Pengenalan Citra Bahasa Isyarat Berdasarkan Sistem Isyarat Bahasa Indonesia Menggunakan Metode Vision Transformer

Renaldy [1] Agus Budi Dharmawan [2]

[1] [2] Teknik Informatika, Fakultas Teknologi Informasi, Universitas Tarumanagara
Jl. Letjen S. Parman No.1, Jakarta
email : renaldy.535200036@stu.untar.ac.id[1], agusd@fti.untar.ac.id[2]

ABSTRACT

Sign language is a form of communication between deaf people. In Indonesia, the formal sign language is Sistem Isyarat Bahasa Indonesia or SIBI for short which is a formal sign language based on American Sign Language. However, automatic sign language recognition still faces various challenges including the complexity of hand gestures, individual variations in sign performance, and the need for real-time interpretation. These challenges make the accuracy and efficiency of sign recognition very important. To address these issues, the Vision Transformer (ViT) method can be implemented, given its advantage in capturing important features of images and its ability in processing complex computer vision tasks. ViT or Vision Transformer is an artificial neural network architecture designed for image processing or computer vision tasks. From the training results with the Vision Transformer model, the training accuracy is 100% and the validation accuracy is 92.30%.

Key words

Indonesian Sign Language System, Artificial Neural Network, Vision Transformer, Image processing, Video processing.

1. Pendahuluan

Bahasa isyarat merupakan bentuk komunikasi antara satu dengan yang lain penderita tunarungu. Di Indonesia bahasa isyarat yang formal adalah Sistem Isyarat Bahasa Indonesia atau disingkat SIBI merupakan bahasa isyarat formal yang dibuat berdasarkan America Sign Language [1]. Selain SIBI ada satu jenis bahasa yang secara alami terbangun dari kelompok tunarungu biasa disebut BISINDO atau Bahasa Isyarat Indonesia [2]. Rancangan aplikasi ini dibuat dengan menggunakan Sistem Isyarat Bahasa Indonesia (SIBI) dikarenakan SIBI adalah sistem isyarat yang telah dibakukan dan disepakati secara nasional, sehingga memudahkan komunikasi antara individu tunarungu di berbagai daerah di Indonesia. Pengenalan bahasa isyarat dibuat agar orang dapat memahami bahasa isyarat dengan mudah, aplikasi akan mendeteksi gerakan tangan dalam bentuk video.

Meskipun begitu, masih ada sejumlah masalah dengan identifikasi bahasa isyarat otomatis, seperti kerumitan gerakan tangan, perbedaan individu dalam kinerja isyarat, dan persyaratan untuk interpretasi *real-time*. Efisiensi dan akurasi identifikasi isyarat sangat penting mengingat kesulitan-kesulitan ini. Metode Vision Transformer (ViT) dapat digunakan untuk mengatasi masalah ini karena dapat menangkap fitur gambar yang signifikan dan kapasitasnya untuk menangani tugas-tugas visi komputer yang menantang.

Oleh karena itu, pengenalan bahasa isyarat dapat dirancang dengan mengimplementasikan metode Vision Transformer [3]. VIT atau Vision Transformer merupakan arsitektur jaringan saraf tiruan yang dirancang untuk pemrosesan citra atau tugas visi komputer. Saat ini model Transformer mulai digunakan di bidang visi komputer yang sebelumnya menggunakan arsitektur Long Short Term Memory atau LSTM. Perbedaan antara Vision Transformer dengan LSTM adalah cara model tersebut memproses data citra. LSTM dirancang untuk mengatasi pemrosesan data urutan dan memiliki mekanisme bawaan untuk mengingat dan memahami hubungan temporal dalam urutan data seperti citra, sedangkan Vision Transformer lebih baik dalam menggambarkan hubungan spasial dalam frame video dan telah dioptimalkan untuk tugas-tugas pengolahan gambar dalam konteks video. Kelebihan Vision Transformer yaitu memproses citra secara global dan mempelajari representasi visual yang lebih abstrak yang memungkinkan model ini memiliki kinerja yang baik dalam tugas pengenalan bahasa isyarat dari data citra. Dibalik kelebihannya, model ini juga memiliki kelemahan yaitu kebutuhan data latih yang banyak dan komputasi yang lebih berat dibanding LSTM [4].

Ada beberapa rancangan yang sudah dibuat oleh pengarang lain dengan topik “Pengenalan Bahasa Isyarat”, yaitu:

1. Rancangan yang diberi judul "SIGNFORMER: DeepVision Transformer for Sign Language Recognition" dikembangkan oleh Deep R. Kothadiya, Chintan M. Bhatt, Tanzila Saba, dan DR. Amjad Rehman. Dengan fokus pada pelatihan model dengan iterasi yang sedikit, rancangan ini mencapai akurasi sebesar 99.29%[5].
2. Kemudian, ada rancangan lain yang berjudul "Spatiotemporal Convolutions and Video Vision Transformers for Signer-Independent Sign Language Recognition" yang dikembangkan oleh Marc Marais, Dane Brown, James Connan,

dan Alden Boby dari Universitas Rhodes. Meskipun memiliki akurasi yang lebih rendah sebesar 72.19%, rancangan ini menunjukkan kemampuan dalam mendeteksi bahasa isyarat secara independen dari pemberi isyarat [6].

3. Terakhir, rancangan yang diberi judul "HGR-ViT: Hand Gesture Recognition with Vision Transformer" dikembangkan oleh Chun Keat Tan, Kian Ming Lim, Roy Kwang Yang Chang, Chin Poo Lee, dan Ali Alqahtani dari beberapa Universitas. Dengan fokus pada pengenalan gerakan tangan, rancangan ini berhasil mencapai akurasi sebesar 99% [7].

Rancangan yang dibuat merupakan sistem rancangan untuk melakukan pengenalan bahasa isyarat dengan menggunakan Vision Transformer. Data yang digunakan yaitu data video gerakan bahasa isyarat yang dikumpulkan dari website kamus SIBI. Data video yang dikumpulkan akan diproses, dilatih, dievaluasi, serta di uji dengan data baru sehingga model dapat mengenali bahasa isyarat dengan baik. Model yang dibangun dapat menerima masukkan berupa video gerakan kosakata SIBI dan keluaran yang dihasilkan berupa teks kosakata.

2. Dasar Teori

2.1 Rancangan Sistem

Sistem rancangan yang dibuat merupakan sebuah aplikasi desktop sederhana untuk melakukan pengenalan bahasa isyarat formal atau SIBI dengan menggunakan metode Vision Transformer. Alur sistem yang dirancang, pertama melakukan pengumpulan data citra video dengan jumlah 11 kosakata. Kemudian citra yang sudah terkumpul akan digunakan pada pelatihan dan evaluasi model Vision Transformer

Aplikasi desktop dibangun menggunakan bahasa Python dengan berbagai macam library python yang digunakan dan Visual Studio Code sebagai Lingkungan Pengembangan Terintegrasi. Beberapa library python yang akan digunakan dalam perancangan aplikasi desktop ini adalah numpy, keras, tensorflow, dan lain lain. Berikut merupakan alur sistem perancangan aplikasi ini bekerja.

Proses dimulai saat memasukkan citra SIBI, kemudian citra akan dilakukan proses preprocessing seperti mengubah ukuran citra akan diubah kemudian membagi tanda isyarat menjadi serangkaian patch yang akan diubah menjadi vektor 1 dimensi. Setiap vektor 1 dimensi masing - masing patch akan di lakukan proses Linear Projection untuk mereduksi dimensi vektor. Hasil Linear Projection dilanjutkan ke proses Positional Encoding untuk memasukan nilai posisi pada vektor yang kemudian akan dikirim ke blok transformer dengan beberapa lapisan self-attention dan jaringan multilayer perceptron, dan sampai ke tahap model. Model yang sudah terlatih akan disimpan dalam bentuk format model dan akan menampilkan hasil output klasifikasi kosakata.

Model dapat menerima masukkan video atau secara langsung, kemudian masukkan yang diterima akan dilakukan ekstrasi fitur agar bisa diterima model. Model yang disimpan akan dipanggil untuk melakukan tugas klasifikasi bahasa isyarat, keluaran yang dihasilkan berupa teks kosakata.

2.2 Sistem Isyarat Bahasa Indonesia (SIBI)

Sistem Isyarat Bahasa Indonesia atau disingkat SIBI merupakan bahasa isyarat formal yang dibuat berdasarkan America Sign Language. Penulis menggunakan beberapa kosakata dari kamus Sistem Isyarat Bahasa Indonesia (SIBI) yaitu saya, anda, ayah, ibu, makan, minum, lihat, nasi, air, kopi, dan kucing. Kosakata ini akan dilatih dan diuji untuk rancangan aplikasi tersebut [8].

Gambar 1. Contoh Kosakata “Air”

Gambar 2. Contoh Kosakata “Anda”

Gambar 3. Contoh Kosakata “Ayah”

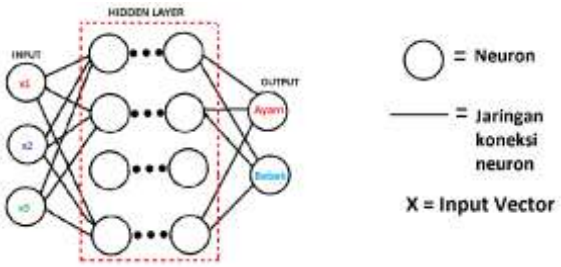
Gambar 4. Contoh Kosakata "Ibu"

Gambar 5. Contoh Kosakata "Kopi"

2.3 Jaringan Saraf Tiruan

Jaringan saraf tiruan merupakan salah satu jenis model untuk pembelajaran mesin. Jaringan saraf tiruan biasa digunakan untuk pengenalan gambar, pemrosesan bahasa alami, dan lain lain [9]. Dasar dari jarangan saraf tiruan ini adalah neuron, dimana setiap neuron saling terhubung satu sama lain layaknya otak manusia dan setiap jaringan neuron memiliki bobot sebagai pengetahuan.

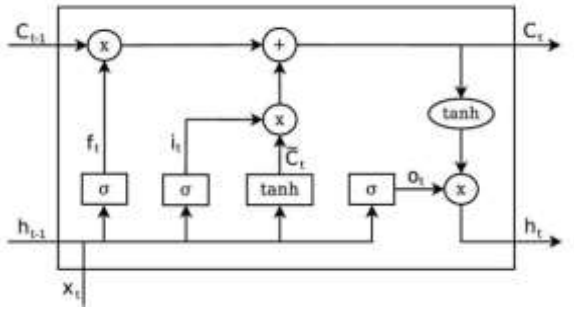
Jaringan saraf tiruan umumnya memiliki tiga lapisan, yaitu lapisan masukkan, lapisan tersembunyi, dan lapisan keluaran. Lapisan masukkan merupakan lapisan awal yang umumnya berisi data masukkan, bentuknya bisa berupa nilai suatu fitur atau piksel gambar. Kemudian pada lapisan tersembunyi data masukkan di ekstrasi dan dapat memahami pola – pola yang rumit. Lapisan keluaran merupakan lapisan akhir yang menampilkan hasil keluaran klasifikasi kelas.



Gambar 6. Jaringan Saraf Tiruan

2.4 Long Short Term Memory (LSTM)

Long Short Term Memory atau LSTM adalah tipe dari jaringan saraf rekuren (RNN) yang memiliki kemampuan untuk menyimpan dan mengakses informasi dalam jangka panjang, sehingga cocok untuk memproses dat berurutan, seperti teks, audio, dan deret waktu. LSTM juga memiliki unit memori yang disebut "cell state" yang memungkinkan mereka untuk mengingat informasi dalam jangka panjang, serta gerbang kontrol yang mengatur aliran informasi dalam dan keluar dari sel LSTM, ini memungkinkan LSTM untuk memahami dan mengingat hubungan jarak jauh dalam data berurutan [11].



Gambar 7. Long Short Term Memory [11]

2.5 Vision Transformer

Vision Transformer merupakan arsitektur jaringan saraf tiruan yang dirancang untuk pemrosesan citra atau tugas visi komputer [4]. Model ini merupakan adaptasi dari Transformer yang pertama kali digunakan dalam pemrosesan bahasa alami. Visual Transformer menerima masukkan berupa citra yang kemudian akan dibagi menjadi beberapa patch atau beberapa bagian kecil dimana setiap patch $(N, P^2 \cdot C)$diubah menjadi vektor (N, D) Setiap patch memiliki nilai piksel seperti nilai intensitas warna, patch umumnya berukuran 16 x 16 piksel. Kemudian patch ini diubah menjadi vektor 1 dimensi yang artinya jika patch berukuran 16 x 16 maka akan menjadi 256 token yang berisi nilai piksel warna. Jika dalam citra memiliki 9 patch maka masing - masing patch akan berisi 256 token sebagai input ke proses selanjutnya. Berikut adalah contoh perhitungan dan langkah - langkahnya :

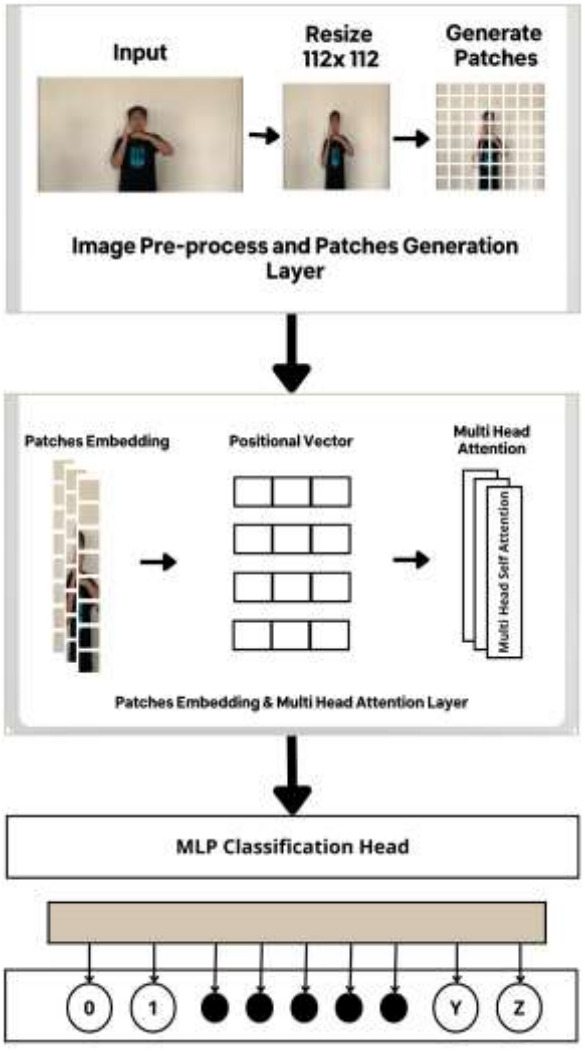
1. Asumsikan citra 2D dengan tinggi (H) dan lebar (W) masing-masing 72x72 piksel, dengan tiga saluran warna (C = 3 untuk citra RGB).
2. Kemudian hitung jumlah total patch (N) yang akan dibuat, diasumsikan bahwa citra akan dibagi menjadi patch berukuran 16x16 piksel. Rumus : $N = (H/P)\ x\ (W/P)$ = (72 / 16) x (72 / 16) = 4 x 4 = 16 patch
3. Setiap patch citra akan diambil dan diubah menjadi vektor. Jadi masing-masing patch akan diubah menjadi vektor dengan dimensi (16 * 16 * 3) = 768, dikali tiga karena memiliki 3 saluran warna (RGB).

4. Hasilnya merupakan representasi urutan 1D dari token embeddings untuk seluruh citra, dan matriks E position akan digabungkan dengan vektor-vektor untuk membentuk representasi akhir.
5. Kemudian vektor – vektor ini dimasukan ke proyeksi linear dengan dikalikan dengan bobot matriks agar dimensi vektor berkurang

Keterangan rumus diatas :

1. X adalah citra.
2. H adalah tinggi.
3. W adalah lebar.
4. C adalah jumlah saluran warna.
5. N adalah adalah jumah total patch.
6. P adalah dimensi dari setiap patch.
7. D adalah dimensi vektor.

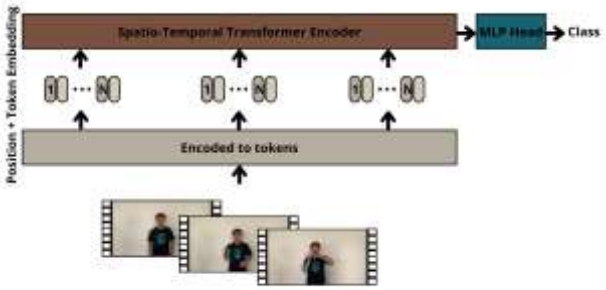
Urutan vektor akan dilanjutkan ke proses Multi-Head Attention dengan aspek yang berbeda, Multi-Head Attention merupakan ekstensi dari Self Attention yang memungkinkan model untuk memproses informasi dari berbagai sudut pandang atau aspek secara bersamaan. Kemudian dilanjutkan ke proses MLP Classification Head untuk mendapatkan hasil kelas [5].



Gambar 8. Algoritma Vision Transformer

Model Vision Transformer memiliki tambahan algoritma bernama Video Vision Transformer (ViViT) yang tugasnya dapat menerima masukkan frame video secara paralel. ViViT mengekstrak token spasial-temporal dari video masukan menggunakan tubelet, yaitu sekelompok frame video yang saling berdekatan. Tubelet adalah sekelompok frame video yang saling berdekatan yang digunakan dalam model ViViT untuk mengubah video masukan menjadi serangkaian token spasial-temporal. Tubelet digunakan untuk mengatasi masalah panjangnya urutan token dalam video dan memungkinkan model untuk mengambil informasi spasial dan temporal dari video secara efisien. Tubelet dapat dihasilkan dengan memilih sekelompok frame video yang saling berdekatan dan membaginya menjadi beberapa segmen. Setiap segmen kemudian dianggap sebagai tubelet dan diubah menjadi serangkaian token menggunakan encoder. Algoritma ini dibuat oleh 5 orang yaitu Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario, Lu˘ci´c† [10], Cordelia Schmid† dari Google Research. Langkah langkah ViViT memproses data yaitu :

1. Input urutan video adalah input ke model ViViT.
2. Ekstraksi Tubelet: Video input dibagi menjadi beberapa tubelet, yang merupakan sekelompok frame video yang berdekatan.
3. Ekstraksi Token: Setiap tubelet diubah menjadi urutan token spasial temporal menggunakan encoder.
4. Pengkodean Token: Token-token ini dikodekan menggunakan beberapa lapisan transformator.
5. Klasifikasi: Representasi video yang dihasilkan dari lapisan transformator terakhir diklasifikasikan menggunakan lapisan terakhir.
6. Output dari model ViViT adalah label kelas yang diprediksi untuk urutan video input.



Gambar 9. Algoritma Video Vision Transformer

3. Hasil Percobaan

Setelah rancangan dan pembuatan dibuat, selanjutnya akan dilakukan pengujian untuk mengecek apakah program yang telah dibuat berjalan dengan baik. Program mengenali citra Sistem Isyarat Bahasa Indonesia (SIBI) dengan input video dan real time gerakan bahasa isyarat.

Bahasa Isyarat yang yang dapat di deteksi yaitu saya, anda, ayah, ibu, makan, minum, lihat, nasi, air, kopi, kucing.

Pengujian model Vision Transformer dalam klasifikasi SIBI dengan cara membuat visualisasi grafik akurasi latih, loss latih, validasi akurasi, dan validasi loss. Apabila terdapat perbandingan antara akurasi dan validasi akurasi maka model Vision Transformer akan di tuning kembali agar lebih optimal dan akan ditambah data latihnya. Ukuran pengujian ini yaitu dengan membagi 2, yaitu train dan validasi dengan ukuran 80 : 20. Kemudian juga menguji modul dengan cara mencoba upload file video, fungsi rekaman webcam.

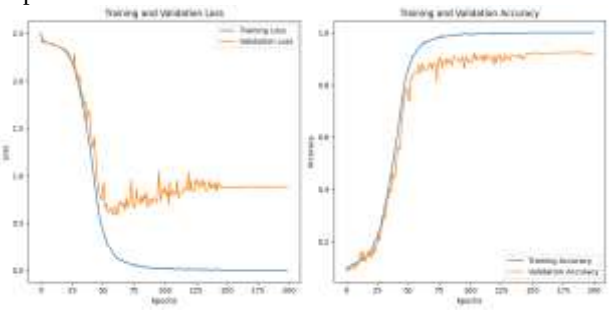
Total data video pada dataset sebanyak 6270 video, data train sebanyak 5020 data, data validasi sebanyak 1250 data. Data dibagi menggunakan library "VideoFrameGenerator". Pada VideoFrameGenerator dilakukan rescale normalisasi dengan membagi 255, mengubah ukuran frame menjadi 112 x 112 piksel, saluran warna abu - abu, dan mengambil 30 frame. Setiap satu gerakan memerlukan 30 frame, hal ini membuat data lebih homogen dan memfasilitasi kemampuan model untuk mempelajari pola gerakan secara efisien dengan menjamin bahwa setiap gerakan diwakili oleh jumlah frame yang sama. Frame yang diambil secara merata dari seluruh durasi video menggunakan metode pengambilan sampel (frame step) yang memastikan interval yang konsisten. Contohnya, jika video memiliki durasi 2 detik dan fps (frame per second) adalah 30 maka total frame adalah 60. Karena setiap satu gerakan memerlukan 30 frame maka dapat dihitung frame step dengan rumus (total durasi video // (jumlah frame dibutuhkan -1) = (60 // 29) = 2 . Dengan nilai frame step adalah 2 maka frame akan diambil 2 langkah dari frame pertama hingga frame akhir, contoh frame ke-1, frame ke-3, frame ke-5, sampai frame ke-59 (total = 30 frame).

Model ViViT dilatih sebanyak 200 epochs dan dilatih dengan berbagai macam ukuran batch size, yaitu ukuran batch size 8, 16, 32, 64, dan 128. Model menerima masukkan frame sebanyak 30 frame, ukuran frame sebesar 112 x 112 dan saluran warna abu - abu agar proses latihan model lebih ringan dan mendapatkan akurasi yang optimal. Setelah model dilatih akan dievaluasi dengan metode learning curve atau memantau proses pelatihan dengan membandingkan loss latih dan akurasi latih dengan loss validasi dan akurasi validasi. Berikut adalah tabel dari tren loss latih, akurasi latih, loss validasi, dan akurasi validasi dari berbagai macam ukuran batch size.

Tabel 1. Hasil pelatihan model

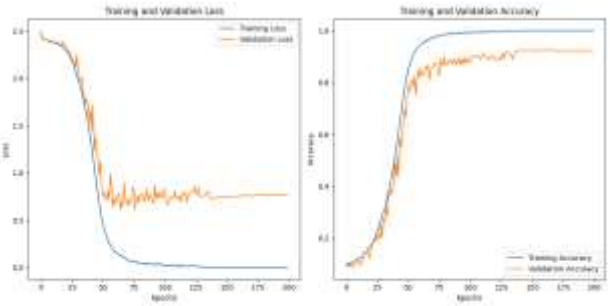
Batch size	Epoch	Akurasi latih	Loss latih	Akurasi validasi	Loss validasi
8	200	100%	0.0000009	92.23%	0.8743
16	200	100%	0.000002	92.23%	0.7725
32	200	100%	0.000008	90.79%	0.9006
64	200	99.32%	0.0276	91.53%	0.5964
128	200	98.68%	0.0487	90.97%	0.5589

Pada tabel diatas menunjukkan bahwa model dilatih dengan berbagai penyetelan batch size untuk membandingkan hasil akurasi latih dengan akurasi validasi, loss latih dengan loss validasi dan dilatih sebanyak 200 epochs. Hasil tabel diatas dapat divisualisasikan dengan tools matplotlib untuk mendapatkan insight lebih baik, grafik validasi dan latih dapat di lihat bawah ini.



Gambar 10. Grafik learning curve model *batch size* 8

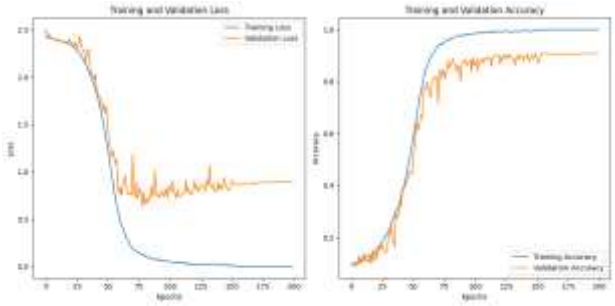
Grafik di atas menunjukkan adanya perbedaan jarak yang cukup signifikan antara loss latih dan loss validasi, di mana nilai loss latih sebesar 0.0000009 dan loss validasi sebesar 0.8743. Pada epoch ke-75, model mulai konvergen pada loss validasi, yang berarti bahwa nilai loss validasi telah mencapai titik stabil dan tidak menunjukkan perubahan signifikan lagi. Sementara itu, perbedaan antara akurasi latih yang mencapai 100% dan akurasi validasi 92.23% yang tidak terlalu signifikan menunjukkan bahwa model ini cukup baik dalam memprediksi gerakan bahasa isyarat.



Gambar 11. Grafik learning curve model *batch size* 16

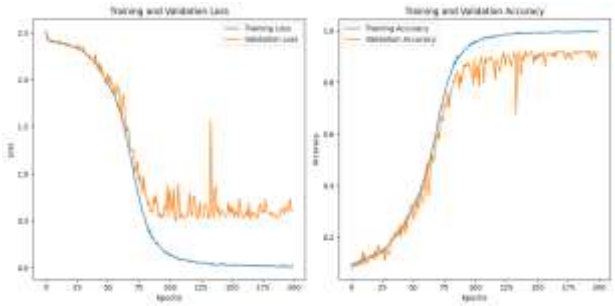
Pada pengujian selanjutnya, diperoleh grafik dengan nilai loss latih sebesar 0.000002 dan loss validasi sebesar

0.7725. Sama seperti sebelumnya pada epoch ke-75, model mulai konvergen pada loss validasi, yang menunjukkan stabilitas. Perbedaan antara akurasi latih 100% dan akurasi validasi 92.23% juga tidak terlalu signifikan, yang kembali menunjukkan bahwa model cukup andal dalam memprediksi gerakan bahasa isyarat.



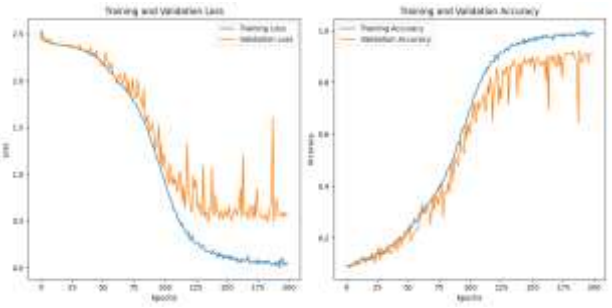
Gambar 12. Grafik learning curve model *batch size* 32

Grafik lain menunjukkan perbedaan jarak yang cukup signifikan antara loss latih dan loss validasi, dengan nilai loss latih sebesar 0.000008 dan loss validasi sebesar 0.9006. Pada epoch ke-100, model mulai konvergen pada loss validasi, yang berarti nilai loss validasi telah mencapai stabilitas. Perbedaan antara akurasi latih 100% dan akurasi validasi 90.79% yang tidak terlalu signifikan juga mengindikasikan bahwa model ini cukup baik dalam memprediksi gerakan bahasa isyarat.



Gambar 13. Grafik learning curve model *batch size* 64

Pada pengujian lainnya, grafik menunjukkan nilai loss latih sebesar 0.0276 dan loss validasi sebesar 0.5964. Pada epoch ke-100, model mulai konvergen pada loss validasi, yang menunjukkan stabilitas nilai loss validasi. Perbedaan antara akurasi latih 99.32% dan akurasi validasi 91.53% yang tidak terlalu signifikan mengindikasikan bahwa model cukup andal dalam memprediksi gerakan bahasa isyarat.



Gambar 14. Grafik learning curve model *batch size* 128

Terakhir, grafik lain menunjukkan nilai loss latih sebesar 0.0487 dan loss validasi sebesar 0.5589. Pada epoch ke-125, model mulai konvergen pada loss validasi, yang berarti nilai loss validasi telah stabil. Perbedaan antara akurasi latih 98.68% dan akurasi validasi 90.97% yang tidak terlalu signifikan menunjukkan bahwa model ini cukup baik dalam memprediksi gerakan bahasa isyarat.

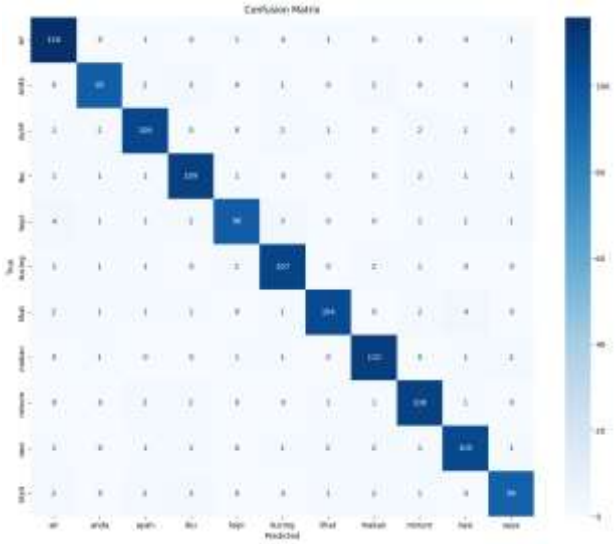
Perbedaan ukuran batch size memiliki dampak signifikan terhadap berbagai aspek dalam proses pelatihan model, termasuk kecepatan konvergensi, waktu proses, akurasi, dan nilai loss. Ukuran batch size yang lebih kecil seperti 8, biasanya menghasilkan update parameter yang lebih sering karena gradien lebih sering dihitung. Hal ini dapat mempercepat konvergensi awal model, tetapi juga membuat proses pelatihan lebih lambat secara keseluruhan karena waktu iterasi per batch yang lebih panjang. Sebaliknya, ukuran batch size yang lebih besar seperti 128, cenderung menghasilkan update parameter yang lebih jarang namun lebih stabil. Hal ini dapat mempercepat waktu per iterasi dan mengurangi total waktu pelatihan, tetapi mungkin membutuhkan lebih banyak epochs untuk mencapai konvergensi.

Tabel 2. Waktu Pelatihan Per-iterasi

Batch Size	Epochs	Rata - rata waktu per-iterasi (detik)	Total Waktu Perlatihan
8	200	52	3 jam
16	200	38	2 jam
32	200	32	2 jam
64	200	31	2 jam
128	200	28	2 jam

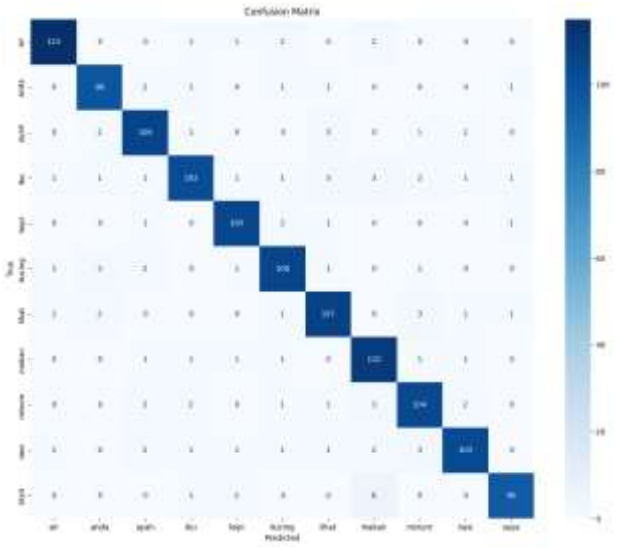
Lamanya proses pelatihan sangat penting karena pelatihan yang terlalu lama dapat menghabiskan waktu dan sumber daya komputasi tanpa menghasilkan peningkatan kinerja yang baik.

Metode Confusion Matrix digunakan untuk melihat seberapa baik model dalam memprediksi target. Confusion Matrix merupakan grafik evaluasi dalam kasus klasifikasi dengan memberikan gambaran menyeluruh tentang hasil prediksi model. Pada dasarnya matriks ini memiliki 4 nilai presentase, yaitu True Positive, True Negative, False Positive, dan False Negative. Ketika model secara akurat memprediksi sampel kelas positif sebagai positif dikenal sebagai True Positive, ketika sampel kelas negatif secara keliru diprediksi oleh model sebagai positif dikenal sebagai False Positive, ketika model secara akurat memprediksi sampel kelas negatif sebagai negatif dikenal sebagai True Negative, dan ketika sampel kelas positif secara keliru diprediksi oleh model sebagai negatif dikenal sebagai False Positive. Berikut merupakan grafik Confusion Matrix dari beberapa batch size model.



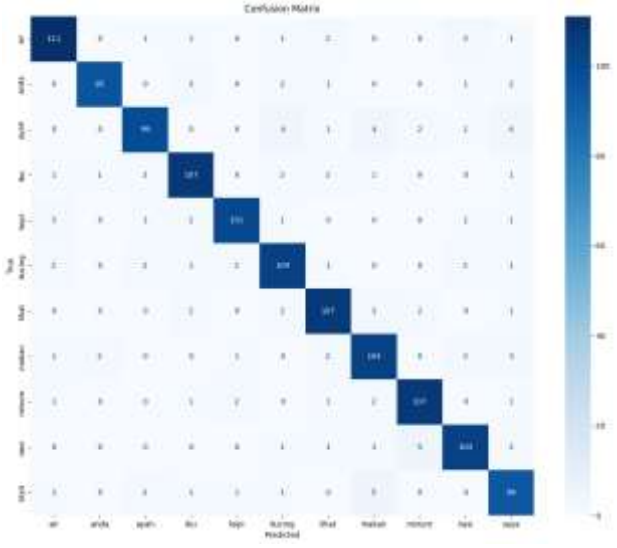
Gambar 15. Grafik confusion matrix model batch size 8

Hasil Confusion Matrix menunjukkan bahwa model berhasil memprediksi kata "air" dengan benar sebanyak 116 kali, kata "anda" sebanyak 95 kali, kata "ayah" sebanyak 106 kali, kata "ibu" sebanyak 109 kali, kata "kopi" sebanyak 96 kali, kata "kucing" sebanyak 107 kali, kata "lihat" sebanyak 104 kali, kata "makan" sebanyak 110 kali, kata "minum" sebanyak 108 kali, kata "nasi" sebanyak 105 kali, dan kata "saya" sebanyak 94 kali. Pada pengujian ini, kata "saya" adalah yang paling sedikit benar untuk model dengan ukuran batch size 8.



Gambar 16. Grafik confusion matrix model *batch size* 16

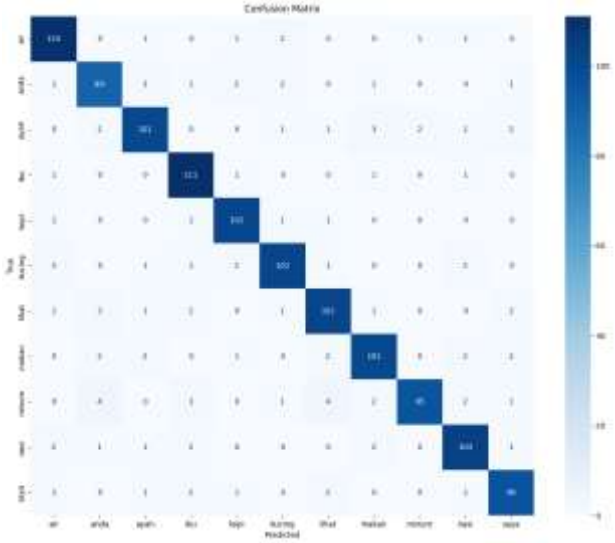
Sementara itu, pada pengujian selanjutnya, model dengan ukuran batch size 16 menunjukkan hasil yang sedikit berbeda. Model berhasil memprediksi kata "air" sebanyak 115 kali, kata "anda" sebanyak 98 kali, kata "ayah" sebanyak 106 kali, kata "ibu" sebanyak 102 kali, kata "kopi" sebanyak 105 kali, kata "kucing" sebanyak 106 kali, kata "lihat" sebanyak 107 kali, kata "makan" sebanyak 110 kali, kata "minum" sebanyak 104 kali, kata "nasi" sebanyak 103 kali, dan kata "saya" sebanyak 96 kali. Gerakan kata "saya" masih merupakan yang paling sedikit benar untuk model ini.



Gambar 17. Grafik confusion matrix model *batch size* 32

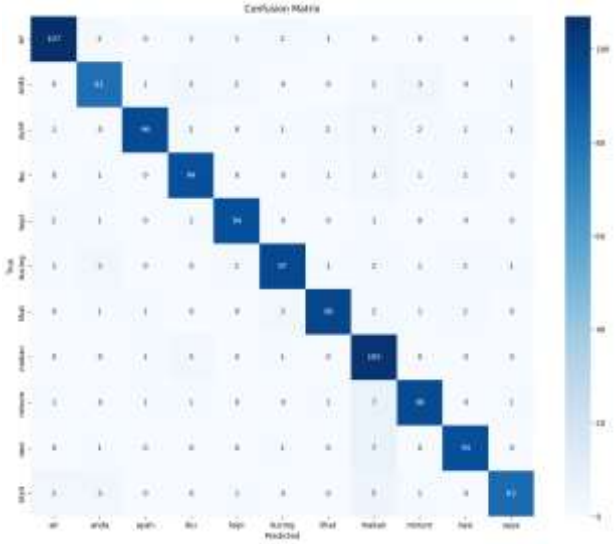
Pada grafik lain, diperoleh hasil untuk model dengan ukuran batch size 32. Model berhasil memprediksi kata "air" sebanyak 111 kali, kata "anda" sebanyak 95 kali, kata "ayah" sebanyak 99 kali, kata "ibu" sebanyak 107 kali, kata "kopi" sebanyak 101 kali, kata "kucing"

sebanyak 104 kali, kata "lihat" sebanyak 107 kali, kata "makan" sebanyak 104 kali, kata "minum" sebanyak 107 kali, kata "nasi" sebanyak 104 kali, dan kata "saya" sebanyak 94 kali. Di sini, gerakan kata "saya" kembali menjadi yang paling sedikit benar.



Gambar 18. Grafik confusion matrix model *batch size* 64

Pada pengujian selanjutnya dengan ukuran batch size 64, model berhasil memprediksi kata "air" sebanyak 110 kali, kata "anda" sebanyak 89 kali, kata "ayah" sebanyak 101 kali, kata "ibu" sebanyak 111 kali, kata "kopi" sebanyak 102 kali, kata "kucing" sebanyak 102 kali, kata "lihat" sebanyak 102 kali, kata "makan" sebanyak 101 kali, kata "minum" sebanyak 95 kali, kata "nasi" sebanyak 104 kali, dan kata "saya" sebanyak 96 kali. Kali ini, gerakan kata "anda" adalah yang paling sedikit benar.



Gambar 19. Grafik confusion matrix model *batch size* 128

Terakhir, hasil Confusion Matrix untuk model dengan ukuran batch size 128 menunjukkan bahwa model berhasil memprediksi kata "air" sebanyak 107 kali, kata "anda" sebanyak 81 kali, kata "ayah" sebanyak 96 kali, kata "ibu" sebanyak 94 kali, kata "kopi" sebanyak 94 kali, kata "kucing" sebanyak 97 kali, kata "lihat" sebanyak 96 kali, kata "makan" sebanyak 105 kali, kata "minum" sebanyak 96 kali, kata "nasi" sebanyak 94 kali, dan kata "saya" sebanyak 83 kali.

Analisis hasil Confusion Matrix menunjukkan bahwa kata "air" dan "makan" memiliki tingkat pengenalan yang sangat tinggi di semua ukuran batch size. Misalnya, kata "air" berhasil dikenali dengan benar sebanyak 116, 115, 111, 110, dan 107 kali untuk batch size 8, 16, 32, 64, dan 128 secara berturut-turut. Sebaliknya, kata "saya" dan "anda" cenderung memiliki tingkat pengenalan yang lebih rendah. Kata "saya" dikenali dengan benar sebanyak 94, 96, 94, 96, dan 83 kali untuk batch size yang sama. Kata "anda" menunjukkan penurunan pengenalan terutama pada batch size yang lebih besar, dengan prediksi benar sebanyak 89 dan 81 kali untuk batch size 64 dan 128.

Ada 1 metode evaluasi untuk mengukur kualitas model, yaitu dengan menghitung akurasi, presisi, recall, dan f1-score. Akurasi merupakan persentase prediksi benar dari total prediksi yang di lakukan oleh model. Presisi mengukur seberapa banyak prediksi positif yang benar dari semua prediksi positif yang dibuat oleh model. Recall mengukur seberapa banyak prediksi positif yang benar dari semua kasus positif aktual yang ada dalam data. F1-score adalah rata-rata harmonis dari presisi dan recall, memberikan keseimbangan antara keduanya dan memberikan gambaran keseluruhan kinerja model dalam menangani ketidakseimbangan data. Berikut tabel metriks evaluasi dibawah ini.

Tabel 3. Nilai akurasi, presisi, recall, F1-score

Batch size	Akurasi	Presisi	Recall	F1 Score
8	92.14%	92.20%	92.14%	92.13%
16	92.30%	92.38%	92.30%	92.31%
32	90.78%	90.88%	90.78%	90.80%
64	91.52%	91.59%	91.52%	91.51%
128	90.71%	91.14%	90.70%	90.78%

Dari tabel di atas, jelas terlihat bahwa ukuran batch berdampak pada performa model. Dari semua ukuran batch size yang ada, ukuran batch size 16 memberikan hasil terbaik dengan akurasi 92.30%, presisi 92.38%, recall 92.30%, dan f1-score 92.31%. Ukuran batch size 8 dan 64 juga memberikan performa yang bagus, ukuran batch size 32 dan 128 memiliki performa akurasi, presisi, recall, dan f1-score terendah.

4. Kesimpulan

Model dibangun dengan beberapa ukuran batch 8, 16, 32, 64, 128 dan dilatih sebanyak 200 epoch untuk mendeteksi 11 gerakan SIBI. Proses pelatihan model berbeda - beda dari segi waktu latih, grafik learning curve, confusion matrix, dan metriks evaluasi (akurasi, presisi, recall, dan f1-score). Waktu pelatihan model tercepat yaitu model dengan ukuran batch 128 dengan rata - rata waktu 28 detik / iterasi, batch 64 dengan rata - rata waktu 31 detik / iterasi, batch 32 dengan rata - rata waktu 32 detik / iterasi, batch 16 dengan rata - rata waktu 38 detik / iterasi, dan batch 8 dengan rata - rata waktu 52 detik / iterasi. Dari kelima model yang proses learning curve cukup baik adalah model dengan ukuran batch size 64 penurunan loss latih dan loss validasi turun lebih stabil dan jarak antara akurasi latih dan akurasi validasi tidak jauh. Dari semua ukuran batch size yang ada, ukuran batch size 16 memberikan hasil terbaik dengan akurasi 92.30%, presisi 92.38%, recall 92.30%, dan f1-score 92.31%. Ukuran batch size 8 dan 64 juga memberikan performa yang bagus, ukuran batch size 32 dan 128 memiliki performa akurasi, presisi, recall, dan f1-score terendah.

REFERENSI

[1] M. Sholawati, K. Auliasari, dan FX. Ariwibisono, "PENGEMBANGAN APLIKASI PENGENALAN BAHASA ISYARAT ABJAD SIBI MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK (CNN)," JATI (Jurnal Mahasiswa Teknik Informatika), vol. 6, no. 1, pp. 134–144, Mar. 2022, doi: https://doi.org/10.36040/jati.v6i1.4507.

[2] M. Bagus, S. Bakti, dan Y. M. Pranoto, "Pengenalan Angka Sistem Isyarat Bahasa Indonesia Dengan Menggunakan Metode Convolutional Neural Network," Prosiding SEMNAS INOTEK (Seminar Nasional Inovasi Teknologi), vol. 3, no. 1, pp. 011–016, 2019, doi: https://doi.org/10.29407/inotek.v3i1.504.

[3] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, dan N. Houlsby, "AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE," Jun. 2021. Available: https://arxiv.org/pdf/2010.11929.pdf

[4] Y. Liu, Y. Zhang, Y. Wang, F. Hou, J. Yuan, J. Tian, Y. Zhang, Z. Shi, J. Fan, dan Z. He, "A Survey of Visual Transformers," Dec. 2022. Accessed: Jun. 03, 2023. [Online]. Available: https://arxiv.org/pdf/2111.06091.pdf

[5] D. R. Kothadiya, C. M. Bhatt, T. Saba, A. Rehman, dan S. A. Bahaj, "SIGNFORMER: DeepVision Transformer for Sign Language Recognition," IEEE Access, vol. 11, pp. 4730–4739, 2023, doi: https://doi.org/10.1109/access.2022.3231130.

[6] M. Marais, D. Brown, J. Connan, dan A. Boby, "Spatiotemporal Convolutions and Video Vision Transformers for Signer-Independent Sign Language Recognition," 2023 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD), Aug. 2023, doi: https://doi.org/10.1109/icabcd59051.2023.10220534.

[7] C. K. Tan, K. M. Lim, R. K. Y. Chang, C. P. Lee, dan A. Alqahtani, "HGR-ViT: Hand Gesture Recognition with Vision Transformer," Sensors, vol. 23, no. 12, p. 5555, Jan. 2023, doi: https://doi.org/10.3390/s23125555.

[8] "Kamus SIBI," pmpk.kemdikbud.go.id. https://pmpk.kemdikbud.go.id/sibi/profil

[9] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, dan H. Arshad, "State-of-the-art in artificial neural network applications: A survey," Heliyon, vol. 4, no. 11, p. e00938, Nov. 2018, doi: https://doi.org/10.1016/j.heliyon.2018.e00938.

[10] Anurag Arnab, M. Dehghani, Georg Heigold, C. Sun, M. Lucic, dan C. Schmid, "ViViT: A Video Vision Transformer," Mar. 2021, doi: https://doi.org/10.48550/arxiv.2103.15691.

[11] S. Mekruksavanich dan A. Jitpattanakul, "LSTM Networks Using Smartphone Data for Sensor-Based Human Activity Recognition in Smart Homes," Sensors, vol. 21, no. 5, p. 1636, Feb. 2021, doi: https://doi.org/10.3390/s21051636.

Renaldy, mahasiswa pada program studi Fakultas Teknologi Informasi di Universitas Tarumanagara

Agus Budi Dharmawan S.kom, M.T., M.sc, memperoleh gelar M.T dari ITS Surabaya. Kemudian memperoleh gelar M.Sc dari Elektronic Engenering FH Darmstad. Saat ini sebagai Dosen Program studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Tarumanagara.