambar 10, dan Gambar 11. Hasil yang diperoleh dari pengujian tersebut didapatkan bahwa ketiga model masih mengalami *overfitting* sehingga akan diberikan pengujian selanjutnya dengan tujuan untuk mengurangi *overfitting*.



Gambar 9. *Loss* dan *Accuracy* dari Model dengan 3 Set *Layer*



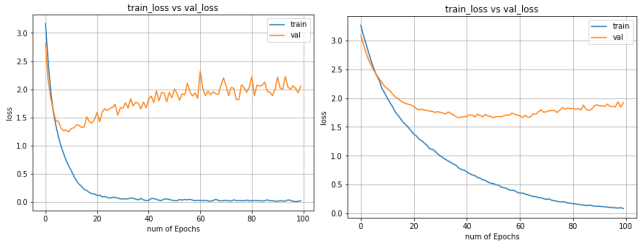
Gambar 10. *Loss* dan *Accuracy* dari Model dengan 4 Set *Layer*



Gambar 11. *Loss* dan *Accuracy* dari Model dengan 5 Set *Layer*

4.2.2 Learning Rate dan Number of Steps (Epoch)

Pada tahap ini dilakukan pengujian terhadap *learning rate* dan jumlah *epoch* (*number of steps*). Nilai *learning rate* yang diuji meliputi 1e-4 dan 1e-5 sementara nilai *number of steps* (*epoch*) yang diuji dimulai dari 1 hingga 100. Hasil *loss* dapat dilihat pada Gambar 12. *Learning rate* dengan nilai 1e-5 menghasilkan nilai *loss* yang lebih stabil dan lebih rendah daripada *learning rate* dengan nilai 1e-4 pada tiap perubahan *epoch* (*number of steps*). Sementara perkembangan *loss* yang terjadi jika dibandingkan dengan *number of steps* (*epoch*), semakin lama perkembangan *loss* semakin naik setelah jangka *steps* tertentu sehingga untuk pengujian selanjutnya diputuskan untuk menggunakan nilai *steps* dengan interval 1 sampai 50 dengan tujuan untuk mengurangi overfitting yang terjadi pada model.



Gambar 12. Hasil *Loss* pada Model dengan *Learning Rate* 1e-4 (Kiri) dan 1e-5 (Kanan)

4.2.3 Pengujian Terhadap Penambahan Dropout dan *Batch Normalization Layer*

Pengujian ini bertujuan untuk mengurangi overfitting pada model-model sebelumnya dengan menambahkan *dropout layer* dan *batch normalization layer*. Pengujian dilakukan dengan 2 model dengan perbedaan nilai *dropout* yaitu 0.5 dan kombinasi 0.5, 0.25, dan 0.25. Hasil perbedaan dapat dilihat pada Tabel 1.

Tabel 1. Perbandingan *Loss* dan *Accuracy* pada *Dropout*

Steps	Dropout 0.5		Dropout 0.5, 0.25, 0.25	
	Loss	Accuracy	Loss	Accuracy
5	2.6628	0.2248	2.5864	0.2519
15	1.7809	0.4031	1.7715	0.4457
20	1.6540	0.4767	1.5961	0.5039
25	1.5725	0.5039	1.4880	0.5233
30	1.5196	0.5271	1.4765	0.5465
35	1.4916	0.5426	**1.3996**	**0.5814**

4.3 Pengujian Model Menggunakan Data *Testing*

Setelah melakukan pengujian model dan parameter untuk menentukan model yang paling baik maka selanjutnya dilakukan pengujian terhadap data *testing* yang sudah disiapkan dari awal. Model akan menggunakan gabungan dari hasil pengujian-pengujian sebelumnya sehingga akan menggunakan model dengan rincian yaitu model terdiri dari 4 set layer, learning rate sebesar 1e-5, dan nilai *dropout* dengan kombinasi 0.5, 0.25, dan 0.25. Hasil dapat dilihat Tabel 2.

Tabel 2. Hasil Pengujian Terhadap Dataset *Testing*

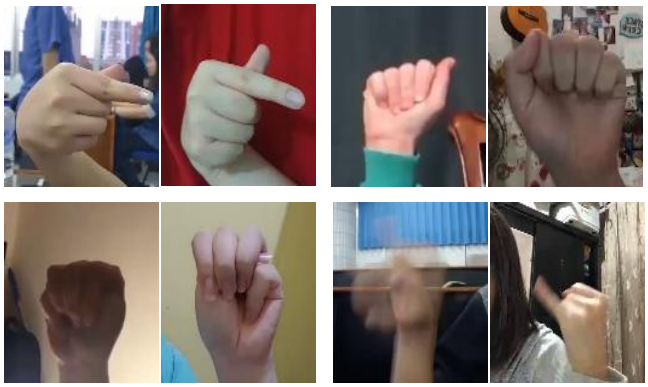
Huruf	Jumlah Video	Benar	Salah	Akurasi
A	10	7	3	70%
B	10	9	1	90%
C	10	10	0	100%
D	10	4	6	40%
E	10	10	0	100%
F	10	6	4	60%
G	10	3	7	30%
H	10	9	1	90%
I	10	6	4	60%
J	9	4	5	44.44%
K	10	4	6	40%

Tabel 2. Hasil Pengujian Terhadap Dataset *Testing* (lanjutan)

Huruf	Jumlah Video	Benar	Salah	Akurasi
L	10	8	2	80%
M	10	2	8	20%
N	10	7	3	70%
O	10	7	3	70%
P	10	2	8	20%
Q	10	9	1	90%
R	10	3	7	30%
S	10	5	5	50%
T	10	5	5	50%
U	10	4	6	40%
V	10	4	6	40%
W	10	8	2	80%
X	10	6	4	60%
Y	10	8	2	80%
Z	9	7	2	77.78%

4.4 Hasil Pengujian dan Analisis

Rata-rata hasil akhir akurasi yang dihasilkan pada pengujian terhadap dataset *testing* adalah 60.58%. Berdasarkan hasil akurasi pengujian, dilakukan analisa penyebab nilai akurasi tersebut. Kondisi dataset merupakan salah satu faktor penting dalam menentukan hasil akhir dan akurasi. Dalam pengujian ini, jumlah dataset yang digunakan cukup terbatas sehingga data yang digunakan untuk *training* tergolong sedikit. Kemudian terdapat dataset dengan *background* yang memiliki warna yang terlalu bervariasi (terlalu kompleks) sehingga menyebabkan model terlalu mempelajari secara teliti (*overfitting*). Terdapat juga dataset dengan pencahayaan yang terlalu gelap sehingga model lebih sulit untuk mempelajari fitur dalam gambar tersebut. Untuk dataset J dan Z yang dinamis, terdapat beberapa hasil gambar yang kurang jelas karena pergerakan terlalu cepat. Terakhir adalah jarak objek dengan kamera. Jarak objek sangat mempengaruhi model dalam belajar karena objek yang seharusnya dipelajari memiliki ukuran yang berbeda-beda sehingga model akan lebih kesusahan untuk mencari tau objek mana yang harus dikenali. Perbandingan kondisi dataset dapat dilihat pada Gambar 13.



Gambar 13. Perbandingan Kondisi Dataset

4.5 Pengujian *Real-Time*

Pada penelitian ini, *real-time* yang seharusnya menjadi salah satu fitur utama yang digunakan justru mengalami kegagalan. Pengujian untuk *real-time* tidak bisa dilakukan dikarenakan teknologi yang digunakan tidak mampu untuk menopang arsitektur yang dibuat. Implementasi sistem dilakukan pada komputer dengan spesifikasi:

- RAM: 8GB
- Memory: 1T HDD
- CPU: Intel Core i5
- GPU: NVDIA GeForce 930MX
- OS Windows 10

Arsitektur yang digunakan untuk *real-time* lebih berat dibandingkan pengujian-pengujian sebelumnya karena *real-time* merupakan sebuah proses yang selalu berjalan terus menerus sehingga akan selalu menerima *input* dan selalu menghasilkan *output*. Pengujian-pengujian yang dilakukan sebelumnya menggunakan 10 frame dikarenakan agar saat *testing* tidak terlalu lama sementara apabila *real-time* maka setiap frame yang diterima harus diproses seluruhnya.

Tabel 3. Perbandingan Jumlah Frame dan Waktu

Jumlah Frame	Waktu
10	32 detik
20	75 detik
30	164 detik

Dapat dilihat pada Tabel 3 bahwa perubahan jumlah frame mempengaruhi waktu *testing*, semakin banyak frame yang digunakan maka semakin lama waktu yang dihasilkan untuk mengeluarkan hasil outputnya. Video yang digunakan memiliki 30 fps dan jika dilihat pada tabel diatas, 30 frame yang sebenarnya merupakan video dengan waktu 1 detik membutuhkan waktu komputasi 2 menit 44 detik sementara jika *real-time*, video harus selalu berjalan setiap detiknya sehingga diputuskan bahwa pengujian tidak akan dilanjutkan.

5. KESIMPULAN

Setelah dilakukan perancangan sistem, pengimplementasian, dan pengujian terhadap aplikasi yang telah dibuat, dapat ditarik kesimpulan sebagai berikut:

- Penggunaan metode *Convolutional Neural Network* dan *Recurrent Neural Network* dalam pengenalan alfabet bahasa isyarat tangan mendapatkan akurasi keseluruhan sebesar 60.58% pada seluruh huruf.
- Berdasarkan pengujian-pengujian diatas, model yang paling baik untuk digunakan pada pengujian ini adalah 4 set layer yang terdiri dari dua *convolutional layer*, *activation layer* ReLU, dan *max pooling* dengan *learning rate* 1e-5 dan *dropout* kombinasi 0.5, 0.25, dan 0.25.
- Untuk membedakan setiap huruf, objek harus terlihat jelas dan objek memiliki warna yang kontras dengan background. Segmentasi yang sesuai untuk gambar juga diperlukan untuk meningkatkan akurasi pada *neural network*.
- Pengujian *real-time* mengalami kegagalan karena teknologi yang digunakan tidak bisa menopang arsitektur yang dibuat.

Saran untuk pengembangan kedepannnya adalah:

- Menambahkan jumlah dan variasi dataset pada *neural network* agar prediksi *neural network* dapat menjadi lebih baik.
- Pengembangan proses segmentasi dalam menghilangkan background dibelakang objek agar dapat dihasilkan hasil yang lebih baik.
- Mencoba pengujian lainnya agar mendapat model yang lebih baik lagi.

6. DAFTAR PUSTAKA

[1] Adityo, M. 2019, March 25. *Sistem Isyarat Bahasa Indonesia (SIBI) atau Bahasa Isyarat Indonesia (BISINDO)?*. Retrieved from Kolom YOTers: https://www.youngontop.com/read/20433/sistem-isyarat-bahasa-indonesia-sibi-atau-bahasa-isyarat-indonesia-bisindo/

[2] Aishwarya. n.d. *Introduction to Recurrent Neural Network*. Retrieved from GeeksforGeeks: https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/

[3] Benafanne, Y. 2019, August 13. Transformer vs RNN and CNN for Translation Task. Retrieved from Medium: https://medium.com/analytics-vidhya/transformer-vs-rnn-and-cnn-18eeefa3602b

[4] Ciaburro, G., & Venkateswaran, B. 2017. Neural Networks with R. Birmingham: Packt Publishing Ltd.

[5] Dinesh, S., Sivaprakash, S., Keshav, M., & Ramya, K. 2018, March. Real-Time American Sign Language Recognition with Faster Regional Convolutional Neuralnetworks. *International Journal of Innovative Reaserch in Science Engineering and Technology (IJIRSET)* (Vol. 7).

[6] Ferlet, P. 2019, July 23. How to work with Time Distributed data in a neural network. Retrieved from Medium: https://medium.com/smileinnovation/how-to-work-with-time-distributed-data-in-a-neural-network-b8b39aa4ce00

[7] Isma, S. T. n.d. MENELITI BAHASA ISYARAT DALAM PERSPEKTIF VARIASI BAHASA.

[8] Lokhande, P., Prajapati, R., & Pansare, S. 2015. Data gloves for sign language recognition system. *International Journal of Computer Applications*, *975*, 8887.

[9] Masood, S., Srivastava, A., Thuwal, H. C., & Ahmad, M. 2018. Real-time sign language gesture (word) recognition from video sequences using CNN and RNN. In *Intelligent Engineering Informatics* (pp. 623-632). Springer, Singapore.

[10] Sofia, N. 2018. CONVOLUTIONAL NEURAL NETWORK. Retrieved from Medium: https://medium.com/@nadhifasofia/1-convolutional-neural-network-convolutional-neural-network-merupakan-salah-satu-metode-machine-28189e17335b

[11] Sommerville, I. 2004. Real-Time Software Design (7th ed., Chapter 15). *Software Engineering*. Retrieved from https://www.coursehero.com/file/17891300/ch15/

[12] Tao, W., Leu, M. C., & Yin, Z. 2018. American Sign Language alphabet recognition using Convolutional Neural Networks with multiview augmentation and inference fusion. *Engineering Applications of Artificial Intelligence*, *76*, 202-213

[13] Zaytar, M. A., & El Amrani, C. 2016. Sequence to sequence weather forecasting with long short-term memory recurrent neural networks. *International Journal of Computer Applications*, *143*(11), 7-1