APLIKASI *FEEDFORWARD NEURAL NETWORK* UNTUK KLASIFIKASI VARIAN SARS-COV-2 BERDASARKAN SEKUENS DNA NUKLEOTIDA

SKRIPSI

OLEH
MUTIARA ALFI
NIM. 220601110071

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2026

APLIKASI *FEEDFORWARD NEURAL NETWORK* UNTUK KLASIFIKASI VARIAN SARS-COV-2 BERDASARKAN SEKUENS DNA NUKLEOTIDA

SKRIPSI

Diajukan Kepada
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
untuk Memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Matematika (S.Mat)

Oleh
Mutiara Alfi
NIM. 220601110071

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2026

APLIKASI *FEEDFORWARD NEURAL NETWORK* UNTUK KLASIFIKASI VARIAN SARS-COV-2 BERDASARKAN SEKUENS DNA NUKLEOTIDA

SKRIPSI

Oleh
Mutiara Alfi
NIM. 220601110071

Telah Disetujui untuk Diuji

Malang, 20 Mei 2026

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Mohammad Jamhuri, M.Si	Dr. Abdussakir, M.Pd
NIP. 19810502 200501 1 004	NIP. 19751006 200312 1 001

Mengetahui,
Ketua Program Studi Matematika

Dr. Fachrur Rozi, M.Si
NIP. 19800527 200801 1 012

APLIKASI *FEEDFORWARD NEURAL NETWORK* UNTUK KLASIFIKASI VARIAN SARS-COV-2 BERDASARKAN SEKUENS DNA NUKLEOTIDA

SKRIPSI

Oleh
Mutiara Alfi
NIM. 220601110071

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Matematika (S.Mat)

Tanggal 12 Juni 2026

Ketua Penguji	: Ari Kusumastuti, M.Pd., M.Si	.....
Anggota Penguji 1	: Juhari, M.Si	.....
Anggota Penguji 2	: Dr. Mohammad Jamhuri, M.Si	.....
Anggota Penguji 3	: Dr. Abdussakir, M.Pd	.....

Mengetahui,
Ketua Program Studi Matematika

Dr. Fachrur Rozi, M.Si
NIP. 19800527 200801 1 012

PERNYATAAN KEASLIAN TULISAN

Saya bertanda tangan di bawah ini

Nama : Mutiara Alfi

NIM : 220601110071

Program Studi : Matematika

Fakultas : Sains dan Teknologi

Judul Skripsi : Aplikasi *Feedforward Neural Network* untuk Klasifikasi Varian SARS-CoV-2 Berdasarkan Sekuens DNA Nukleotida

Menyatakan bahwa skripsi yang saya susun merupakan hasil karya sendiri dan tidak mengambil atau menyalin tulisan maupun pemikiran orang lain. Seluruh isi skripsi ini adalah hasil pemikiran saya, kecuali bagian yang telah disebutkan sumbernya dalam daftar rujukan pada halaman terakhir. Apabila di kemudian hari terbukti bahwa skripsi ini merupakan hasil plagiasi atau meniru karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 12 Juni 2026

Mutiara Alfi

NIM. 220601110071

MOTO

"Sesungguhnya sesudah kesulitan itu ada kemudahan, maka apabila kamu telah selesai (dari satu urusan), tetaplah bekerja keras (untuk urusan yang lain)."

(Q.S Al-Insyirah: 6-7)

PERSEMBAHAN

Bismillahirrahmanirrahim

Skripsi ini penulis persembahkan kepada:

Kedua orang tua tercinta, Ayah Moh Nafe' dan Ibu Ninik, yang dengan ketulusan, kesabaran, dan kasih sayang tidak terhitung telah membimbing penulis. Terima kasih atas setiap doa yang dipanjatkan, setiap nasihat yang diberikan, setiap pengorbanan yang tidak ternilai mengiringi langkah penulis, serta keyakinan yang tidak pernah pudar terhadap impian penulis hingga dimampukan meraih gelar sarjana.

Kakak tersayang Mar'atur Roikha dan Aizatul Aini, yang menjadi bagian penting dalam perjalanan penulis dengan selalu memberikan dukungan, dan semangat, hingga penulis dimampukan untuk menyelesaikan skripsi ini.

KATA PENGANTAR

Segala puji bagi Allah SWT atas limpahan rahmat, taufik, serta hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi ini yang berjudul "Aplikasi *Feedforward Neural Network* untuk Klasifikasi Varian SARS-CoV-2 Berdasarkan Sekuens DNA Nukleotida" sebagai salah satu syarat untuk memperoleh gelar sarjana pada Program Studi Matematika Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Dalam proses penyusunan skripsi ini, penulis banyak memperoleh dukungan, bimbingan, dan bantuan dari berbagai pihak, baik secara langsung maupun tidak langsung. Oleh karena itu, penulis menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

1. Prof. Dr. Hj. Ilfi Nur Diana, M.Si., CAHRM., CRMP selaku Rektor Universitas Islam Negeri Maulana Malik Ibrahim.
2. Dr. Agus Mulyono, M.Kes., selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim.
3. Dr. Fachrur Rozi, M.Si., selaku Ketua Program Studi Matematika, Universitas Islam Negeri Maulana Malik Ibrahim.
4. Dr. Mohammad Jamhuri, M.Si., selaku Dosen Pembimbing I yang telah memberikan berbagai ilmu, bimbingan, saran, motivasi, dan arahan kepada penulis dalam proses penyusunan skripsi ini.
5. Dr. Abdussakir, M.Pd., selaku Dosen Pembimbing II yang telah memberikan ilmu, bimbingan, saran, dan arahan kepada penulis dalam pengintegrasian nilai-nilai keislaman pada skripsi ini.
6. Ari Kusumastuti, M.Pd., M.Si., selaku Ketua Penguji yang telah berkenan memberikan arahan serta saran yang membangun untuk penyusunan skripsi.
7. Juhari, M.Si., selaku Anggota Penguji I yang telah memberikan arahan dan saran yang bermanfaat kepada penulis dalam penyusunan skripsi.
8. Seluruh Dosen Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim.
9. Cinta pertama, Ayah Moh Nafe' dan pintu surga Ibu Ninik Fardah, yang senantiasa menjadi fondasi utama dalam setiap langkah kehidupan penulis. Izinkan penulis mengucapkan terima kasih yang sebesar-besarnya, atas segala doa yang tiada henti, kesabaran, keikhlasan, kasih sayang, serta dukungan

yang tulus. Terima kasih telah berjuang bersama, mengorbankan waktu, tenaga, dan upaya demi mendukung penulis menyelesaikan studinya sampai meraih gelar sarjana. Sehat selalu dan panjang umur karena ayah dan ibu harus selalu ada di setiap perjuangan dan pencapaian hidup penulis.

10. Kakak pertama penulis, Mar'atur Roikha dan suaminya Hanif Risdianto. Terima kasih banyak atas dukungannya secara moril maupun materil yang dengan tulus diberikan. Selalu menyediakan waktu dan materi saat penulis sangat membutuhkannya, serta menjadi contoh dalam setiap fase kehidupan penulis. Di saat penulis merasa ragu, kata-kata dari kakak yang mampu mengembalikan semangat yang sempat hilang. Terima kasih sudah menjadi kakak yang baik dan selalu ada.
11. Kakak kedua penulis, Aizatul Aini yang tiada hentinya memberikan perhatian, motivasi, dukungan dan doa serta menjadi contoh dalam setiap fase kehidupan penulis. Dengan penuh keikhlasan telah membantu orang tua untuk memenuhi kebutuhan dan fasilitas penulis tanpa mengeluh sedikit pun. Di saat penulis merasa ragu, kata-kata dari kakak yang mampu mengembalikan semangat yang sempat hilang. Terima kasih sudah menjadi kakak yang baik dan selalu ada.
12. Keluarga tercinta, khususnya Nenek Sunarsih. Terima kasih yang dengan tulus memberikan perhatian, kasih sayang, doa, materi dan dukungan yang tiada hentinya. Kehadiran nenek menjadi sumber motivasi dan kekuatan bagi penulis dalam menjalani setiap fase kehidupan, termasuk dalam menyelesaikan studi ini. Keluarga lainnya yang telah memberikan dukungan, kasih sayang, dan doa.
13. Seseorang yang tak kalah penting kehadirannya, Zidan Ramadhan Yulian yang telah menjadi bagian dari perjalanan penulis. Terima kasih telah berkontribusi banyak baik tenaga, waktu, maupun materi kepada penulis. Telah menjadi rumah paling tenang di antara keramaian, menjadi telinga paling sabar di antara banyak yang memilih bungkam, yang menemani, menghibur dalam kesedihan serta selalu memberi semangat untuk penulis pantang menyerah. Terima kasih telah berjuang bersama sampai detik ini.

14. Teman-teman seperjuangan yaitu Ayu Habibatun Nurwahyu Zainuri, Amira Adelia Putri, Rossima Eva Yuliana, Nurus Syafi'ah, dan Do'aul Isma Mufidah yang sudah setia menemani, memberikan semangat, saran, dan dukungan selama proses penyusunan skripsi ini. Terima kasih atas hiburan untuk penulis agar tetap menjalani perkuliahan dengan rasa bahagia. Kehadiran kalian bukan hanya mengisi hari-hari selama perkuliahan, tetapi juga meninggalkan jejak hangat. Kalian bukan sekadar teman, tetapi bagian dari perjalanan yang akan selalu hidup dalam ingatan penulis.
15. Seluruh mahasiswa Program Studi Matematika angkatan 2022.

Semoga segala bentuk dukungan yang diberikan kepada penulis dari semua pihak mendapat balasan rahmat dan kebaikan dari Allah SWT, serta menjadi amal yang memberikan manfaat di dunia maupun di akhirat.

Malang, 12 Juni 2026

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR SIMBOL

D	: *Dataset* pelatihan
n	: Banyak data
d	: Banyak fitur *input*
m	: Banyak neuron pada *hidden layer*
p	: Banyak kelas *output*
$\mathbf{x}_i$	: Vektor fitur data ke-i
$\mathbf{W}$	: Matriks bobot dari lapisan *input* ke *hidden*
$\mathbf{b}$	: Vektor bias pada *hidden layer*
$\mathbf{a}_i$	: Hasil transformasi linier pada *hidden layer*
$\mathbf{z}_i$	: Keluaran *hidden layer* setelah fungsi ReLU
$\mathbf{V}$	: Matriks bobot dari *hidden* ke *output*
$\mathbf{c}$	: Vektor bias pada *output layer*
$\mathbf{s}_i$	: Masukan total pada *output layer*
$\mathbf{o}_i$	: Vektor probabilitas hasil *softmax*
$\hat{y}_i$	: Hasil prediksi kelas data ke-i
$\mathbf{y}_i$	: Label target (*one-hot encoding*)
E	: *Loss* rata-rata
L_i	: *Loss* untuk data ke-i
$\boldsymbol{\delta}_i$	: Sinyal kesalahan pada *output layer*
$\boldsymbol{\gamma}_i$	: Sinyal kesalahan pada *hidden layer*
$\nabla_{\mathbf{W}} L_i$	: Gradien *loss* terhadap bobot $\mathbf{W}$
$\nabla_{\mathbf{V}} L_i$	: Gradien *loss* terhadap bobot $\mathbf{V}$
$\nabla_{\mathbf{b}} L_i$	: Gradien *loss* terhadap bias $\mathbf{b}$
$\nabla_{\mathbf{c}} L_i$	: Gradien *loss* terhadap bias $\mathbf{c}$
η	: *Learning rate*
X_{new}	: Nilai fitur setelah normalisasi
X_{min}	: Nilai minimum fitur
X_{max}	: Nilai maksimum fitur
M_i	: Nilai metrik evaluasi kelas ke-i
$Support_i$	: Banyak data aktual pada kelas ke-i

DAFTAR LAMPIRAN

ABSTRAK

Alfi, Mutiara. 2026. **Aplikasi *Feedforward Neural Network* untuk Klasifikasi Varian SARS-CoV-2 Berdasarkan Sekuens DNA Nukleotida**. Skripsi. Program Studi Matematika. Fakultas Sains dan Teknologi. Universitas Islam Negeri Maulana Malik Ibrahim Malang.
Pembimbing: (I) Dr. Mohammad Jamhuri, M.Si. (II) Dr. Abdussakir, M.Pd.

Kata Kunci: *Feedforward Neural Network*, Klasifikasi, *K*-Mer, SARS-CoV-2, Sekuens cDNA, *Variants of Concern*.

Klasifikasi varian SARS-CoV-2 diperlukan untuk membantu mengidentifikasi perbedaan pola genom antarvarian, khususnya pada varian yang termasuk dalam kategori *Variants of Concern* (VoC). Penelitian ini bertujuan untuk menerapkan algoritma *Feedforward Neural Network* (FFNN) dalam mengklasifikasikan varian SARS-CoV-2 berdasarkan sekuens nukleotida hasil representasi *complementary* DNA (cDNA). Data yang digunakan berupa *complete genome* SARS-CoV-2 dari basis data NCBI dengan lima kelas varian, yaitu Alpha, Beta, Delta, Gamma, dan Omicron. *Dataset* yang digunakan didominasi oleh sekuens dengan *host* manusia, dengan sebagian kecil data berasal dari *host* lain. Tahapan penelitian meliputi *preprocessing* data, ekstraksi fitur menggunakan metode k-mer, normalisasi data menggunakan *Min-Max Scaling*, pembagian data menjadi data *training*, *validation*, dan *testing*, serta klasifikasi menggunakan algoritma FFNN. Evaluasi model dilakukan menggunakan *confusion matrix*, *classification report*, dan waktu komputasi. Hasil pengujian pada konfigurasi utama menunjukkan bahwa model FFNN memperoleh nilai akurasi, presisi, *recall*, dan F1-*score* sebesar 0,9983 dalam waktu pelatihan selama 8,7880 detik. Hasil tersebut menunjukkan bahwa FFNN berpotensi efektif diterapkan dalam klasifikasi multikelas varian SARS-CoV-2 berdasarkan pola sekuens nukleotida hasil representasi cDNA.

ABSTRACT

Alfi, Mutiara. 2026. **Application of Feedforward Neural Network for Classification of SARS-CoV-2 Variants Based on Nucleotide DNA Sequences**. Undergraduate Thesis. Mathematics Study Program, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang.
Advisors: (I) Dr. Mohammad Jamhuri, M.Si. (II) Dr. Abdussakir, M.Pd.

Keywords: Feedforward Neural Network, Classification, K-Mer, SARS-CoV-2, cDNA Sequences, Variants of Concern.

Classification of SARS-CoV-2 variants is essential for identifying differences in genomic patterns between variants, especially those categorized as Variants of Concern (VoC). This study aims to apply the Feedforward Neural Network (FFNN) algorithm to classify SARS-CoV-2 variants based on nucleotide sequences obtained from complementary DNA (cDNA) representation. The data used consists of the complete genome of SARS-CoV-2 sourced from the NCBI database, including five variant categories: Alpha, Beta, Delta, Gamma, and Omicron. The collection is primarily made up of sequences from human hosts, with only a minor amount coming from other hosts. The research process involves stages such as data preprocessing, feature extraction through the k-mer technique, normalization of data using Min-Max Scaling, partitioning data into training, validation, and testing sets, along with classification using the FFNN algorithm. The model's performance was assessed using a confusion matrix, classification report, and computing time. The findings indicate that the FFNN algorithm is capable of classifying SARS-CoV-2 variants with outstanding performance. The model reached an accuracy, precision, recall, and F1-score values of $0{,}9983$, completing training $8{,}7880$ seconds. These results indicate that FFNN has the potential to be effectively applied in the multiclass classification of SARS-CoV-2 variants based on nucleotide sequence patterns from cDNA representation.

مستخلص البحث

ألفي, موتيارا.٢٠٢٦. **تطبيق الشبكة العصبية ذات التغذية الأمامية لتصنيف متحورات سارس-كوف-٢ ب ناءً على تسلسلات النيوكليوتيدات في الحمض النووي**. البحث العلمي. قسم الرياضيات,كلية العلوم والتكنولوجيا .جامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج

المشرف :)١ (د. محمد جمهوري, الماجستير في العلوم)٢ (د. عبد الشكر, ماجستير في العلوم.

الكلمات الأساسية: الشبكة العصبية ذات التغذية الأمامية التصنيف،كي-مير، فيروس سارس-كوف-٢، تسلسلات الحمض النووي، المتحورات المثيرة للقلق

يعد تصنيف متغيرات فيروس سارس-كوف-٢ ضروريًا للمساعدة في تحديد الاختلافات في أنماط الجينوم بين المتغيرات، لا سيما تلك التي تندرج ضمن فئة المتغيرات المثيرة للقلق (VoC). هدفت هذه الدراسة إلى تطبيق خوارزمية شبكة العصبية التقدمية ($FFNN$) في تصنيف متغيرات سارس-كوف-٢ بناءً على تسلسل النوكليوتيدات الناتج عن تمثيل الحمض النووي التكميلي ($cDNA$). البيانات المستخدمة هي الجينوم الكامل سارس-كوف-٢ من قاعدة بيانات $NCBI$ مع خمس فئات من المتغيرات، وهي ألفا، وبيتا، ودلتا، وجاما، وأوميكرون. تهيمن على مجموعة البيانات المستخدمة تسلسلات مع مضيف بشري، مع جزء صغير من البيانات من مضيفات أخرى. شملت مراحل البحث المعالجة المسبقة للبيانات، واستخراج الميزات باستخدام طريقة $k-mer$، وتطبيع البيانات باستخدام $Min-Max\ Scaling$، وتقسيم البيانات إلى بيانات تدريب، وتحقق، واختبار، بالإضافة إلى التصنيف باستخدام خوارزمية $FFNN$. أُجري تقييم النموذج باستخدام مصفوفة الارتباك، وتقرير التصنيف، ووقت الحساب. أظهرت نتائج الاختبار على التكوين الرئيسي أن نموذج الشبكة العصبية ذات التغذية الأمامية ($FFNN$) حقق دقةً، وضبطًا، واستدعاءً، $F1-score$ تبلغ ٠،٩٩٨٣ في وقت تدريب قدره ٨،٧٨٨٠ ثانية. تشير هذه النتائج إلى أن نموذج $FFNN$ لديه القدرة على التطبيق الفعال في التصنيف متعدد الفئات لمتغيرات فيروس $SARS-CoV-2$ بناءً على أنماط تسلسل النيوكليوتيدات من تمثيلات الحمض النووي المكمل ($cDNA$).

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi di bidang kecerdasan buatan (*Artificial Intelligence*) telah memberikan dampak besar dalam pengolahan data. Data menjadi komponen utama bagi berbagai sektor karena mampu menjadi dasar dalam proses pengambilan keputusan yang efisien. Namun, semakin banyaknya data yang dihasilkan, muncul tantangan dalam proses pengolahannya. Oleh karena itu, dibutuhkan pendekatan yang mampu mengolah data dalam jumlah besar. Salah satu pendekatan yang banyak digunakan untuk permasalahan tersebut adalah *machine learning*. Menurut Menarianti dkk. (2025), *machine learning* merupakan cabang kecerdasan buatan berfokus pada pengembangan metode yang mampu belajar dari data besar dan kompleks untuk melakukan prediksi, klasifikasi, serta pengambilan keputusan tanpa diprogram secara eksplisit. Pendekatan ini memungkinkan sistem meningkatkan kinerja melalui proses pembelajaran, sehingga menghasilkan analisis yang akurat dan efisien.

Berbagai metode telah dikembangkan dalam bidang *machine learning* untuk permasalahan klasifikasi dan prediksi, seperti algoritma *Feedforward Neural Network* (FFNN). Algoritma FFNN dipilih karena kemampuannya dalam mempelajari hubungan nonlinier antar variabel *input* dan *output* melalui proses pembelajaran *backpropagation* (Zaki & Meira, 2020). FFNN bekerja dengan memberikan informasi satu arah dari lapisan *input* menuju lapisan *output* tanpa adanya umpan balik (*feedback loop*). Melalui proses pembelajaran berbasis *backpropagation*, FFNN mampu menyesuaikan bobot jaringan dengan

meminimalkan kesalahan menggunakan pendekatan *gradient descent*.

Algoritma FFNN banyak diterapkan untuk proses klasifikasi. Klasifikasi merupakan teknik yang digunakan untuk pengolahan data dengan mengelompokkan berdasarkan karakteristik yang dimiliki objek tersebut (Ramadhani & Rosyid, 2022). Klasifikasi bertujuan membangun algoritma yang mampu mempelajari pola hubungan antara fitur sebagai data masukan (*input*) dan label sebagai kelas target (*output*) menggunakan *dataset* yang berlabel pada data pelatihan sehingga dapat melakukan prediksi secara akurat terhadap data baru (Daqiqil, 2021). Keberhasilan klasifikasi sangat bergantung pada representasi fitur dan arsitektur algoritma yang digunakan (Putra, 2020).

Bidang informatika memanfaatkan FFNN untuk mendeteksi, mengklasifikasi, dan memprediksi varian genetik berdasarkan data sekuens SARS-CoV-2. Virus penyebab COVID-19 yang mengancam populasi manusia di seluruh dunia. Seiring berjalannya waktu, SARS-CoV-2 terus mengalami mutasi dan menghasilkan varian baru dengan karakteristik berbeda-beda. Munculnya berbagai varian dapat memengaruhi tingkat penularan, keparahan penyakit, serta efektivitas vaksin dan pengobatan yang ada (Jamhuri dkk., 2025). Pada pertengahan tahun 2022, WHO telah menetapkan lima varian yang dikategorikan sebagai VoC yaitu Alpha (B.1.1.7), Beta (B.1.351), Delta (B.1.617.2), Gamma (P.1), dan Omicron (B.1.1.529).

Perkembangan mutasi barunya dapat diamati dengan mengklasifikasikan variannya untuk membantu adanya kemunculan dan penyebaran virus (Jamhuri dkk., 2025). Hingga September 2025, WHO melaporkan terdapat 114.882 kasus baru COVID-19 dalam periode 28 hari terakhir tersebar di 69 negara, dengan 3.149

sampel (6,5%) dari 48.737 sampel yang diuji terdeteksi positif SARS-CoV-2 (WHO, 2025). Varian baru virus SARS-CoV-2 yang muncul menimbulkan tantangan besar bagi kesehatan global yang mengakibatkan beban sektor kesehatan dalam melakukan tugas dan layanannya kepada masyarakat. Namun, memberikan peluang bagi teknologi informasi untuk melakukan klasifikasi varian khususnya dalam bidang kecerdasan buatan (*Artificial Intelligence*) dalam kondisi pandemi di mana jumlah pasien terpapar virus sangat banyak. Klasifikasi varian menjadi aspek penting dalam mengamati perkembangan virus dan pengendalian pandemi karena memengaruhi kebijakan kesehatan masyarakat yang lebih efektif (Jamhuri dkk., 2025).

Penggunaan FFNN dalam analisis data telah diterapkan pada berbagai bidang, termasuk bidang kesehatan dan bioinformatika. Penelitian oleh Gayathri dkk. (2021) menggabungkan metode CNN, *sparse autoencoder*, dan FFNN dalam mengembangkan analisis sistem citra X-ray yang dirancang untuk tenaga medis, yang menggunakan FFNN sebagai klasifikator akhir yang membedakan pneumonia akibat COVID-19 dan non-COVID-19, dengan hasil menunjukkan bahwa metode tersebut memberikan akurasi klasifikasi yang tinggi. Aljaaf dkk. (2021) menggunakan *ensemble* FFNN yang dikombinasikan dengan metode analisis data untuk memprediksi banyak kasus harian COVID-19, hasil menunjukkan bahwa FFNN mampu memberikan akurasi prediksi yang baik, ditunjukkan oleh nilai metrik evaluasi *Mean Absolute Percentage Error* (MAPE) yang rendah. Vilain & Aris-Brosou (2023) membandingkan FFNN dan *random forest* dalam memprediksi banyak kasus COVID-19 berdasarkan data genom SARS-CoV-2. Hasil menunjukkan bahwa kedua model tersebut mampu mencapai akurasi yang tinggi,

namun FFNN lebih banyak menghubungkan data berdasarkan varian virus secara keseluruhan dibandingkan mengidentifikasi mutasi tertentu yang memiliki dampak biologis signifikan.

Klasifikasi varian SARS-CoV-2 juga banyak dikembangkan menggunakan bebagai metode komputasional. Penelitian Mwanga dkk. (2023) menggunakan *deep learning* (CNN)-LSTM untuk mengklasifikasikan lima varian SARS-CoV-2 yaitu Alpha, Beta, Delta, Gamma, dan Omicron berdasarkan *protein spike* hingga mencapai akurasi tinggi. Selanjutnya, penelitian Correia dkk. (2025) memberikan pendekatan baru untuk klasifikasi *spesies* dan varian virus SARS-CoV-2 menggunakan *Chaos Game Representation* (CGR) dan 2D *Multifractal Detrended Fluctuation Analysis* (2D MF-DFA), menghasilkan fitur yang kemudian diklasifikasikan menggunakan algoritma *Support Vector Machine* (SVM) sehingga mampu membedakan varian SARS-CoV-2 secara efektif meskipun terdapat kemiripan genetik antar varian. Ullah dkk. (2022) mengembangkan metode *Temporal Convolution Network* (TCN) yang digunakan untuk mengidentifikasi dan mengklasifikasi berbagai varian virus SARS-CoV-2 berdasarkan urutan genom, dengan menunjukkan bahwa TCN mampu menangkap hubungan sekuensial antar fitur dan menghasilkan akurasi yang tinggi.

Secara lebih spesifik pada pengembangan metode jaringan saraf, Jamhuri dkk. (2025) menunjukkan bahwa metode optimasi *Gauss–Newton* berbasis QR *factorization* pada FFNN untuk klasifikasi varian SARS-CoV-2 berhasil meningkatkan efisiensi pelatihan FFNN. Dalam penelitian tersebut, fokus utama penelitian terletak pada pengembangan metode optimasi untuk meningkatkan proses pelatihan jaringan saraf. Namun, penelitian-penelitian tersebut belum

membahas penerapan berbasis fitur k-mer menggunakan sekuens genom lengkap (*whole genome*) SARS-CoV-2. Selain itu, aspek keberagaman *host* dalam *dataset* klasifikasi varian SARS-CoV-2 juga belum banyak dibahas pada penelitian sebelumnya. Oleh karena itu, diperlukan penelitian yang membahas kemampuan FFNN dalam mengklasifikasikan varian SARS-CoV-2 berdasarkan fitur k-mer dari sekuens genom lengkap dengan karakteristik data.

Berdasarkan tinjauan terhadap penelitian-penelitian sebelumnya, setiap metode memiliki kelebihan dan keterbatasan masing-masing. Dalam penelitian ini, penulis mengusulkan penerapan algoritma FFNN untuk mengklasifikasi varian SARS-CoV-2 yaitu Alpha, Beta, Delta, Gamma, dan Omicron, menggunakan fitur k-mer menggunakan sekuens genom lengkap (*whole genome*) hasil representasi cDNA yang diperoleh dari *database* publik NCBI. FFNN dipilih karena memiliki kemampuan dalam mempelajari hubungan nonlinier dan menangkap pola kompleks pada data genom. *Dataset* yang digunakan didominasi oleh *host human*, dengan sebagian kecil data berasal dari *host* lain. Oleh karena itu, penelitian ini tidak secara khusus mengevaluasi kinerja model berdasarkan masing-masing *host*, melainkan berfokus pada klasifikasi varian SARS-CoV-2 berdasarkan pola sekuens genom. Dengan demikian, penelitian ini bertujuan untuk mengevaluasi kinerja FFNN berbasis fitur k-mer dalam mengklasifikasikan varian SARS-CoV-2 berdasarkan sekuens genom lengkap (*whole genome*) hasil representasi cDNA.

Penerapan metode matematika menggunakan algoritma komputasional terhadap data dalam pandangan Islam merupakan pengamalan ilmu yang berkontribusi nyata kepada masyarakat. Hal ini merupakan wujud ibadah dan tanggung jawab kepada Allah SWT yang sejalan dengan ajaran Islam tentang

keutamaan menuntut ilmu dan kewajiban mengamalkannya. Dalam khazanah Islam klasik, Sayyidina Ali bin Abi Thalib r.a. menyampaikan sebuah ungkapan hikmah (al-Sakhāwī, 1987), yaitu:

ثَمَرٍ بِلَا كَالشَّجَرِ عَمَلٍ بِلَا العِلْمُ

Artinya: *"Ilmu tanpa amal, bagaikan pohon yang tidak berbuah." (al-Sakhāwī dalam al-Maqāsid al-Hasanah fī Bayān Kathīr min al-Ahādīth al-Mushtaharah 'alā al-Alsinah, no. 1030; Dār al-Kutub al-'Ilmiyyah, Beirut, 1987)*

Menurut al-Sakhāwī, ungkapan tersebut merupakan salah satu perkataan masyhur yang dikenal di kalangan ulama. Dalam al-Maqāsid al-Hasanah, menegaskan bahwa ilmu yang tidak diamalkan maka tidak akan bermanfaat, karena ilmu merupakan sarana untuk beramal dengan benar, sedangkan amal merupakan tujuan dari ilmu itu sendiri. Sebagaimana pohon yang tidak berbuah tidak akan memberi keuntungan untuk lingkungannya. Dengan demikian, seseorang yang berilmu hendaknya menyampaikan dan mengamalkan agar mendapatkan manfaat serta mempertanggungjawabkan di hadapan Allah SWT. Prinsip ini menjadi landasan dalam penelitian ini sebagai bentuk pengamalan ilmu melalui penerapan algoritma FFNN dalam proses klasifikasi.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, rumusan masalah dalam penelitian ini adalah:

1. Bagaimana penerapan algoritma *Feedforward Neural Network* (FFNN) dalam mengklasifikasikan varian SARS-CoV-2 berdasarkan data sekuens cDNA nukleotida?
2. Bagaimana kinerja algoritma *Feedforward Neural Network* (FFNN) dalam mengklasifikasi varian SARS-CoV-2 berdasarkan data sekuens cDNA nukleotida?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah di atas maka tujuan penelitian ini adalah:

1. Menerapkan algoritma *Feedforward Neural Network* (FFNN) dalam klasifikasi varian SARS-CoV-2 berdasarkan sekuens cDNA nukleotida.
2. Mengevaluasi kinerja algoritma *Feedforward Neural Network* (FFNN) berdasarkan akurasi, presisi, *recall*, F1-*score*, *confusion matrix* dan efisiensi waktu komputasi.

1.4 Manfaat Penelitian

Adapun beberapa manfaat yang diharapkan dari penelitian ini adalah:

1. Manfaat Teoritis:
   a. Memberikan kontribusi ilmiah terhadap pengembangan metode klasifikasi berbasis *Feedforward Neural Network* (FFNN) pada sekuens genom untuk mengidentifikasi varian SARS-CoV-2 berdasarkan sekuens cDNA nukleotida.
   b. Menjadi referensi akademik mengenai kemampuan arsitektur FFNN dalam mempelajari pola kompleks pada bentuk numerik hasil pengolahan sekuens cDNA, khususnya dalam proses klasifikasi multikelas varian SARS-CoV-2.
2. Manfaat Praktis:
   a. Menghasilkan model klasifikasi yang dapat mengidentifikasi varian secara akurat ke dalam varian yang telah ditentukan.
   b. Memberikan pendekatan komputasional yang lebih efisien dalam pengolahan dan analisis data sekuens, sehingga dapat mengurangi waktu dan kompleksitas proses identifikasi varian.

1.5 Batasan Masalah

Agar penelitian ini lebih terfokus, maka penulis menetapkan batasan sebagai berikut:

1. Fokus penelitian hanya klasifikasi multikelas varian SARS-CoV-2, tanpa membahas aspek prediksi keparahan infeksi, efektivitas vaksin, atau penularan antar populasi.
2. Data yang digunakan dalam penelitian ini mencakup sekuens dari beberapa spesies atau sumber lain, yaitu *Human*, Tikus, Virus Rekayasa Lab, Kucing, dan Monyet. Hal ini menunjukkan bahwa data yang digunakan masih bersifat heterogen karena belum dilakukan penyaringan secara ketat berdasarkan jenis *host*.
3. Penelitian ini hanya menggunakan ekstraksi fitur berbasis k-mer tanpa penggunaan teknik *feature selection*.

1.6 Definisi Istilah

1. *Variants of Concern* (VoC)

   Varian virus yang memengaruhi sifat biologisnya, seperti meningkatkan kemampuan penularan, tingkat keparahan penyakit, menurunkan efektivitas vaksin, serta menunjukkan peningkatan tingkat penyebaran yang lebih signifikan dibandingkan varian sebelumnya.

2. Sekuens

   Komponen utama penyusun gen yang mengodekan karakteristik tertentu pada sel yang tersusun oleh asam nukleat berbentuk nukleotida Adenin (A), Timin (T), Sitosin (C), dan Guanin (G).

3. Genom

   Keseluruhan materi genetik suatu organisme yang tersusun atas DNA atau RNA serta mengandung informasi biologis yang diperlukan untuk fungsi, pertumbuhan, dan pewarisan sifat organisme tersebut.

4. FASTA

   FASTA adalah format standar untuk menyimpan sekuens cDNA yang memudahkan proses ekstraksi fitur menggunakan *k*-mer, sehingga data sekuens dapat diubah menjadi vektor numerik untuk digunakan sebagai *input*.

5. *Cross*-e*ntropy Loss*

   Fungsi kerugian (*loss function*) yang digunakan untuk mengukur perbedaan antara distribusi probabilitas yang diperoleh dari fungsi aktivasi *softmax* pada lapisan *output* dengan distribusi target yang direpresentasikan dalam bentuk *one-hot encoding*.

6. *K*-Mer

   Proses ekstraksi fitur yang menyatakan *substring* sepanjang *k* dengan mengubah data sekuens yang tersusun atas nukleotida, yaitu (A, T, C, G) menjadi numerik dalam bentuk vektor fitur.

7. *One-Hot Encoding*

   Teknik mengubah urutan label kelas menjadi bentuk numerik berupa vektor biner yang akan menghasilkan sebuah matriks dua dimensi terdiri dari kombinasi angka 0 dan 1 dalam seluruh sekuens.

8. *Confusion Matrix*

   Matriks evaluasi yang menunjukkan banyak prediksi benar dan salah untuk memudahkan mengukur kinerja algoritma, terutama ketika data bersifat tidak

seimbang.

9. *TensorFlow*

   Framework yang digunakan untuk membangun dan melatih dengan menghitung turunan fungsi kerugian (*loss function*) terhadap seluruh parameter algoritma, kemudian memperbarui bobot berdasarkan aturan *gradient descent*.

BAB II

KAJIAN TEORI

2.1 *Machine Learning*

Machine learning menjadi salah satu cabang dari kecerdasan buatan (AI) karena efisien untuk penggunaan data dalam jumlah besar (Menarianti dkk., 2025). Menurut Putra (2020), *Machine learning* adalah teknik yang digunakan dalam pengolahan data dengan pendekatan matematis untuk membuat model yang mengidentifikasi pola-pola data. Komponen yang membangun *machine learning* adalah pengalaman dan kemampuan sistem untuk belajar dari data tanpa perlu diprogram berulang kali oleh manusia. Kemampuan ini bertujuan untuk membangun algoritma yang memungkinkan sistem untuk belajar dari data dan memprediksi tanpa diprogram secara eksplisit (Daqiqil, 2021).

Penerapan *machine learning* tidak berfokus pada pemilihan algoritma saja, tetapi juga melibatkan tahapan yang dibagi oleh *dataset* pelatihan (*training*), validasi (*validation*) dan pengujian (*testing*). Tahapan ini berperan dalam menentukan kualitas algoritma, karena tahapan *training* bertujuan untuk membangun algoritma dalam merepresentasikan pola, *validation* mengevaluasi hyperparameter dan *early stopping*, sedangkan *testing* digunakan dalam tahap pengujian kemampuan algoritma. Pembagian *dataset* menjadi *training* dan *testing* pada umumnya memiliki rasio 90%:10%, 80%:20%, atau 70%:30% (Putra, 2020).

Machine learning secara umum dibagi menjadi tiga kategori utama, yaitu *supervised learning*, *unsupervised learning*, *reinforcement learning*. *Supervised learning* merupakan metode pembelajaran yang memiliki hubungan antara *input* dan *output* berdasarkan data yang sudah ada untuk dipelajari algoritma berdasarkan

data *training* yang berlabel dengan tujuan melakukan generalisasi data input. Berbeda dengan itu, *unsupervised learning* digunakan ketika data tidak memiliki label, sehingga sistem berusaha menemukan pola secara acak (Menarianti dkk., 2025). Sementara itu, *reinforcement learning* merupakan metode ketika sistem atau komputer yang belajar melalui interaksi langsung dengan suatu lingkungan secara dinamis. Dalam metode ini, sistem ditugaskan untuk melakukan serangkaian tindakan agar dapat mencapai suatu tujuan tertentu (Daqiqil, 2021). Proses pembelajaran terjadi melalui mekanisme *trial and error* dengan memperoleh umpan balik berupa *reward* atau *penalty* (Menarianti dkk., 2025).

Metode *Supervised Learning* merupakan salah satu pendekatan yang digunakan dalam klasifikasi. Menurut Menarianti dkk. (2025), teknik *Supervised Learning* melakukan pendekatan pada *dataset* berlabel. Metode ini memanfaatkan data yang sudah memiliki label, yaitu *input* atau *output* dengan memetakannya agar dapat meminimalkan kesalahan saat melakukan prediksi terhadap data baru yang belum diketahui kelasnya dengan mengacu pada pola yang sudah dipelajari. Dalam penelitian ini mengimplementasikan salah satu algoritma yang terdapat dalam *supervised learning* yaitu *Feedforward Neural Network* (FFNN).

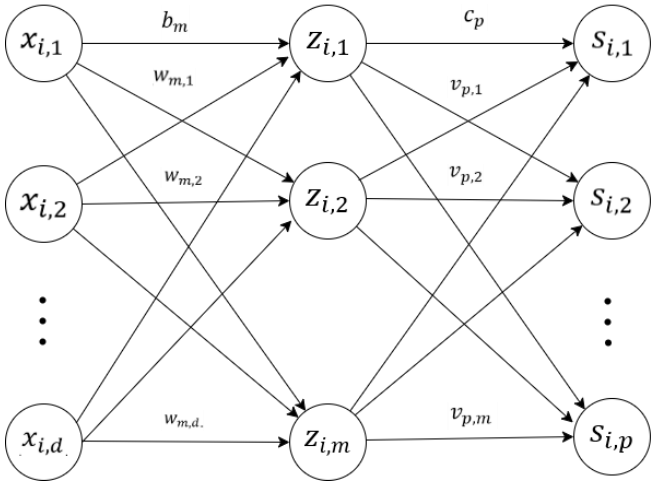
2.2 *Feedforward Neural Network* (FFNN)

Artificial Neural Network (ANN) merupakan sistem yang memiliki banyak unit pemrosesan sederhana. Prinsip kerja dari ANN terinspirasi dari cara kerja sistem jaringan saraf manusia. Jaringan ini memiliki kemampuan alami untuk menyimpan pengetahuan berdasarkan pengalaman sehingga dapat melakukan pengenalan pola, ekstraksi fitur, dan transformasi masukan menjadi keluaran. Unit pemrosesan sederhana dalam ANN disebut sebagai neuron, yang terhubung melalui

bobot tertentu. Setiap koneksi memiliki nilai bobot yang menentukan kekuatan sinyal antar neuron (Haykin, 2009).

Salah satu bentuk arsitektur ANN yang paling umum digunakan adalah *Feedforward Neural Network* (FFNN) atau juga dikenal sebagai *Multilayer Perceptron* (MLP). Penjelasan mengenai arsitektur, alur *feedforward*, formulasi matematis, fungsi aktivasi, fungsi *loss*, dan *backpropagation* algoritma FFNN pada bagian ini mengikuti konsep pada Zaki & Meira (2020), dengan penyesuaian terhadap kebutuhan analisis pada penelitian ini. Pada penelitian ini, FFNN terdiri atas tiga lapisan utama, yaitu *input*, tersembunyi (*hidden*), dan *output*. Jaringan ini disebut *feedforward* karena informasi hanya bergerak satu arah dari lapisan input menuju lapisan *output* tanpa adanya umpan balik (*feedback*). Setiap neuron pada suatu lapisan terhubung dengan semua neuron pada lapisan berikutnya.

Pada lapisan *input* terdapat neuron d yang menyatakan banyak fitur hasil ekstraksi dari *k*-mer, lapisan *hidden* memiliki neuron m yang berfungsi mengubah nilai keluaran sebelum diproses oleh fungsi aktivasi seperti ReLu, sedangkan pada lapisan *output* terdapat neuron p untuk memprediksi banyak kelas yaitu 5 varian SARS-CoV-2. Sehingga terdapat parameter yang harus dipelajari selama proses pelatihan (*training*), yaitu parameter bobot ($\mathbf{W}$ dan $\mathbf{V}$) dengan total bobot yakni $m \times d + p \times m$ dan parameter bias ($\mathbf{b}$ dan $\mathbf{c}$) dengan total bias $m + p$. Dengan demikian, total parameter yaitu $(m \times d) + (p \times m) + (m + p)$. Berdasarkan penjelasan tersebut, bentuk umum jaringan dengan satu lapisan *hidden* dapat dilihat pada Gambar 2.1, dengan setiap neuron menerima seluruh sinyal dari lapisan sebelumnya.



Gambar 2.1 *Feedforward Neural Network* Satu *Hidden Layer*

Berikut dijelaskan tahapan proses pada metode *Feedforward Neural Network* (FFNN).

1. Lapisan *Input*

   Pada tahap *feedforward*, lapisan *input* tidak menggunakan fungsi aktivasi karena tidak memiliki bobot dan bias. Lapisan ini hanya meneruskan nilai masukan ke lapisan berikutnya tanpa melakukan transformasi. Setiap neuron pada lapisan *input* menerima data mentah berupa fitur dari setiap sampel dan meneruskan secara langsung ke lapisan *hidden*. Setiap data masukan dinyatakan sebagai suatu vektor fitur berukuran $1 \times d$, yang dituliskan dalam bentuk:

   $$\mathbf{x}_i = [x_{i,1} \quad x_{i,2} \quad \dots \quad x_{i,d}], \qquad \mathbf{x}_i \in \mathbb{R}^d \tag{2.1}$$

   Persamaan (2.1) memuat nilai fitur dari satu data *training*, dengan setiap $\mathbf{x}_i$ menunjukkan urutan data dalam *dataset,* dengan setiap nilai i merepresentasikan satu sekuens. Sedangkan $x_{i,d}$ menyatakan nilai fitur ke-d pada data ke-i, dan d merupakan banyak fitur. Dalam data *training* $D\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, $\mathbf{x}_i$ merupakan satu baris dengan d fitur, sedangkan y_i merupakan label target dan n merupakan banyak data.

2. Lapisan Tersembunyi (*Hidden Layer*)

Lapisan *hidden* merupakan lapisan pertama yang melakukan proses terhadap data masukan. Lapisan ini memiliki parameter berupa bobot ($\mathbf{W}$) dan bias ($\mathbf{b}$) pada setiap neuron untuk melakukan transformasi linier terhadap vektor *input* sebelum menggunakan fungsi aktivasi. Setiap neuron pada lapisan *hidden* menerima sinyal dari semua neuron lapisan *input*. Pada lapisan *hidden* mengikuti prinsip perkalian matriks yaitu jika banyak kolom pada matriks pertama sama dengan banyak baris pada matriks kedua. Jika A berukuran $m \times r$ dan B berukuran $r \times n$, maka hasil kali AB adalah matriks $m \times n$ (Anton & Rorres, 2013). Prinsip ini bertujuan untuk memastikan operasi linier antara bobot dan masukan.

Semua bobot antara lapisan *input* dan lapisan *hidden* disusun ke dalam matriks bobot $\mathbf{W} \in \mathbb{R}^{m \times d}$, dengan d menyatakan banyak neuron *input* berupa fitur dan m menyatakan banyak neuron *hidden*. Setiap baris pada $\mathbf{W}$ menyatakan bobot yang menghubungkan semua neuron *input* menuju satu neuron *hidden*. Matriks bobot didefinisikan sebagai:

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,d} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,d} \\ \vdots & \vdots & & \vdots \\ w_{m,1} & w_{m,2} & \cdots & w_{m,d} \end{bmatrix} \tag{2.2}$$

Persamaan (2.1) pada lapisan *input* didefinisikan sebagai vektor baris, maka pada tahap *feedforward* di lapisan *hidden* diubah menjadi vektor kolom melalui *transpose*:

$$(\mathbf{x_i})^T = \begin{bmatrix} x_{i,1} \\ x_{i,2} \\ \vdots \\ x_{i,d} \end{bmatrix} \in \mathbb{R}^{d \times 1} \tag{2.3}$$

Pada setiap neuron *hidden* memiliki vektor bias $\mathbf{b} \in \mathbb{R}^{m\times 1}$, yaitu:

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \tag{2.4}$$

Total masukan untuk setiap neuron *hidden* diperoleh dari hasil perkalian matriks antara bobot dan vektor *input* kemudian ditambahkan dengan vektor bias, yang dapat dinyatakan sebagai:

$$\mathbf{a}_i = \mathbf{W}(\mathbf{x}_i)^\top + \mathbf{b}$$

$$\mathbf{a}_i = \begin{bmatrix} a_{i,1} \\ a_{i,2} \\ \vdots \\ a_{i,m} \end{bmatrix} = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,d} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,d} \\ \vdots & \vdots & & \vdots \\ w_{m,1} & w_{m,2} & \cdots & w_{m,d} \end{bmatrix} \begin{bmatrix} x_{i,1} \\ x_{i,2} \\ \vdots \\ x_{i,d} \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \tag{2.5}$$

Kemudian, diterapkan fungsi aktivasi *Rectified Linear Unit* (ReLU) untuk menghasilkan keluaran z dari setiap neuron *hidden*:

$$\mathrm{ReLU}(a) = \begin{cases} a, & \text{jika } a > 0 \\ 0, & \text{jika } a \leq 0 \end{cases} \tag{2.6}$$

Persamaan (2.6) menghasilkan keluaran nol untuk nilai negatif dan mempertahankan nilai positif tanpa perubahan. Ketika diterapkan pada seluruh elemen hasil perhitungan a, maka menghasilkan vektor keluaran lapisan *hidden*:

$$\mathbf{z}_i = \begin{bmatrix} z_{i,1} \\ z_{i,2} \\ \vdots \\ z_{i,m} \end{bmatrix} = \begin{bmatrix} \mathrm{ReLU}(a_{i,1}) \\ \mathrm{ReLU}(a_{i,2}) \\ \vdots \\ \mathrm{ReLU}(a_{i,m}) \end{bmatrix} \tag{2.7}$$

Persamaan (2.7) menyatakan nilai keluaran dari neuron *hidden* berukuran $m \times 1$, dengan setiap $\mathbf{z}_i$ merupakan hasil ReLU keluaran neuron *hidden* ke-i.

3. Lapisan *Output*

Lapisan *output* menerima masukan dari semua neuron dari lapisan *hidden*.

Setiap neuron melakukan kombinasi linier terhadap semua nilai keluaran neuron *hidden* yang terhubung dan ditambah dengan bias ($\mathbf{c}$). Proses ini mengikuti prinsip dasar perkalian matriks yaitu hasil kali antara matriks berukuran $p \times m$ dengan vektor berukuran $m \times 1$ akan menghasilkan vektor berukuran $p \times 1$ (Anton & Rorres, 2013). Dengan demikian, setiap neuron *output* mendapatkan sinyal yang merupakan gabungan dari seluruh neuron *hidden* yang bekerja terhadap prediksi akhir jaringan.

Semua bobot antara lapisan *hidden* dan lapisan *output* disusun ke dalam matriks bobot $\mathbf{V} \in \mathbb{R}^{p \times m}$, dengan p menyatakan banyak neuron *output* untuk memprediksi banyak kelas dan m menyatakan banyak neuron *hidden*. Setiap baris $\mathbf{V}$ menyatakan bobot yang menghubungkan semua neuron *hidden* menuju satu neuron *output*. Matriks bobot $\mathbf{V}$ dan vektor *hidden* z didefinisikan sebagai:

$$\mathbf{V} = \begin{bmatrix} v_{1,1} & v_{1,2} & \cdots & v_{1,m} \\ v_{2,1} & v_{2,2} & \cdots & v_{2,m} \\ \vdots & \vdots & & \vdots \\ v_{p,1} & v_{p,2} & \ldots & v_{p,m} \end{bmatrix}, \tag{2.8}$$

$$\mathbf{z}_i = \begin{bmatrix} z_{i,1} \\ z_{i,2} \\ \vdots \\ z_{i,m} \end{bmatrix} \in \mathbb{R}^{m \times} 1 \tag{2.9}$$

Setiap neuron *output* memiliki vektor bias $\mathbf{c} \in \mathbb{R}^{p \times 1}$, yaitu:

$$\mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_p \end{bmatrix} \tag{2.10}$$

Proses ini menghasilkan total masukan untuk setiap neuron *output* yang dihitung melalui perkalian matriks antara bobot dan vektor keluaran *hidden* kemudian ditambahkan dengan vektor bias, dapat dinyatakan sebagai:

$$\mathbf{s}_i = \mathbf{V}\mathbf{z}_i + \boldsymbol{c}$$

$$\mathbf{s}_i = \begin{bmatrix} s_{i,1} \\ s_{i,2} \\ \vdots \\ s_{i,p} \end{bmatrix} = \begin{bmatrix} v_{1,1} & v_{1,2} & \cdots & v_{1,m} \\ v_{2,1} & v_{2,2} & \cdots & v_{2,m} \\ \vdots & \vdots & & \vdots \\ v_{p,1} & v_{p,2} & \ldots & v_{p,m} \end{bmatrix} \begin{bmatrix} z_{i,1} \\ z_{i,2} \\ \vdots \\ z_{i,m} \end{bmatrix} + \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_p \end{bmatrix} \tag{2.11}$$

Persamaan (2.11) menghasilkan keluaran awal dari setiap neuron *output* sebelum proses normalisasi probabilitas dilakukan.

Fungsi aktivasi *softmax* digunakan pada lapisan *output* untuk klasifikasi multikelas. Fungsi ini mengubah nilai keluaran $\mathbf{s}_i$ menjadi distribusi probabilitas yang totalnya satu, sehingga setiap keluaran dapat dinyatakan sebagai peluang dari masing-masing kelas. *Softmax* memetakan setiap nilai pada rentang $[0, 1]$ dan memastikan jumlah semua probabilitas mencapai 1. Persamaannya ditulis sebagai:

$$\mathbf{o}_{i,j} = \frac{e^{s_{i,j}}}{\sum_{k=1}^{p} e^{s_{i,k}}}, \qquad j = 1, 2, \ldots, p \tag{2.12}$$

Setelah fungsi *softmax* diterapkan, vektor $\mathbf{s}_i$ diubah menjadi probabilitas $\mathbf{o}_i$ dengan semua nilai $0 \leq \mathbf{o}_i \leq 1$ dan $\sum_{j=1}^{p} \mathbf{o}_{i,j} = 1$. Hasil transformasi menghasilkan vektor keluaran o pada Persamaan (2.13) berikut:

$$\mathbf{o}_i = \begin{bmatrix} o_{i,1} \\ o_{i,2} \\ \vdots \\ o_{i,p} \end{bmatrix} = \begin{bmatrix} \dfrac{e^{s_{i,1}}}{\sum_{k=1}^{p} e^{s_{i,k}}} \\ \dfrac{e^{s_{i,2}}}{\sum_{k=1}^{p} e^{s_{i,k}}} \\ \vdots \\ \dfrac{e^{s_{i,p}}}{\sum_{k=1}^{p} e^{s_{i,k}}} \end{bmatrix} \tag{2.13}$$

Setiap elemen $\mathbf{o}_i$ pada Persamaan (2.13), menyatakan probabilitas prediksi data ke-i terhadap kelas ke-j berdasarkan hasil kombinasi linier bobot $\mathbf{V}$, keluaran lapisan *hidden* $\mathbf{z}_i$, dan bias $\mathbf{c}$ yang telah ditransformasi oleh fungsi aktivasi *softmax*. Setelah diperoleh nilai probabilitas dari fungsi *softmax*, tahap

selanjutnya adalah penentuan kelas prediksi dengan memilih nilai probabilitas terbesar dari vektor keluaran *softmax* menggunakan Persamaan (2.14):

$$\hat{y}_i = \arg\max_j \left(o_{i,j}\right) \tag{2.14}$$

Fungsi arg max adalah untuk memilih kelas dengan nilai probabilitas tertinggi, sehingga kelas tersebut menjadi hasil prediksi akhir dari algoritma.

4. Fungsi *Loss*

   Fungsi *loss* (*error function*) digunakan untuk menghitung besarnya perbedaan antara nilai prediksi jaringan dengan target sebenarnya. Pada permasalahan klasifikasi multikelas, fungsi *loss* yang digunakan adalah *cross-entropy loss* karena dapat mengukur perbedaan antara distribusi probabilitas hasil keluaran *softmax* dengan distribusi target. Jika terdapat kelas sebanyak p, maka setiap neuron pada lapisan *output* merepresentasikan satu kelas. Sehingga, banyak neuron pada lapisan *output* sama dengan banyak kelas yang diprediksi.

   Keluaran jaringan pada setiap data masukan $\mathbf{x}_i \in \mathbb{R}^d$ dinyatakan dalam bentuk vektor probabilitas hasil aktivasi fungsi *softmax* pada lapisan *output*, yaitu $\mathbf{o}_i \in \mathbb{R}^{p\times 1}$ dengan $o_{i,j}$ menyatakan probabilitas prediksi untuk kelas ke-j. Sementara itu, label target untuk data dinyatakan dalam bentuk vektor *one-hot* encoding $\mathbf{y}_i = [y_{i,1} \quad y_{i,2} \quad \cdots \quad y_{i,p}]^T$ dengan satu elemen bernilai 1 sebagai kelas sebenarnya dan elemen lainnya 0. Vektor target ini digunakan untuk menilai seberapa baik jaringan mengenali kelas benar dari setiap data.

   Pada data *training* terdapat n dalam himpunan $D = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots (\mathbf{x}_n, \mathbf{y}_n)\}$, maka fungsi *cross-entropy loss* untuk seluruh data didefinisikan sebagai:

$$E = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{p} y_{i,j}\ln(o_{i,j})$$

$$= -\frac{1}{n}\sum_{i=1}^{n}(y_{i,1}\ln(o_{i,1}) + y_{i,2}\ln(o_{i,2}) + \cdots + y_{i,p}\ln(o_{i,p})) \quad (2.15)$$

Persamaan (2.15) mengukur tingkat kesalahan rata-rata jaringan dalam memprediksi semua data *training*. Jika algoritma memberikan probabilitas yang tinggi pada kelas yang benar sehingga menghasilkan nilai *loss* yang kecil. Sebaliknya, jika probabilitas pada kelas yang benar rendah, ln (o_{i,j^*}) bernilai negatif besar, sehingga $-\ln$ (o_{i,j^*}) menjadi besar.

Fungsi *loss* untuk setiap data ke-i dapat dituliskan sebagai:

$$L_i = -\sum_{j=1}^{p} y_{i,j}\ln(o_{i,j}) \quad (2.16)$$

Karena label target menggunakan *one-hot encoding*, maka hanya satu elemen bernilai 1:

$$L_i = -\ln(o_{i,j^*}) \quad (2.17)$$

Dengan j^* merupakan kelas benar. Sehingga hubungan antara *loss* total dan untuk setiap data adalah:

$$E = \frac{1}{n}\sum_{i=1}^{n} L_i \quad (2.18)$$

Simbol n menyatakan banyak sampel pada *dataset*, sedangkan p menyatakan banyak kelas yang diprediksi oleh jaringan. Komponen $y_{i,j}$ adalah elemen dari vektor *one-hot* yang menunjukkan kelas benar untuk sampel ke-i, di mana nilainya 1 untuk kelas yang sesuai dan 0 untuk kelas lainnya. Sementara itu, $o_{i,j}$ adalah probabilitas prediksi jaringan untuk kelas ke-j dari sampel ke-i, yang dihasilkan melalui fungsi *softmax* pada lapisan *output*. Dengan demikian,

semakin akurat prediksi jaringan terhadap seluruh n data pelatihan, semakin kecil nilai *loss* total E.

5. *Backpropagation*

   Backpropagation merupakan metode untuk menghitung gradien dari fungsi *loss* terhadap semua parameter algoritma. Nilai kesalahan L_i yang diperoleh digunakan sebagai dasar dalam melakukan *backpropagation*. Algoritma memiliki dua matriks bobot, yaitu matriks bobot *input* menuju *hidden* pada Persamaan (2.2) dan matriks bobot *hidden* menuju *output* pada Persamaan (2.8). Sehingga, gradien fungsi *loss* terhadap parameter memiliki bentuk matriks dengan ukuran yang sama:

$$\nabla_{\mathbf{W}} L_i \in \mathbb{R}^{m \times d}, \qquad \nabla_{\mathbf{V}} L_i \in \mathbb{R}^{p \times m} \tag{2.19}$$

   Setiap neuron *output* ke-j menerima masukan total berupa kombinasi linier antara bobot $\mathbf{V}$, keluaran *hidden* $\mathbf{z}_i$, dan bias yang dinyatakan pada Persamaan (2.11). Selanjutnya, fungsi *softmax* digunakan untuk mengubah nilai $\boldsymbol{s}_i$ menjadi probabilitas prediksi dengan menggunakan Persamaan (2.12).

   Gradien pada lapisan *hidden* dihitung dengan menentukan sinyal kesalahan pada lapisan *output*, yaitu:

$$\boldsymbol{\delta}_i = \mathbf{o}_i - \mathbf{y}_i \tag{2.20}$$

   Persamaan (2.20) diteruskan ke setiap neuron *hidden* dengan menghitung turunan fungsi aktivasi ReLU pada lapisan *hidden*. Nilai ini menunjukkan selisih antara hasil prediksi model dan label sebenarnya. Semakin besar selisihnya, semakin besar kontribusi neuron tersebut terhadap kesalahan total. Melalui aturan rantai, turunan terhadap bobot pada lapisan *hidden* menuju *output* dinyatakan sebagai:

$$\nabla_{\mathbf{V}} L_i = (\mathbf{o}_i - \mathbf{y}_i)\mathbf{z}_i^T \tag{2.21}$$

Selain bobot, parameter bias pada setiap lapisan juga diperbarui menggunakan nilai gradien yang diperoleh dari proses *backpropagation*. Pada lapisan *output*, gradien terhadap bias pada neuron ke-j ditentukan oleh nilai kesalahan pada neuron tersebut, sehingga diperoleh bahwa turunan fungsi *loss* terhadap bias *output* sama dengan sinyal kesalahan pada *output*, yaitu:

$$\nabla_{\mathbf{c}} L_i = \boldsymbol{\delta}_i \tag{2.22}$$

Kesalahan pada *output* diteruskan ke *hidden* untuk menghitung kontribusi setiap neuron *hidden* terhadap kesalahan total. Sinyal kesalahan pada neuron *hidden* untuk data ke-i dinyatakan sebagai:

$$\boldsymbol{\gamma}_i = (\mathbf{V}^{\mathrm{T}}\boldsymbol{\delta}_i) \odot f'(\mathbf{a}_i) \tag{2.23}$$

Gradien terhadap bobot antara lapisan input dan lapisan hidden dihitung setelah sinyal kesalahan pada neuron hidden diperoleh. Pada perhitungan gradien bobot $\mathbf{W}$, bentuk $\mathbf{x}_i$ menggunakan bentuk baris, karena merujuk pada Persamaan (2.1). Sementara itu, bentuk $\mathbf{x}_i^T$ digunakan pada proses *feedforward* untuk menyesuaikan Persamaan (2.5). Berdasarkan aturan rantai, diperoleh:

$$\nabla_{\mathbf{W}} L_i = \boldsymbol{\gamma}_i \, \mathbf{x}_i \tag{2.24}$$

Keterangan pada Persamaan (2.14) yaitu $\boldsymbol{\gamma}_i \in \mathbb{R}^{m\times 1}$, $\mathbf{x}_i \in \mathbb{R}^{1\times d}$, $\nabla_{\mathbf{W}} L_i \in \mathbb{R}^{m\times d}$. Persamaan (2.24) menunjukkan bahwa perubahan bobot dipengaruhi oleh nilai kesalahan pada neuron *hidden* serta nilai *input* yang masuk ke neuron tersebut. Selanjutnya, pada lapisan *hidden*, gradien terhadap bias dihitung berdasarkan sinyal kesalahan dari lapisan *output* melalui bobot yang menghubungkan kedua lapisan tersebut. Dengan demikian, turunan fungsi *loss* terhadap bias pada neuron *hidden* diperoleh sebagai:

$$\nabla_{\mathbf{b}} L_i = \boldsymbol{\gamma}_i \tag{2.25}$$

Persamaan (2.25) menunjukkan bahwa perubahan bias pada *hidden* sepenuhnya ditentukan oleh sinyal kesalahan yang diterima neuron tersebut, sehingga setiap bias akan menyesuaikan kontribusinya terhadap pengurangan nilai kesalahan secara keseluruhan.

6. *Update* Parameter

Backpropagation bekerja dengan menyebarkan nilai kesalahan dari *output* menuju *hidden* secara terstruktur, sehingga setiap bobot diperbarui sesuai kesalahan terhadap kesalahan prediksi. Perhitungan gradien dalam bentuk operasi matriks memungkinkan seluruh proses dilakukan secara efisien, terutama dalam perangkat lunak. Salah satu yang paling umum digunakan adalah *TensorFlow*, yang mampu melakukan perhitungan *backpropagation* dan pembaruan bobot secara otomatis. *TensorFlow* bekerja melalui *reverse-mode automatic differentiation* (*autodiff*) berdasarkan aturan *gradient descent*. Dengan laju pembelajaran η yang mengatur besar kecilnya langkah pembaruan parameter, pembaruan kedua matriks bobot dan bias dinyatakan sebagai:

$$\mathbf{W}^{baru} = \mathbf{W}^{lama} - \eta \nabla_{\mathbf{W}} E, \qquad \mathbf{V}^{baru} = \mathbf{V}^{lama} - \eta \nabla_{\mathbf{V}} E \tag{2.26}$$

$$\mathbf{b}^{baru} = \mathbf{b}^{lama} - \eta \nabla_{\mathbf{b}} E, \qquad \mathbf{c}^{baru} = \mathbf{c}^{lama} - \eta \nabla_{\mathbf{c}} E \tag{2.27}$$

TensorFlow memungkinkan proses pelatihan berjalan secara efisien, terutama ketika dijalankan pada perangkat GPU (Géron, 2019).

2.3 Klasifikasi

Klasifikasi merupakan salah satu bentuk *supervised learning* untuk menemukan suatu algoritma yang dapat menjelaskan kelas data penting. Klasifikasi bertujuan untuk membangun algoritma yang dapat menjelaskan dan membedakan

label kelas berdasarkan analisis data *training* dengan karakteristik yang dimiliki oleh masing-masing objek (Han dkk., 2011). Pada proses ini, algoritma mempelajari hubungan antara fitur dan label kelas untuk mengidentifikasi kategori dari objek baru yang belum pernah dilihat sebelumnya. Kemampuan algoritma dalam mengenali pola secara akurat sangat dipengaruhi oleh kualitas fitur yang digunakan (Daqiqil, 2021).

Penelitian ini menggunakan jenis *multi-class classification*. *Multi-class classification* merupakan jenis klasifikasi yang mempunyai lebih dari dua kelas untuk diidentifikasi karena bertujuan untuk membedakan beberapa varian seperti Alpha, Beta, Delta, Gamma, dan Omicron. Setiap data genom dapat dikategorikan ke dalam salah satu varian, sehingga proses pelatihan memungkinkan algoritma mempelajari pola dan karakteristik yang berbeda antar varian. Dengan pendekatan ini, algoritma dapat mengenali perbedaan ciri genom antar varian lebih akurat.

2.4 *K*-Mer

Metode k-mer banyak digunakan dalam analisis sekuens biologis dengan merepresentasikan *substring* sepanjang k dalam suatu sekuens biologis. K-mer tersusun atas nukleotida, yaitu Adenin (A), Timin (T), Sitosin (C), dan Guanin (G) yang umumnya digunakan dalam konteks genom komputasional dan analisis sekuens (Alshayeji dkk., 2023). Metode ini diterapkan untuk ekstraksi fitur yang bertujuan untuk membagi data sekuens menjadi satu atau lebih kelompok yang memiliki karakteristik. Ekstraksi fitur dilakukan dengan mengubah sekuens menjadi representasi numerik dalam bentuk vektor fitur (Putra, 2020).

Pada sekuens seperti ATCG terdapat empat monomer (A, T, C, G), tiga 2-mer (AT, TC, CG), dua 3-mer (ATC, TCG), dan satu 4-mer (ATCG). Banyaknya

kombinasi fitur yang terbentuk adalah 4^k dengan nilai $k \geq 1$. Meskipun konsep ini terlihat sederhana, perhitungan k-mer pada data sekuens yang berukuran besar menyebabkan kelebihan kapasitas memori komputer karena menghasilkan banyaknya *substring*. Oleh karena itu, pemilihan nilai k perlu disesuaikan dengan karakteristik data dan kebutuhan analisis, karena tidak terdapat nilai k yang optimal secara universal serta penggunaan k-mer yang lebih panjang dapat meningkatkan kompleksitas fitur dan kebutuhan komputasi (Zielezinski dkk., 2017).

2.5 *One-Hot Encoding*

One-hot encoding adalah teknik yang digunakan untuk merepresentasikan data kategori menjadi bentuk numerik berupa vektor biner agar dapat diproses untuk algoritma komputasional. Dalam konteks klasifikasi, *one-hot encoding* digunakan sebagai target untuk merepresentasikan label kelas dari setiap sampel cDNA. Teknik ini menghasilkan matriks dua dimensi yang terdiri dari kombinasi angka 0 dan 1 dalam seluruh sekuens. Dengan setiap elemen vektor mewakili satu kategori untuk variabel dengan kategori label yaitu Alpha, Beta, Delta, Gamma, Omicron yang direpresentasikan dalam Tabel 2.1 berikut:

Tabel 2.1 Representasi *One-Hot Encoding*

Label	Target
Alpha	[1, 0, 0, 0, 0]
Beta	[0, 1, 0, 0, 0]
Delta	[0, 0, 1, 0, 0]
Gamma	[0, 0, 0, 1, 0]
Omicron	[0, 0, 0, 0, 1]

Tabel 2.1 menunjukkan setiap label direpresentasikan dalam bentuk *one-hot* digunakan sebagai target pada algoritma (Mwanga dkk., 2023).

2.6 *Min-Max Scaling*

Min-max scaling adalah teknik normalisasi data yang digunakan untuk

mengubah skala nilai pada setiap fitur (variabel) dalam *dataset* agar berada dalam rentang [0, 1]. Teknik ini melakukan transformasi linier terhadap nilai asli untuk menjaga proporsi antarfitur, sehingga hubungan antar variabel tetap terjaga setelah proses normalisasi. Dengan menyamakan skala antar fitur, algoritma dapat melakukan proses pembelajaran secara optimal tanpa bias terhadap fitur yang memiliki rentang nilai lebih besar, sehingga kinerja algoritma menjadi lebih akurat dan stabil (Dullah dkk., 2025).

Persamaan yang digunakan untuk melakukan normalisasi dengan metode *min-max scaling* adalah sebagai berikut:

$$X_{new} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{2.28}$$

dengan:

X : Nilai asli,

X_{new} : Nilai fitur setelah dinormalisasi,

X_{min} : Nilai minimum dari fitur,

X_{max} : Nilai maksimum dari fitur.

Normalisasi menggunakan *min-max scaling* penting dalam konteks *machine learning* karena semua nilai dalam *dataset* akan terdistribusi dalam rentang yang sama, sehingga menghindari masalah perbedaan skala yang dapat memengaruhi hasil analisis dan kinerja algoritma (Dullah dkk., 2025).

2.7 *Confusion Matrix*

Evaluasi model merupakan tahapan penting dalam pengembangan model klasifikasi untuk mengukur seberapa baik model dapat melakukan prediksi. Teknik yang banyak digunakan untuk mengevaluasi kinerja model klasifikasi adalah *confusion matrix* dan *classification report*. *Confusion matrix* juga sangat

bermanfaat ketika data bersifat tidak seimbang (*imbalanced dataset*). *Confusion matrix* merupakan matriks yang digunakan untuk menunjukkan kinerja algoritma klasifikasi dengan membandingkan hasil prediksi terhadap nilai aktual, sehingga memudahkan analisis terhadap banyak prediksi yang benar maupun salah untuk setiap kelas. Matriks ini berfungsi untuk menilai bagaimana algoritma mengenali kelas positif dan negatif dengan baik, serta untuk mendeteksi kemungkinan kesalahan klasifikasi (Dullah dkk., 2025).

Menurut Daqiqil (2021), kinerja model klasifikasi akan diukur menggunakan 4 komponen utama dalam *confusion matrix* pada Tabel 2.2:

Tabel 2.2 *Confusion Matrix*

		Data Prediksi	
		Positif	**Negatif**
Data Aktual	**Positif**	*True Positive* (TP)	*False Negative* (FN)
	Negatif	*False Positive* (FP)	*True Negative* (TN)

Tiap komponen dari *confusion matrix* pada Tabel 2.2 adalah:

1. *True Positive* (TP), yaitu data kelas positif yang diprediksi dengan benar sebagai positif.
2. *False Negative* (FN), yaitu data kelas positif yang diprediksi salah sebagai negatif.
3. *True Negative* (TN), yaitu data kelas negatif yang diprediksi dengan benar sebagai kelas negatif.
4. *False Positive* (FP), yaitu data kelas negatif yang diprediksi salah sebagai kelas positif.

Pada klasifikasi multikelas, perhitungan TP, FP, FN, dan TN dilakukan dengan pendekatan *one-versus-rest*. Setiap kelas secara bergantian dianggap sebagai kelas positif, sedangkan kelas lainnya dianggap sebagai kelas negatif. Dengan demikian,

nilai TP, FP, FN, dan TN dapat dihitung untuk setiap kelas, sehingga metrik evaluasi seperti akurasi, presisi, *recall*, dan F1-*score* dapat diperoleh pada masing-masing kelas.

Classification report menampilkan berbagai metrik penting yang digunakan untuk menilai kinerja model klasifikasi, seperti akurasi, presisi*, recall*, dan F1-*score*. Kategori-kategori pada *confusion matrix* digunakan sebagai dasar perhitungan metrik-metrik tersebut (Dullah dkk., 2025). Daqiqil Id (2021) menjelaskan bahwa metrik evaluasi pada model klasifikasi, seperti akurasi, presisi dihitung berdasarkan hubungan antara TP, TN, FP, dan FN. Akurasi merupakan perbandingan banyak prediksi yang benar terhadap seluruh prediksi yang dilakukan oleh model.

$$\text{Akurasi} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.29}$$

Persamaan (2.29) menunjukkan tingkat akurasi keseluruhan model dalam mengklasifikasikan data ke label yang benar. Sementara itu, presisi merupakan proporsi prediksi positif yang sesuai dengan kondisi sebenarnya.

$$\text{Presisi} = \frac{TP}{TP + FP} \tag{2.30}$$

Persamaan (2.30) menunjukkan bahwa semakin tinggi nilai presisi, semakin sedikit kesalahan prediksi positif yang dilakukan oleh model. Kemudian, *recall* menunjukkan kemampuan model dalam menemukan seluruh data kelas positif.

$$Recall = \frac{TP}{TP + FN} \tag{2.31}$$

Persamaan (2.31) menunjukkan seberapa banyak data positif yang berhasil dikenali oleh model, sehingga semakin tinggi nilai *recall* semakin sedikit data positif yang tidak terdeteksi. Sementara itu, F1-*score* merupakan rata-rata dari presisi dan *recall*

yang memberikan penilaian lebih seimbang terhadap kinerja model, terutama ketika distribusi kelas tidak seimbang.

$$\text{F1} - Score = 2 \times \frac{\text{Presisi} \times Recall}{\text{Presisi} + Recall} \tag{2.32}$$

Nilai F1-*score* yang mendekati 1 menunjukkan bahwa model memiliki keseimbangan yang baik antara ketepatan dan kemampuan mendeteksi data positif (Dullah dkk., 2025).

Pada setiap varian, nilai presisi, *recall*, dan *F1-score* dihitung untuk mengetahui kemampuan model dalam mengklasifikasikan masing-masing varian SARS-CoV-2 secara spesifik. Perhitungan pada setiap varian dilakukan karena kinerja model dapat berbeda pada setiap kelas. Setelah nilai evaluasi pada setiap kelas diperoleh, nilai tersebut dapat digunakan untuk menghitung nilai evaluasi secara keseluruhan. Nilai keseluruhan dihitung menggunakan *weighted average*, yaitu rata-rata yang memberikan bobot berdasarkan banyak data aktual atau *support* pada setiap kelas. Pendekatan ini digunakan karena jumlah data pada setiap kelas tidak selalu sama, sehingga setiap kelas memberikan kontribusi sesuai dengan proporsi jumlah datanya. Rumus *weighted average* dinyatakan sebagai berikut, (Géron, 2019):

$$Weighted\ Average = \frac{\sum_{i=1}^{p}(M_i \times Support_i)}{\sum_{i=1}^{p} Support_i} \tag{2.33}$$

Pada Persamaan (2.33), M_i menyatakan nilai metrik evaluasi pada varian ke-i, $Support_i$ menyatakan banyak data aktual pada varian ke$i$, dan p menyatakan banyaknya kelas. Dengan demikian, nilai *weighted average* dapat digunakan untuk menggambarkan kinerja model secara keseluruhan dengan tetap mempertimbangkan perbedaan banyak data pada setiap kelas.

2.8 Sekuens Nukleotida

Sekuens nukleotida merupakan urutan basa nitrogen penyusun materi genetik yang menyimpan informasi biologis suatu organisme atau virus (Jones & Pevzner, 2004). Pada virus SARS-CoV-2 informasi genetik tersimpan dalam genom RNA untai tunggal positif yang tersusun atas basa Adenin (A), Urasil (U), Sitosin (C), dan Guanin (G). Urutan basa tersebut menentukan struktur, fungsi, serta karakteristik biologis virus, termasuk kemampuan replikasi, infektivitas, dan proses evolusi melalui mutasi genetik. Perubahan pada urutan nukleotida dapat menyebabkan perubahan karakteristik virus dan menjadi salah satu faktor utama yang membedakan suatu varian SARS-CoV-2 dengan varian lainnya.

Sekuens virus RNA sering direpresentasikan dalam bentuk *complementary* DNA (cDNA) pada analisis bioinformatika untuk kebutuhan penyimpanan, pengolahan, dan analisis komputasi. Dalam representasi cDNA, basa Urasil (U) digantikan oleh Timin (T) sehingga sekuens dinyatakan menggunakan empat basa nukleotida, yaitu Adenin (A), Sitosin (C), dan Guanin (G) (Yang & Rao, 2021). Dengan demikian, sekuens tersebut dinyatakan sebagai rangkaian simbol A, T, C, dan G atau dalam istilah bioinformatika disebut *string* nukleotida (Jones & Pevzner, 2004).

Menurut Jamhuri dkk. (2025), dalam bidang bioinformatika, sekuens yang tersusun atas (A, T, C, G) tidak dapat langsung diproses oleh algoritma pembelajaran mesin termasuk jaringan saraf tiruan, karena model-model tersebut hanya dapat menerima *input* berupa data numerik. Tahap ini dapat dilakukan dengan berbagai teknik, seperti *k*-mer dan *one-hot encoding*. Representasi ini memungkinkan setiap posisi basa nukleotida dalam genom ditransformasikan

menjadi fitur numerik yang dapat dianalisis oleh algoritma komputasi. Dalam konteks penelitian bioinformatika, terutama pada klasifikasi varian SARS-CoV-2, tahap representasi ini berperan penting karena menentukan seberapa baik algoritma, seperti FFNN, karena dapat mengenali pola dan perbedaan antar varian berdasarkan variasi sekuens genom yang dianalisis.

2.9 Varian SARS-CoV-2

Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2) merupakan jenis virus yang menyebabkan penyakit *Coronavirus Disease* 2019 (COVID-19). Partikelnya memiliki bentuk bulat dengan diameter sekitar 80–160 nm (Yang dkk., 2020), namun pada permukaannya tertutupi oleh protein Spike (S) yang menonjol menyerupai mahkota (Singh dkk., 2021). Virus ini termasuk dalam genus *Betacoronavirus* dari keluarga *Coronaviridae* dan subfamili *Orthocoronavirinae* (Salehi-Vaziri dkk., 2022). Dengan panjang genom sekitar 26-32 kilobase (kb) yang menjadikan salah satu genom terpanjang di antara virus RNA lainnya (Yang & Rao, 2021).

Seiring berjalannya waktu, SARS-CoV-2 terus mengalami mutasi gen. Mutasi gen merupakan perubahan gen secara spontan dari partikel virus induk yang menjadi partikel virus turunannya (Parwanto, 2021). SARS-CoV-2 memiliki tingkat mutasi yang tinggi. Mutasi pada bagian-bagian tertentu dari genom, terutama pada gen *spike* (S), dapat menghasilkan varian baru yang merupakan target utama sistem imun dan vaksin (Jamhuri dkk., 2025). Setiap varian mengalami dan memiliki mutasi dengan karakteristik berbeda pada protein *spike* yang dapat memengaruhi ikatan virus ke reseptor enzim pengubah angiotensin 2 (ACE2) pada sel manusia, meningkatkan daya infeksi dan kemampuan virus untuk menghindar

dari netralisasi antibodi (Choi & Smith, 2021).

Mutasi pada SARS-CoV-2 juga memengaruhi penularan pada manusia dan kemampuan virus dalam menginfeksi berbagai spesies lain. Penelitian Li dkk. (2024), menunjukkan bahwa interaksi antara protein *spike* dan reseptor ACE2 berperan penting dalam penularan antar spesies. Hal ini menyebabkan SARS-CoV-2 tidak hanya ditemukan pada manusia, tetapi juga pada berbagai jenis mamalia lain. Penyebaran SARS-CoV-2 yang luas dan bersifat dua arah antara manusia dan hewan meningkatkan kemungkinan munculnya varian baru yang mampu beradaptasi pada berbagai spesies inang (*host*). Selain itu, penelitian tersebut juga melakukan pengujian terhadap kultur sel dari 49 spesies mamalia yang berbeda dan menunjukkan bahwa beberapa mutasi tertentu mampu meningkatkan kemampuan virus dalam berikatan dengan reseptor ACE2 pada berbagai spesies. Temuan ini menunjukkan bahwa mutasi SARS-CoV-2 dapat memengaruhi jangkauan inang (*host range*) virus. Oleh karena itu, klasifikasi varian SARS-CoV-2 tidak hanya relevan pada manusia, tetapi juga pada berbagai spesies mamalia lain serta menjadi perhatian penting dalam pemantauan perkembangan SARS-CoV-2.

Genom SARS-CoV-2 dari berbagai spesies *host* dapat memiliki variasi karakteristik sekuens nukleotida akibat proses mutasi dan adaptasi virus pada masing-masing inang. Coronavirus memiliki tingkat mutasi dan rekombinasi genetik yang tinggi sehingga mampu beradaptasi dan menginfeksi berbagai spesies mamalia (Yang dkk., 2020). Beberapa mutasi pada protein *spike* SARS-CoV-2 mampu meningkatkan kemampuan virus dalam berikatan dengan reseptor ACE2 pada berbagai spesies mamalia serta memengaruhi *host tropism* dan penularan antar spesies (Li dkk., 2024). Mutasi yang terjadi selama proses adaptasi virus dapat

menyebabkan perubahan komposisi dan pola sekuens nukleotida pada genom SARS-CoV-2. Perbedaan kemampuan adaptasi virus terhadap berbagai *host* tersebut menunjukkan bahwa data genom SARS-CoV-2 dari manusia dan hewan memungkinkan memiliki variasi karakteristik pola sekuens nukleotida.

WHO mengelompokkan varian tersebut menjadi tiga kategori untuk diutamakan pemantauan dan penelitiannya yaitu *Variants of Interest* (VoI), *Variants under Monitoring* (VUM), dan *Variants of Concern* (VoC) (Susilo dkk., 2022). VoI atau varian yang menjadi perhatian didefinisikan sebagai varian SARS-CoV-2 yang mengalami perubahan genetik dengan memengaruhi pada karakteristik virus. Perubahan ini menyebabkan adanya peningkatan penularan, keparahan penyakit, kemampuan menghindari kekebalan tubuh. Penularan ini berpusat di komunitas yang signifikan atau beberapa kelompok orang yang terinfeksi di berbagai negara disertai peningkatan prevalensi secara relatif dari waktu ke waktu (Choi & Smith, 2021). Varian Lambda dan Mu ditetapkan sebagai VoI.

VUM adalah jenis varian dengan perubahan genetik yang belum diketahui mengenai adanya peningkatan penularan, dampak morbiditas dan mortalitas, serta virulensinya. Kappa, Iota, Eta, Epsilon, Zeta dan Theta ditetapkan sebagai dari VUM. Kelompok varian ini terbukti dapat menurunkan efektivitas terapi dengan menggunakan antibodi monoklonal dan plasma konvalesen (Susilo dkk., 2022). Varian SARS-CoV-2 dikategorikan sebagai VoC apabila memenuhi kriteria VoI dan telah terbukti memberikan dampak signifikan terhadap kesehatan masyarakat. Dampak tersebut meliputi adanya peningkatan kemampuan penularan, peningkatan virulensi atau perubahan pada manifestasi klinis penyakit, dan adanya penurunan efektivitas tindakan kesehatan (Choi & Smith, 2021).

VoC merupakan varian dengan *fenotipe* yang memberikan dampak negatif terhadap prognosis penyakit disertai peningkatan transmisi yang lebih signifikan dibandingkan dengan VoI. Kondisi ini menimbulkan kekhawatiran mengenai penurunan efektivitas vaksin yang ada (Susilo dkk., 2022). Rentang inang (*host range*) pada varian-varian baru SARS-CoV-2, terutama VoC dan VoI, masih terus diteliti karena kemampuan adaptasi virus pada berbagai spesies belum sepenuhnya dipahami. Hal ini menunjukkan bahwa mutasi pada varian VoC tidak hanya berdampak pada peningkatan penularan antar manusia, tetapi juga berpotensi memengaruhi kemampuan penularan antar spesies (Li dkk., 2024).

Varian yang termasuk dalam kategori VoC antara lain:

1. Alpha (B.1.1.7) terdeteksi pertama kali di Inggris pada 14 September 2020 dilengkapi mutasi seperti N501Y dan P681H pada protein *spike* yang dapat meningkatkan afinitas terhadap reseptor ACE2 dan efisiensi penularan (Parwanto, 2021).
2. Beta (B.1.351) pertama kali diidentifikasi di Afrika Selatan pada Mei 2020 dengan memiliki mutasi K417N, E484K, dan N501Y yang dapat menurunkan efikasi antibodi penetral (Susilo dkk., 2022).
3. Delta (B.1.617.2) terdeteksi pertama kali di India pada Oktober 2020 ditandai dengan adanya mutasi L452R dan P681R yang berkontribusi terhadap peningkatan daya infeksi dan penurunan efektivitas beberapa terapi antibodi monoklonal (Susilo dkk., 2022).
4. Gamma (P.1) ditemukan di Brazil pada Januari 2021 dan memiliki mutasi seperti beta yakni K417T, E484K, serta N501Y yang dapat menyebabkan respons imun, termasuk menurunkan efektivitas antibodi penetral

monoklonal (Susilo dkk., 2022).

5. Omicron (B.1.1.529) terdeteksi di Afrika Selatan pada November 2021, memiliki lebih dari 30 mutasi pada protein spike yang menyebabkan *immune escape* sehingga mengurangi perlindungan dari vaksin maupun infeksi sebelumnya (Susilo dkk., 2022).

VoC dikategorikan berbahaya karena memiliki mutasi genetik yang secara signifikan meningkatkan kemampuan virus untuk menular, menginfeksi ulang, dan menghindari respons imun tubuh (Susilo dkk., 2022). Menurut Özüdoğru dkk. (2022), varian seperti Delta dan Omicron memiliki kemampuan kekebalan yang tinggi serta potensi infeksi ulang yang jauh lebih besar dibandingkan varian sebelumnya. Selain itu, penelitian Choi & Smith (2021) dan Susilo dkk. (2022) menjelaskan bahwa varian VoC seperti Beta, Gamma, dan Delta memiliki mutasi seperti E484K, N501Y, L452R, dan P681R yang menyebabkan peningkatan afinitas terhadap reseptor ACE2 serta penurunan efektivitas vaksin dan terapi antibodi monoklonal. Varian Omicron (B.1.1.529) memiliki lebih dari 30 mutasi pada gen S (*spike*), menjadikannya varian dengan tingkat transmisi paling tinggi dan risiko reinfeksi terbesar sepanjang pandemi. Kombinasi mutasi tersebut tidak hanya meningkatkan daya infeksi virus, tetapi juga menurunkan kemampuan sistem imun tubuh untuk mengenali dan menetralisasi virus.

2.10 Kajian Integrasi Topik dengan Hadis

Islam menempatkan ilmu sebagai sarana yang harus diikuti dengan amal. Ilmu harus diwujudkan dalam bentuk penerapan yang dapat memberikan manfaat bagi manusia. Sebagaimana telah dijelaskan pada bagian latar belakang mengenai keutamaan menuntut ilmu dan perintah untuk mengamalkannya sebagai sarana

untuk mewujudkan nilai-nilai kebaikan dan memperkuat keimanan kepada Allah SWT. Dalam konteks penelitian ini, penerapan algoritma FFNN merupakan bentuk pengamalan ilmu pengetahuan. Penerapan ini memberikan manfaat bagi manusia, terutama dalam bidang kesehatan, deteksi penyakit, dan pengambilan kebijakan medis yang lebih cepat dan akurat.

Sejalan dengan hal tersebut, terdapat sebuah perkataan masyhur yang dinukil oleh Imam al-Nawawī dari Imam al-Syāfi'ī, yaitu:

بِالْعِلْمِ فَعَلَيْهِ أَرَادَهُمَا وَمَنْ ،بِالْعِلْمِ فَعَلَيْهِ الْأَخِرَةَ أَرَادَ وَمَنْ ،بِالْعِلْمِ فَعَلَيْهِ الدُّنْيَا أَرَادَ مَنْ

Artinya: "*Barangsiapa yang menginginkan dunia maka hendaklah berilmu. Barangsiapa yang menginginkan akhirat, maka hendaklah dengan ilmu. Barangsiapa yang menginginkan keduanya, maka hendaklah dengan ilmu*." (*Imam al-Syāfi'ī, dinukil oleh Imam al-Nawawī dalam al-Majmū' Syarh al-Muhadzdzab, Muqaddimah)*

Perkataan ini merupakan nasihat Imam al-Syāfi'ī yang menekankan pentingnya ilmu sebagai pondasi seluruh aspek kehidupan. Menurut Imam al-Nawawī dalam *al-Majmū' Syarh al-Muhadzdzab* menjelaskan bahwa ilmu merupakan jalan yang mengantarkan setiap amal agar diterima oleh Allah SWT. Ilmu menjadi pembimbing, sedangkan amal menjadi buah dari ilmu tersebut. Dalam konteks penelitian, algoritma FFNN bekerja dengan proses pembelajaran, pengenalan pola, dan klasifikasi. Proses tersebut sejalan dengan konsep ilmu yang membimbing tindakan, dengan FFNN berperan dalam proses pengelompokan sekuens genom sehingga dapat mengidentifikasi perbedaan karakteristik varian SARS-CoV-2 seperti Alpha, Beta, Delta, Gamma, dan Omicron secara akurat. Ini menunjukkan bagaimana ilmu teknologi dapat diimplementasikan untuk menyelesaikan persoalan kemanusiaan.

Perkataan tersebut memiliki relevansi yang kuat dengan penelitian ini karena

bukan hanya aktivitas ilmiah, tetapi merupakan penerapan dan aksi nyata sebagai wujud keimanan dan pengabdian kepada Allah SWT. Hal ini sejalan dengan sabda Rasulullah SAW (al-Tabarānī, 2012), yaitu:

لِلنَّاسِ أَنْفَعُهُمْ النَّاسِ خَيْرُ

Artinya: *"Sebaik-baik manusia adalah yang paling bermanfaat bagi manusia (lainnya).'' (HR. al-Tabarānī, dalam al-Mu'jam al-Awsat, no. 5787; al-Haitsami, Majma' al-Zawāid, 8/16)*

Hadis ini diriwayatkan oleh al-Ṭabarānī yang dikomentari oleh *al-Haitsami, Majma' al-Zawāid* yang menegaskan bahwa kemuliaan manusia di sisi Allah SWT tidak hanya diukur dari ilmunya, melainkan dari sejauh mana ilmu tersebut memberi manfaat untuk orang lain. Dengan mengaplikasikan pengetahuan kecerdasan buatan, penelitian ini mencerminkan amal dengan ilmu yang bernilai ibadah. Namun, Islam mengingatkan bahwa tidak semua ilmu membawa keberkahan. Ilmu yang tidak diamalkan atau tidak memberikan manfaat dapat menjadi penyebab kerugian bagi pemiliknya. Oleh karena itu, Rasulullah SAW memohon perlindungan dari ilmu yang tidak bermanfaat, sebagaimana dalam sabdanya (Ibn Majāh, 2019) yaitu:

يُسْتَجَابُ لاَ دَعْوَةٍ وَمِنْ تَشْبَعُ اَ نَفْسٍ وَمِنْ يَخْشَعُ لاَ قَلْبٍ وَمِنْ يَنْفَعُ لاَ عِلْمٍ مِنْ بِكَ أَعُوذُ إِنِّي اللَّهُمَّ لَهَا

Artinya: *"Ya Allah, aku berlindung kepada-Mu dari ilmu yang tidak bermanfaat, dari hati yang tidak khusyu', dari jiwa yang tidak pernah puas, dan dari doa yang tidak dikabulkan." (HR. Ibn Majāh, Sunan Ibn Mājah, Kitāb al-Muqaddimah, no.250)*

Hadis ini menunjukkan bahwa Nabi Muhammad SAW tidak hanya meminta tambahan ilmu, tetapi juga memohon perlindungan dari ilmu yang tidak memberikan manfaat. Dalam Sharh Sahīh Muslim, al-Nawawī menjelaskan bahwa ilmu yang tidak bermanfaat adalah ilmu yang tidak mendekatkan seseorang kepada

ketaatan, tidak menambah kedekatan kepada Allah, serta tidak memberi manfaat bagi manusia. Dengan demikian, integrasi nilai Islam dalam penelitian ini menegaskan bahwa ilmu pengetahuan tidak hanya pada teori, tetapi diwujudkan dalam implementasi nyata dengan menerapkan algoritma FFNN dalam mendukung deteksi dan analisis sekuens genom untuk klasifikasi varian SARS-CoV-2.

BAB III

METODE PENELITIAN

3.1 Jenis Penelitian

Penelitian ini menggunakan pendekatan kuantitatif eksperimental yang berfokus pada pengujian kinerja algoritma *Feedforward Neural Network* dalam melakukan klasifikasi dengan mengidentifikasi dan membedakan varian SARS-CoV-2 berdasarkan fitur sekuens genom. Pendekatan kuantitatif dipilih karena seluruh proses analisis melibatkan data numerik hasil transformasi dari sekuens cDNA melalui tahapan *preprocessing* dan ekstraksi fitur.

3.2 Data dan Sumber Data

Data yang digunakan dalam penelitian ini berupa data sekunder, yaitu sekuens genom virus SARS-CoV-2 yang diperoleh dari *National Center for Biotechnology Information* (NCBI). Data yang diambil merupakan sekuens genom lengkap (*complete genome*) virus SARS-CoV-2 yang dikategorikan sebagai *Variants of Concern* (VoC), yaitu Alpha, Beta, Delta, Gamma, dan Omicron. Banyak data yang digunakan 9.104 sekuens dengan pembagian pada setiap varian sebanyak 2.000 sekuens untuk varian Alpha, Delta, dan Gamma. Sementara itu, varian Beta menggunakan sebanyak 1.235 sekuens dan varian Omicron menggunakan sebanyak 1.869 sekuens.

Seluruh data sekuens disimpan dalam format FASTA yang terdiri atas dua bagian utama, yaitu header dan sekuens nukleotida. Bagian header diawali dengan simbol “>” yang memuat informasi identitas sekuens, seperti *accession number*, nama isolat, inang, lokasi dan waktu pengambilan sampel, serta informasi gen penyusun genom virus. Sementara itu, bagian sekuens nukleotida berisi urutan basa

penyusun genom SARS-CoV-2 yang terdiri atas Adenin (A), Timin (T), Sitosin (C), dan Guanin (G). Meskipun SARS-CoV-2 merupakan virus RNA untai tunggal positif, data genom yang tersedia pada basis data NCBI umumnya direpresentasikan dalam bentuk *complementary* DNA (cDNA) (Singh dkk., 2021).

3.3 Tahapan Penelitian

1. Analisis Deskripsi *Dataset*

   Menjelaskan sumber data, jumlah data yang digunakan, dan proses pengunduhan data.

2. *Exploratory* Data *Analysis* (EDA)

   EDA merupakan tahap awal yang dilakukan untuk memahami karakteristik *dataset* sebelum *preprocessing* dan pemodelan. Analisis yang dilakukan meliputi distribusi banyak data untuk melihat keseimbangan antar kelas, identifikasi struktur data, analisis sekuens duplikat, identifikasi karakter ambigu, identifikasi karakter tidak valid, serta pemeriksaan *missing value*.

3. *Preprocessing* Data

   Preprocessing dilakukan untuk memastikan bahwa data dalam kondisi bersih, lengkap, dan sesuai dengan format input yang dibutuhkan untuk analisis lebih lanjut. Langkah-langkah *preprocessing* meliputi:

   a. *Cleaning* Data

      Terdapat beberapa proses *cleaning* yang dilakukan yaitu penghapusan sekuens duplikat dan penghapusan karakter nukleotida tidak valid.

   b. Ekstraksi Fitur Menggunakan K-Mer

      Setiap sekuens akan diubah menjadi kumpulan k-mer dengan panjang (k) tertentu yang merepresentasikan pola dari urutan nukleotida dari sekuens

cDNA secara berurutan.

4. *One-Hot Encoding*

   Mengodekan setiap kelas ke dalam sebuah vektor dengan panjang yang sama dengan jumlah yang tersedia. Vektor ini berfungsi sebagai target atau *output*. Setiap elemen vektor terdiri dari angka 0 dan 1, dengan angka 1 menandakan kelas target dari sekuens tersebut yang sesuai berdasarkan Tabel 2.1.

5. Pembagian Data

   Tahap ini dilakukan setelah proses *preprocessing* dengan pembagian data dibagi menjadi 3 bagian utama, yaitu data *training*, *validation*, dan *testing* berdasarkan penelitian Jamhuri dkk. (2025). Pembagian data dilakukan dengan rasio 70% data *training*, 15% data *validation* dan 15% data *testing*.

6. Normalisasi Data

   Digunakan metode *min-max scaling* untuk mengubah rentang nilai setiap fitur k-mer agar berada pada rentang $[0, 1]$. Normalisasi ini bertujuan agar skala antar fitur sama, sehingga tidak terjadi bias karena adanya perbedaan rentang nilai antar fitur. Rumus yang digunakan dalam proses normalisasi merujuk pada Persamaan (2.28). Hasil dari proses ini digunakan sebagai dasar dalam pembentukan fitur numerik yang akan menjadi *input* pada algoritma FFNN.

7. Pengujian Parameter Model

   Pada penelitian ini, model FFNN dilatih menggunakan beberapa parameter pelatihan (*hyperparameter*) untuk mengatur proses pembelajaran model. Parameter yang digunakan meliputi nilai *k* pada ekstraksi fitur *k*-mer, banyak neuron *input layer*, banyak neuron *hidden layer,* banyak neuron *output layer*, aktivasi *hidden layer*, aktivasi *output layer*, jumlah *epoch*, *batch size*,

learning rate, *optimizer*, fungsi *loss*, *early stopping* dan *verbose*.

8. Implementasi *Feedforward Neural Network* (FFNN)

   Langkah-langkah algoritma FFNN dalam penelitian ini sebagai berikut:

   a. Lapisan *Input*

      Nilai neuron pada lapisan *input* disesuaikan dengan banyaknya fitur hasil ekstraksi *k*-mer dari sekuens genom SARS-CoV-2 yang telah dinormalisasi. Setiap neuron pada lapisan *input* hanya berfungsi untuk meneruskan nilai masukan ke lapisan berikutnya tanpa fungsi aktivasi, sebagaimana dinyatakan pada Persamaan (2.1).

   b. Lapisan Tersembunyi

      Setiap neuron pada lapisan *hidden* menerima sinyal dari seluruh neuron pada lapisan sebelumnya dan menghitung kombinasi linier antara nilai masukan, bobot, serta bias, menggunakan Persamaan (2.2) hingga (2.5). Hasil kombinasi linier kemudian diproses menggunakan fungsi aktivasi *Rectified Linear Unit* (ReLU) pada Persamaan (2.6) yang menghasilkan keluaran seperti pada Persamaan (2.7).

   c. Lapisan *Output*

      Setiap neuron pada lapisan *output* menghitung nilai kombinasi linier dari seluruh keluaran lapisan *hidden* menggunakan Persamaan (2.8) hingga Persamaan (2.11). Karena penelitian ini merupakan klasifikasi multikelas, fungsi aktivasi *softmax* digunakan sesuai Persamaan (2.12) dan (2.14) untuk menghasilkan distribusi probabilitas pada setiap kelas.

   d. Perhitungan Fungsi *Loss*

      Menghitung kesalahan antara hasil prediksi dengan nilai target sebenarnya

menggunakan fungsi *cross-entropy loss* pada Persamaan (2.15) hingga Persamaan (2.18). Fungsi ini membandingkan distribusi probabilitas keluaran dari fungsi *softmax* dengan label sebenarnya yang telah direpresentasikan dalam bentuk *one-hot encoding*. Nilai *loss* yang semakin kecil menunjukkan bahwa model semakin baik dalam melakukan prediksi.

e. Proses *Backpropagation*

Proses *backpropagation* menghitung gradien dari fungsi *loss* terhadap setiap bobot menggunakan aturan rantai (*chain rule*). Perhitungan menggunakan Persamaan (2.19) hingga Persamaan (2.25).

f. *Update* Parameter

Pembaruan bobot menggunakan *TensorFlow* pada Persamaan (2.26) dan Persamaan (2.27) untuk meminimalkan nilai fungsi *loss*.

Algoritma FFNN dilatih menggunakan data *training*, sedangkan data *validation* digunakan untuk mengamati kinerja model selama proses pelatihan, menyesuaikan hiperparameter, serta menerapkan *early stopping* untuk mencegah *overfitting*.

9. Evaluasi Model

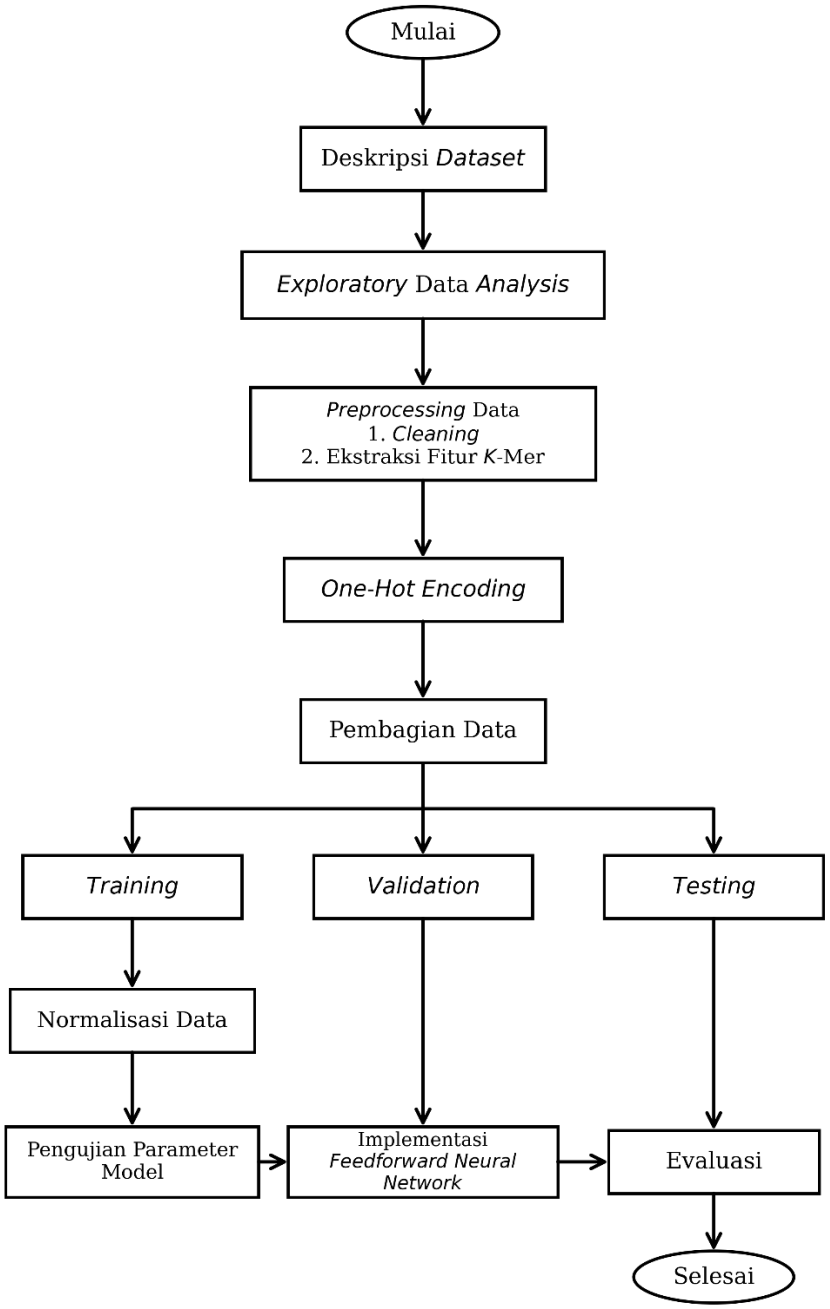
Model yang telah melalui proses pelatihan dievaluasi menggunakan data *testing* yang tidak digunakan pada proses *training*, sehingga hasil evaluasi yang diperoleh bersifat objektif dan menjelaskan kemampuan model dalam menghasilkan prediksi. Evaluasi dilakukan untuk mengetahui seberapa baik algoritma dalam mengklasifikasikan sekuens cDNA. *Confusion Matrix* digunakan untuk mengevaluasi kinerja model klasifikasi dengan membandingkan hasil prediksi model terhadap label. Kinerja model diukur

berdasarkan Tabel 2.2. Hasil perhitungan *Confusion Matrix* dengan menampilkan berbagai metrik penting yang digunakan untuk menilai kinerja model klasifikasi menggunakan Persamaan (2.29) hingga Persamaan (2.33). Keberhasilan model ditentukan berdasarkan beberapa indikator hasil evaluasi. Menurut Géron (2019), evaluasi model klasifikasi tidak hanya ditinjau dari nilai akurasi, tetapi juga dari nilai presisi, *recall*, F1-*score*, *confusion matrix* dan efisiensi waktu komputasi. Selain itu, Gorunescu (2011) mengelompokkan tingkat akurasi klasifikasi ke dalam beberapa kategori, yaitu *excellent classification* (90%–100%), *good classification* (80%–90%), *fair classification* (70%–80%), *poor classification* (60%–70%), dan *failure* (50%–60%). Berdasarkan referensi tersebut, model pada penelitian ini dikatakan berhasil apabila memenuhi kriteria berikut:

1. Nilai *training loss* dan *validation loss* menurun secara konvergen selama proses pelatihan serta tidak menunjukkan indikasi *overfitting* yang signifikan.
2. Nilai akurasi, presisi, *recall*, dan *F1-score* menunjukkan hasil yang tinggi dan konsisten pada data pengujian.
3. Berdasarkan kategori Gorunescu (2011), model mencapai kategori *good classification* atau lebih tinggi, yaitu memperoleh akurasi minimal 80%.
4. *Confusion matrix* menunjukkan *false positive* dan *false negative* yang relatif rendah dibandingkan banyak data pengujian.

3.4 Diagram Alir Penelitian

Tahapan penelitian yang dilakukan pada penelitian ini disajikan dalam bentuk diagram alir sebagaimana ditunjukkan pada Gambar 3.1:



Gambar 3.1 Diagram Alir Penelitian

BAB IV

HASIL DAN PEMBAHASAN

4.1 Deskripsi Data

Dataset yang digunakan dalam penelitian ini berupa data sekunder yaitu sekuens genom virus SARS-CoV-2 yang diperoleh dari *National Center for Biotechnology Information* (NCBI). *Dataset* tersebut merupakan data mentah dengan sekuens genom lengkap (*complete genome*) SARS-CoV-2 yang terdiri dari lima varian yang termasuk dalam kategori *Variants of Concern* (VoC), yaitu Alpha, Beta, Gamma, Delta, dan Omicron. Seluruh *dataset* yang digunakan sebanyak 9.104 sekuens yang terdiri dari 2.000 sekuens pada masing-masing varian Alpha, Delta, dan Gamma. Sementara itu, untuk varian Beta sebanyak 1.235 sekuens dan varian Omicron 1.869 sekuens. Seluruh *dataset* disimpan dalam format FASTA.

Pada proses pengambilan data tidak dilakukan penyaringan khusus terhadap inang (*host*). Hal ini karena fokus utama penelitian adalah mengklasifikasikan varian SARS-CoV-2 berdasarkan karakteristik genom virus, bukan berdasarkan jenis *host*. Namun, d*ataset* pada penelitian ini didominasi oleh *host* manusia, sedangkan data dari *host* lain hanya sebagian kecil. *Host* merupakan organisme tempat virus diisolasi, seperti manusia, hewan atau organisme lain yang terinfeksi. SARS-CoV-2 memiliki kemampuan penularan antar spesies dan dapat menginfeksi berbagai mamalia melalui interaksi antara protein *spike* dan reseptor ACE2 (Li dkk., 2024). Selain itu, sekuens genom yang berasal dari *host* berbeda memungkinkan adanya variasi karakteristik data, termasuk panjang dan pola sekuens genom. Dengan tidak dibatasinya jenis *host*, model klasifikasi yang

dibangun diharapkan mampu mengenali pola genom yang menjadi karakteristik setiap varian SARS-CoV-2.

Kesesuaian *dataset* tetap dijaga melalui penerapan beberapa penyaringan lain pada tahap pengambilan data. Penyaringan yang diterapkan antara lain berdasarkan *Virus*/*Taxonomy* dan *Pango lineage*. *Virus*/*Taxonomy* digunakan untuk memastikan bahwa seluruh data yang diunduh merupakan virus SARS-CoV-2. Sementara itu, *Pango lineage* digunakan untuk mengidentifikasi dan mengelompokkan dengan memasukkan kode *lineage* spesifik untuk masing-masing varian. Sebagai contoh, varian Alpha diidentifikasi menggunakan kode *lineage* B.1.1.7, varian Beta menggunakan B.1.351, varian Gamma menggunakan P.1, varian Delta menggunakan B.1.617.2, serta varian Omicron menggunakan B.1.1.529. Penerapan ini bertujuan untuk memperoleh *dataset* yang terstruktur berdasarkan varian sehingga dapat digunakan sebagai kelas dalam proses klasifikasi.

Pengambilan data juga memperhatikan karakter ambigu pada sekuens nukleotida dengan menerapkan filter *Ambiguous Characters*. Karakter ambigu seperti huruf "N" yang menunjukkan basa nukleotida tidak dapat diidentifikasi secara spesifik dalam hasil sekuensing. Dalam penelitian ini, dipilih nilai $N = 0$ agar sekuens yang digunakan merupakan kualitas data yang terjamin. Namun, filter ini hanya menghilangkan karakter "N", dan tidak otomatis menghapus seluruh karakter ambigu IUPAC yang tidak digunakan dalam ekstraksi fitur penelitian ini.

SARS-CoV-2 merupakan virus dengan genom RNA untai tunggal positif. Namun, data pada NCBI umumnya telah dikonversi ke dalam bentuk *complementary* DNA (cDNA) untuk memudahkan penyimpanan dan analisis komputasi (Singh dkk., 2021). Selain itu, Yang & Rao (2021) menjelaskan bahwa

genom RNA dikonversi menjadi DNA komplementer agar lebih stabil dengan metode analisis genom berbasis DNA. Oleh karena itu, sekuens yang digunakan pada penelitian ini berupa sekuens nukleotida hasil representasi cDNA yang tersusun atas empat basa nukleotida, yaitu Adenin (A), Timin (T), Sitosin (C), dan Guanin (G).

Dataset yang digunakan dalam penelitian ini disajikan pada Tabel 4.1 melalui contoh data awal sebelum tahap pengolahan data. Setiap baris pada *dataset* memuat identitas sekuens, urutan nukleotida, panjang genom, serta label varian yang digunakan sebagai kelas dalam proses klasifikasi.

Tabel 4.1 Cuplikan *Dataset* Setiap Varian

ID Sekuens	Sekuens	Panjang Sekuens	Varian
MW904718.1	AGATCTGTTCTCTAAACGAACTT…	29763	Alpha
ON647499.1	TTGTAGATCTGTTCTCTAAACGA…	29758	Beta
MZ863014.1	ATAACTAATTACTGTCGTTGACA…	29638	Delta
OM897974.1	TCGTCCGTGTTGCAGCCGATCAT…	29531	Gamma
PV800150.1	ATTAAAGGTTTATACCTTCCCAG…	29885	Omicron

4.2 Pengolahan Data

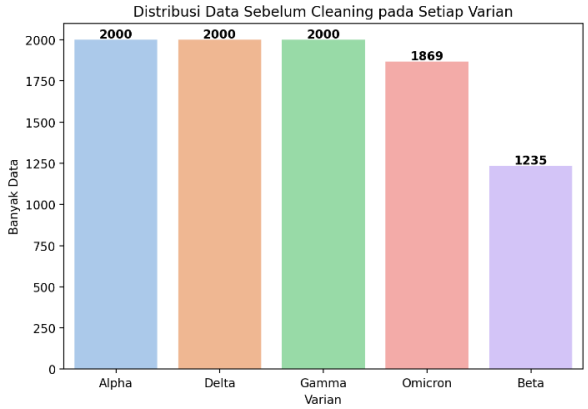
4.2.1 *Exploratory* Data *Analysis* (EDA)

EDA merupakan tahapan awal dalam pengolahan data untuk mengidentifikasi kualitas data, struktur data, serta kesesuaian *dataset* dengan kebutuhan proses klasifikasi yang meliputi beberapa analisis, yaitu:

1. Distribusi Data

   Tahap awal EDA dilakukan dengan membaca dan menggabungkan seluruh data sekuens genom SARS-CoV-2. Data yang digunakan dalam penelitian ini terdiri dari lima file FASTA yang masing-masing merepresentasikan varian Alpha, Beta, Gamma, Delta, dan Omicron. Hasil penggabungan data menunjukkan bahwa *dataset* awal yang digunakan dalam penelitian ini

sebanyak 9.104 sekuens genom SARS-CoV-2. Distribusi data pada masing-masing varian ditunjukkan pada Gambar 4.1.



Gambar 4.1 *Distribusi Data Sebelum* Cleaning *pada Setiap Varian*

Pada Gambar 4.1, menunjukkan bahwa terdapat perbedaan data pada setiap varian. Perbedaan distribusi data tersebut menunjukkan bahwa *dataset* yang digunakan belum sepenuhnya seimbang, karena jumlah sekuens pada varian Beta lebih sedikit dibandingkan dengan varian lainnya. Namun setiap kelas masih memiliki banyak data yang cukup untuk digunakan pada proses pelatihan dan evaluasi model. Distribusi ini hanya menggambarkan kondisi awal *dataset* sebelum dilakukan pengolahan data dan *preprocessing*.

2. Identifikasi Struktur Data

   Identifikasi struktur data dilakukan untuk memahami format dan isi data yang digunakan. Data disimpan dalam format FASTA yang terdiri dari *header* dan sekuens nukleotida. *Header* pada setiap sekuens diawali dengan simbol “>” yang memuat informasi seperti *accession number*, nama isolat virus, inang (*host*), lokasi geografis, waktu pengambilan sampel, serta keterangan terkait jenis sekuens. Berdasarkan hasil terhadap beberapa data sampel diketahui terdapat variasi penulisan informasi pada *header*, yang ditunjukkan pada Tabel 4.2.

Tabel 4.2 Informasi Sekuens SARS-CoV-2

Varian	Header
Alpha	>MW904718.1 |Severe acute respiratory syndrome coronavirus 2 isolate SARS-CoV-2/human/USA/TX-CDC-QDX23014358/2021, complete genome
Beta	>ON647499.1 |Severe acute respiratory syndrome coronavirus 2 isolate SARS-CoV-2/human/SouthAfrica/NHLS-UCT-GS-9348/2020, complete genome
Delta	>MZ863014.1 |Severe acute respiratory syndrome coronavirus 2 isolate SARS-CoV-2/human/USA/CA-CDC-LC0145899/2021 ORF1ab polyprotein (ORF1ab), ORF1a polyprotein (ORF1ab), surface glycoprotein (S), ORF3a protein (ORF3a), envelope protein (E), membrane glycoprotein (M), ORF6 protein (ORF6), ORF7a protein (ORF7a), and ORF7b (ORF7b) genes, complete cds; ORF8 gene, complete sequence; and nucleocapsid phosphoprotein (N) and ORF10 protein (ORF10) genes, complete cds
Gamma	>OM897974.1 |Severe acute respiratory syndrome coronavirus 2 isolate SARS-CoV-2/human/USA/UW21021968527/2021 ORF1ab polyprotein (ORF1ab), ORF1a polyprotein (ORF1ab), surface glycoprotein (S), ORF3a protein (ORF3a), envelope protein (E), membrane glycoprotein (M), ORF6 protein (ORF6), ORF7a protein (ORF7a), ORF7b (ORF7b), ORF8 protein (ORF8), nucleocapsid phosphoprotein (N), and ORF10 protein (ORF10) genes, complete cds
Omicron	> PV800150.1 |Severe acute respiratory syndrome coronavirus 2 isolate SARS-CoV-2/human/USA/WA1/2020-MA clone SARS-CoV-2-BA.5_MA, complete genome

Pada Tabel 4.2, diketahui varian Alpha, Beta, dan Omicron secara langsung memberikan keterangan *complete genome* yang menunjukkan bahwa sekuens merupakan genom virus secara utuh (*whole genome*). Sementara itu, pada varian Delta dan Gamma *header* tidak secara langsung memberikan istilah *complete genome*. Perbedaan penulisan pada *header* tersebut menunjukkan adanya variasi format penyajian data pada NCBI, dengan sebagian data disubmit dalam bentuk genom utuh (*whole genome*) dan sebagian lainnya bentuk genom teranotasi (*annotated genome*). Kedua format tersebut tetap merepresentasikan sekuens secara lengkap, sehingga dapat digunakan dalam analisis yang sama.

Informasi pada *header* juga menunjukkan inang (*host*) atau keterangan sumber isolat virus. Sebagian besar data menunjukkan *human* sebagai inang. Berdasarkan hasil analisis, ditunjukkan distribusi *host* pada Tabel 4.3.

Tabel 4.3 Distribusi *Host*

***Host*/Keterangan**	**Banyak Sekuens**
Human	9.082
Tikus	9
Virus Rekayasa Lab	11
Kucing	1
Monyet	1

Berdasarkan Tabel 4.3, diketahui bahwa sebagian besar data berasal dari *host human* sebanyak 9.082 sekuens. Selain itu, terdapat sebagian kecil data yang berasal dari *host* lain, yaitu tikus, kucing, dan sel vero yang berasal dari monyet. Terdapat kategori ”virus rekayasa labolatorium” yang bukan *host* biologis, melainkan keterangan sumber atau kondisi isolat pada data. Oleh karena itu, kategori tersebut dipisahkan secara makna dari *host* biologis. Meskipun sebagian besar data didominasi oleh *host human*, keberadaan data dari *host* lain memberikan variasi karakteristik genom yang dapat digunakan untuk menguji kemampuan model dalam mengenali pola genom SARS-CoV-2 pada data dengan karakteristik yang lebih beragam.

3. Identifikasi Sekuens Duplikat

   Dilakukan pemeriksaan terhadap kemungkinan adanya sekuens yang memiliki urutan nukleotida identik atau memiliki urutan yang sama lebih dari satu kali. Hal ini disebabkan adanya data yang memiliki urutan nukleotida yang sama tetapi baris yang berbeda. Berdasarkan hasil analisis, ditemukan sebanyak 525 sekuens yang memiliki urutan nukleotida identik meskipun memiliki ID yang berbeda. Hasil tersebut menunjukkan bahwa varian Gamma

memiliki sekuens duplikat terbanyak yaitu 278 sekuens. Selanjutnya, pada varian Delta ditemukan sebanyak 110 sekuens, varian Beta sebanyak 100 sekuens, varian Alpha sebanyak 20 sekuens, dan varian Omicron sebanyak 17 sekuens duplikat. Untuk memberikan gambaran, pada Tabel 4.4 disajikan beberapa sekuens duplikat.

Tabel 4.4 Cuplikan Sekuens Duplikat

ID Sekuens	Varian	Sekuens
MZ161284.1	Alpha	AGATCTGTTCTCTAAACGAACTTT...
MZ226125.1	Alpha	AGATCTGTTCTCTAAACGAACTTT...
MW841904.1	Alpha	AGATCTGTTCTCTAAACGAACTTT...
…	…	…
OM245815.1	Omicron	ACTTTCGATCTCTTGTAGATCTGT…
OM245825.1	Omicron	ACTTTCGATCTCTTGTAGATCTGT…
OM245832.1	Omicron	ACTTTCGATCTCTTGTAGATCTGT…

4. Identifikasi Karakter Nukleotida Ambigu

   Dilakukan identifikasi karakter nukleotida ambigu berupa huruf N pada *dataset*. Karakter N tidak memberikan informasi spesifik mengenai jenis basa nukleotida pada posisi tersebut. Berdasarkan hasil analisis terhadap seluruh *dataset*, tidak ditemukan adanya karakter ambigu berupa N. Hal ini ditunjukkan oleh tidak adanya sekuens yang berisi karakter tersebut.

5. Identifikasi Karakter Nukleotida Tidak Valid

   Berdasarkan hasil analisis, ditemukan sekuens yang mengandung karakter selain A, T, C, G, yaitu W, H, V, S, M, Y, R, dan K sebanyak 677 sekuens dari total *dataset* awal. Secara biologis, karakter-karakter tersebut bukan merupakan karakter yang salah, melainkan termasuk kode ambiguitas nukleotida berdasarkan standar IUPAC (*International Union of Pure and Applied Chemistry*). Kode tersebut digunakan untuk merepresentasikan ketidakpastian dalam hasil *sekuensing*, misalnya ketika suatu posisi

nukleotida dapat mewakili lebih dari satu kemungkinan basa nitrogen (Cornish-Bowden, 1985). Dalam penelitian ini, karakter W, H, V, S, M, Y, R, dan K dikategorikan sebagai karakter tidak valid secara komputasional karena proses ekstraksi fitur *k*-mer hanya menggunakan empat basa nukleotida utama, yaitu A, T, C, dan G.

Frekuensi kemunculan karakter tidak valid dalam *dataset* menunjukkan bahwa karakter Y muncul sebanyak 2.242 kali, karakter R sebanyak 1.215 kali, karakter K sebanyak 674 kali, karakter M sebanyak 428 kali, karakter W sebanyak 283 kali, karakter S sebanyak 116 kali, serta karakter V dan H masing-masing sebanyak 2 kali. Analisis juga dilakukan berdasarkan setiap varian. Hasil tersebut menunjukkan varian Beta memiliki sekuens dengan adanya karakter tidak valid di dalamnya sebanyak 196 sekuens, varian Delta sebanyak 191 sekuens, varian Alpha sebanyak 149 sekuens, varian Gamma sebanyak 91 sekuens, dan varian Omicron sebanyak 50 sekuens. Pada Tabel 4.5 disajikan distribusi sekuens yang berisi karakter tidak valid yang tersebar pada seluruh varian.

Tabel 4.5 Cuplikan Sekuens Berisi Karakter Tidak Valid

ID Sekuens	Varian	Sekuens
OU112910.1	Alpha	…GTAATSATCCAT…
OU361647.1	Alpha	…CCACCGAAGASA…
OU075296.1	Alpha	…GTTCTWAATCAC…
…	…	…
OM812069.1	Omicron	… TAGCTATYGTAG…
OV666124.1	Omicron	… ACCTTCCYAGGT…

6. Identifikasi *Missing Value*

Missing value bertujuan untuk mengidentifikasi keberadaan data yang hilang atau tidak terisi pada *dataset*. *Missing value* dapat memengaruhi kualitas data dan berpotensi menurunkan kinerja model. Pada penelitian ini, pemeriksaan

missing value dilakukan terhadap seluruh atribut *dataset* dengan menghitung jumlah nilai kosong (*null values*) pada setiap kolom *dataset*. Hasil pemeriksaan menunjukkan bahwa seluruh atribut memiliki nilai nol (0) untuk *missing value*. Hal ini menunjukkan bahwa tidak terdapat data yang kosong atau hilang pada *dataset* yang digunakan dalam penelitian.

4.2.2 *Preprocessing* Data

Tahap ini dilakukan untuk meningkatkan kualitas *dataset* sebelum digunakan pada tahap selanjutnya. Tahapan ini memastikan bahwa seluruh sekuens berada dalam kondisi lengkap, bersih, serta dalam format yang sesuai. Proses *preprocessing* dalam penelitian ini meliputi beberapa langkah sebagai berikut:

1. Penghapusan Sekuens Duplikat

   Dilakukan pemeriksaan terhadap kemungkinan adanya sekuens yang memiliki urutan nukleotida identik atau memiliki urutan yang sama lebih dari satu kali. Hasil pemeriksaan terhadap 9.104 sekuens awal, ditemukan 525 sekuens duplikat dalam *dataset*. Untuk menjaga keunikan data, salah satu dari sekuens yang memiliki urutan identik tersebut dihapus sehingga hanya satu sekuens yang dipertahankan. Dengan demikian, *dataset* berkurang menjadi 8.579 sekuens setelah proses penghapusan dilakukan. Beberapa sekuens yang teridentifikasi sebagai duplikat sudah ditampilkan pada Tabel 4.4.

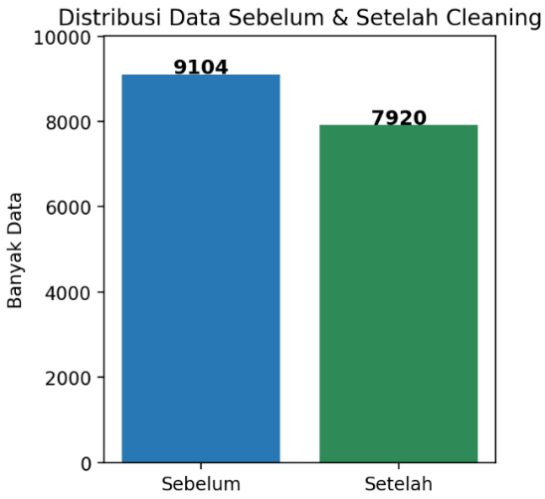
2. Penghapusan Karakter Nukleotida Tidak Valid

   Dataset hasil penghapusan duplikat menjadi 8.579 sekuens, dilakukan pembersihan terhadap sekuens yang berisi karakter nukleotida tidak valid. Hasil analisis pada tahap EDA sebelumnya, ditemukan 677 sekuens berisi

karakter nukleotida tidak valid yang tersebar pada seluruh varian. Namun, karena proses penghapusan sekuens duplikat dilakukan dahulu sebagian sekuens yang mengandung karakter tidak valid juga ikut terhapus. Jadi, banyaknya sekuens yang dihapus pada tahap penghapusan karakter tidak valid menjadi sebanyak 659 sekuens.

Berdasarkan Tabel 4.5, adanya karakter nukleotida tidak valid dapat menyebabkan ketidakonsistenan dalam proses analisis komputasi, khususnya pada tahap ekstraksi fitur, karena proses tersebut membutuhkan nukleotida yang terdefinisi. Oleh karena itu, seluruh sekuens yang mengandung karakter selain A, T, C, dan G dihapus dari *dataset*. Setelah proses penghapusan dilakukan, banyak *dataset* berkurang dari yang awalnya 8579 sekuens menjadi 7.920 sekuens.

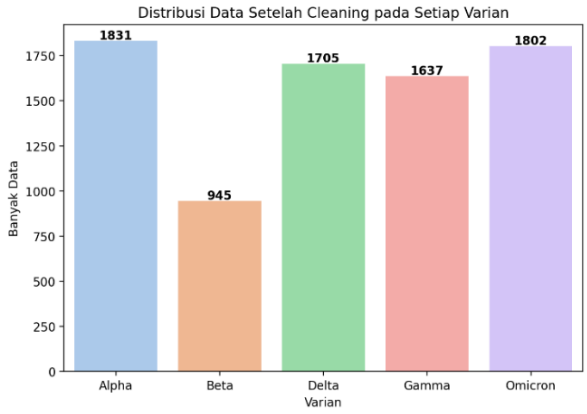
Perubahan distribusi data sebelum dan setelah proses *cleaning* disajikan pada Gambar 4.2.



Gambar 4.2 Distribusi Data Sebelum dan Setelah *Cleaning*

Berdasarkan Gambar 4.2, data awal yang digunakan sebanyak 9.104 sekuens. Setelah proses *cleaning*, data berkurang menjadi 7.920 sekuens. Hal ini

menunjukkan bahwa proses *cleaning* berhasil menghilangkan data yang tidak sesuai sehingga *dataset* bersih dan dapat digunakan pada tahap selanjutnya. Distribusi data pada setiap varian setelah proses *cleaning* disajikan pada Gambar 4.3.



Gambar 4.3 Distribusi Data Setelah *Cleaning* pada Setiap Varian

Berdasarkan Gambar 4.3, menunjukkan bahwa varian Alpha memiliki data paling banyak yaitu 1.831 sekuens, diikuti varian Omicron sebanyak 1.802 sekuens, varian Delta sebanyak 1.705 sekuens, dan varian Gamma sebanyak 1.637 sekuens. Sementara itu, varian Beta memiliki jumlah data paling sedikit yaitu 945 sekuens. Perbedaan banyak data pada setiap varian menunjukkan adanya variasi distribusi data antar kelas setelah proses *cleaning* dilakukan. *Dataset* hasil *cleaning* ini selanjutnya digunakan pada tahap ekstraksi fitur menggunakan metode k-mer serta proses pelatihan model FFNN.

3. Ekstraksi Fitur Menggunakan K-Mer

   Dataset akhir setelah melalui proses *cleaning* diperoleh sebanyak 7.920 sekuens yang selanjutnya digunakan pada tahap ekstraksi fitur menggunakan metode k-mer. Pada tahap ini, metode k-mer digunakan untuk mengubah setiap sekuens genom ke dalam bentuk numerik berupa angka. Proses ini dilakukan dengan memecah sekuens menjadi potongan-potongan *substring*

sepanjang k, dengan nilai $k = 4$. Sehingga, setiap sekuens menjadi kumpulan pola berurutan yang terdiri dari empat basa nukleotida (4-mer).

Berdasarkan konsep yang telah dijelaskan pada Bab II, seluruh kemungkinan kombinasi k-mer dihitung menggunakan rumus 4^k, sehingga diperoleh:

$$4^k = 4^4 = 256$$

Setiap sekuens direpresentasikan ke dalam 256 fitur yang menunjukkan jumlah kemunculan masing-masing kombinasi 4-mer. Penggunaan hitungan tersebut tetap dapat digunakan karena data yang dianalisis merupakan sekuens genom lengkap, sehingga variasi panjang antar sekuens relatif kecil. Oleh karena itu, pengaruh perbedaan panjang sekuens terhadap jumlah kemunculan k-mer dianggap terbatas.

Proses ekstraksi k-mer disajikan pada Tabel 4.6 untuk memberikan gambaran dari hasil.

Tabel 4.6 Cuplikan Hasil Ekstraksi Fitur *K*-Mer ($k = 4$)

No	AAAA	AAAT	AAAC	AAAG	…	GGGC	GGGG
1	250	245	191	204	…	31	14
2	280	247	192	205	…	31	14
3	250	246	190	203	…	31	14
4	249	245	190	204	…	31	14
5	246	242	190	205	…	31	14
…	…	…	…	…	…	…	…
7.920	249	244	191	204	…	31	14

Pada Tabel 4.6 ditunjukkan bahwa banyak data yang digunakan dalam penelitian ini sebanyak 7.920 sekuens, dengan setiap sekuens direpresentasikan dalam bentuk vektor fitur hasil ekstraksi 4-mer. Proses ini menghasilkan 256 fitur yang merepresentasikan kemungkinan kombinasi nukleotida yang menghasilkan matriks fitur berukuran 7.920×256 dan vektor label berukuran 7.920×1.

4.2.3 *One-Hot Encoding*

Pada tahap ini, dilakukan transformasi data terhadap variabel target berupa data kategorikal yaitu varian SARS-CoV-2. Variabel target tersebut diubah ke dalam bentuk numerik agar dapat digunakan dalam proses klasifikasi menggunakan metode OHE. Dalam penelitian ini, OHE digunakan untuk merepresentasikan label kelas varian SARS-CoV-2 ke dalam bentuk vektor biner, sedangkan fitur *input* diperoleh dari hasil ekstraksi fitur k-mer sebanyak 256 fitur. Dengan demikian, proses OHE hanya diterapkan pada label kelas (*output*) dan tidak mengubah fitur hasil ekstraksi k-mer.

OHE merupakan teknik transformasi data kategorikal menjadi vektor biner, di mana setiap kelas direpresentasikan sebagai sebuah vektor dengan panjang sesuai banyak kelas. Dalam vektor tersebut, satu elemen bernilai 1 yang menunjukkan posisi kelas, sedangkan elemen lainnya bernilai 0. Dalam penelitian ini, terdapat lima kelas varian SARS-CoV-2, yaitu Alpha, Beta, Delta, Gamma, dan Omicron. Melalui proses OHE, setiap kelas diubah menjadi vektor berdimensi lima. Hasil pemetaan label ke dalam bentuk vektor ditunjukkan pada Tabel 4.7.

Tabel 4.7 Representasi Label Menggunakan *One-Hot Encoding*

Label	**Kode**
Alpha	[1, 0, 0, 0, 0]
Beta	[0, 1, 0, 0, 0]
Delta	[0, 0, 1, 0, 0]
Gamma	[0, 0, 0, 1, 0]
Omicron	[0, 0, 0, 0, 1]

4.2.4 Pembagian Data

Dataset yang telah melalui tahap *preprocessing* dan OHE dibagi menjadi tiga subset dengan proporsi 70% data pelatihan (*training*), 15% data validasi (*validation*), dan 15% data pengujian (*testing*). Proses pembagian data dilakukan

menggunakan metode *stratified split*, yaitu teknik pemisahan data yang mempertahankan proporsi distribusi kelas pada setiap subset. Pembagian data dilakukan dalam dua tahap menggunakan fungsi *train_test_split* dari pustaka *Scikit-Learn*. Pada tahap pertama, pembagian data dilakukan dengan parameter *test_size* $= 0{,}30$, dan *random_state* $= 42$, sehingga sebesar 30% data dialokasikan sebagai data sementara (*remaining data*), sedangkan 70% data digunakan sebagai data pelatihan awal. Penggunaan parameter *stratify* bertujuan untuk menjaga proporsi distribusi kelas pada setiap subset tetap konsisten, sehingga masing-masing varian SARS-CoV-2 tetap terwakili secara seimbang.

Pada tahap kedua data sementara tersebut kembali dibagi menggunakan fungsi yang sama dengan parameter *test_size* $= 0{,}5$, *stratify* = *y_rem*, dan *random_state* $= 42$. Pembagian ini menghasilkan dua subset dengan proporsi yang sama besar, yaitu masing-masing 50% dari data sementara. Karena data sementara hanya merupakan 30% dari total *dataset*, maka hasil pembagian ini setara dengan 15% data validasi dan 15% data pengujian dari keseluruhan *dataset*. Berikut adalah hasil pembagian data yang disajikan pada Tabel 4.8.

Tabel 4.8 Distribusi Pembagian Data

Subset	**Alpha**	**Beta**	**Delta**	**Gamma**	**Omicron**	**Total**
Train	1.282	661	1.194	1.146	1.261	5.544
Validation	274	142	256	245	271	1.188
Test	275	142	255	246	270	1.188
Total	1.831	945	1.705	1.637	1.802	7.920

Berdasarkan Tabel 4.8, menunjukkan bahwa sekuens untuk *train* sebanyak 5.544, *validation* sebanyak 1.188 sekuens, dan 1.188 sekuens untuk *test*, dengan total keseluruhan data sebanyak 7.920 sekuens.

Distribusi kelas pada setiap subset menunjukkan bahwa penggunaan metode *stratified split* berhasil mempertahankan keseimbangan jumlah data antar varian.

Hal ini terlihat dari distribusi jumlah sekuens untuk setiap varian yang relatif proporsional pada data *train*, *validation*, dan *test*. Pembagian data yang seimbang membantu model memperoleh pola pembelajaran yang lebih baik selama proses *training*. Data *validation* digunakan untuk memantau kinerja model selama pelatihan berlangsung, sedangkan data *testing* digunakan untuk mengukur kemampuan generalisasi terhadap data yang belum pernah dilihat sebelumnya.

4.2.5 Normalisasi Data

Pada tahap ini dilakukan proses normalisasi data menggunakan metode *min-max scaling* yang bertujuan untuk mengubah rentang nilai fitur hasil ekstraksi k-mer. Normalisasi penting untuk dilakukan karena fitur k-mer memiliki rentang nilai yang berbeda-beda yang dapat menyebabkan algoritma FFNN menjadi bias terhadap fitur dengan nilai yang lebih besar. Oleh karena itu, nilai fitur dinormalisasikan ke dalam skala yang sama yaitu antara 0 hingga 1.

Perhitungan *min-max scaling* merujuk pada Persamaan (2.28). Pada tahap awal, nilai minimum (X_{min}) dan maksimum (X_{max}) ditentukan hanya berdasarkan data *train* (X_{train}) untuk menghindari data *leakage*. Pada Tabel 4.9 disajikan proses normalisasi, data *train* hasil ekstraksi k-mer yang terdiri dari dua fitur yaitu AAAA dan AAAT pada lima data awal.

Tabel 4.9 Hasil Ekstraksi Fitur K-Mer Lima Sekuens

No	AAAA	AAAT
1	250	245
2	280	247
3	250	246
4	249	245
5	246	242

Pada Tabel 4.9, dihitung nilai X_{min} dan X_{max} untuk setiap fitur, yang ditunjukkan pada Tabel 4.10.

Tabel 4.10 Nilai Maksimum dan Minimum Fitur *K*-Mer Data *Train*

Fitur	X_{min}	X_{max}
AAAA	246	280
AAAT	242	247

Tabel 4.10, menunjukkan nilai X_{min} dan X_{max} setiap fitur k-mer pada data yang digunakan dalam proses normalisasi. Kemudian, dilakukan perhitungan normalisasi pada sekuens dengan data ke-1 sampai ke-5, yaitu:

$$X_{new} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

1. Fitur AAAA

$$x_{1,1} = \frac{250 - 246}{280 - 246} = 0{,}1176$$

$$x_{2,1} = \frac{280 - 246}{280 - 246} = 1{,}0000$$

$$x_{3,1} = \frac{250 - 246}{280 - 246} = 0{,}1176$$

$$x_{4,1} = \frac{249 - 246}{280 - 246} = 0{,}0882$$

$$x_{5,1} = \frac{246 - 246}{280 - 246} = 0$$

2. Fitur AAAT

$$x_{1,2} = \frac{245 - 242}{247 - 242} = 0{,}6000$$

$$x_{2,2} = \frac{247 - 242}{247 - 242} = 1{,}0000$$

$$x_{3,2} = \frac{246 - 242}{247 - 242} = 0{,}8000$$

$$x_{4,2} = \frac{245 - 242}{247 - 242} = 0{,}6000$$

$$x_{5,2} = \frac{242 - 242}{247 - 242} = 0$$

Hasil setelah melalui proses *min-max scaling* disajikan pada Tabel 4.11.

Tabel 4.11 Hasil Normalisasi Fitur Menggunakan *Min-Max Scaling*

No	AAAA	AAAT
1	0,1176	0,6000
2	1,0000	1,0000
3	0,1176	0,8000
4	0,0882	0,6000
5	0,0000	0,0000

Berdasarkan Tabel 4.11, data hasil normalisasi digunakan sebagai data masukan pada lapisan *input* untuk proses pelatihan menggunakan algoritma FFNN.

4.3 Implementasi *Feedforward Neural Network* (FFNN)

Pada tahapan ini dilakukan eksperimen dalam dua skala, yaitu perhitungan manual pada contoh kecil dengan menggunakan lima data dan dua fitur untuk menunjukkan proses *feedforward* hingga *update* parameter secara bertahap. Pada eksperimen penuh menggunakan Kaggle pada seluruh *dataset* yang telah melalui tahapan pengolahan data. Pada proses penulisan perhitungan secara manual dan program, hanya menggunakan empat angka di belakang koma dengan proses pembulatan ke atas

4.3.1 Perhitungan Manual Menggunakan Lima Data Awal

1. Lapisan *input*

   Lapisan pertama yang menerima data hasil *preprocessing* sebagai data masukan. Lapisan ini menggunakan neuron sebanyak $d = 2$, yang sesuai dengan hasil fitur hasil ekstraksi k-mer. Hasil tersebut diperoleh dari banyaknya kemungkinan kombinasi k-mer yang telah melalui proses normalisasi *min-max scaling* yang diperoleh dari data *train*. Setiap data masukan dinyatakan berdasarkan Persamaan (2.1) sebagai berikut:

$$\mathbf{x}_i = [x_{i,1} \quad x_{i,2}]$$

Setiap $\mathbf{x}_i$ menyatakan data ke-i dengan $x_{i,1}$ merupakan nilai fitur ke-1 data ke-i dan $x_{i,2}$ merupakan nilai fitur ke-2 dari data ke-i. Nilai i merepresentasikan satu data sekuens. Dalam penelitian ini setelah melalui tahap *preprocessing,* jumlah data yang digunakan sebanyak 5 sekuens, dengan indeks $i = 1, 2, 3, 4, 5$. Sebagai ilustrasi, berdasarkan Tabel 4.10 digunakan vektor *input* pada data ke-1 hingga ke-5 dituliskan sebagai:

$$\mathbf{x}_1 = [0{,}1176 \quad 0{,}6000]$$

$$\mathbf{x}_2 = [1{,}0000 \quad 1{,}0000]$$

$$\mathbf{x}_3 = [0{,}1176 \quad 0{,}8000]$$

$$\mathbf{x}_4 = [0{,}0882 \quad 0{,}6000]$$

$$\mathbf{x}_5 = [0{,}0000 \quad 0{,}0000]$$

Seluruh representasi ini digunakan sebagai data masukan pada lapisan *hidden*.

2. Lapisan Tersembunyi (*Hidden Layer*)

Tahap selanjutnya, lapisan *hidden* menerima data masukan berupa vektor fitur dari lapisan *input*. Data masukan tersebut ditransformasi linier menggunakan parameter bobot (**W**) dan bias (**b**). Pada tahap awal, parameter diinisialisasi secara acak yang bertujuan untuk mencegah agar tidak ada nilai yang sama antar neuron, sehingga setiap neuron dapat mempelajari pola yang berbeda selama proses pelatihan. Berdasarkan Persamaan (2.2), bobot yang menghubungkan lapisan *input* dengan lapisan *hidden* disusun dalam matriks bobot $\mathbf{W} \in \mathbb{R}^{m \times d}$. Dalam penelitian ini, neuron pada lapisan input yaitu $d = 2$ dan neuron pada lapisan *hidden* sebanyak $m = 5$. Matriks bobot $\mathbf{W}$ memiliki ukuran 5×2 yang dituliskan sebagai:

$$\mathbf{W} \in \mathbb{R}^{5 \times 2}$$

Elemen-elemen pada matriks bobot W dinyatakan sebagai:

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \\ w_{3,1} & w_{3,2} \\ w_{4,1} & w_{4,2} \\ w_{5,1} & w_{5,2} \end{bmatrix}$$

Setiap baris pada matriks **W** merepresentasikan satu neuron pada lapisan *hidden*, yang menerima masukan dari seluruh neuron lapisan *input*.

Pada Persamaan (2.3) data masukan diubah menjadi vektor kolom melalui operasi *transpose*:

$$(\mathbf{x}_i)^T = \begin{bmatrix} x_{i,1} \\ x_{i,2} \end{bmatrix} \in \mathbb{R}^{2\times 1}$$

Setiap neuron *hidden* memiliki bias $\mathrm{b} \in \mathbb{R}^{m\times 1}$ seperti yang ditunjukkan pada Persamaan (2.4):

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix}$$

Proses total masukan neuron *hidden* mengikuti Persamaan (2.5):

$$\mathbf{a}_i = \mathbf{W}(\mathbf{x}_i)^\top + \mathbf{b}$$

Sehingga dalam bentuk vektor diperoleh:

$$\mathbf{a}_i = \begin{bmatrix} a_{i,1} \\ a_{i,2} \\ a_{i,3} \\ a_{i,4} \\ a_{i,5} \end{bmatrix} = \begin{bmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \\ w_{3,1} & w_{3,2} \\ w_{4,1} & w_{4,2} \\ w_{5,1} & w_{5,2} \end{bmatrix} \begin{bmatrix} x_{i,1} \\ x_{i,2} \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix}$$

Sebagai gambaran, digunakan nilai parameter yang diinisialisasi secara acak menggunakan Kaggle dinyatakan sebagai berikut:

$$\mathbf{W} = \begin{bmatrix} 0{,}0312 & 0{,}0946 \\ 0{,}0705 & 0{,}0559 \\ 0{,}0072 & 0{,}0072 \\ -0{,}0036 & 0{,}0853 \\ 0{,}0561 & 0{,}0679 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} -0{,}0077 \\ 0{,}0967 \\ 0{,}0816 \\ 0{,}0134 \\ 0{,}0100 \end{bmatrix}$$

Dengan menggunakan data masukan hasil normalisasi:

$\mathbf{x}_1 = [0{,}1176, 0{,}6000]$, yaitu:

$$a_{1,1} = w_{1,1}x_{1,1} + w_{1,2}x_{1,2} + b_1$$

$$= (0{,}0312)(0{,}1176) + (0{,}0946)(0{,}6000) + (-0{,}0077)$$

$$= 0{,}0037 + 0{,}0568 + (-0{,}0077) = 0{,}0528$$

$$a_{1,2} = w_{2,1}x_{1,1} + w_{2,2}x_{1,2} + b_2$$

$$= (0{,}0705)(0{,}1176) + (0{,}0559)(0{,}6000) + 0{,}0967$$

$$= 0{,}0083 + 0{,}0335 + 0{,}0967 = 0{,}1385$$

$$a_{1,3} = w_{3,1}x_{1,1} + w_{3,2}x_{1,2} + b_3$$

$$= (0{,}0072)(0{,}1176) + (0{,}0072)(0{,}6000) + 0{,}0816$$

$$= 0{,}0008 + 0{,}0043 + 0{,}0816 = 0{,}0867$$

$$a_{1,4} = w_{4,1}x_{1,1} + w_{4,2}x_{1,2} + b_4$$

$$= (-0{,}0036)(0{,}1176) + (0{,}0853)(0{,}6000) + 0{,}0134$$

$$= (-0.0004) + 0{,}0512 + 0{,}0134 = 0{,}0642$$

$$a_{1,5} = w_{5,1}x_{1,1} + w_{5,2}x_{1,2} + b_5$$

$$= (0{,}0561)(0{,}1176) + (0{,}0679)(0{,}6000) + 0{,}0100$$

$$= 0{,}0066 + 0{,}0407 + 0{,}0100 = 0{,}0573$$

$\mathbf{x}_2 = [1{,}0000, 1{,}0000]$, yaitu:

$$a_{2,1} = w_{1,1}x_{2,1} + w_{1,2}x_{2,2} + b_1$$

$$= (0{,}0312)(1{,}000) + (0{,}0946)(1{,}000) + (-0{,}0077)$$

$$= 0{,}0312 + 0{,}0946 + (-0{,}0077) = 0{,}1181$$

$$a_{2,2} = w_{2,1}x_{2,1} + w_{2,2}x_{2,2} + b_2$$

$$= (0{,}0705)(1{,}0000) + (0{,}0559)(1{,}0000) + 0{,}0967$$

$$= 0{,}0705 + 0{,}0559 + 0{,}0967 = 0{,}2231$$

$$a_{2,3} = w_{3,1}x_{2,1} + w_{3,2}x_{2,2} + b_3$$

$$= (0{,}0072)(1{,}0000) + (0{,}0072)(1{,}0000) + 0{,}0816$$

$$= 0{,}0072 + 0{,}0072 + 0{,}0816 = 0{,}0960$$

$$a_{2,4} = w_{4,1}x_{2,1} + w_{4,2}x_{2,2} + b_4$$

$$= (-0{,}0036)(1{,}0000) + (0{,}0853)(1{,}0000) + 0{,}0134$$

$$= (-0.0036) + 0{,}0853 + 0{,}0134 = 0{,}0951$$

$$a_{2,5} = w_{5,1}x_{2,1} + w_{5,2}x_{2,2} + b_5$$

$$= (0{,}0561)(1{,}0000) + (0{,}0679)(1{,}0000) + 0{,}0100$$

$$= 0{,}0561 + 0{,}0679 + 0{,}0100 = 0{,}1340$$

$\mathbf{x}_3 = [0{,}1176, 0{,}8000]$, yaitu:

$$a_{3,1} = w_{1,1}x_{3,1} + w_{1,2}x_{3,2} + b_1$$

$$= (0{,}0312)(0{,}1176) + (0{,}0946)(0{,}8000) + (-0{,}0077)$$

$$= 0{,}0037 + 0{,}0757 + (-0{,}0077) = 0{,}0717$$

$$a_{3,2} = w_{2,1}x_{3,1} + w_{2,2}x_{3,2} + b_2$$

$$= (0{,}0705)(0{,}1176) + (0{,}0559)(0{,}8000) + 0{,}0967$$

$$= 0{,}0083 + 0{,}0447 + 0{,}0967 = 0{,}1497$$

$$a_{3,3} = w_{3,1}x_{3,1} + w_{3,2}x_{3,2} + b_3$$

$$= (0{,}0072)(0{,}1176) + (0{,}0072)(0{,}8000) + 0{,}0816$$

$$= 0{,}0008 + 0{,}0058 + 0{,}0816 = 0{,}0882$$

$$a_{3,4} = w_{4,1}x_{3,1} + w_{4,2}x_{3,2} + b_4$$

$$= (-0{,}0036)(0{,}1176) + (0{,}0853)(0{,}8000) + 0{,}0134$$

$$= (-0{,}0004) + 0{,}0682 + 0{,}0134 = 0{,}0812$$

$$a_{3,5} = w_{5,1}x_{3,1} + w_{5,2}x_{3,2} + b_5$$

$$= (0{,}0561)(0{,}1176) + (0{,}0679)(0{,}8000) + 0{,}0100$$

$$= 0{,}0066 + 0{,}0543 + 0{,}0100 = 0{,}0709$$

$\mathbf{x}_4 = [0{,}0882, 0{,}6000]$, yaitu:

$$a_{4,1} = w_{1,1}x_{4,1} + w_{1,2}x_{4,2} + b_1$$

$$= (0{,}0312)(0{,}0882) + (0{,}0946)(0{,}6000) + (-0{,}0077)$$

$$= 0{,}0028 + 0{,}0568 + (-0{,}0077) = 0{,}0519$$

$$a_{4,2} = w_{2,1}x_{4,1} + w_{2,2}x_{4,2} + b_2$$

$$= (0{,}0705)(0{,}0882) + (0{,}0559)(0{,}6000) + 0{,}0967$$

$$= 0{,}0062 + 0{,}0335 + 0{,}0967 = 0{,}1364$$

$$a_{4,3} = w_{3,1}x_{4,1} + w_{3,2}x_{4,2} + b_3$$

$$= (0{,}0072)(0{,}0882) + (0{,}0072)(0{,}6000) + 0{,}0816$$

$$= 0{,}0006 + 0{,}0043 + 0{,}0816 = 0{,}0865$$

$$a_{4,4} = w_{4,1}x_{4,1} + w_{4,2}x_{4,2} + b_4$$

$$= (-0{,}0036)(0{,}0882) + (0{,}0853)(0{,}6000) + 0{,}0134$$

$$= (-0.0003) + 0{,}0512 + 0{,}0134 = 0{,}0643$$

$$a_{4,5} = w_{5,1}x_{4,1} + w_{5,2}x_{4,2} + b_5$$

$$= (0{,}0561)(0{,}0882) + (0{,}0679)(0{,}6000) + 0{,}0100$$

$$= 0{,}0049 + 0{,}0407 + 0{,}0100 = 0{,}0556$$

$\mathbf{x}_5 = [0{,}0000, 0{,}0000]$, yaitu:

$$a_{5,1} = w_{1,1}x_{5,1} + w_{1,2}x_{5,2} + b_1$$

$$= (0{,}0312)(0) + (0{,}0946)(0) + (-0{,}0077)$$

$$= 0 + 0 + (-0{,}0077) = 0 - 0{,}0077$$

$$a_{5,2} = w_{2,1}x_{5,1} + w_{2,2}x_{5,2} + b_2$$
$$= (0{,}0705)(0) + (0{,}0559)(0) + 0{,}0967$$
$$= 0 + 0 + 0{,}0967 = 0{,}0967$$
$$a_{5,3} = w_{3,1}x_{5,1} + w_{3,2}x_{5,2} + b_3$$
$$= (0{,}0072)(0) + (0{,}0072)(0) + 0{,}0816$$
$$= 0 + 0 + 0{,}0816 = 0{,}0816$$
$$a_{5,4} = w_{4,1}x_{5,1} + w_{4,2}x_{5,2} + b_4$$
$$= (-0{,}0036)(0) + (0{,}0853)(0) + 0{,}0134$$
$$= 0 + 0 + 0{,}0134 = 0{,}0134$$
$$a_{5,5} = w_{5,1}x_{5,1} + w_{5,2}x_{5,2} + b_5$$
$$= (0{,}0561)(0) + (0{,}0679)(0) + 0{,}0100$$
$$= 0 + 0 + 0{,}0100 = 0{,}0100$$

Nilai aktivasi $\mathbf{a}_i$ pada setiap neuron *hidden* digunakan sebagai masukan untuk menerapkan ReLU sesuai Persamaan (2.6) dengan mempertahankan nilai positif dan mengubah nilai negatif menjadi nol. Penerapan ReLU menghasilkan vektor keluaran $\mathbf{z}_i \in \mathbb{R}^{5\times 1}$ merujuk pada Persamaan (2.7):

$$\mathbf{z}_i = \begin{bmatrix} z_{i,1} \\ z_{i,2} \\ z_{i,3} \\ z_{i,4} \\ z_{i,5} \end{bmatrix} = \begin{bmatrix} \mathrm{ReLU}(a_{i,1}) \\ \mathrm{ReLU}(a_{i,2}) \\ \mathrm{ReLU}(a_{i,3}) \\ \mathrm{ReLU}(a_{i,4}) \\ \mathrm{ReLU}(a_{i,5}) \end{bmatrix}$$

Untuk data ke-1

$$\mathbf{z}_1 = \begin{bmatrix} \mathrm{ReLU}(a_{1,1}) \\ \mathrm{ReLU}(a_{1,2}) \\ \mathrm{ReLU}(a_{1,3}) \\ \mathrm{ReLU}(a_{1,4}) \\ \mathrm{ReLU}(a_{1,5}) \end{bmatrix} = \begin{bmatrix} 0{,}0528 \\ 0{,}1385 \\ 0{,}0867 \\ 0{,}0642 \\ 0{,}0573 \end{bmatrix}$$

Untuk data ke-2

$$\mathbf{z}_2 = \begin{bmatrix} \text{ReLU}(a_{2,1}) \\ \text{ReLU}(a_{2,2}) \\ \text{ReLU}(a_{2,3}) \\ \text{ReLU}(a_{2,4}) \\ \text{ReLU}(a_{2,5}) \end{bmatrix} = \begin{bmatrix} 0{,}1181 \\ 0{,}2231 \\ 0{,}0960 \\ 0{,}0951 \\ 0{,}1340 \end{bmatrix}$$

Untuk data ke-3

$$\mathbf{z}_3 = \begin{bmatrix} \text{ReLU}(a_{3,1}) \\ \text{ReLU}(a_{3,2}) \\ \text{ReLU}(a_{3,3}) \\ \text{ReLU}(a_{3,4}) \\ \text{ReLU}(a_{3,5}) \end{bmatrix} = \begin{bmatrix} 0{,}0717 \\ 0{,}1497 \\ 0{,}0882 \\ 0{,}0812 \\ 0{,}0709 \end{bmatrix}$$

Untuk data ke-4

$$\mathbf{z}_4 = \begin{bmatrix} \text{ReLU}(a_{4,1}) \\ \text{ReLU}(a_{4,2}) \\ \text{ReLU}(a_{4,3}) \\ \text{ReLU}(a_{4,4}) \\ \text{ReLU}(a_{4,5}) \end{bmatrix} = \begin{bmatrix} 0{,}0519 \\ 0{,}1364 \\ 0{,}0865 \\ 0{,}0643 \\ 0{,}0556 \end{bmatrix}$$

Untuk data ke-5

$$\mathbf{z}_5 = \begin{bmatrix} \text{ReLU}(a_{5,1}) \\ \text{ReLU}(a_{5,2}) \\ \text{ReLU}(a_{5,3}) \\ \text{ReLU}(a_{5,4}) \\ \text{ReLU}(a_{5,5}) \end{bmatrix} = \begin{bmatrix} 0{,}0000 \\ 0{,}0967 \\ 0{,}0816 \\ 0{,}0134 \\ 0{,}0100 \end{bmatrix}$$

Berdasarkan hasil tersebut, nilai aktivasi yang bernilai positif tetap dipertahankan oleh fungsi ReLU. Sementara itu, nilai $a_{5,1} = -0{,}0077$ diubah menjadi nol. Dengan demikian, seluruh hasil transformasi ReLU membentuk vektor keluaran pada lapisan *hidden*. Hasil keluaran tersebut diteruskan ke lapisan *output,* untuk memperoleh vektor skor keluaran.

3. Lapisan *Output*

Digunakan untuk memprediksi kelas berdasarkan keluaran dari lapisan *hidden.* Pada lapisan ini, neuron ditentukan berdasarkan banyaknya kelas yang diklasifikasikan sebanyak $p = 5$. Setiap neuron pada lapisan ini merepresentasikan satu varian SARS-CoV-2, yaitu Alpha, Beta, Delta, Gamma, Omicron.

Setiap data masukan pada lapisan *output* merupakan vektor keluaran dari lapisan *hidden* yaitu $\mathbf{z}_i \in \mathbb{R}^{5\times 1}$. Data tersebut ditransformasi linier menggunakan parameter bobot (**V**) dan bias (**c**). Berdasarkan Persamaan (2.8), bobot yang menghubungkan lapisan *hidden* dengan lapisan *output* disusun dalam matriks bobot $\mathbf{V} \in \mathbb{R}^{p\times m}$ dengan banyak neuron pada lapisan *hidden* yaitu $m = 5$ dan banyak neuron pada lapisan *output* sebanyak $p = 5$. Matriks bobot **V** memiliki ukuran 5×5 yang dinyatakan sebagai:

$$\mathbf{V} \in \mathbb{R}^{5\times 5}$$

Elemen-elemen pada matriks bobot **W** dinyatakan sebagai:

$$\mathbf{V} = \begin{bmatrix} v_{1,1} & v_{1,2} & v_{1,3} & v_{1,4} & v_{1,5} \\ v_{2,1} & v_{2,2} & v_{2,3} & v_{2,4} & v_{2,5} \\ v_{3,1} & v_{3,2} & v_{3,3} & v_{3,4} & v_{3,5} \\ v_{4,1} & v_{4,2} & v_{4,3} & v_{4,4} & v_{4,5} \\ v_{5,1} & v_{5,2} & v_{5,3} & v_{5,4} & v_{5,5} \end{bmatrix}$$

Keluaran dari lapisan *hidden* yang diperoleh yakni

$$\mathbf{z}_i = \begin{bmatrix} z_{i,1} \\ z_{i,2} \\ z_{i,3} \\ z_{i,4} \\ z_{i,5} \end{bmatrix} \in \mathbb{R}^{5\times 1}$$

Setiap neuron *output* memiliki vektor bias $\mathbf{c} \in \mathbb{R}^{p\times 1}$, seperti yang ditunjukkan pada Persamaan (2.10):

$$\mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix}$$

Total masukan untuk setiap neuron *output* dihitung mengikuti Persamaan (2.11), yaitu:

$$\mathbf{s}_i = \mathbf{V}\mathbf{z}_i + \mathbf{c}$$

$$\mathbf{s}_i = \begin{bmatrix} s_{i,1} \\ s_{i,2} \\ s_{i,3} \\ s_{i,4} \\ s_{i,5} \end{bmatrix} = \begin{bmatrix} v_{1,1} & v_{1,2} & v_{1,3} & v_{1,4} & v_{1,5} \\ v_{2,1} & v_{2,2} & v_{2,3} & v_{2,4} & v_{2,5} \\ v_{3,1} & v_{3,2} & v_{3,3} & v_{3,4} & v_{3,5} \\ v_{4,1} & v_{4,2} & v_{4,3} & v_{4,4} & v_{4,5} \\ v_{5,1} & v_{5,2} & v_{5,3} & v_{5,4} & v_{5,5} \end{bmatrix} \begin{bmatrix} z_{i,1} \\ z_{i,2} \\ z_{i,3} \\ z_{i,4} \\ z_{i,5} \end{bmatrix} + \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix}$$

Digunakan nilai parameter yang diinisialisasi secara acak menggunakan Kaggle dinyatakan sebagai berikut:

$$\mathbf{V} = \begin{bmatrix} 0{,}0102 & 0{,}0235 & 0{,}0477 & 0{,}0375 & 0{,}0220 \\ 0{,}0573 & 0{,}0053 & 0{,}0221 & 0{,}0303 & 0{,}0402 \\ 0{,}0764 & 0{,}0120 & 0{,}0466 & 0{,}0552 & -0{,}0049 \\ 0{,}0568 & 0{,}0088 & -0{,}0028 & 0{,}0944 & 0{,}0962 \\ 0{,}0789 & 0{,}0235 & 0{,}0007 & 0{,}0653 & 0{,}0384 \end{bmatrix}, \mathbf{c} = \begin{bmatrix} 0{,}0034 \\ 0{,}0445 \\ -0{,}0062 \\ 0{,}0900 \\ 0{,}0185 \end{bmatrix}$$

Menggunakan data keluaran lapisan *hidden*, maka:

Data ke-1, yaitu $\mathbf{z}_1 = \begin{bmatrix} 0{,}0528 \\ 0{,}1385 \\ 0{,}0867 \\ 0{,}0642 \\ 0{,}0573 \end{bmatrix}$

$$s_{1,1} = v_{1,1}z_{1,1} + v_{1,2}z_{1,2} + v_{1,3}z_{1,3} + v_{1,4}z_{1,4} + v_{1,5}z_{1,5} + c_1$$

$$= (0{,}0102)(0{,}0528) + (0{,}0235)(0{,}1385) + (0{,}0477)(0{,}0867) + (0{,}0375)(0{,}0642) + (0{,}0220)(0{,}0573) + 0{,}0034$$

$$= 0{,}0005 + 0{,}0033 + 0{,}0041 + 0{,}0024 + 0{,}0013 + 0{,}0034$$

$$= 0{,}0150$$

$$s_{1,2} = v_{2,1}z_{1,1} + v_{2,2}z_{1,2} + v_{2,3}z_{1,3} + v_{2,4}z_{2,4} + v_{2,5}z_{2,5} + c_2$$

$$= (0{,}0573)(0{,}0528) + (0{,}0053)(0{,}1385) + (0{,}0221)(0{,}0867) +$$
$$(0{,}0303)(0{,}0642) + (0{,}0402)(0{,}0573) + 0{,}0445$$
$$= 0{,}0030 + 0{,}0007 + 0{,}0019 + 0{,}0019 + 0{,}0023 + 0{,}0445$$
$$= 0{,}0543$$

$$s_{1,3} = v_{3,1}z_{1,1} + v_{3,2}z_{1,2} + v_{3,3}z_{1,3} + v_{3,4}z_{3,4} + v_{3,5}z_{3,5} + c_3$$
$$= (0{,}0764)(0{,}0528) + (0{,}0120)(0{,}1385) + (0{,}0466)(0{,}0867) +$$
$$(0{,}0552)(0{,}0642) + (-0{,}0049)(0{,}0573) + (-0{,}0062)$$
$$= 0{,}0040 + 0{,}0017 + 0{,}0040 + 0{,}0035 + (-0{,}0003) + (-0{,}0062)$$
$$= 0{,}0067$$

$$s_{1,4} = v_{4,1}z_{1,1} + v_{4,2}z_{1,2} + v_{4,3}z_{1,3} + v_{4,4}z_{4,4} + v_{4,5}z_{4,5} + c_4$$
$$= (0{,}0568)(0{,}0528) + (0{,}0088)(0{,}1385) + (-0{,}0028)(0{,}0867) +$$
$$(0{,}0944)(0{,}0642) + (0{,}0962)(0{,}0573) + 0{,}0900$$
$$= 0{,}0030 + 0{,}0012 + (-0{,}0002) + 0{,}0061 + 0{,}0055 + +0{,}0900$$
$$= 0{,}1056$$

$$s_{1,5} = v_{5,1}z_{1,1} + v_{5,2}z_{1,2} + v_{5,3}z_{1,3} + v_{5,4}z_{5,4} + v_{5,5}z_{5,5} + c_5$$
$$= (0{,}0789)(0{,}0528) + (0{,}0235)(0{,}1385) + (0{,}0007)(0{,}0867) +$$
$$(0{,}0653)(0{,}0642) + (0{,}0384)(0{,}0573) + 0{,}0185$$
$$= 0{,}0042 + 0{,}0033 + 0{,}0001 + 0{,}0042 + 0{,}0022 + 0{,}0185$$
$$= 0{,}0325$$

Data ke-2, yaitu $\mathbf{z}_2 = \begin{bmatrix} 0{,}1181 \\ 0{,}2231 \\ 0{,}0960 \\ 0{,}0951 \\ 0{,}1340 \end{bmatrix}$

$$s_{2,1} = v_{1,1}z_{2,1} + v_{1,2}z_{2,2} + v_{1,3}z_{2,3} + v_{1,4}z_{2,4} + v_{1,5}z_{2,5} + c_1$$
$$= (0{,}0102)(0{,}1181) + (0{,}0235)(0{,}2231) + (0{,}0477)(0{,}0960) +$$

$$(0{,}0375)(0{,}0951) + (0{,}0220)(0{,}1340) + 0{,}0034$$

$$= 0{,}0012 + 0{,}0052 + 0{,}0046 + 0{,}0036 + 0{,}0029 + 0{,}0034$$

$$= 0{,}0209$$

$$s_{2,2} = v_{2,1}z_{2,1} + v_{2,2}z_{2,2} + v_{2,3}z_{2,3} + v_{2,4}z_{2,4} + v_{2,4}z_{2,5} + c_2$$

$$= (0{,}0573)(0{,}1181) + (0{,}0053)(0{,}2231) + (0{,}0221)(0{,}0960) +$$

$$(0{,}0303)(0{,}0951) + (0{,}0402)(0{,}1340) + 0{,}0445$$

$$= 0{,}0068 + 0{,}0012 + 0{,}0021 + 0{,}0029 + 0{,}0054 + 0{,}0445$$

$$= 0{,}0629$$

$$s_{2,3} = v_{3,1}z_{2,1} + v_{3,2}z_{2,2} + v_{3,3}z_{2,3} + v_{3,4}z_{2,4} + v_{3,5}z_{2,5} + c_3$$

$$= (0{,}0764)(0{,}1181) + (0{,}0120)(0{,}2231) + (0{,}0466)(0{,}0960) +$$

$$(0{,}0552)(0{,}0951) + (-0{,}0049)(0{,}1340) + (-0{,}0062)$$

$$= 0{,}0090 + 0{,}0027 + 0{,}0045 + 0{,}0052 + (-0{,}0007) + (-0{,}0062)$$

$$= 0{,}0145$$

$$s_{2,4} = v_{4,1}z_{2,1} + v_{4,2}z_{2,2} + v_{4,3}z_{2,3} + v_{4,4}z_{2,4} + v_{4,5}z_{2,5} + c_4$$

$$= (0{,}0568)(0{,}1181) + (0{,}0088)(0{,}2231) + (-0{,}0028)(0{,}0960) +$$

$$(0{,}0944)(0{,}0951) + (0{,}0962)(0{,}1340) + 0{,}0900$$

$$= 0{,}0067 + 0{,}0019 + (-0{,}0002) + 0{,}0090 + 0{,}0129 + 0{,}0900$$

$$= 0{,}1203$$

$$s_{2,5} = v_{5,1}z_{2,1} + v_{5,2}z_{2,2} + v_{5,3}z_{2,3} + v_{5,4}z_{2,4} + v_{5,5}z_{2,5} + c_5$$

$$= (0{,}0789)(0{,}1181) + (0{,}0235)(0{,}2231) + (0{,}0007)(0{,}0960) +$$

$$(0{,}0653)(0{,}0951) + (0{,}0384)(0{,}1340) + 0{,}0185$$

$$= 0{,}0093 + 0{,}0052 + 0{,}0001 + 0{,}0062 + 0{,}0051 + 0{,}0185$$

$$= 0{,}0444$$

Data ke-3, yaitu $\mathbf{z}_3 = \begin{bmatrix} 0{,}0717 \\ 0{,}1497 \\ 0{,}0882 \\ 0{,}0812 \\ 0{,}0709 \end{bmatrix}$

$$s_{3,1} = v_{1,1}z_{3,1} + v_{1,2}z_{3,2} + v_{1,3}z_{3,3} + v_{1,4}z_{3,4} + v_{1,5}z_{3,5} + c_1$$

$$= (0{,}0102)(0{,}0717) + (0{,}0235)(0{,}1497) + (0{,}0477)(0{,}0882) +$$

$$(0{,}0375)(0{,}0812) + (0{,}0220)(0{,}0709) + 0{,}0034$$

$$= 0{,}0007 + 0{,}0035 + 0{,}0042 + 0{,}0030 + 0{,}0016 + 0{,}0034$$

$$= 0{,}0164$$

$$s_{3,2} = v_{2,1}z_{3,1} + v_{2,2}z_{3,2} + v_{2,3}z_{3,3} + v_{2,4}z_{3,4} + v_{2,5}z_{3,5} + c_2$$

$$= (0{,}0573)(0{,}0717) + (0{,}0053)(0{,}1497) + (0{,}0221)(0{,}0882) +$$

$$(0{,}0303)(0{,}0812) + (0{,}0402)(0{,}0709) + 0{,}0445$$

$$= 0{,}0041 + 0{,}0008 + 0{,}0019 + 0{,}0025 + 0{,}0029 + 0{,}0445$$

$$= 0{,}0567$$

$$s_{3,3} = v_{3,1}z_{3,1} + v_{3,2}z_{3,2} + v_{3,3}z_{3,3} + v_{3,4}z_{3,4} + v_{3,5}z_{3,5} + c_3$$

$$= (0{,}0764)(0{,}0717) + (0{,}0120)(0{,}1497) + (0{,}0466)(0{,}0882) +$$

$$(0{,}0552)(0{,}0812) + (-0{,}0049)(0{,}0709) + (-0{,}0062)$$

$$= 0{,}0055 + 0{,}0018 + 0{,}0041 + 0{,}0045 + (-0{,}0003) + (-0{,}0062)$$

$$= 0{,}0094$$

$$s_{3,4} = v_{4,1}z_{3,1} + v_{4,2}z_{3,2} + v_{4,3}z_{3,3} + v_{4,4}z_{3,4} + v_{4,5}z_{3,5} + c_4$$

$$= (0{,}0568)(0{,}0717) + (0{,}0088)(0{,}1497) + (-0{,}0028)(0{,}0882) +$$

$$(0{,}0944)(0{,}0812) + (0{,}0962)(0{,}0709) + 0{,}0900$$

$$= 0{,}0041 + 0{,}0013 + (-0{,}0002) + 0{,}0077 + 0{,}0068 + 0{,}0900$$

$$= 0{,}1097$$

$$s_{3,5} = v_{5,1}z_{3,1} + v_{5,2}z_{3,2} + v_{5,3}z_{3,3} + v_{5,4}z_{3,4} + v_{5,5}z_{3,5} + c_5$$

$$= (0{,}0789)(0{,}0717) + (0{,}0235)(0{,}1497) + (0{,}0007)(0{,}0882) +$$
$$(0{,}0653)(0{,}0812) + (0{,}0384)(0{,}0709) + 0{,}0185$$
$$= 0{,}0057 + 0{,}0035 + 0{,}0001 + 0{,}0053 + 0{,}0027 + 0{,}0185$$
$$= 0{,}0358$$

Data ke-4, yaitu $\mathbf{z}_4 = \begin{bmatrix} 0{,}0519 \\ 0{,}1364 \\ 0{,}0865 \\ 0{,}0643 \\ 0{,}0556 \end{bmatrix}$

$$s_{4,1} = v_{1,1}z_{4,1} + v_{1,2}z_{4,2} + v_{1,3}z_{4,3} + v_{1,4}z_{4,4} + v_{1,5}z_{4,5} + c_1$$
$$= (0{,}0102)(0{,}0519) + (0{,}0235)(0{,}1364) + (0{,}0477)(0{,}0865) +$$
$$(0{,}0375)(0{,}0643) + (0{,}0220)(0{,}0556) + 0{,}0034$$
$$= 0{,}0005 + 0{,}0032 + 0{,}0041 + 0{,}0024 + 0{,}0012 + 0{,}0034$$
$$= 0{,}0148$$

$$s_{4,2} = v_{2,1}z_{4,1} + v_{2,2}z_{4,2} + v_{2,3}z_{4,3} + v_{2,4}z_{4,4} + v_{2,5}z_{4,5} + c_2$$
$$= (0{,}0573)(0{,}0519) + (0{,}0053)(0{,}1364) + (0{,}0221)(0{,}0865) +$$
$$(0{,}0303)(0{,}0643) + (0{,}0402)(0{,}0556) + 0{,}0445$$
$$= 0{,}0030 + 0{,}0007 + 0{,}0019 + 0{,}0019 + 0{,}0022 + 0{,}0445$$
$$= 0{,}0542$$

$$s_{4,3} = v_{3,1}z_{4,1} + v_{3,2}z_{4,2} + v_{3,3}z_{4,3} + v_{3,4}z_{4,4} + v_{3,5}z_{4,5} + c_3$$
$$= (0{,}0764)(0{,}0519) + (0{,}0120)(0{,}1364) + (0{,}0466)(0{,}0865) +$$
$$(0{,}0552)(0{,}0643) + (-0{,}0049)(0{,}0556) + (-0{,}0062)$$
$$= 0{,}0040 + 0{,}0016 + 0{,}0040 + 0{,}0035 + (-0{,}0003) + (-0{,}0062)$$
$$= 0{,}0066$$

$$s_{4,4} = v_{4,1}z_{4,1} + v_{4,2}z_{4,2} + v_{4,3}z_{4,3} + v_{4,4}z_{4,4} + v_{4,5}z_{4,5} + c_4$$
$$= (0{,}0568)(0{,}0519) + (0{,}0088)(0{,}1364) + (-0{,}0028)(0{,}0865) +$$

$$(0{,}0944)(0{,}0643) + (0{,}0962)(0{,}0556) + 0{,}0900$$

$$= 0{,}0029 + 0{,}0012 + (-0{,}0002) + 0{,}0061 + 0{,}0053 + 0{,}0900$$

$$= 0{,}1053$$

$$s_{4,5} = v_{5,1}z_{5,1} + v_{5,2}z_{5,2} + v_{5,3}z_{5,3} + v_{5,4}z_{5,4} + v_{5,5}z_{5,5} + c_5$$

$$= (0{,}0789)(0{,}0519) + (0{,}0235)(0{,}1364) + (0{,}0007)(0{,}0865) +$$

$$(0{,}0653)(0{,}0643) + (0{,}0384)(0{,}0556) + 0{,}0185$$

$$= 0{,}0041 + 0{,}0032 + 0{,}0001 + 0{,}0042 + 0{,}0021 + 0{,}0185$$

$$= 0{,}0322$$

Data ke-5, yaitu $\mathbf{z}_5 = \begin{bmatrix} 0{,}0000 \\ 0{,}0967 \\ 0{,}0816 \\ 0{,}0134 \\ 0{,}0100 \end{bmatrix}$

$$s_{5,1} = v_{1,1}z_{5,1} + v_{1,2}z_{5,2} + v_{1,3}z_{5,3} + v_{1,4}z_{5,4} + v_{1,5}z_{5,5} + c_1$$

$$= (0{,}0102)(0{,}000) + (0{,}0235)(0{,}0967) + (0{,}0477)(0{,}0816) +$$

$$(0{,}0375)(0{,}0134) + (0{,}0220)(0{,}0100) + 0{,}0034$$

$$= 0{,}0000 + 0{,}0023 + 0{,}0039 + 0{,}0005 + 0{,}0002 + 0{,}0034$$

$$= 0{,}0103$$

$$s_{5,2} = v_{2,1}z_{5,1} + v_{2,2}z_{5,2} + v_{2,3}z_{5,3} + v_{2,4}z_{5,4} + v_{2,5}z_{5,5} + c_2$$

$$= (0{,}0573)(0{,}0000) + (0{,}0053)(0{,}0967) + (0{,}0221)(0{,}0816) +$$

$$(0{,}0303)(0{,}0134) + (0{,}0402)(0{,}0100) + 0{,}0445$$

$$= 0{,}0000 + 0{,}0005 + 0{,}0018 + 0{,}0004 + 0{,}0004 + 0{,}0445$$

$$= 0{,}0476$$

$$s_{5,3} = v_{3,1}z_{5,1} + v_{3,2}z_{5,2} + v_{3,3}z_{5,3} + v_{3,4}z_{5,4} + v_{3,5}z_{5,5} + c_3$$

$$= (0{,}0764)(0{,}000) + (0{,}0120)(0{,}0967) + (0{,}0466)(0{,}0816) +$$

$$(0{,}0552)(0{,}0134) + (-0{,}0049)(0{,}0100) + (-0{,}0062)$$

$$= 0{,}0000 + 0{,}0012 + 0{,}0038 + 0{,}0007 + 0 + (-0{,}0062)$$

$$= (-0{,}0005)$$

$$s_{5,4} = v_{4,1}z_{5,1} + v_{4,2}z_{5,2} + v_{4,3}z_{5,3} + v_{4,4}z_{5,4} + v_{4,5}z_{5,5} + c_4$$

$$= (0{,}0568)(0{,}0000) + (0{,}0088)(0{,}0967) + (-0{,}0028)(0{,}0816) + (0{,}0944)(0{,}0134) + (0{,}0962)(0{,}0100) + 0{,}0900$$

$$= 0{,}0000 + 0{,}0009 + (-0{,}0002) + 0{,}0013 + 0{,}0010 + 0{,}0900$$

$$= 0{,}0930$$

$$s_{5,5} = v_{5,1}z_{5,1} + v_{5,2}z_{5,2} + v_{5,3}z_{5,3} + v_{5,4}z_{5,4} + v_{5,5}z_{5,5} + c_5$$

$$= (0{,}0789)(0{,}0000) + (0{,}0235)(0{,}0967) + (0{,}0007)(0{,}0816) + (0{,}0653)(0{,}0134) + (0{,}0384)(0{,}0100) + 0{,}0185$$

$$= 0{,}0000 + 0{,}0023 + 0{,}0001 + 0{,}0009 + 0{,}0004 + 0{,}0185$$

$$= 0{,}0222$$

Dari hasil perhitungan pada setiap data ke-i, diperoleh vektor keluaran lapisan *output*:

$$\mathbf{s}_i = \begin{bmatrix} s_{i,1} \\ s_{i,2} \\ s_{i,3} \\ s_{i,4} \\ s_{i,5} \end{bmatrix} \in \mathbb{R}^{5\times 1}, i = 1,2,3,4,5$$

Diberikan cuplikan setelah diperoleh nilai $\mathbf{s}_i$ untuk seluruh nilai vektor keluaran lapisan *output* berupa skor awal 5 data yang dinyatakan sebagai:

$$\mathbf{S} = \begin{bmatrix} s_{1,1} & s_{1,2} & s_{1,3} & s_{1,4} & s_{1,5} \\ s_{2,1} & s_{2,2} & s_{2,3} & s_{2,4} & s_{2,5} \\ s_{3,1} & s_{3,2} & s_{3,3} & s_{3,4} & s_{3,5} \\ s_{4,1} & s_{4,2} & s_{4,3} & s_{4,4} & s_{4,5} \\ s_{5,1} & s_{5,2} & s_{5,3} & s_{5,4} & s_{5,5} \end{bmatrix} \in \mathbb{R}^{5\times 5}$$

$$\mathbf{S} = \begin{bmatrix} 0{,}0150 & 0{,}0543 & 0{,}0067 & 0{,}1056 & 0{,}0325 \\ 0{,}0209 & 0{,}0629 & 0{,}0145 & 0{,}1203 & 0{,}0444 \\ 0{,}0164 & 0{,}0567 & 0{,}0094 & 0{,}1097 & 0{,}0358 \\ 0{,}0148 & 0{,}0542 & 0{,}0066 & 0{,}1053 & 0{,}0322 \\ 0{,}0103 & 0{,}0476 & -0{,}0005 & 0{,}0930 & 0{,}0222 \end{bmatrix}$$

Nilai $\mathbf{s}_i$ tersebut belum dapat digunakan secara langsung sebagai probabilitas, karena masih merupakan hasil transformasi linier. Oleh karena itu, digunakan fungsi aktivasi *softmax* menggunakan Persamaan (2.12):

$$\mathbf{o}_{i,j} = \frac{e^{s_{i,j}}}{\sum_{k=1}^{p} e^{s_{i,k}}}, \qquad j = 1,2,\dots,p$$

$$\mathbf{o}_{i,j} = \frac{e^{s_{i,j}}}{\sum_{k=1}^{5} e^{s_{i,k}}}, \qquad j = 1,2,3,4,5$$

Proses transformasi nilai keluaran menjadi probabilitas, dilakukan perhitungan fungsi *softmax* pada lima data awal. Berdasarkan hasil sebelumnya, diperoleh vektor keluaran lapisan *output* yaitu:

Data ke-1, $\mathbf{s}_1 = \begin{bmatrix} 0{,}0150 \\ 0{,}0543 \\ 0{,}0067 \\ 0{,}1056 \\ 0{,}0325 \end{bmatrix}$

Setiap elemen pada vektor $\mathbf{s}_1$ ditransformasikan ke dalam bentuk eksponensial menggunakan konstanta Euler dengan $e = 2{,}71828$, sehingga diperoleh:

$$\mathbf{e}^{\mathbf{s}_1} = \begin{bmatrix} e^{0{,}0150} \\ e^{0{,}0543} \\ e^{0{,}0067} \\ e^{0{,}1056} \\ e^{0{,}0325} \end{bmatrix} = \begin{bmatrix} 1{,}0151 \\ 1{,}0558 \\ 1{,}0067 \\ 1{,}1114 \\ 1{,}0330 \end{bmatrix}$$

Seluruh nilai eksponensial dijumlahkan:

$$\sum_{j=1}^{5} \mathbf{e}^{\mathbf{s}_{1,j}} = 5{,}2222$$

Vektor keluaran fungsi *softmax* untuk data ke-1 dinyatakan sebagai:

$$\mathbf{o}_1 = \begin{bmatrix} \frac{e^{s_{1,1}}}{\sum_{j=1}^{5} e^{s_{1,j}}} \\ \frac{e^{s_{1,2}}}{\sum_{j=1}^{5} e^{s_{1,j}}} \\ \frac{e^{s_{1,3}}}{\sum_{j=1}^{5} e^{s_{1,j}}} \\ \frac{e^{s_{1,4}}}{\sum_{j=1}^{5} e^{s_{1,j}}} \\ \frac{e^{s_{1,5}}}{\sum_{j=1}^{5} e^{s_{1,j}}} \end{bmatrix} = \begin{bmatrix} \frac{1,0151}{5,2222} \\ \frac{1,0558}{5,2222} \\ \frac{1,0067}{5,2222} \\ \frac{1,1114}{5,2222} \\ \frac{1,0330}{5,2222} \end{bmatrix} = \begin{bmatrix} 0,1944 \\ 0,2022 \\ 0,1928 \\ 0,2128 \\ 0,1978 \end{bmatrix}$$

Kemudian menentukan kelas prediksi menggunakan Persamaan (2.14):

$$\hat{y}_1 = \arg\max_j(o_{1,j})$$

Berdasarkan hasil perhitungan *softmax*, nilai probabilitas terbesar terdapat pada neuron *output* ke-4 sebesar $0{,}2128$. Hal ini menunjukkan bahwa algoritma memprediksi data ke-1 sebagai kelas ke-4. Sebagai ilustrasi, data ke-1 memiliki label asli pada kelas ke-4, maka representasi OHE:

$$\mathbf{y}_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \text{maka } \hat{\mathbf{y}}_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Data ke-2, $\mathbf{s}_2 = \begin{bmatrix} 0,0209 \\ 0,0629 \\ 0,0145 \\ 0,1203 \\ 0,0444 \end{bmatrix}$

Setiap elemen pada vektor $\mathbf{s}_2$ ditransformasikan ke dalam bentuk eksponensial menggunakan konstanta Euler dengan $e = 2{,}71828$, sehingga diperoleh:

$$\mathbf{e}^{\mathbf{s}_2} = \begin{bmatrix} e^{0,0209} \\ e^{0,0629} \\ e^{0,0145} \\ e^{0,1203} \\ e^{0,0444} \end{bmatrix} = \begin{bmatrix} 1,0211 \\ 1,0649 \\ 1,0146 \\ 1,1278 \\ 1,0454 \end{bmatrix}$$

Selanjutnya seluruh nilai eksponensial dijumlahkan:

$$\sum_{j=1}^{5} \mathbf{e}^{\mathbf{s}_{2,j}} = 5{,}2738$$

Sehingga, vektor keluaran fungsi *softmax* untuk data ke-2 dinyatakan sebagai:

$$\mathbf{o}_2 = \begin{bmatrix} \frac{e^{s_{2,1}}}{\sum_{j=1}^{5} e^{s_{2,j}}} \\ \frac{e^{s_{2,2}}}{\sum_{j=1}^{5} e^{s_{2,j}}} \\ \frac{e^{s_{2,3}}}{\sum_{j=1}^{5} e^{s_{2,j}}} \\ \frac{e^{s_{2,4}}}{\sum_{j=1}^{5} e^{s_{2,j}}} \\ \frac{e^{s_{2,5}}}{\sum_{j=1}^{5} e^{s_{2,j}}} \end{bmatrix} = \begin{bmatrix} \frac{1{,}0211}{5{,}2738} \\ \frac{1{,}0649}{5{,}2738} \\ \frac{1{,}0146}{5{,}2738} \\ \frac{1{,}1278}{5{,}2738} \\ \frac{1{,}0454}{5{,}2738} \end{bmatrix} = \begin{bmatrix} 0{,}1936 \\ 0{,}2019 \\ 0{,}1924 \\ 0{,}2138 \\ 0{,}1982 \end{bmatrix}$$

Kemudian menentukan kelas prediksi menggunakan Persamaan (2.14):

$$\hat{y}_2 = \arg\max_j(o_{2,j})$$

Berdasarkan hasil perhitungan *softmax*, nilai probabilitas terbesar terdapat pada neuron *output* ke-4 sebesar $0{,}2138$. Hal ini menunjukkan bahwa algoritma memprediksi data ke-2 sebagai kelas ke-4. Sebagai ilustrasi, data ke-2 memiliki label asli pada kelas ke-4, maka representasi OHE dinyatakan sebagai:

$$\mathbf{y}_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \text{maka } \hat{\mathbf{y}}_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Data ke-3, $\mathbf{s}_3 = \begin{bmatrix} 0{,}0164 \\ 0{,}0567 \\ 0{,}0094 \\ 0{,}1097 \\ 0{,}0358 \end{bmatrix}$

Setiap elemen pada vektor $\mathbf{s}_3$ ditransformasikan ke dalam bentuk eksponensial menggunakan konstanta Euler dengan $e = 2{,}71828$, maka:

$$\mathbf{e}^{\mathbf{s}_3} = \begin{bmatrix} e^{0,0164} \\ e^{0,0567} \\ e^{0,0094} \\ e^{0,1097} \\ e^{0,0358} \end{bmatrix} = \begin{bmatrix} 1,0165 \\ 1,0583 \\ 1,0094 \\ 1,1159 \\ 1,0364 \end{bmatrix}$$

Selanjutnya seluruh nilai eksponensial dijumlahkan:

$$\sum_{j=1}^{5} \mathbf{e}^{\mathbf{s}_{3,j}} = 5,2365$$

Sehingga, vektor keluaran fungsi *softmax* untuk data ke-3 dinyatakan sebagai:

$$\mathbf{o}_3 = \begin{bmatrix} \frac{e^{s_{3,1}}}{\sum_{j=1}^{5} e^{s_{3,j}}} \\ \frac{e^{s_{3,2}}}{\sum_{j=1}^{5} e^{s_{3,j}}} \\ \frac{e^{s_{3,3}}}{\sum_{j=1}^{5} e^{s_{3,j}}} \\ \frac{e^{s_{3,4}}}{\sum_{j=1}^{5} e^{s_{3,j}}} \\ \frac{e^{s_{3,5}}}{\sum_{j=1}^{5} e^{s_{3,j}}} \end{bmatrix} = \begin{bmatrix} \frac{1,0165}{5,2365} \\ \frac{1,0583}{5,2365} \\ \frac{1,0094}{5,2365} \\ \frac{1,1159}{5,2365} \\ \frac{1,0364}{5,2365} \end{bmatrix} = \begin{bmatrix} 0,1941 \\ 0,2021 \\ 0,1928 \\ 0,2131 \\ 0,1979 \end{bmatrix}$$

Kemudian menentukan kelas prediksi menggunakan Persamaan (2.14):

$$\hat{y}_3 = \arg\max_j (o_{3,j})$$

Berdasarkan hasil perhitungan *softmax*, nilai probabilitas terbesar terdapat pada neuron *output* ke-4 sebesar $0,2131$. Hal ini menunjukkan bahwa algoritma memprediksi data ke-3 sebagai kelas ke-4. Sebagai ilustrasi, data ke-3 memiliki label asli pada kelas ke-4, maka representasi OHE:

$$\mathbf{y}_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \text{maka } \hat{\mathbf{y}}_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Data ke-4, $\mathbf{s}_4 = \begin{bmatrix} 0,0148 \\ 0,0542 \\ 0,0066 \\ 0,1053 \\ 0,0322 \end{bmatrix}$

Setiap elemen pada vektor $\mathbf{s}_4$ ditransformasikan ke dalam bentuk eksponensial menggunakan konstanta Euler dengan $e = 2{,}71828$, sehingga diperoleh:

$$\mathbf{e}^{\mathbf{s}_4} = \begin{bmatrix} e^{0{,}0148} \\ e^{0{,}0542} \\ e^{0{,}0066} \\ e^{0{,}1053} \\ e^{0{,}0322} \end{bmatrix} = \begin{bmatrix} 1{,}0149 \\ 1{,}0557 \\ 1{,}0066 \\ 1{,}1110 \\ 1{,}0327 \end{bmatrix}$$

Selanjutnya seluruh nilai eksponensial dijumlahkan:

$$\sum_{j=1}^{5} \mathbf{e}^{\mathbf{s}_{4,j}} = 5{,}2209$$

Sehingga, vektor keluaran fungsi *softmax* untuk data ke-4 dinyatakan sebagai:

$$\mathbf{o}_4 = \begin{bmatrix} \frac{e^{s_{4,1}}}{\sum_{j=1}^{5} e^{s_{4,j}}} \\ \frac{e^{s_{4,2}}}{\sum_{j=1}^{5} e^{s_{4,j}}} \\ \frac{e^{s_{4,3}}}{\sum_{j=1}^{5} e^{s_{4,j}}} \\ \frac{e^{s_{4,4}}}{\sum_{j=1}^{5} e^{s_{4,j}}} \\ \frac{e^{s_{4,5}}}{\sum_{j=1}^{5} e^{s_{4,j}}} \end{bmatrix} = \begin{bmatrix} \frac{1{,}0149}{5{,}2209} \\ \frac{1{,}0557}{5{,}2209} \\ \frac{1{,}0066}{5{,}2209} \\ \frac{1{,}1110}{5{,}2209} \\ \frac{1{,}0327}{5{,}2209} \end{bmatrix} = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ 0{,}2128 \\ 0{,}1978 \end{bmatrix}$$

Kemudian menentukan kelas prediksi menggunakan Persamaan (2.14):

$$\hat{y}_4 = \arg\max_j (o_{4,j})$$

Berdasarkan hasil perhitungan *softmax*, nilai probabilitas terbesar terdapat pada neuron *output* ke-4 sebesar $0{,}2128$. Hal ini menunjukkan bahwa algoritma memprediksi data ke-4 sebagai kelas ke-4. Sebagai ilustrasi, data ke-4 memiliki label asli pada kelas ke-4, maka representasi OHE dinyatakan sebagai:

$$\mathbf{y}_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \text{maka } \hat{\mathbf{y}}_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Data ke-5, $\mathbf{s}_5 = \begin{bmatrix} 0{,}0103 \\ 0{,}0476 \\ -0{,}0005 \\ 0{,}0930 \\ 0{,}0222 \end{bmatrix}$

Setiap elemen pada vektor $\mathbf{s}_5$ ditransformasikan ke dalam bentuk eksponensial menggunakan konstanta Euler dengan $e = 2{,}71828$, sehingga diperoleh:

$$\mathbf{e}^{\mathbf{s}_5} = \begin{bmatrix} e^{0{,}0103} \\ e^{0{,}0476} \\ e^{(-0{,}0005)} \\ e^{0{,}0930} \\ e^{0{,}0222} \end{bmatrix} = \begin{bmatrix} 1{,}0104 \\ 1{,}0488 \\ 0{,}9995 \\ 1{,}0975 \\ 1{,}0224 \end{bmatrix}$$

Selanjutnya seluruh nilai eksponensial dijumlahkan:

$$\sum_{j=1}^{5} \mathbf{e}^{\mathbf{s}_{5,j}} = 5{,}1786$$

Sehingga, vektor keluaran fungsi *softmax* untuk data ke-5 dinyatakan sebagai:

$$\mathbf{o}_5 = \begin{bmatrix} \frac{e^{s_{5,1}}}{\sum_{j=1}^{5} e^{s_{5,j}}} \\ \frac{e^{s_{5,2}}}{\sum_{j=1}^{5} e^{s_{5,j}}} \\ \frac{e^{s_{5,3}}}{\sum_{j=1}^{5} e^{s_{5,j}}} \\ \frac{e^{s_{5,4}}}{\sum_{j=1}^{5} e^{s_{5,j}}} \\ \frac{e^{s_{5,5}}}{\sum_{j=1}^{5} e^{s_{5,j}}} \end{bmatrix} = \begin{bmatrix} \frac{1{,}0104}{5{,}1786} \\ \frac{1{,}0488}{5{,}1786} \\ \frac{0{,}9995}{5{,}1786} \\ \frac{1{,}0975}{5{,}1786} \\ \frac{1{,}0224}{5{,}1786} \end{bmatrix} = \begin{bmatrix} 0{,}1951 \\ 0{,}2025 \\ 0{,}1930 \\ 0{,}2119 \\ 0{,}1974 \end{bmatrix}$$

Kemudian menentukan kelas prediksi menggunakan Persamaan (2.14):

$$\hat{y}_5 = \arg\max_j (o_{5,j})$$

Berdasarkan hasil perhitungan *softmax*, nilai probabilitas terbesar terdapat

pada neuron *output* ke-4 sebesar 0,2119. Hal ini menunjukkan bahwa algoritma memprediksi data ke-5 sebagai kelas ke-4. Sebagai ilustrasi, data ke-5 memiliki label asli pada kelas ke-4, maka representasi OHE dinyatakan sebagai:

$$\mathbf{y}_5 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \text{maka } \hat{\mathbf{y}}_5 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Berdasarkan hasil perhitungan, algoritma memprediksi kelas berdasarkan nilai tertinggi pada vektor keluaran *softmax*. Secara keseluruhan, lima data pertama diprediksi sebagai kelas ke-4 yaitu Gamma yang menunjukkan bahwa *softmax* memberikan probabilitas tertinggi pada kelas tersebut di seluruh data yang dihitung.

4. Perhitungan Fungsi *Loss*

   Nilai probabilitas dari fungsi *softmax* pada lapisan *output* digunakan untuk menghitung tingkat kesalahan (*loss*) antara hasil prediksi model dan label sebenarnya. Pada penelitian ini digunakan fungsi *cross-entropy loss* untuk mengukur perbedaan antara distribusi probabilitas hasil prediksi dengan label target. Berdasarkan hasil perhitungan pada lapisan *output,* diperoleh vektor probabilitas hasil fungsi *softmax* untuk setiap data yaitu $\mathbf{o}_i \in \mathbb{R}^{p\times 1}$.

   Banyak kelas yang digunakan dalam penelitian ini sebanyak $p = 5$, dengan representasi label menggunakan OHE. Fungsi *loss* untuk setiap data ke-i merujuk pada Persamaan (2.16):

$$L_i = -\sum_{j=1}^{5} y_{i,j} \ln(o_{i,j})$$

$$L_i = -\left(y_{i,1}\ln\left(o_{i,1}\right) + y_{i,2}\ln\left(o_{i,2}\right) + y_{i,3}\ln\left(o_{i,3}\right) + y_{i,4}\ln\left(o_{i,4}\right) + y_{i,5}\ln\left(o_{i,5}\right)\right)$$

Karena menggunakan OHE, maka hanya satu elemen bernilai 1, mengikuti Persamaan (2.17):

$$L_i = -\ln\left(o_{i,j^*}\right)$$

Untuk memberikan gambaran proses fungsi *loss*, dilakukan perhitungan berdasarkan hasil pada lapisan *output*, yang diperoleh vektor keluaran lapisan *output*.

Pada perhitungan *loss* untuk data ke-1, dengan menggunakan hasil perhitungan sebelumnya pada fungsi *softmax*, yaitu:

$$\mathbf{o}_1 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ 0{,}2128 \\ 0{,}1978 \end{bmatrix}, \mathbf{y}_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Substitusikan ke dalam rumus, maka:

$$L_1 = -((0\ln(0{,}1944)) + 0(\ln(0{,}2022)) + 0(\ln(0{,}1928)) + 1(\ln(0{,}2128)) + (0(\ln(0{,}1978))$$

Hal ini menunjukkan bahwa data ke-1 termasuk ke dalam kelas ke-4. Oleh karena itu, nilai probabilitas yang digunakan dalam perhitungan *loss* adalah probabilitas pada kelas ke-4, yaitu $\mathbf{o}_1 = 0{,}2128$. Selanjutnya, nilai *loss* dihitung sebagai berikut:

$$L_i = -\ln\left(o_{i,j^*}\right)$$

$$L_1 = -\ln\left(0{,}2128\right)$$

$$L_1 = -(-1{,}5474) = 1{,}5474$$

Perhitungan *loss* untuk data ke-2 menggunakan hasil *softmax* yaitu:

$$\mathbf{o}_2 = \begin{bmatrix} 0{,}1936 \\ 0{,}2019 \\ 0{,}1924 \\ 0{,}2138 \\ 0{,}1982 \end{bmatrix}, \mathbf{y}_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Substitusikan ke dalam rumus, maka:

$$L_2 = -(0(\ln(0{,}1936)) + 0(\ln(0{,}2019)) + 0(\ln(0{,}1924)) +$$
$$1(\ln(0{,}2138)) + (0(\ln(0{,}1982))$$

Hal ini menunjukkan bahwa data ke-2 termasuk ke dalam kelas ke-4, sehingga probabilitas yang digunakan adalah $\mathbf{o}_2 = 0{,}2138$, maka:

$$L_i = -\ln(o_{i,j^*})$$

$$L_2 = -\ln(0{,}2138)$$

$$L_2 = -(-1{,}5427) = 1{,}5427$$

Pada data ke-3 menggunakan hasil dari *softmax* yaitu:

$$\mathbf{o}_3 = \begin{bmatrix} 0{,}1941 \\ 0{,}2021 \\ 0{,}1928 \\ 0{,}2131 \\ 0{,}1979 \end{bmatrix}, \mathbf{y}_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Substitusikan ke dalam rumus, maka:

$$L_3 = -(0(\ln(0{,}1941)) + 0(\ln(0{,}2021)) + 0(\ln(0{,}1928)) +$$
$$1(\ln(0{,}2131)) + 0(\ln(0{,}1979))$$

Hal ini menunjukkan bahwa data ke-3 termasuk ke dalam kelas ke-4, sehingga probabilitas yang digunakan adalah $\mathbf{o}_3 = 0{,}2131$, maka:

$$L_i = -\ln(o_{i,j^*})$$

$$L_3 = -\ln(0{,}2131)$$

$$L_3 = -(-1{,}5460) = 1{,}5460$$

Pada perhitungan *loss* untuk data ke-4 menggunakan hasil *softmax* yaitu:

$$\mathbf{o}_4 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ 0{,}2128 \\ 0{,}1978 \end{bmatrix}, \mathbf{y}_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Substitusikan ke dalam rumus, maka:

$$L_4 = -(0(\ln(0{,}1944)) + 0(\ln(0{,}2022)) + 0(\ln(0{,}1928)) +$$
$$1(\ln(0{,}2128)) + (0(\ln(0{,}1978))$$

Hal ini menunjukkan bahwa data ke-4 termasuk ke dalam kelas ke-4, sehingga probabilitas yang digunakan adalah $\mathbf{o}_4 = 0{,}2128$, maka:

$$L_i = -\ln\,(o_{i,j^*})$$

$$L_4 = -\ln\,(0{,}2128)$$

$$L_4 = -(-1{,}5474) = 1{,}5474$$

Selanjutnya, untuk perhitungan *loss* untuk data ke-5 menggunakan hasil dari *softmax* yaitu:

$$\mathbf{o}_5 = \begin{bmatrix} 0{,}1951 \\ 0{,}2025 \\ 0{,}1930 \\ 0{,}2119 \\ 0{,}1974 \end{bmatrix}, \mathbf{y}_5 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Substitusikan ke dalam rumus, maka:

$$L_5 = -(0(\ln(0{,}1951)) + 0(\ln(0{,}2025)) + 0(\ln(0{,}1930)) +$$
$$1(\ln(0{,}2119)) + 0(\ln(0{,}1974))$$

Hal ini menunjukkan bahwa data ke-5 termasuk ke dalam kelas ke-4, sehingga probabilitas yang digunakan adalah $\mathbf{o}_5 = 0{,}2119$, maka:

$$L_i = -\ln\,(o_{i,j^*})$$

$$L_5 = -\ln\,(0{,}2119)$$

$$L_5 = -(-1{,}5516) = 1{,}5516$$

Setelah diperoleh nilai *loss* masing-masing data, maka dilakukan penjumlahan berdasarkan Persamaan (2.15):

$$E = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{p} y_{i,j} \ln(o_{i,j})$$

Pada perhitungan di atas sudah diperoleh nilai *loss* setiap data L_i dengan mengikuti Persamaan (2.18):

$$E = \frac{1}{n}\sum_{i=1}^{n} L_i$$

Diketahui bahwa $n = 5$, maka:

$$E = \frac{1}{5}\sum_{i=1}^{5}(L_1 + L_2 + L_3 + L_4 + L_5)$$

$$E = \frac{1}{5}(1{,}5474 + 1{,}5427 + 1{,}5460 + 1{,}5474 + 1{,}5516)$$

$$E = \frac{1}{5}(7{,}7351) = 1{,}5470$$

Berdasarkan hasil perhitungan, diperoleh nilai *loss* masing-masing data yaitu:

$$L_1 = 1{,}5474, L_2 = 1{,}5427, L_3 = 1{,}5460, L_4 = 1{,}5474, L_5 = 1{,}5516$$

Nilai rata-rata *loss* sebesar $E = 1{,}5470$ dari ke-5 data tersebut. Nilai rata-rata *loss* ini merepresentasikan tingkat kesalahan model secara keseluruhan dalam melakukan prediksi. Semakin kecil nilai *loss*, maka semakin baik kemampuan model dalam menghasilkan prediksi yang mendekati nilai target. Selain itu, nilai *loss* digunakan pada proses *backpropagation* untuk menghitung gradien terhadap setiap parameter. Nilai *loss* yang diperoleh menunjukkan bahwa pada tahap awal pelatihan model masih memiliki tingkat kesalahan prediksi yang cukup besar. Oleh karena itu, diperlukan proses *backpropagation* untuk

memperbarui parameter agar kesalahan prediksi dapat dikurangi secara bertahap pada iterasi berikutnya.

5. *Backpropagation*

Dilakukan proses *backpropagation* untuk menghitung gradien fungsi *loss* terhadap seluruh parameter. Proses ini dimulai dari lapisan *output*, kemudian ke lapisan *hidden* dengan menggunakan aturan rantai (*chain rule*). Berdasarkan Persamaan (2.20), sinyal kesalahan pada lapisan *output* untuk data ke-i dinyatakan sebagai:

$$\boldsymbol{\delta}_i = \mathbf{o}_i - \mathbf{y}_i$$

Menggunakan persamaan tersebut, diperoleh perhitungan, yaitu:

Untuk data ke-1

$$\mathbf{o}_1 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ 0{,}2128 \\ 0{,}1978 \end{bmatrix}, \mathbf{y}_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Maka:

$$\boldsymbol{\delta}_1 = \mathbf{o}_1 - \mathbf{y}_1$$

$$\boldsymbol{\delta}_1 = \begin{bmatrix} o_{1,1} - y_{1,1} \\ o_{1,2} - y_{1,2} \\ o_{1,3} - y_{1,3} \\ o_{1,4} - y_{1,4} \\ o_{1,5} - y_{1,5} \end{bmatrix} = \begin{bmatrix} 0{,}1944 - 0 \\ 0{,}2022 - 0 \\ 0{,}1928 - 0 \\ 0{,}2128 - 1 \\ 0{,}1978 - 0 \end{bmatrix} = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7872 \\ 0{,}1978 \end{bmatrix}$$

Untuk data ke-2

$$\mathbf{o}_2 = \begin{bmatrix} 0{,}1936 \\ 0{,}2019 \\ 0{,}1924 \\ 0{,}2138 \\ 0{,}1982 \end{bmatrix}, \mathbf{y}_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Maka:

$$\boldsymbol{\delta}_2 = \mathbf{o}_2 - \mathbf{y}_2$$

$$\boldsymbol{\delta}_2 = \begin{bmatrix} o_{2,1} - y_{2,1} \\ o_{2,2} - y_{2,2} \\ o_{2,3} - y_{2,3} \\ o_{2,4} - y_{2,4} \\ o_{2,5} - y_{2,5} \end{bmatrix} = \begin{bmatrix} 0{,}1936 - 0 \\ 0{,}2019 - 0 \\ 0{,}1924 - 0 \\ 0{,}2138 - 1 \\ 0{,}1982 - 0 \end{bmatrix} = \begin{bmatrix} 0{,}1936 \\ 0{,}2019 \\ 0{,}1924 \\ -0{,}7862 \\ 0{,}1982 \end{bmatrix}$$

Untuk data ke-3

$$\mathbf{o}_3 = \begin{bmatrix} 0{,}1941 \\ 0{,}2021 \\ 0{,}1928 \\ 0{,}2131 \\ 0{,}1979 \end{bmatrix}, \mathbf{y}_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Maka:

$$\boldsymbol{\delta}_3 = \mathbf{o}_3 - \mathbf{y}_3$$

$$\boldsymbol{\delta}_3 = \begin{bmatrix} o_{3,1} - y_{3,1} \\ o_{3,2} - y_{3,2} \\ o_{3,3} - y_{3,3} \\ o_{3,4} - y_{3,4} \\ o_{3,5} - y_{3,5} \end{bmatrix} = \begin{bmatrix} 0{,}1941 - 0 \\ 0{,}2021 - 0 \\ 0{,}1928 - 0 \\ 0{,}2131 - 1 \\ 0{,}1979 - 0 \end{bmatrix} = \begin{bmatrix} 0{,}1941 \\ 0{,}2021 \\ 0{,}1928 \\ -0{,}7869 \\ 0{,}1979 \end{bmatrix}$$

Untuk data ke-4

$$\mathbf{o}_4 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ 0{,}2128 \\ 0{,}1978 \end{bmatrix}, \mathbf{y}_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Maka:

$$\boldsymbol{\delta}_4 = \mathbf{o}_4 - \mathbf{y}_4$$

$$\boldsymbol{\delta}_4 = \begin{bmatrix} o_{4,1} - y_{4,1} \\ o_{4,2} - y_{4,2} \\ o_{4,3} - y_{4,3} \\ o_{4,4} - y_{4,4} \\ o_{4,5} - y_{4,5} \end{bmatrix} = \begin{bmatrix} 0{,}1944 - 0 \\ 0{,}2022 - 0 \\ 0{,}1928 - 0 \\ 0{,}2128 - 1 \\ 0{,}1978 - 0 \end{bmatrix} = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7872 \\ 0{,}1978 \end{bmatrix}$$

Untuk data ke-5

$$\mathbf{o}_5 = \begin{bmatrix} 0{,}1951 \\ 0{,}2025 \\ 0{,}1930 \\ 0{,}2119 \\ 0{,}1974 \end{bmatrix}, \mathbf{y}_5 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Maka:

$$\boldsymbol{\delta}_5 = \mathbf{o}_5 - \mathbf{y}_5$$

$$\boldsymbol{\delta}_5 = \begin{bmatrix} o_{5,1} - y_{5,1} \\ o_{5,2} - y_{5,2} \\ o_{5,3} - y_{5,3} \\ o_{5,4} - y_{5,4} \\ o_{5,5} - y_{5,5} \end{bmatrix} = \begin{bmatrix} 0{,}1951 - 0 \\ 0{,}2025 - 0 \\ 0{,}1930 - 0 \\ 0{,}2119 - 1 \\ 0{,}1974 - 0 \end{bmatrix} = \begin{bmatrix} 0{,}1951 \\ 0{,}2025 \\ 0{,}1930 \\ -0{,}7881 \\ 0{,}1974 \end{bmatrix}$$

Nilai $\boldsymbol{\delta}_i$ menunjukkan besarnya selisih antara probabilitas prediksi dan label sebenarnya untuk setiap kelas.

Gradien terhadap bobot $\mathbf{V}$ pada lapisan *hidden* menuju *output* dihitung menggunakan Persamaan (2.21):

$$\nabla_{\mathbf{V}} L_i = (\mathbf{o}_i - \mathbf{y}_i)\mathbf{z}_i^T$$

$$\nabla_{\mathbf{V}} L_i = \boldsymbol{\delta}_i \mathbf{z}_i^T$$

$\mathbf{z}_i$ merupakan vektor keluaran pada lapisan *hidden*. Agar dapat dilakukan perkalian matriks, vektor $\mathbf{z}_i$ *ditranspose* menjadi:

$$\boldsymbol{z}_i^T = \begin{bmatrix} 0{,}0528 & 0{,}1385 & 0{,}0867 & 0{,}0642 & 0{,}0573 \\ 0{,}1181 & 0{,}2231 & 0{,}0960 & 0{,}0951 & 0{,}1340 \\ 0{,}0717 & 0{,}1497 & 0{,}0882 & 0{,}0812 & 0{,}0709 \\ 0{,}0519 & 0{,}1364 & 0{,}0865 & 0{,}0643 & 0{,}0556 \\ 0{,}0000 & 0{,}0967 & 0{,}0816 & 0{,}0134 & 0{,}0100 \end{bmatrix}$$

Diketahui nilai kesalahan *output* pada data ke-1 hingga ke-5:

$$\boldsymbol{\delta}_1 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7872 \\ 0{,}1978 \end{bmatrix}, \boldsymbol{\delta}_2 = \begin{bmatrix} 0{,}1936 \\ 0{,}2019 \\ 0{,}1924 \\ -0{,}7862 \\ 0{,}1982 \end{bmatrix}, \boldsymbol{\delta}_3 = \begin{bmatrix} 0{,}1941 \\ 0{,}2021 \\ 0{,}1928 \\ -0{,}7869 \\ 0{,}1979 \end{bmatrix}, \boldsymbol{\delta}_4 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7872 \\ 0{,}1978 \end{bmatrix}, \boldsymbol{\delta}_5 = \begin{bmatrix} 0{,}1951 \\ 0{,}2025 \\ 0{,}1930 \\ -0{,}7881 \\ 0{,}1974 \end{bmatrix}$$

Pada gradien terhadap bobot V dihitung untuk data ke-1:

$$\nabla_{\mathbf{V}} L_1 = \boldsymbol{\delta}_1 \mathbf{z}_1^T$$

$$\nabla_{\mathbf{V}} L_1 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7872 \\ 0{,}1978 \end{bmatrix} [0{,}0528 \quad 0{,}1385 \quad 0{,}0867 \quad 0{,}0642 \quad 0{,}0573]$$

$$= [(0{,}1944(0{,}0528)), (0{,}1944(0{,}1385)), (0{,}1944(0{,}0867)),$$

$$(0{,}1944(0{,}0642)), 0{,}1944(0{,}0573))],$$

$$[(0{,}2022(0{,}0528)), (0{,}2022(0{,}1385)), (0{,}2022(0{,}0867)),$$

$$(0{,}2022(0{,}0642)), 0{,}2022(0{,}0573))],$$

$$[(0{,}1928(0{,}0528)), (0{,}1928(0{,}1385)), (0{,}1928(0{,}0867)),$$

$$(0{,}1944(0{,}0642)), 0{,}1944(0{,}0573))],$$

$$[(-0{,}7872(0{,}0528)), (-0{,}7872(0{,}1385)), (-0{,}7872(0{,}0867)),$$

$$(-0{,}7872(0{,}0642)), (-0{,}7872(0{,}0573))],$$

$$[(0{,}1978(0{,}0528)), (0{,}1978(0{,}1385)), (0{,}1978(0{,}0867)),$$

$$(0{,}1978(0{,}0642)), 0{,}1978(0{,}0573))]$$

$$\nabla_{\mathbf{V}} L_1 = \begin{bmatrix} 0{,}0103 & 0{,}0269 & 0{,}0169 & 0{,}0125 & 0{,}0111 \\ 0{,}0107 & 0{,}0280 & 0{,}0175 & 0{,}0130 & 0{,}0116 \\ 0{,}0102 & 0{,}0267 & 0{,}0167 & 0{,}0124 & 0{,}0110 \\ -0{,}0416 & -0{,}1090 & -0{,}0683 & -0{,}0505 & -0{,}0451 \\ 0{,}0104 & 0{,}0274 & 0{,}0171 & 0{,}0127 & 0{,}0113 \end{bmatrix}$$

Pada gradien terhadap bobot V dihitung untuk data ke-2:

$$\nabla_{\mathbf{V}} L_2 = \boldsymbol{\delta}_2 \mathbf{z}_2^T$$

$$\nabla_{\mathbf{V}} L_2 = \begin{bmatrix} 0{,}1936 \\ 0{,}2019 \\ 0{,}1924 \\ -0{,}7862 \\ 0{,}1982 \end{bmatrix} [0{,}1181 \quad 0{,}2231 \quad 0{,}0960 \quad 0{,}0951 \quad 0{,}1340]$$

$$= [(0{,}1936(0{,}1181)), (0{,}1936(0{,}2231)), (0{,}1936(0{,}0960)),$$

$$(0{,}1936(0{,}0951)), 0{,}1936(0{,}1340))],$$

$$[(0{,}2019(0{,}1181)), (0{,}2019(0{,}2231)), (0{,}2019(0{,}0960)),$$

$$(0{,}2019(0{,}0951)), 0{,}2019(0{,}1340))],$$

$$[(0{,}1924(0{,}1181)), (0{,}1924(0{,}2231)), (0{,}1924(0{,}0960)),$$

$$(0{,}1924(0{,}0951)), 0{,}1924(0{,}1340))],$$

$$[(-0{,}7862(0{,}1181)), (-0{,}7862(0{,}2231)), (-0{,}7862(0{,}0960)),$$

$$(-0{,}7862(0{,}0951)), (-0{,}7862(0{,}1340))],$$

$$[(0{,}1982(0{,}1181)), (0{,}1982(0{,}2231)), (0{,}1982(0{,}0960)),$$

$$(0{,}1982(0{,}0951)), 0{,}1982(0{,}1340))]$$

$$\nabla_{\mathbf{V}} L_2 = \begin{bmatrix} 0{,}0229 & 0{,}0432 & 0{,}0186 & 0{,}0184 & 0{,}0259 \\ 0{,}0238 & 0{,}0450 & 0{,}0194 & 0{,}0192 & 0{,}0271 \\ 0{,}0227 & 0{,}0429 & 0{,}0185 & 0{,}0183 & 0{,}0258 \\ -0{,}0929 & -0{,}1754 & -0{,}0755 & -0{,}0748 & -0{,}1054 \\ 0{,}0234 & 0{,}0442 & 0{,}0190 & 0{,}0188 & 0{,}0266 \end{bmatrix}$$

Pada gradien terhadap bobot V dihitung untuk data ke-3:

$$\nabla_{\mathbf{V}} L_3 = \boldsymbol{\delta}_3 \mathbf{z}_3^T$$

$$\nabla_{\mathbf{V}} L_3 = \begin{bmatrix} 0{,}1941 \\ 0{,}2021 \\ 0{,}1928 \\ -0{,}7869 \\ 0{,}1979 \end{bmatrix} [0{,}0717 \quad 0{,}1497 \quad 0{,}0882 \quad 0{,}0812 \quad 0{,}0709]$$

$$= [(0{,}1941(0{,}0717)), (0{,}1941(0{,}1497)), (0{,}1941(0{,}0882)),$$

$$(0{,}1941(0{,}0812)), (0{,}1941(0{,}0709))],$$

$$[(0{,}2021(0{,}0717)), (0{,}2021(0{,}1497)), (0{,}2021(0{,}0882)),$$

$$(0{,}2021(0{,}0812)), (0{,}2021(0{,}0709))],$$

$$[(0{,}1928(0{,}0717)), (0{,}1928(0{,}1497)), (0{,}1928(0{,}0882)),$$

$$(0{,}1928(0{,}0812)), (0{,}1928(0{,}0709))],$$

$$[(-0{,}7869(0{,}0717)), (-0{,}7869(0{,}1497)), (-0{,}7869(0{,}0882)),$$

$$(-0{,}7869(0{,}0812)), (-0{,}7869(0{,}0709))],$$

$$[(0{,}1979(0{,}0717)), (0{,}1979(0{,}1497)), (0{,}1979(0{,}0882)),$$

$$(0{,}1979(0{,}0812)), (0{,}1979(0{,}0709))]$$

$$\nabla_{\mathbf{V}} L_3 = \begin{bmatrix} 0{,}0139 & 0{,}0291 & 0{,}0171 & 0{,}0158 & 0{,}0138 \\ 0{,}0145 & 0{,}0303 & 0{,}0178 & 0{,}0164 & 0{,}0143 \\ 0{,}0138 & 0{,}0289 & 0{,}0170 & 0{,}0157 & 0{,}0137 \\ -0{,}0564 & -0{,}1178 & -0{,}0694 & -0{,}0639 & -0{,}0558 \\ 0{,}0142 & 0{,}0296 & 0{,}0175 & 0{,}0161 & 0{,}0140 \end{bmatrix}$$

Pada gradien terhadap bobot V dihitung untuk data ke-4:

$$\nabla_{\mathbf{V}} L_4 = \boldsymbol{\delta}_4 \mathbf{z}_4^T$$

$$\nabla_{\mathbf{V}} L_4 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7872 \\ 0{,}1978 \end{bmatrix} [0{,}0519 \quad 0{,}1364 \quad 0{,}0865 \quad 0{,}0643 \quad 0{,}0556]$$

$$= [(0{,}1944(0{,}0519)), (0{,}1944(0{,}1364)), (0{,}1944(0{,}0865)),$$

$$(0{,}1944(0{,}0643)), 0{,}1944(0{,}0556))],$$

$$[(0{,}2022(0{,}0519)), (0{,}2022(0{,}1364)), (0{,}2022(0{,}0865)),$$

$$(0{,}2022(0{,}0643)), 0{,}2022(0{,}0556))],$$

$$[(0{,}1928(0{,}0519)), (0{,}1928(0{,}1364)), (0{,}1928(0{,}0865)),$$

$$(0{,}1928(0{,}0643)), 0{,}1928(0{,}0556))],$$

$$[(-0{,}7872(0{,}0519)), (-0{,}7872(0{,}1364)), (-0{,}7872(0{,}0865)),$$

$$(-0{,}7872(0{,}0643)), (-0{,}7872(0{,}0556))],$$

$$[(0{,}1978(0{,}0519)), (0{,}1978(0{,}1364)), (0{,}1978(0{,}0865)),$$

$$(0{,}1978(0{,}0643)), 0{,}1978(0{,}0556))]$$

$$\nabla_{\mathbf{V}} L_4 = \begin{bmatrix} 0{,}0101 & 0{,}0265 & 0{,}0168 & 0{,}0125 & 0{,}0108 \\ 0{,}0105 & 0{,}0276 & 0{,}0175 & 0{,}0130 & 0{,}0112 \\ 0{,}0100 & 0{,}0263 & 0{,}0167 & 0{,}0124 & 0{,}0107 \\ -0{,}0409 & -0{,}1074 & -0{,}0681 & -0{,}0506 & -0{,}0438 \\ 0{,}0103 & 0{,}0270 & 0{,}0171 & 0{,}0127 & 0{,}0110 \end{bmatrix}$$

Pada gradien terhadap bobot V dihitung untuk data ke-5:

$$\nabla_{\mathbf{V}} L_5 = \boldsymbol{\delta}_5 \mathbf{z}_5^T$$

$$\nabla_{\mathbf{V}} L_5 = \begin{bmatrix} 0{,}1951 \\ 0{,}2025 \\ 0{,}1930 \\ -0{,}7881 \\ 0{,}1974 \end{bmatrix} [0{,}0000 \quad 0{,}0967 \quad 0{,}0816 \quad 0{,}0134 \quad 0{,}0100]$$

$$= [(0{,}1951(0{,}0000)), (0{,}1951(0{,}0967)), (0{,}1951(0{,}0816)),$$

$$(0{,}1951(0{,}0134)), (0{,}1951(0{,}0100))],$$

$$[(0{,}2025(0{,}0000)), (0{,}2025(0{,}0967)), (0{,}2025(0{,}0816)),$$

$$(0{,}2025(0{,}0134)), (0{,}2025(0{,}0100))],$$

$$[(0{,}1930(0{,}0000)), (0{,}1930(0{,}0967)), (0{,}1930(0{,}0816)),$$

$$(0{,}1930(0{,}0134)), (0{,}1930(0{,}0100))],$$

$$[(-0{,}7881(0{,}0000)), (-0{,}7881(0{,}0967)), (-0{,}7881(0{,}0816)),$$

$$(-0{,}7881(0{,}0134)), (-0{,}7881(0{,}0100))],$$

$$[(0{,}1974(0{,}0000)), (0{,}1974(0{,}0967)), (0{,}1974(0{,}0816)),$$

$$(0{,}1974(0{,}0134)), (0{,}1974(0{,}0100))]$$

$$\nabla_{\mathbf{V}} L_5 = \begin{bmatrix} 0{,}0000 & 0{,}0189 & 0{,}0159 & 0{,}0026 & 0{,}0020 \\ 0{,}0000 & 0{,}0196 & 0{,}0165 & 0{,}0027 & 0{,}0020 \\ 0{,}0000 & 0{,}0187 & 0{,}0157 & 0{,}0026 & 0{,}0019 \\ 0{,}0000 & -0{,}0762 & -0{,}0643 & -0{,}0106 & -0{,}0079 \\ 0{,}0000 & 0{,}0191 & 0{,}0161 & 0{,}0026 & 0{,}0020 \end{bmatrix}$$

Gradien $\nabla_{\mathbf{V}} L_i$ menunjukkan seberapa besar perubahan bobot $\mathbf{V}$ memiliki pengaruh signifikan terhadap nilai kesalahan, sehingga memerlukan penyesuaian yang lebih besar pada proses pembaruan parameter.

Pembaruan parameter pada lapisan output tidak hanya melibatkan bobot, tetapi juga bias. Gradien terhadap bias dihitung berdasarkan sinyal kesalahan pada lapisan output yang telah diperoleh sebelumnya. Berdasarkan turunan fungsi *loss* terhadap bias menggunakan Persamaan (2.22):

$$\nabla_{\mathbf{c}} L_i = \boldsymbol{\delta}_i$$

Persamaan tersebut, diperoleh gradien bias *output* sebagai berikut:

$$\nabla_{\mathbf{c}} L_1 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7872 \\ 0{,}1978 \end{bmatrix}, \nabla_{\mathbf{c}} L_2 = \begin{bmatrix} 0{,}1936 \\ 0{,}2019 \\ 0{,}1924 \\ -0{,}7862 \\ 0{,}1982 \end{bmatrix}, \nabla_{\mathbf{c}} L_3 = \begin{bmatrix} 0{,}1941 \\ 0{,}2021 \\ 0{,}1928 \\ -0{,}7869 \\ 0{,}1979 \end{bmatrix}, \nabla_{\mathbf{c}} L_4 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7872 \\ 0{,}1978 \end{bmatrix},$$

$$\nabla_{\mathbf{c}} L_5 = \begin{bmatrix} 0{,}1951 \\ 0{,}2025 \\ 0{,}1930 \\ -0{,}7881 \\ 0{,}1974 \end{bmatrix}$$

Gradien bias pada setiap data memiliki nilai yang sama dengan sinyal kesalahan pada lapisan *output*.

Nilai kesalahan pada *output* diteruskan ke *hidden* agar dilakukan substitusi nilai untuk menghitung sinyal kesalahan pada lapisan *hidden*. Berdasarkan Persamaan (2.23):

$$\boldsymbol{\gamma}_i = (\mathbf{V}^{\mathrm{T}} \boldsymbol{\delta}_i) \odot f'(\mathbf{a}_i)$$

Menggunakan bobot *hidden* menuju *output* V:

$$\mathbf{V} = \begin{bmatrix} 0{,}0102 & 0{,}0235 & 0{,}0477 & 0{,}0375 & 0{,}0220 \\ 0{,}0573 & 0{,}0053 & 0{,}0221 & 0{,}0303 & 0{,}0402 \\ 0{,}0764 & 0{,}0120 & 0{,}0466 & 0{,}0552 & -0{,}0049 \\ 0{,}0568 & 0{,}0088 & -0{,}0028 & 0{,}0944 & 0{,}0962 \\ 0{,}0789 & 0{,}0235 & 0{,}0007 & 0{,}0653 & 0{,}0384 \end{bmatrix}$$

Pada perhitungan lapisan *hidden* tidak semua nilai $\mathbf{a}_i$ menghasilkan nilai aktivasi $\mathbf{z}_i \leq 0$ sehingga, $f'(\mathbf{a}_i) = 0$. Sedangkan, untuk nilai aktivasi $\mathbf{z}_i > 0$ maka $f'(\mathbf{a}_i) = 1$ yang sesuai dengan Persamaan (2.6).

Diketahui nilai kesalahan *output* pada lima data awal yaitu:

$$\boldsymbol{\delta}_1 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7872 \\ 0{,}1978 \end{bmatrix}, \boldsymbol{\delta}_2 = \begin{bmatrix} 0{,}1936 \\ 0{,}2019 \\ 0{,}1924 \\ -0{,}7862 \\ 0{,}1982 \end{bmatrix}, \boldsymbol{\delta}_3 = \begin{bmatrix} 0{,}1941 \\ 0{,}2021 \\ 0{,}1928 \\ -0{,}7869 \\ 0{,}1979 \end{bmatrix}, \boldsymbol{\delta}_4 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7872 \\ 0{,}1978 \end{bmatrix}, \boldsymbol{\delta}_5 = \begin{bmatrix} 0{,}1951 \\ 0{,}2025 \\ 0{,}1930 \\ -0{,}7881 \\ 0{,}1974 \end{bmatrix}$$

Pada data ke-1 dihitung sinyal kesalahan, maka:

$$\boldsymbol{\gamma}_1 = (v_{1,1}\delta_{1,1} + v_{2,1}\delta_{1,2} + v_{3,1}\delta_{1,3} + v_{4,1}\delta_{1,4} + +v_{5,1}\delta_{1,5})$$

$$\gamma_{1,1} = (0{,}0102)(0{,}1944) + (0{,}0573)(0{,}2022) + (0{,}0764)(0{,}1928) + (0{,}0568)(-0{,}7872) + (0{,}0789)(0{,}1978)$$

$$= 0{,}0020 + 0{,}0116 + 0{,}0147 + (-0{,}0447) + 0{,}0156 = -0{,}0008$$

$$\gamma_{1,2} = (0{,}0235)(0{,}1944) + (0{,}0053)(0{,}2022) + (0{,}0120)(0{,}1928) + (0{,}0088)(-0{,}7872) + (0{,}0235)(0{,}1978)$$

$$= 0{,}0046 + 0{,}0011 + 0{,}0023 + (-0{,}0069) + 0{,}0046 = 0{,}0057$$

$$\gamma_{1,3} = (0{,}0477)(0{,}1944) + (0{,}0221)(0{,}2022) + (0{,}0466)(0{,}1928) + (-0{,}0028)(-0{,}7872) + (0{,}0007)(0{,}1978)$$

$$= 0{,}0093 + 0{,}0045 + 0{,}0090 + 0{,}0022 + 0{,}0001 = 0{,}0251$$

$$\gamma_{1,4} = (0{,}0375)(0{,}1944) + (0{,}0303)(0{,}2022) + (0{,}0552)(0{,}1928) + (0{,}0944)(-0{,}7872) + (0{,}0653)(0{,}1978)$$

$$= 0{,}0073 + 0{,}0061 + 0{,}0106 + (-0{,}0743) + 0{,}0129 = -0{,}0374$$

$$\gamma_{1,5} = (0{,}0220)(0{,}1944) + (0{,}0402)(0{,}2022) + (-0{,}0049)(0{,}1928) + (0{,}0962)(-0{,}7872) + (0{,}0384)(0{,}1978)$$

$$= 0{,}0043 + 0{,}0081 + (-0{,}0009) + (-0{,}0757) + 0{,}0076 = -0{,}0566$$

Untuk $z_{1,1}, z_{1,2}, \dots, z_{1,5}$ menghasilkan nilai aktivasi $z_i > 0$ maka $f'(a_i) = 1$.

Sehingga, diperoleh:

$$\boldsymbol{\gamma}_1 = \begin{bmatrix}\gamma_{1,1}\\ \gamma_{1,2}\\ \gamma_{1,3}\\ \gamma_{1,4}\\ \gamma_{1,5}\end{bmatrix} = \begin{bmatrix}(-0{,}0008)1\\ (0{,}0057)1\\ (0{,}0251)1\\ (-0{,}0374)1\\ (-0{,}0566)1\end{bmatrix} = \begin{bmatrix}-0{,}0008\\ 0{,}0057\\ 0{,}0251\\ -0{,}0374\\ -0{,}0566\end{bmatrix}$$

Pada data ke-2 dihitung sinyal kesalahan, maka:

$$\boldsymbol{\gamma}_2 = (v_{1,1}\delta_{2,1} + v_{2,1}\delta_{2,2} + v_{3,1}\delta_{2,3} + v_{4,1}\delta_{2,4} + v_{5,1}\delta_{2,5})$$

$$\gamma_{2,1} = (0{,}0102)(0{,}1936) + (0{,}0573)(0{,}2019) + (0{,}0764)(0{,}1924) + (0{,}0568)(-0{,}7862) + (0{,}0789)(0{,}1982)$$

$$= 0{,}0020 + 0{,}0116 + 0{,}0147 + (-0{,}0447) + 0{,}0156 = -0{,}0008$$

$$\gamma_{2,2} = (0{,}0235)(0{,}1936) + (0{,}0053)(0{,}2019) + (0{,}0120)(0{,}1924) + (0{,}0088)(-0{,}7862) + (0{,}0235)(0{,}1982)$$

$$= 0{,}0045 + 0{,}0011 + 0{,}0023 + (-0{,}0069) + 0{,}0047 = 0{,}0057$$

$$\gamma_{2,3} = (0{,}0477)(0{,}1936) + (0{,}0221)(0{,}2019) + (0{,}0466)(0{,}1924) + (-0{,}0028)(-0{,}7862) + (0{,}0007)(0{,}1982)$$

$$= 0{,}0092 + 0{,}0045 + 0{,}0090 + 0{,}0022 + 0{,}0001 = 0{,}0251$$

$$\gamma_{2,4} = (0{,}0375)(0{,}1936) + (0{,}0303)(0{,}2019) + (0{,}0552)(0{,}1924) + (0{,}0944)(-0{,}7862) + (0{,}0653)(0{,}1982)$$

$$= 0{,}0073 + 0{,}0061 + 0{,}0106 + (-0{,}0742) + 0{,}0129 = -0{,}0373$$

$$\gamma_{2,5} = (0{,}0220)(0{,}1936) + (0{,}0402)(0{,}2019) + (-0{,}0049)(0{,}1924) + (0{,}0962)(-0{,}7862) + (0{,}0384)(0{,}1982)$$

$$= 0{,}0043 + 0{,}0081 + (-0{,}0009) + (-0{,}0756) + 0{,}0076 = -0{,}0565$$

Untuk $z_{2,1}, z_{2,2}, \dots, z_{2,5}$ menghasilkan nilai aktivasi $\mathbf{z}_i > 0$ maka $f'(\mathbf{a}_i) = 1$.

Sehingga, diperoleh:

$$\boldsymbol{\gamma}_2 = 1\begin{bmatrix}\gamma_{2,1}\\ \gamma_{2,2}\\ \gamma_{2,3}\\ \gamma_{2,4}\\ \gamma_{2,5}\end{bmatrix} = \begin{bmatrix}(-0{,}0008)1\\ (0{,}0057)1\\ (0{,}0250)1\\ (-0{,}0373)1\\ (-0{,}0565)1\end{bmatrix} = \begin{bmatrix}-0{,}0008\\ 0{,}0057\\ 0{,}0250\\ -0{,}0373\\ -0{,}0565\end{bmatrix}$$

Pada data ke-3 dihitung sinyal kesalahan, maka:

$$\boldsymbol{\gamma}_3 = (v_{1,1}\delta_{3,1} + v_{2,1}\delta_{3,2} + v_{3,1}\delta_{3,3} + v_{4,1}\delta_{3,4} + v_{5,1}\delta_{3,5})$$

$$\gamma_{3,1} = (0{,}0102)(0{,}1941) + (0{,}0573)(0{,}2021) + (0{,}0764)(0{,}1928) + (0{,}0568)(-0{,}7869) + (0{,}0789)(0{,}1979)$$

$$= 0{,}0020 + 0{,}0116 + 0{,}0147 + (-0{,}0447) + 0{,}0156 = -0{,}0008$$

$$\gamma_{3,2} = (0{,}0235)(0{,}1941) + (0{,}0053)(0{,}2021) + (0{,}0120)(0{,}1928) +$$

$$(0{,}0088)(-0{,}7869) + (0{,}0235)(0{,}1979)$$

$$= 0{,}0046 + 0{,}0011 + 0{,}0023 + (-0{,}0069) + 0{,}0047 = 0{,}0058$$

$$\gamma_{3,3} = (0{,}0477)(0{,}1941) + (0{,}0221)(0{,}2021) + (0{,}0466)(0{,}1928) +$$

$$(-0{,}0028)(-0{,}7869) + (0{,}0007)(0{,}1979)$$

$$= 0{,}0093 + 0{,}0045 + 0{,}0090 + 0{,}0022 + 0{,}0001 = 0{,}0251$$

$$\gamma_{3,4} = (0{,}0375)(0{,}1941) + (0{,}0303)(0{,}2021) + (0{,}0552)(0{,}1928) +$$

$$(0{,}0944)(-0{,}7869) + (0{,}0653)(0{,}1979)$$

$$= 0{,}0073 + 0{,}0061 + 0{,}0106 + (-0{,}0743) + 0{,}0129 = -0{,}0374$$

$$\gamma_{3,5} = (0{,}0220)(0{,}1941) + (0{,}0402)(0{,}2021) + (-0{,}0049)(0{,}1928) +$$

$$(0{,}0962)(-0{,}7869) + (0{,}0384)(0{,}1979)$$

$$= 0{,}0043 + 0{,}0081 + (-0{,}0009) + (-0{,}0757) + 0{,}0076 = -0{,}0566$$

Untuk $z_{3,1}, z_{3,2}, \dots, z_{3,5}$ menghasilkan nilai aktivasi $\mathbf{z}_i > 0$ maka $f'(\mathbf{a}_i) = 1$.

Sehingga, diperoleh:

$$\boldsymbol{\gamma}_3 = \begin{bmatrix} \gamma_{3,1} \\ \gamma_{3,2} \\ \gamma_{3,3} \\ \gamma_{3,4} \\ \gamma_{3,5} \end{bmatrix} = \begin{bmatrix} (-0{,}0008)1 \\ (0{,}0058)1 \\ (0{,}0251)1 \\ (-0{,}0374)1 \\ (-0{,}0566)1 \end{bmatrix} = \begin{bmatrix} -0{,}0008 \\ 0{,}0058 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix}$$

Pada data ke-4 dihitung sinyal kesalahan, maka:

$$\boldsymbol{\gamma}_4 = (v_{1,1}\delta_{4,1} + v_{2,1}\delta_{4,2} + v_{3,1}\delta_{4,3} + v_{4,1}\delta_{4,4} + v_{5,1}\delta_{4,5})$$

$$\gamma_{4,1} = (0{,}0102)(0{,}1944) + (0{,}0573)(0{,}2022) + (0{,}0764)(0{,}1928) +$$

$$(0{,}0568)(-0{,}7872) + (0{,}0789)(0{,}1978)$$

$$= 0{,}0020 + 0{,}0116 + 0{,}0147 + (-0{,}0447) + 0{,}0156 = -0{,}0008$$

$$\gamma_{4,2} = (0{,}0235)(0{,}1944) + (0{,}0053)(0{,}2022) + (0{,}0120)(0{,}1928) +$$

$$(0{,}0088)(-0{,}7872) + (0{,}0235)(0{,}1978)$$

$$= 0{,}0046 + 0{,}0011 + 0{,}0023 + (-0{,}0069) + 0{,}0046 = 0{,}0057$$

$$\gamma_{4,3} = (0{,}0477)(0{,}1944) + (0{,}0221)(0{,}2022) + (0{,}0466)(0{,}1928) +$$

$$(-0{,}0028)(-0{,}7872) + (0{,}0007)(0{,}1978)$$

$$= 0{,}0093 + 0{,}0045 + 0{,}0090 + 0{,}0022 + 0{,}0001 = 0{,}0251$$

$$\gamma_{4,4} = (0{,}0375)(0{,}1944) + (0{,}0303)(0{,}2022) + (0{,}0552)(0{,}1928) +$$

$$(0{,}0944)(-0{,}7872) + (0{,}0653)(0{,}1978)$$

$$= 0{,}0073 + 0{,}0061 + 0{,}0106 + (-0{,}0743) + 0{,}0129 = -0{,}0374$$

$$\gamma_{4,5} = (0{,}0220)(0{,}1944) + (0{,}0402)(0{,}2022) + (-0{,}0049)(0{,}1928) +$$

$$(0{,}0962)(-0{,}7872) + (0{,}0384)(0{,}1978)$$

$$= 0{,}0043 + 0{,}0081 + (-0{,}0009) + (-0{,}0757) + 0{,}0076 = -0{,}0566$$

Untuk $z_{4,1}, z_{4,2}, \dots, z_{4,5}$ menghasilkan nilai aktivasi $\mathbf{z}_i > 0$ maka $f'(\mathbf{a}_i) = 1$.

Sehingga, diperoleh:

$$\boldsymbol{\gamma}_4 = \begin{bmatrix} \gamma_{4,1} \\ \gamma_{4,2} \\ \gamma_{4,3} \\ \gamma_{4,4} \\ \gamma_{4,5} \end{bmatrix} = \begin{bmatrix} (-0{,}0008)1 \\ (0{,}0057)1 \\ (0{,}0251)1 \\ (-0{,}0374)1 \\ (-0{,}0566)1 \end{bmatrix} = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix}$$

Pada data ke-5 dihitung sinyal kesalahan, maka:

$$\boldsymbol{\gamma}_5 = (v_{1,1}\delta_{5,1} + v_{2,1}\delta_{5,2} + v_{3,1}\delta_{5,3} + v_{4,1}\delta_{5,4} + v_{5,1}\delta_{5,5})$$

$$\gamma_{5,1} = (0{,}0102)(0{,}1951) + (0{,}0573)(0{,}2025) + (0{,}0764)(0{,}1930) +$$

$$(0{,}0568)(-0{,}7881) + (0{,}0789)(0{,}1974)$$

$$= 0{,}0020 + 0{,}0116 + 0{,}0147 + (-0{,}0448) + 0{,}0156 = -0{,}0009$$

$$\gamma_{5,2} = (0{,}0235)(0{,}1951) + (0{,}0053)(0{,}2025) + (0{,}0120)(0{,}1930) +$$

$$(0{,}0088)(-0{,}7881) + (0{,}0235)(0{,}1974)$$

$$= 0{,}0046 + 0{,}0011 + 0{,}0023 + (-0{,}0069) + 0{,}0046 = 0{,}0057$$

$$\gamma_{5,3} = (0{,}0477)(0{,}1951) + (0{,}0221)(0{,}2025) + (0{,}0466)(0{,}1930) + (-0{,}0028)(-0{,}7881) + (0{,}0007)(0{,}1974)$$
$$= 0{,}0093 + 0{,}0045 + 0{,}0090 + 0{,}0022 + 0{,}0001 = 0{,}0251$$
$$\gamma_{5,4} = (0{,}0375)(0{,}1951) + (0{,}0303)(0{,}2025) + (0{,}0552)(0{,}1930) + (0{,}0944)(-0{,}7881) + (0{,}0653)(0{,}1974)$$
$$= 0{,}0073 + 0{,}0061 + 0{,}0106 + (-0{,}0744) + 0{,}0129 = -0{,}0374$$
$$\gamma_{5,5} = (0{,}0220)(0{,}1951) + (0{,}0402)(0{,}2025) + (-0{,}0049)(0{,}1930) + (0{,}0962)(-0{,}7881) + (0{,}0384)(0{,}1974)$$
$$= 0{,}0043 + 0{,}0081 + (-0{,}0009) + (-0{,}0758) + 0{,}0076 = -0{,}0567$$

Untuk $z_{5,1}$ menghasilkan nilai aktivasi $\mathbf{z}_i = 0$ maka $f'(\mathbf{a}_i) = 0$. Sedangkan, pada $z_{5,2}, z_{5,3}, \dots, z_{5,5}$ menghasilkan nilai aktivasi $\mathbf{z}_i > 0$ maka $f'(\mathbf{a}_i) = 1$. Sehingga, diperoleh:

$$\boldsymbol{\gamma}_5 = \begin{bmatrix} \gamma_{5,1} \\ \gamma_{5,2} \\ \gamma_{5,3} \\ \gamma_{5,4} \\ \gamma_{5,5} \end{bmatrix} = \begin{bmatrix} (0{,}0000)0 \\ (0{,}0057)1 \\ (0{,}0251)1 \\ (-0{,}0375)1 \\ (-0{,}0567)1 \end{bmatrix} = \begin{bmatrix} 0{,}0000 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0375 \\ -0{,}0567 \end{bmatrix}$$

Proses ini merupakan tahapan penting dalam *backpropagation* karena menjadi dasar dalam menghitung gradien bobot antara lapisan *input* dan *hidden* yang merujuk pada Persamaan (2.24):

$$\nabla_{\mathbf{W}} L_i = \boldsymbol{\gamma}_i \, \mathbf{x}_i$$

Diketahui sinyal kesalahan *hidden* pada lima data awal yaitu:

$$\boldsymbol{\gamma}_1 = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix}, \boldsymbol{\gamma}_2 = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0250 \\ -0{,}0373 \\ -0{,}0565 \end{bmatrix}, \boldsymbol{\gamma}_3 = \begin{bmatrix} -0{,}0008 \\ 0{,}0058 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix}, \boldsymbol{\gamma}_4 = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix}, \boldsymbol{\gamma}_5 = \begin{bmatrix} 0{,}0000 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0567 \end{bmatrix}$$

Dilakukan perhitungan gradien terhadap bobot **W** untuk data ke-1:

$$\nabla_{\mathbf{W}} L_1 = \boldsymbol{\gamma}_1 \, \mathbf{x}_1$$

$$\nabla_{\mathbf{W}} L_1 = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix} [0{,}1176 \quad 0{,}6000]$$

$$\nabla_{\mathbf{W}} L_1 = \begin{bmatrix} -0{,}0008(0{,}1176) & -0{,}0008(0{,}6000) \\ 0{,}0057(0{,}1176) & 0{,}0057(0{,}6000) \\ 0{,}0251(0{,}1176) & 0{,}0251(0{,}6000) \\ -0{,}0374(0{,}1176) & -0{,}0374(0{,}6000) \\ -0{,}0566(0{,}1176) & -0{,}0566(0{,}6000) \end{bmatrix}$$

$$\nabla_{\mathbf{W}} L_1 = \begin{bmatrix} -0{,}0001 & -0{,}0005 \\ 0{,}0007 & 0{,}0034 \\ 0{,}0030 & 0{,}0151 \\ -0{,}0044 & -0{,}0224 \\ -0{,}0067 & -0{,}0340 \end{bmatrix}$$

Pada data ke-2 dilakukan perhitungan gradien sebagai berikut:

$$\nabla_{\mathbf{W}} L_2 = \boldsymbol{\gamma}_2 \, \mathbf{x}_2$$

$$\nabla_{\mathbf{W}} L_2 = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0250 \\ -0{,}0373 \\ -0{,}0565 \end{bmatrix} [1{,}0000 \quad 1{,}0000]$$

$$\nabla_{\mathbf{W}} L_2 = \begin{bmatrix} -0{,}0008(1{,}0000) & -0{,}0008(1{,}0000) \\ 0{,}0057(1{,}0000) & 0{,}0057(1{,}0000) \\ 0{,}0250(1{,}0000) & 0{,}0250(1{,}0000) \\ -0{,}0373(1{,}0000) & -0{,}0373(1{,}0000) \\ -0{,}0565(1{,}0000) & -0{,}0565(1{,}0000) \end{bmatrix}$$

$$\nabla_{\mathbf{W}} L_2 = \begin{bmatrix} -0{,}0008 & -0{,}0008 \\ 0{,}0057 & 0{,}0057 \\ 0{,}0250 & 0{,}0250 \\ -0{,}0373 & -0{,}0373 \\ -0{,}0565 & -0{,}0565 \end{bmatrix}$$

Pada data ke-3 dilakukan perhitungan gradien sebagai berikut:

$$\nabla_{\mathbf{W}} L_3 = \boldsymbol{\gamma}_3 \, \mathbf{x}_3$$

$$\nabla_{\mathbf{W}} L_3 = \begin{bmatrix} -0{,}0008 \\ 0{,}0058 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix} [0{,}1176 \quad 0{,}8000]$$

$$\nabla_{\mathbf{W}} L_3 = \begin{bmatrix} -0{,}0008(0{,}1176) & -0{,}0008(0{,}8000) \\ 0{,}0058(0{,}1176) & 0{,}0058(0{,}8000) \\ 0{,}0251(0{,}1176) & 0{,}0251(0{,}8000) \\ -0{,}0374(0{,}1176) & -0{,}0374(0{,}8000) \\ -0{,}0566(0{,}1176) & -0{,}0566(0{,}8000) \end{bmatrix}$$

$$\nabla_{\mathbf{W}} L_3 = \begin{bmatrix} -0{,}0001 & -0{,}0006 \\ 0{,}0007 & 0{,}0046 \\ 0{,}0030 & 0{,}0201 \\ -0{,}0044 & -0{,}0299 \\ -0{,}0067 & -0{,}0453 \end{bmatrix}$$

Pada untuk data ke-4 dilakukan perhitungan gradien sebagai berikut:

$$\nabla_{\mathbf{W}} L_4 = \boldsymbol{\gamma}_4 \, \mathbf{x}_4$$

$$\nabla_{\mathbf{W}} L_4 = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix} [0{,}0882 \quad 0{,}6000]$$

$$\nabla_{\mathbf{W}} L_4 = \begin{bmatrix} -0{,}0008(0{,}0882) & -0{,}0008(0{,}6000) \\ 0{,}0057(0{,}0882) & 0{,}0057(0{,}6000) \\ 0{,}0251(0{,}0882) & 0{,}0251(0{,}6000) \\ -0{,}0374(0{,}0882) & -0{,}0374(0{,}6000) \\ -0{,}0566(0{,}0882) & -0{,}0566(0{,}6000) \end{bmatrix}$$

$$\nabla_{\mathbf{W}} L_4 = \begin{bmatrix} -0{,}0001 & -0{,}0005 \\ 0{,}0005 & 0{,}0034 \\ 0{,}0022 & 0{,}0151 \\ -0{,}0033 & -0{,}0224 \\ -0{,}0050 & -0{,}0340 \end{bmatrix}$$

Pada data ke-5 dilakukan perhitungan gradien sebagai berikut:

$$\nabla_{\mathbf{W}} L_5 = \boldsymbol{\gamma}_5 \, \mathbf{x}_5$$

$$\nabla_{\mathbf{W}} L_5 = \begin{bmatrix} 0{,}0000 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0567 \end{bmatrix} [0{,}0000 \quad 0{,}0000]$$

$$\nabla_{\mathbf{W}} L_5 = \begin{bmatrix} 0{,}0000(0{,}0000) & 0{,}0000(0{,}0000) \\ 0{,}0057(0{,}0000) & 0{,}0057(0{,}0000) \\ 0{,}0251(0{,}0000) & 0{,}0251(0{,}0000) \\ -0{,}0375(0{,}0000) & -0{,}0374(0{,}0000) \\ -0{,}0567(0{,}0000) & -0{,}0567(0{,}0000) \end{bmatrix}$$

$$\nabla_{\mathbf{W}} L_5 = \begin{bmatrix} 0{,}0000 & 0{,}0000 \\ 0{,}0000 & 0{,}0000 \\ 0{,}0000 & 0{,}0000 \\ 0{,}0000 & 0{,}0000 \\ 0{,}0000 & 0{,}0000 \end{bmatrix}$$

Langkah selanjutnya adalah menghitung gradien terhadap bias pada lapisan *hidden*. Berdasarkan turunan fungsi *loss* terhadap bias pada Persamaan (2.25):

$$\nabla_{\mathbf{b}} L_i = \boldsymbol{\gamma}_i$$

Menggunakan persamaan tersebut, diperoleh gradien bias *hidden* sebagai berikut:

$$\nabla_{\mathbf{b}} L_1 = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix}, \nabla_{\mathbf{b}} L_2 = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0250 \\ -0{,}0373 \\ -0{,}0565 \end{bmatrix}, \nabla_{\mathbf{b}} L_3 = \begin{bmatrix} -0{,}0008 \\ 0{,}0058 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix}, \nabla_{\mathbf{b}} L_4 = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix},$$

$$\nabla_{\mathbf{b}} L_5 = \begin{bmatrix} 0{,}0000 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0567 \end{bmatrix}$$

Gradien bias *hidden* pada setiap data memiliki nilai yang sama dengan sinyal kesalahan pada lapisan *hidden*. Nilai rata-rata gradien ini digunakan sebagai dasar dalam proses *update* parameter, sehingga arah dan besar perubahan parameter menjadi lebih stabil.

6. *Update* Parameter

Proses ini dilakukan menggunakan metode *gradient descent* yang bertujuan untuk meminimalkan nilai fungsi *loss*. Pada tahap ini, seluruh parameter

meliputi bobot dan bias diperbarui berdasarkan nilai gradien yang telah diperoleh dari proses *backpropagation*. Aturan pembaruan parameter merujuk pada Persamaan (2.26) dan Persamaan (2.27). Pada perhitungan ini digunakan $\eta = 0{,}1$. Nilai learning rate tersebut digunakan untuk menentukan besarnya langkah pembaruan parameter pada setiap iterasi.

a. *Update* Bobot *Hidden* menuju *Output*

Terdapat $n = 5$, maka gradien total diperoleh dengan menghitung rata-rata:

$$\nabla_{\mathbf{V}} E = \frac{1}{5}\sum_{i=1}^{5}(\nabla_{\mathbf{V}} L_i)$$

Selanjutnya, dilakukan perhitungan elemen per elemen. Sebagai ilustrasi:

$$\nabla_{\mathbf{V}} E = \tfrac{1}{5}(\nabla_V L_1 + \nabla_V L_2 + \nabla_V L_3 + \nabla_V L_4 + \nabla_V L_5)$$

$$(\nabla_V E)_{1,1} = \frac{0{,}0103+0{,}0229+0{,}0139+0{,}0101+0{,}0000}{5} = 0{,}0114$$

$$(\nabla_V E)_{2,1} = \frac{0{,}0107+0{,}0238+0{,}0145+0{,}0105+0{,}0000}{5} = 0{,}0119$$

$$(\nabla_V E)_{3,1} = \frac{0{,}0102+0{,}0227+0{,}0138+0{,}0100+0{,}0000}{5} = 0{,}0113$$

$$(\nabla_V E)_{4,1} = \frac{(-0{,}0416)+(-0{,}0929)+(-0{,}0564)+(-0{,}0409)+(0{,}0000)}{5} = -0{,}0464$$

$$(\nabla_V E)_{5,1} = \frac{0{,}0104+0{,}0234+0{,}0142+0{,}0103+0{,}0000}{5} = 0{,}0117$$

Dengan cara yang sama untuk seluruh elemen, diperoleh:

$$\nabla_{\mathbf{V}} E = \begin{bmatrix} 0{,}0114 & 0{,}0289 & 0{,}0171 & 0{,}0124 & 0{,}0127 \\ 0{,}0119 & 0{,}0301 & 0{,}0177 & 0{,}0129 & 0{,}0132 \\ 0{,}0113 & 0{,}0287 & 0{,}0169 & 0{,}0123 & 0{,}0126 \\ -0{,}0464 & -0{,}1172 & -0{,}0691 & -0{,}0501 & -0{,}0516 \\ 0{,}0117 & 0{,}0295 & 0{,}0174 & 0{,}0126 & 0{,}0130 \end{bmatrix}$$

Selanjutnya, bobot diperbarui menggunakan:

$$\mathbf{V}^{baru} = \mathbf{V}^{lama} - \eta \nabla_{\mathbf{V}} E$$

Diketahui:

$$\mathbf{V}^{lama} = \begin{bmatrix} 0{,}0102 & 0{,}0235 & 0{,}0477 & 0{,}0375 & 0{,}0220 \\ 0{,}0573 & 0{,}0053 & 0{,}0221 & 0{,}0303 & 0{,}0402 \\ 0{,}0764 & 0{,}0120 & 0{,}0466 & 0{,}0552 & -0{,}0049 \\ 0{,}0568 & 0{,}0088 & -0{,}0028 & 0{,}0944 & 0{,}0962 \\ 0{,}0789 & 0{,}0235 & 0{,}0007 & 0{,}0653 & 0{,}0384 \end{bmatrix}$$

Maka, diberikan cuplikan perhitungan V^{baru}:

$$\mathbf{V}^{baru} = \mathbf{V}^{lama} - 0{,}1(\nabla_{\mathbf{V}} E)$$

$$V_{1,1}^{baru} = 0{,}0102 - 0{,}1(0{,}0114) = 0{,}0091$$

$$V_{2,1}^{baru} = 0{,}0573 - 0{,}1(0{,}0119) = 0{,}0561$$

$$V_{3,1}^{baru} = 0{,}0764 - 0{,}1(0{,}0113) = 0{,}0753$$

Dengan cara yang sama untuk seluruh elemen, diperoleh hasil pembaruan:

$$\mathbf{V}^{baru} = \begin{bmatrix} 0{,}0091 & 0{,}0206 & 0{,}0460 & 0{,}0363 & 0{,}0207 \\ 0{,}0561 & 0{,}0023 & 0{,}0203 & 0{,}0290 & 0{,}0389 \\ 0{,}0753 & 0{,}0091 & 0{,}0449 & 0{,}0540 & -0{,}0062 \\ 0{,}0614 & 0{,}0205 & 0{,}0041 & 0{,}0994 & 0{,}1014 \\ 0{,}0777 & 0{,}0206 & -0{,}0010 & 0{,}0640 & 0{,}0371 \end{bmatrix}$$

b. *Update* Bias *Output* (**c**)

Gradien bias *output* dihitung sebagai rata-rata dari $\boldsymbol{\delta}_i$:

$$\nabla_{\boldsymbol{c}} L_i = \frac{1}{5} \sum_{i=1}^{5} \boldsymbol{\delta}_i$$

Selanjutnya, diketahui nilai kesalahan *output* pada data ke-1 hingga ke-5:

$$\boldsymbol{\delta}_1 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7872 \\ 0{,}1978 \end{bmatrix}, \boldsymbol{\delta}_2 = \begin{bmatrix} 0{,}1936 \\ 0{,}2019 \\ 0{,}1924 \\ -0{,}7862 \\ 0{,}1982 \end{bmatrix}, \boldsymbol{\delta}_3 = \begin{bmatrix} 0{,}1941 \\ 0{,}2021 \\ 0{,}1928 \\ -0{,}7869 \\ 0{,}1979 \end{bmatrix}, \boldsymbol{\delta}_4 = \begin{bmatrix} 0{,}1944 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7872 \\ 0{,}1978 \end{bmatrix}, \boldsymbol{\delta}_5 = \begin{bmatrix} 0{,}1951 \\ 0{,}2025 \\ 0{,}1930 \\ -0{,}7881 \\ 0{,}1974 \end{bmatrix}$$

$$\nabla_{\boldsymbol{c}} L_i = \frac{1}{5}(\boldsymbol{\delta}_1 + \boldsymbol{\delta}_2 + \boldsymbol{\delta}_3 + \boldsymbol{\delta}_4 + \boldsymbol{\delta}_5)$$

$$\nabla_{\boldsymbol{c}} L_1 = \frac{0{,}1944 + 0{,}1936 + 0{,}1941 + 0{,}1944 + 0{,}1951}{5} = 0{,}1943$$

$$\nabla_{\boldsymbol{c}} L_2 = \frac{0{,}2022 + 0{,}2019 + 0{,}2021 + 0{,}2022 + 0{,}2025}{5} = 0{,}2022$$

$$\nabla_{\boldsymbol{c}} L_3 = \frac{0{,}1928+0{,}1924+0{,}1928+0{,}1928+0{,}1930}{5} = 0{,}1928$$

$$\nabla_{\boldsymbol{c}} L_4 = \frac{(-0{,}7872)+(-0{,}7862)+(-0{,}7869)+(-0{,}7872)+(-0{,}7881)}{5} = 0{,}7871$$

$$\nabla_{\boldsymbol{c}} L_5 = \frac{0{,}1978+0{,}1982+0{,}1979+0{,}1978+0{,}1974}{5} = 0{,}1978$$

Sehingga diperoleh:

$$\nabla_{\boldsymbol{c}} L_i = \begin{bmatrix} 0{,}1943 \\ 0{,}2022 \\ 0{,}1928 \\ -0{,}7871 \\ 0{,}1978 \end{bmatrix}$$

Kemudian dilakukan pembaruan dengan menggunakan:

$$\mathbf{c}^{lama} = \begin{bmatrix} 0{,}0034 \\ 0{,}0445 \\ -0{,}0062 \\ 0{,}0900 \\ 0{,}0185 \end{bmatrix}$$

Maka:

$$\mathbf{c}^{baru} = \mathbf{c}^{lama} - \eta \nabla_{\mathbf{c}} L_i$$

Diberikan cuplikan perhitungan c^{baru}:

$$\mathbf{c}^{baru} = \mathbf{c}^{lama} - 0{,}1(\nabla_{\mathbf{c}} L_i)$$

$$c_{1,1}^{baru} = 0{,}0034 - 0{,}1(0{,}1943) = -0{,}0160$$

$$c_{2,1}^{baru} = 0{,}0445 - 0{,}1(0{,}2022) = 0{,}0243$$

$$c_{3,1}^{baru} = (-0{,}0062) - 0{,}1(0{,}1928) = -0{,}0255$$

Dengan cara yang sama untuk seluruh elemen, diperoleh hasil pembaruan sebagai berikut:

$$\mathbf{c}^{baru} = \begin{bmatrix} -0{,}0160 \\ 0{,}0243 \\ -0{,}0255 \\ 0{,}1687 \\ -0{,}0013 \end{bmatrix}$$

c. *Update* Bobot *Input* menuju *Hidden*

Karena terdapat $n = 5$, maka gradien total diperoleh dengan menghitung rata-rata:

$$\nabla_{\mathbf{W}} E = \frac{1}{5}\sum_{i=1}^{5}(\nabla_{\mathbf{W}} L_i)$$

Selanjutnya, dilakukan perhitungan elemen per elemen. Sebagai ilustrasi

$\nabla_{\mathbf{W}} E = \frac{1}{5}(\nabla_W L_1 + \nabla_W L_2 + \nabla_W L_3 + \nabla_W L_4 + \nabla_W L_5)$

$\nabla_W E_{1,1} = \frac{-0{,}0001+(-0{,}0008)+(-0{,}0001)+(-0{,}0001)+0{,}0000}{5} = -0{,}0002$

$\nabla_W E_{2,1} = \frac{0{,}0007+0{,}0057+0{,}0007+0{,}0005+0{,}0000}{5} = 0{,}0015$

$\nabla_W E_{3,1} = \frac{0{,}0030+0{,}0250+0{,}0030+0{,}0022+0{,}0000}{5} = 0{,}0066$

Dengan cara yang sama untuk seluruh elemen, diperoleh hasil pembaruan sebagai berikut:

$$\nabla_{\mathbf{W}} E = \begin{bmatrix} -0{,}0002 & -0{,}0005 \\ 0{,}0015 & 0{,}0034 \\ 0{,}0066 & 0{,}0151 \\ -0{,}0099 & -0{,}0224 \\ -0{,}0150 & -0{,}0340 \end{bmatrix}$$

Selanjutnya, bobot diperbarui menggunakan:

$$\mathbf{W}^{baru} = \mathbf{W}^{lama} - \eta \nabla_{\mathbf{W}} E$$

dengan $\mathbf{W}^{lama} = \begin{bmatrix} 0{,}0312 & 0{,}0946 \\ 0{,}0705 & 0{,}0559 \\ 0{,}0072 & 0{,}0072 \\ -0{,}0036 & 0{,}0853 \\ 0{,}0561 & 0{,}0679 \end{bmatrix}$, maka:

Diberikan cuplikan perhitungan $\mathbf{W}^{baru}$:

$$\mathbf{W}^{baru} = \mathbf{W}^{lama} - 0{,}1(\nabla_{\mathbf{W}} E)$$

$W_{1,1}^{baru} = 0{,}0312 - 0{,}1(0{,}0002) = 0{,}0312$

$W_{2,1}^{baru} = 0{,}0705 - 0{,}1(0{,}0015) = 0{,}0704$

$W_{3,1}^{baru} = 0{,}0072 - 0{,}1(0{,}0066) = 0{,}0065$

Perhitungan yang sama dilakukan untuk perhitungan seluruh elemen, diperoleh hasil pembaruan sebagai berikut:

$$\mathbf{W}^{baru} = \begin{bmatrix} 0{,}0312 & 0{,}0946 \\ 0{,}0704 & 0{,}0556 \\ 0{,}0065 & 0{,}0057 \\ -0{,}0026 & 0{,}0875 \\ 0{,}0576 & 0{,}0713 \end{bmatrix}$$

d. *Update* Bias *Hidden* (**b**)

Gradien bias *hidden* dihitung sebagai rata-rata dari $\boldsymbol{\gamma}_i$

$$\nabla_{\mathbf{b}} L_i = \frac{1}{5}\sum_{i=1}^{5} \boldsymbol{\gamma}_i$$

Diketahui sinyal kesalahan *hidden* pada lima data awal yaitu:

$$\boldsymbol{\gamma}_1 = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix}, \boldsymbol{\gamma}_2 = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0250 \\ -0{,}0373 \\ -0{,}0565 \end{bmatrix}, \boldsymbol{\gamma}_3 = \begin{bmatrix} -0{,}0008 \\ 0{,}0058 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix}, \boldsymbol{\gamma}_4 = \begin{bmatrix} -0{,}0008 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix},$$

$$\boldsymbol{\gamma}_5 = \begin{bmatrix} 0{,}0000 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0567 \end{bmatrix}$$

$\nabla_{\mathbf{b}} L_i = \frac{1}{5}(\boldsymbol{\gamma}_1 + \boldsymbol{\gamma}_2 + \boldsymbol{\gamma}_3 + \boldsymbol{\gamma}_4 + \boldsymbol{\gamma}_5)$

$\nabla_b L_1 = \frac{-0{,}0008+(-0{,}0008)+(-0{,}0008)+(-0{,}0008)+(-0{,}0008)}{5} = -0{,}0006$

$\nabla_b L_2 = \frac{0{,}0057+0{,}0057+0{,}0058+0{,}0057+0{,}0057}{5} = 0{,}0057$

$\nabla_b L_3 = \frac{0{,}0251+0{,}0250+0{,}0251+0{,}0251+0{,}0251}{5} = 0{,}0251$

$\nabla_b L_4 = \frac{(-0{,}0374)+(-0{,}0373)+(-0{,}0374)+(-0{,}074)+(-0{,}0374)}{5} = -0{,}0374$

$$\nabla_b L_5 = \frac{(-0{,}0566)+(-0{,}0565)+(-0{,}0566)+(-0{,}0566)+(-0{,}0567)}{5} = -0{,}0566$$

Sehingga, diperoleh:

$$\nabla_{\mathbf{b}} L_i = \begin{bmatrix} -0{,}0006 \\ 0{,}0057 \\ 0{,}0251 \\ -0{,}0374 \\ -0{,}0566 \end{bmatrix}$$

Kemudian dilakukan pembaruan dengan menggunakan dengan menggunakan $\mathbf{b}^{lama} = \begin{bmatrix} -0{,}0077 \\ 0{,}0967 \\ 0{,}0816 \\ 0{,}0134 \\ 0{,}0100 \end{bmatrix}$, maka:

$$\mathbf{b}^{baru} = \mathbf{b}^{lama} - \eta \nabla_{\mathbf{b}} L_i$$

Diberikan cuplikan perhitungan b^{baru}:

$$\mathbf{b}^{baru} = \mathbf{b}^{lama} - 0{,}1(\nabla_{\mathbf{b}} L_i)$$

$$b_{1,1}^{baru} = -0{,}0077 - 0{,}1(-0{,}0006) = -0{,}0076$$

$$b_{2,1}^{baru} = 0{,}0967 - 0{,}1(0{,}0057) = 0{,}0961$$

$$b_{3,1}^{baru} = 0{,}0816 - 0{,}1(0{,}0251) = 0{,}0791$$

$$b_{4,1}^{baru} = 0{,}0134 - 0{,}1(-0{,}0374) = 0{,}0171$$

$$b_{5,1}^{baru} = 0{,}0100 - 0{,}1(-0{,}0566) = 0{,}0157$$

Sehingga, diperoleh:

$$\mathbf{b}^{baru} = \begin{bmatrix} -0{,}0076 \\ 0{,}0961 \\ 0{,}0791 \\ 0{,}0171 \\ 0{,}0157 \end{bmatrix}$$

Vektor yang diperoleh merupakan hasil pembaruan parameter dalam satu iterasi. Setelah pembaruan ini, langkah selanjutnya adalah mengevaluasi

kembali model pada lima contoh data. Proses ini melibatkan perhitungan skor *output*, mengubah skor menjadi probabilitas, dan kemudian membandingkan selisihnya dengan label sebenarnya untuk memastikan seluruh nilai kesalahan mengecil. Selanjutnya, nilai *loss* rata-rata dihitung kembali dan dibandingkan dengan nilai *loss* sebelum pembaruan untuk memeriksa apakah terjadi perbaikan. Setelah itu, dilakukan pengecekan terhadap kriteria berhenti yang telah ditetapkan, misalnya perubahan parameter sudah sangat kecil atau penurunan *loss* sudah tidak signifikan lagi. Jika kriteria tersebut terpenuhi, iterasi pada contoh ini dianggap konvergen. Jika belum, maka seluruh proses perhitungan di atas diulang.

Pada perhitungan manual, proses hanya dapat dilakukan untuk satu iterasi sehingga belum mampu menunjukkan pembaruan parameter secara berulang hingga mencapai kondisi konvergen. Oleh karena itu, proses yang sama kemudian diterapkan pada eksperimen pada seluruh data di Subbab 4.3.2, dengan menggunakan program.

4.3.2 Implementasi pada Seluruh Data

Perhitungan manual pada lima data awal memberikan gambaran mengenai tahapan *feedforward*, perhitungan *loss*, *backpropagation*, dan *update* parameter. Selanjutnya, dilakukan implementasi pada seluruh *dataset* menggunakan program Kaggle. Melalui implementasi program, proses pembaruan parameter dapat dilakukan secara berulang pada setiap *epoch* sehingga model mampu mempelajari pola data secara lebih optimal. Implementasi FFNN dilakukan menggunakan data hasil *preprocessing* yang telah melalui tahapan *cleaning*, ekstraksi fitur menggunakan metode *k*-mer, *one-hot encoding*, pembagian data dan normalisasi

data. Pada penelitian ini, implementasi utama model menggunakan *random seed* 42. Penggunaan *random seed* 42 diterapkan pada hasil utama, pengujian parameter, evaluasi *confusion matrix*, *classification report* serta perbandingan metode.

Dilakukan pengujian pada beberapa parameter sebelum dilakukan proses pelatihan secara keseluruhan. Parameter yang diuji meliputi nilai k-mer dan banyak neuron pada *hidden layer* berdasarkan nilai akurasi, presisi, *recall*, F1-*score*, dan waktu pelatihan. Seluruh pengujian parameter dilakukan menggunakan konfigurasi *random seed* 42 agar perbandingan antarparameter berada pada kondisi yang sama. Dengan demikian, hasil pengujian parameter digunakan untuk menentukan konfigurasi model, sedangkan pengujian beberapa *random seed* berbeda digunakan sebagai evaluasi tambahan untuk mengetahui kestabilan model terhadap variasi parameter acak, bukan sebagai dasar utama pemilihan konfigurasi akhir model.

Pada nilai k-mer dilakukan pengujian menggunakan nilai k, yaitu $k = 3, k = 4,$ dan $k = 5$. Pemilihan nilai k merujuk pada penelitian Zhang dkk. (2011) yang menerapkan penggunaan variasi panjang *substring* k-mer dalam representasi fitur sekuens yang bertujuan untuk mengetahui pengaruh panjang substring terhadap kinerja klasifikasi dan jumlah fitur yang dihasilkan. Pengujian dilakukan dengan konfigurasi dan pembagian data yang sama menggunakan *random seed* 42. Lima kali *running* dilakukan untuk memperoleh rata-rata waktu pelatihan, sedangkan nilai metrik evaluasi cenderung tetap karena konfigurasi model, pembagian data, dan *random seed* yang digunakan sama. Hasil pengujian rata-rata dari setiap nilai k disajikan pada Tabel 4.12, sedangkan hasil pengujian lengkap

dari lima kali *running* disajikan pada Lampiran 8.

Tabel 4.12 Hasil Pengujian Variasi Nilai *K*-Mer

Nilai k	Jumlah Fitur	Akurasi	Presisi	*Recall*	F1-*score*	Waktu Pelatihan (detik)
3	64	0,9091	0,9276	0,9091	0,8949	8,3980
4	256	0,9983	0,9983	0,9983	0,9983	8,6040
5	1024	0,9983	0,9983	0,9983	0,9983	10,7329

Tabel 4.12 menunjukkan beberapa nilai k-mer yang berpengaruh yang menghasilkan jumlah fitur berbeda. Tabel tersebut menunjukkan bahwa semakin besar nilai k, maka jumlah fitur yang terbentuk semakin banyak karena kombinasi k-mer mengikuti rumus 4^k.

Pada $k = 3$, menghasilkan nilai lebih rendah dibandingkan $k = 4$ dan $k = 5$ dengan nilai akurasi dan *recall* sebesar $0{,}9091$, presisi sebesar $0{,}9276$, dan F1-*score* sebesar $0{,}8949$. Jumlah fitur yang dihasilkan sebanyak 64 menyebabkan informasi pola sekuens yang digunakan masih terbatas. Selain itu *substring* yang terbentuk masih terlalu pendek, sehingga informasi karakteristik antarvarian belum tertangkap dengan baik.

Pada nilai $k = 4$, terjadi peningkatan jumlah fitur menjadi 256 yang diikuti peningkatan kinerja yang signifikan yaitu nilai akurasi, presisi, *recall*, F1-*score* sebesar $0{,}9983$. Selain itu, waktu pelatihan selama $8{,}6040$ detik masih efisien meskipun jumlah fitur meningkat. Hasil ini menunjukkan bahwa nilai $k = 4$ mampu menghasilkan fitur yang cukup baik dalam membedakan karakteristik antar varian.

Pada $k = 5$ jumlah fitur meningkat secara signifikan menjadi 1024 fitur. Meskipun jumlah fitur meningkat dibandingkan $k = 4$, hasil tersebut tidak menghasilkan peningkatan performa klasifikasi. Selain itu, waktu pelatihan juga

menjadi lebih lama karena kompleksitas fitur yang semakin tinggi. Hal ini menunjukkan bahwa peningkatan nilai k tidak selalu menghasilkan performa yang lebih baik dan menyebabkan proses komputasi menjadi lebih kompleks.

Berdasarkan hasil pengujian tersebut, nilai $k = 4$ dipilih sebagai parameter terbaik karena mampu menghasilkan kinerja tinggi dengan waktu pelatihan yang lebih efisien. Pemilihan $k = 4$ juga sejalan dengan kajian teori yang menyatakan bahwa tidak terdapat nilai k yang optimal secara universal dan penggunaan k-mer yang lebih panjang dapat meningkatkan kompleksitas fitur serta kebutuhan komputasi (Zielezinski dkk., 2017).

Pengujian juga dilakukan terhadap banyak neuron pada *hidden layer*. Pemilihan banyak neuron ini merujuk pada penelitian Doukim dkk. (2010) yang menggunakan pendekatan *binary search mode* dengan pola *power of two*. Hasil pengujian rata-rata banyak neuron *hidden* disajikan pada Tabel 4.13, sedangkan hasil pengujian lengkap dari lima kali *running* disajikan pada Lampiran 9.

Tabel 4.13 Hasil Pengujian Variasi Banyak Neuron pada *Hidden Layer*

Jumlah m	**Akurasi**	**Presisi**	***Recall***	**F1-*score***	**Waktu Pelatihan (detik)**
32	0,9983	0,9983	0,9983	0,9983	7,8240
64	0,9983	0,9983	0,9983	0,9983	8,0000
128	0,9975	0,9975	0,9975	0,9975	8,5640
256	0,9975	0,9975	0,9975	0,9975	9,0080

Berdasarkan Tabel 4.13, hasil pengujian menunjukkan bahwa banyak neuron *hidden layer* memberikan pengaruh terhadap kinerja model dan waktu pelatihan yang dihasilkan.

Pada $m = 32$, model menghasilkan nilai akurasi, presisi, *recall*, dan F1-sebesar $0{,}9983$ dengan waktu pelatihan selama $7{,}8240$ detik. Hasil tersebut menunjukkan bahwa model mampu memberikan kapasitas pembelajaran yang

baik dalam mengenali pola k-mer pada data sekuens SARS-CoV-2. Selain itu, waktu pelatihan yang dihasilkan merupakan yang paling cepat dibandingkan variasi neuron lainnya. Pada $m = 64$, model juga menghasilkan nilai akurasi, presisi, *recall*, dan F1-*score* sebesar $0{,}9983$. Nilai tersebut sama dengan hasil pada $m = 32$, namun waktu pelatihan meningkat menjadi selama $8{,}0000$ detik. Hal ini menunjukkan bahwa penambahan jumlah neuron dari 32 menjadi 64 tidak memberikan peningkatan terhadap nilai kinerja klasifikasi, tetapi menyebabkan waktu pelatihan menjadi sedikit lebih lama..

Pada $m = 128$ dan $m = 256$, kinerja model mengalami sedikit penurunan dengan menghasilkan nilai akurasi, presisi, *recall*, dan F1-*score* yang sama, yaitu sebesar $0{,}9975$. Jika dibandingkan dengan $m = 32$ dan $m = 64$ yang memperoleh nilai $0{,}9983$, terjadi penurunan sebesar $0{,}0008$ pada masing-masing metrik evaluasi. Selain itu, waktu pelatihan meningkat seiring bertambahnya neuron, yaitu $8{,}5640$ detik pada $m = 128$ dan $9{,}0080$ detik pada $m = 256$. Hasil ini menunjukkan bahwa penambahan jumlah neuron pada hidden layer tidak selalu meningkatkan kinerja model. Sebaliknya, jumlah neuron yang lebih besar dapat meningkatkan kompleksitas jaringan dan waktu komputasi tanpa memberikan peningkatan terhadap hasil klasifikasi. Hal ini menunjukkan bahwa penambahan banyak neuron dari 64 menjadi 256 tidak meningkatkan kinerja klasifikasi dan menambah waktu komputasi. Menurut Géron (2019), penambahan banyak neuron pada *hidden layer* dapat meningkatkan kompleksitas model dan kebutuhan komputasi selama proses pelatihan.

Berdasarkan hasil pengujian di atas, dipilih $m = 32$ sebagai parameter pada penelitian ini. Pemilihan tersebut didasarkan pada kemampuan model dalam

menghasilkan nilai kinerja klasifikasi paling optimal dengan waktu pelatihan yang lebih efisien. Selain itu, penggunaan banyak neuron yang lebih sedikit juga membuat model menjadi lebih sederhana.

Hasil pengujian variasi nilai *k-mer* dan banyak neuron *hidden layer* digunakan sebagai dasar dalam penyusunan arsitektur model FFNN. Arsitektur model yang digunakan dalam penelitian ini disajikan pada Tabel 4.14.

Tabel 4.14 Arsitektur Model *Feedforward Neural Network*

Arsitektur Model	**Keterangan**
Banyak Neuron *Input Layer* (d)	256
Banyak Neuron *Hidden Layer* (m)	32
Aktivasi *Hidden Layer*	*Rectified Linear Unit* (ReLU)
Banyak Neuron *Output Layer* (p)	5
Aktivasi *Output Layer*	*Softmax*

Berdasarkan Tabel 4.14, arsitektur FFNN terdiri dari *input layer*, *hidden layer*, dan *output layer* dengan mempertimbangkan kompleksitas data dan efisiensi komputasi. Pada *input layer* digunakan sebanyak 256 neuron dari jumlah fitur hasil ekstraksi k-mer terbaik, yaitu $k = 4$ yang merepresentasikan seluruh kombinasi kemungkinan 4-mer yang digunakan sebagai data masukan.

Pada penelitian ini menggunakan satu *hidden layer* dengan banyak neuron sebanyak 32 neuron. Penggunaan satu *hidden layer* dipilih karena mampu mempelajari pola nonlinier pada data dengan baik serta menghasilkan kinerja klasifikasi yang optimal. Selain itu, penggunaan satu *hidden layer* model menjadi lebih sederhana sehingga proses komputasi dan waktu pelatihan menjadi lebih efisien. Pada *hidden layer* digunakan fungsi aktivasi ReLU karena mampu membantu model mempelajari hubungan nonlinier antar fitur dengan proses komputasi yang lebih stabil dan membantu mengurangi permasalahan *vanishing gradient* dengan mengubah nilai *input* negatif menjadi 0 pada proses pelatihan.

Pada *output layer*, menggunakan 5 neuron yang merepresentasikan jumlah kelas varian pada penelitian ini dengan menggunakan fungsi *softmax*, karena penelitian ini termasuk ke dalam klasifikasi multikelas. *Softmax* berfungsi untuk mengubah *output* menjadi nilai probabilitas pada setiap kelas sehingga model dapat menentukan kelas dengan probabilitas tertinggi sebagai hasil prediksi.

Pelatihan model secara keseluruhan dilakukan berdasarkan arsitektur FFNN yang telah ditentukan. Proses pelatihan bertujuan untuk memperoleh parameter berupa bobot dan bias yang baik, sehingga model mampu menghasilkan prediksi yang akurat terhadap data yang digunakan. Parameter yang digunakan pada penelitian ini disajikan pada Tabel 4.15.

Tabel 4.15 Parameter Pelatihan Model

Parameter	**Nilai**
Jumlah *Epoch*	30
Batch Size	64
Learning Rate	0,01
Optimizer	*Stochastic Gradient Descent* (SGD)
Fungsi *Loss*	*Categorical Crossentropy*
Early Stopping	5
Verbose	0

Pada Tabel 4.15, proses pelatihan model menggunakan algoritma optimasi SGD dipilih karena mampu memperbarui parameter model secara bertahap berdasarkan nilai gradien yang diperoleh dari proses *backpropagation*. Nilai *learning rate* sebesar 0,01 berfungsi untuk menjaga kestabilan proses pembelajaran pada pembaruan parameter pada setiap iterasi agar model dapat mencapai kondisi konvergen tanpa menyebabkan perubahan parameter yang terlalu besar.

Pelatihan model dilakukan selama maksimal 30 *epoch* bertujuan untuk memberikan kesempatan model mempelajari pola data secara baik tanpa menyebabkan *overfitting*, sedangkan *batch size* sebesar 64 dipilih karena mampu

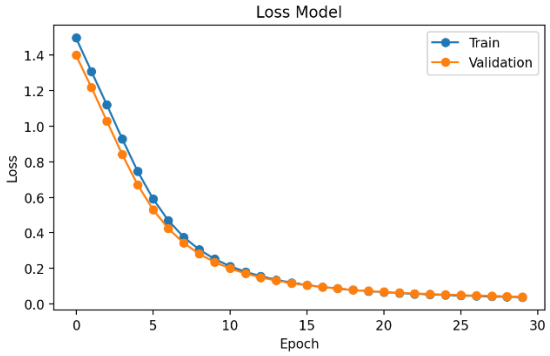
menjaga keseimbangan antara kecepatan komputasi dan kestabilan pembaruan gradien selama proses pelatihan. Selain itu, digunakan fungsi *loss categorical crossentropy* karena sesuai dengan penelitian ini yaitu klasifikasi multikelas dengan *output* berbentuk probabilitas yang dihasilkan dari fungsi *softmax*.

Teknik *early stopping* diterapkan untuk membantu mengurangi risiko *overfitting* selama proses pelatihan. Pada penelitian ni, *early stopping* menggunakan nilai *patience* sebesar 5, yang berfungsi untuk menghentikan proses pelatihan secara otomatis apabila *validation loss* tidak mengalami perbaikan dalam lima *epoch* berturut-turut sehingga dapat mencegah *overfitting*. Kemudian, parameter *verbose* bernilai 0 berfungsi untuk menonaktifkan tampilan proses pelatihan pada setiap *epoch* sehingga *output* pelatihan tidak ditampilkan secara detail selama model dijalankan. Secara keseluruhan, kombinasi parameter pelatihan seperti *learning rate*, *batch size*, jumlah *epoch*, optimizer, fungsi *loss*, dan *early stopping* dipilih untuk memastikan bahwa model belajar secara efisien, dengan keseimbangan antara kecepatan konvergensi dan stabilitas, sehingga menghasilkan prediksi yang akurat pada data *test*.

Berdasarkan hasil pengujian parameter, konfigurasi akhir model FFNN yang digunakan dalam penelitian ini adalah nilai $k = 4$, $m = 32$, fungsi aktivasi ReLU pada *hidden layer*, fungsi aktivasi *Softmax* pada *output layer*, *optimizer* SGD, *learning rate* sebesar $0{,}01$, *batch size* 64, jumlah *epoch* maksimum 30, dan *early stopping* dengan *patience* 5. Konfigurasi ini dijalankan menggunakan *random seed* 42 yang digunakan pada hasil utama, *confusion matrix*, *classification report*, serta perbandingan dengan metode pembanding.

Seluruh parameter berperan dalam menghasilkan model FFNN yang mampu

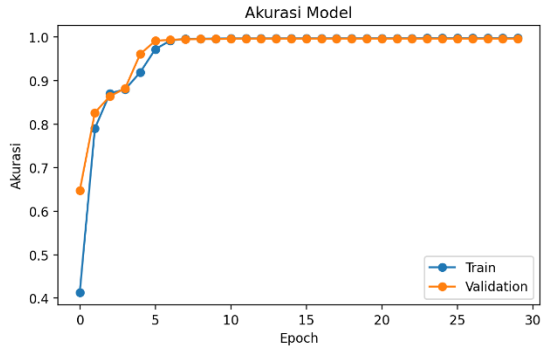
melakukan klasifikasi yang baik terhadap data yang belum pernah dilihat sebelumnya. Untuk memberikan gambaran mengenai hasil pelatihan model, kurva *training loss* dan *validation loss* pada setiap *epoch* ditunjukkan pada Gambar 4.4.



Gambar 4.4 Kurva *Loss* pada Setiap *Epoch*

Pada Gambar 4.4, kurva menunjukkan penurunan secara bertahap selama proses pelatihan. Pada *epoch* awal, nilai *loss* masih relatif tinggi, yaitu sekitar 1,5 pada data *training* dan 1,4 pada data *validation* yang menunjukkan model belum mampu mempelajari pola data secara optimal. Namun, seiring bertambahnya *epoch* nilai *loss* pada data *training* dan *validation* terus mengalami penurunan hingga mencapai sekitar $0{,}04 - 0{,}05$ pada akhir pelatihan. Penurunan nilai *loss* ini menunjukkan bahwa model berhasil mempelajari pola data secara bertahap melalui proses pembaruan bobot dan bias pada setiap iterasi.

Kurva akurasi pelatihan dan akurasi validasi juga ditunjukkan pada Gambar 4.5 untuk melihat perkembangan kinerja model selama proses pelatihan.



Gambar 4.5 Kurva Akurasi pada Setiap *Epoch*

Pada Gambar 4.5, kurva menunjukkan peningkatan nilai akurasi pada data *training* dan *validation* secara bertahap selama proses pelatihan berlangsung. Pada *epoch* awal, nilai akurasi masih relatif rendah, yaitu akurasi *training* pada kisaran 0,41, sedangkan akurasi *validation* berada pada kisaran 0,65. Kondisi tersebut menunjukkan bahwa pada awal pelatihan kemampuan model dalam mengenali pola data masih terbatas. Namun, seiring bertambahnya *epoch*, nilai akurasi pada data *training* maupun *validation* terus mengalami peningkatan pada epoch ke-5 hingga ke-6 mendekati 0,99. Selanjutnya, akurasi cenderung stabil hingga akhir pelatihan dan mendekati nilai 1.

Berdasarkan hasil pada Gambar 4.4 dan Gambar 4.5, kurva *training* dan *validation* terlihat memiliki pola yang saling berdekatan searah dengan selisih yang relatif kecil selama proses pelatihan. Kondisi tersebut menunjukkan bahwa model memiliki kemampuan generalisasi yang baik terhadap data *validation* dan tidak mengalami *overfitting* yang signifikan. *Overfitting* ditandai dengan meningkatnya kurva *validation loss* atau menurunnya *validation* akurasi dibandingkan *training* pada *epoch* tertentu (Géron, 2019). Namun, pada grafik tersebut kurva *loss* tetap mengalami penurunan secara konsisten, sedangkan kurva akurasi terus mengalami peningkatan hingga akhir pelatihan.

Kurva juga menunjukkan bahwa proses pelatihan dengan menggunakan *optimizer* SGD dengan *learning rate* sebesar 0,01 mampu menjaga kestabilan proses pembelajaran model. Selain itu, perubahan nilai *loss* dan akurasi yang semakin kecil pada *epoch* akhir menunjukkan bahwa model mulai mendekati kondisi konvergen, yaitu kondisi ketika pembaruan parameter tidak lagi memberikan perubahan kinerja yang signifikan. Peningkatan akurasi pada data

training dan *validation* menunjukkan bahwa parameter pelatihan yang dipilih, efektif dalam menyesuaikan bobot model secara stabil.

Pada Tabel 4.16 disajikan proses pelatihan model pada setiap *epoch* untuk memberikan gambaran yang lebih rinci.

Tabel 4.16 Nilai *Loss* dan Akurasi Setiap *Epoch*

Epoch	***Train Loss***	***Val Loss***	***Train* Akurasi**	***Val* Akurasi**
1	1,4981	1,4004	0,4133	0,6474
2	1,3095	1,2195	0,7903	0,8266
3	1,1220	1,0303	0,8711	0,8645
4	0,9296	0,8416	0,8801	0,8822
5	0,7475	0,6711	0,9192	0,9613
6	0,5918	0,5313	0,9725	0,9916
7	0,4689	0,4234	0,9925	0,9942
⋮	⋮	⋮	⋮	⋮
24	0,0524	0,0546	0,9981	0,9967
25	0,0491	0,0514	0,9981	0,9967
26	0,0461	0,0486	0,9981	0,9967
27	0,0435	0,0461	0,9981	0,9967
28	0,0411	0,0438	0,9981	0,9967
29	0,0390	0,0418	0,9981	0,9967
30	0,0370	0,0399	0,9981	0,9967

Berdasarkan Tabel 4.16, terlihat bahwa pada *epoch* awal nilai *training loss* dan *validation loss* masih tinggi sebesar 1,4981 dan 1,4004. Hal ini sesuai dengan nilai *training* akurasi dan *validation* akurasi yang masih rendah, yaitu sebesar 0,4133 dan 0,6474 yang menunjukkan bahwa model belum mampu mengenali pola data dengan baik. Dengan terus bertambahnya *epoch*, nilai *training loss* dan *validation loss* terus mengalami penurunan secara bertahap.

Pada *epoch* ke-5, *training loss* menurun menjadi 0,7475 dan *validation loss* menjadi 0,6711. Pada tahap ini, nilai *training* akurasi dan *validation* akurasi juga mengalami peningkatan menjadi 0,9192 dan 0,9613. Selanjutnya, pada *epoch* ke-6 dan ke-7, kinerja model meningkat cukup signifikan dengan *training loss* sebesar 0,5918 dan *validation loss* sebesar 0,5313, sedangkan *training* akurasi

dan *validation* akurasi masing-masing mencapai 0,9725 dan 0,9916. Pada *epoch* ke-7, *training loss* kembali menurun menjadi 0,4689 dan *validation loss* menjadi 0,4234, dengan *training* akurasi sebesar 0,9925 dan *validation* akurasi sebesar 0,9942. Pada *epoch* akhir, yaitu *epoch* ke-30, diperoleh nilai *training loss* sebesar 0,0370 dan *validation loss* sebesar 0,0399. Sementara itu, nilai *training* akurasi mencapai 0,9981 dan *validation* akurasi sebesar 0,9967.

Hasil tersebut menunjukkan model telah mencapai kinerja yang sangat baik dalam melakukan klasifikasi varian SARS-CoV-2. Selain itu, selisih antara nilai *training loss*, *validation loss*, *training* akurasi dan *validation* akurasi terlihat relatif kecil sepanjang proses pelatihan. Kondisi ini menunjukkan bahwa model mampu mempertahankan kinerja yang stabil pada data *training* dan *validation* sehingga tidak mengalami *overfitting* yang signifikan. Secara keseluruhan, hasil implementasi program menunjukkan model berhasil mempelajari pola data dengan baik melalui proses pembaruan bobot dan bias pada setiap *epoch*.

4.4 Evaluasi dan Validasi Hasil

Evaluasi model digunakan untuk mengukur kinerja FFNN dalam mengklasifikasikan varian SARS-CoV-2. Tahap ini menggunakan data uji yang tidak digunakan dalam proses pelatihan. Pengukuran kinerja dilakukan menggunakan *confusion matrix* dan *classification report* yang menghasilkan metrik evaluasi berupa akurasi, presisi, *recall*, dan F1-*score* menggunakan konfigurasi akhir model dengan *random seed* 42.

Evaluasi tambahan dilakukan menggunakan beberapa nilai *random seed* dengan konfigurasi akhir yang digunakan tetap sama. Tujuan pengujian ini adalah untuk memastikan kestabilan, konsistensi kinerja model, dan melihat pengaruh

variasi parameter acak terhadap kinerja model. Hasil pengujian ini tidak digunakan sebagai pengganti hasil utama, melainkan sebagai evaluasi tambahan untuk mengetahui apakah kinerja model tetap konsisten ketika nilai *random seed* berbeda. Menurut Colas dkk. (2018) penggunaan *random seed* yang berbeda pada algoritma dan *hyperparameter* yang sama dapat menghasilkan variasi kinerja model akibat pengaruh proses inisialisasi acak selama pelatihan.

Nilai *random seed* yang digunakan, yaitu $42, 43, 44, 45$, dan 46. Pada setiap nilai *random seed*, proses pelatihan dan evaluasi dijalankan sebanyak lima kali, kemudian hasil dari lima *running* tersebut dirata-ratakan untuk memperoleh nilai evaluasi pada masing-masing *random seed*. Selanjutnya, nilai rata-rata dari kelima *random seed* tersebut dihitung kembali untuk memperoleh rata-rata keseluruhan dan simpangan baku. Pengujian digunakan untuk melihat sejauh mana perubahan nilai *random seed* memengaruhi hasil evaluasi model. Hasil pengujian rata-rata menggunakan *random seed* disajikan pada Tabel 4.17, sedangkan hasil lengkap dari lima kali percobaan disajikan pada Lampiran 10.

Tabel 4.17 Hasil Evaluasi Kinerja Model Menggunakan Lima *Random Seed*

Random seed	**Akurasi**	**Presisi**	***Recall***	**F1-*score***	**Waktu Pelatihan (detik)**
42	0,9983	0,9983	0,9983	0,9983	9,2560
43	0,9975	0,9975	0,9975	0,9975	9,0680
44	0,9891	0,9893	0,9891	0,9890	9,2020
45	0,9907	0,9909	0,9907	0,9907	9,1240
46	0,9975	0,9975	0,9975	0,9975	9,2260
Rata-rata	**0,9946**	**0,9947**	**0,9946**	**0,9946**	**9,1752**
Simpangan Baku	**0,0044**	**0,0042**	**0,0044**	**0,0044**	**0,0774**

Pada Tabel 4.17, dilihat bahwa penggunaan nilai *random seed* yang berbeda menghasilkan variasi kinerja model pada setiap percobaan. Nilai akurasi berada pada rentang 0,9891 hingga 0,9983. Nilai presisi berada pada rentang 0,9893

hingga 0,9983, nilai *recall* berada pada rentang 0,9891 hingga 0,9983, dan nilai F1-*score* berada pada rentang 0,9890 hingga 0,9983. Hasil terbaik diperoleh pada *random seed* 42 dengan seluruh nilai metrik evaluasi sebesar 0,9983. Hasil yang hampir sama juga diperoleh pada *random seed* 43 dan 46 dengan seluruh nilai metrik evaluasi sebesar 0,9975.

Pada *random seed* 44 dan 45 menghasilkan kinerja yang lebih rendah dibandingkan *random seed* lainnya. Pada *random seed* 44 diperoleh nilai akurasi dan *recall* sebesar 0,9891 dengan nilai presisi sebesar 0,9893 serta F1-*score* sebesar 0,9890. Sementara itu, pada *random seed* 45 diperoleh nilai akurasi, *recall*, F1-*score* sebesar 0,9907 dengan nilai presisi sebesar 0,9909. Meskipun terjadi penurunan, nilai metrik evaluasi pada kedua *random seed* tersebut masih berada pada kategori sangat tinggi. Hal ini menunjukkan bahwa perbedaan *random seed* memberikan sedikit pengaruh terhadap hasil pelatihan.

Berdasarkan seluruh pengujian, diperoleh rata-rata akurasi sebesar 0,9946 dengan simpangan baku sebesar 0,0044. Rata-rata presisi, *recall*, dan F1-*score* masing-masing sebesar 0,9947, 0,9946, dan 0,9946 dengan simpangan baku berturut-turut sebesar 0,0042, 0,0044, dan 0,0044. Sementara itu, rata-rata waktu pelatihan selama 9,1752 detik dengan simpangan baku selama 0,1880 detik. Nilai simpangan baku yang relatif kecil pada waktu pelatihan menunjukkan bahwa waktu komputasi model cenderung stabil pada seluruh pengujian. Secara keseluruhan, hasil pengujian menunjukkan bahwa model memiliki kinerja yang baik dan stabil terhadap variasi *random seed*. Nilai akurasi terbaik sebesar 0,9983 diperoleh pada *random seed* 42, sedangkan rata-rata akurasi dari lima *random seed* berbeda adalah 0,9946.

Evaluasi menggunakan *confusion matrix* dan *classification report* dilakukan berdasarkan hasil utama model pada konfigurasi akhir dengan *random seed* 42. *Confusion matrix* digunakan pada evaluasi model untuk menggambarkan kinerja model dalam mengklasifikasikan data ke dalam kelas yang benar, sehingga diketahui banyak prediksi yang benar maupun salah untuk setiap kelas. Berdasarkan hasil pengujian yang merujuk pada Tabel 2.2, diperoleh nilai *True Positive* (TP), *False Positive* (FP), *False Negative* (FN), dan *True Negative* (TN) yang disajikan dalam Tabel 4.18. Nilai-nilai tersebut digunakan untuk menghitung metrik evaluasi berupa akurasi, presisi, *recall*, dan F1-*score* pada setiap varian.

Tabel 4.18 Hasil *Confusion Matrix*

Varian	TP	FP	FN	TN
Alpha	275	0	0	913
Beta	142	0	0	1046
Delta	253	0	2	933
Gamma	246	1	0	941
Omicron	270	1	0	917

Pada Tabel 4.18, sebagian besar data berhasil diklasifikasikan dengan benar, yang terlihat dari nilai TP yang tinggi, nilai FP dan FN yang relatif rendah pada setiap kelas.

Pada varian Alpha dan Beta model menunjukkan kinerja terbaik dengan hasil klasifikasi sempurna karena tidak terdapat kesalahan prediksi FP dan FN. Pada varian Delta yang tidak berhasil dikenali sebagai Delta dan diprediksi ke kelas lain. Pada varian Gamma seluruh data aktual berhasil dikenali dengan benar yang ditunjukkan oleh nilai FN sebesar 0, namun masih terdapat satu data dari kelas lain yang salah diprediksi sebagai Gamma sehingga menghasilkan FP sebesar 1. Sementara itu, pada varian Omicron terdapat satu data dari kelas lain yang salah diprediksi sebagai Omicron.

Perhitungan metrik evaluasi dilakukan berdasarkan nilai TP, FP, FN, dan TN yang diperoleh dari Tabel 4.18 menggunakan Persamaan (2.29) hingga Persamaan (2.32). Perhitungan presisi, *recall*, dan *F1-score* dilakukan pada masing-masing kelas, sedangkan akurasi dihitung sebagai nilai keseluruhan model berdasarkan jumlah prediksi benar terhadap seluruh data uji. Berdasarkan Tabel 4.18, data uji yang digunakan pada konfigurasi akhir dengan *random seed* 42 sebanyak 1188 data. Banyak prediksi benar diperoleh dari penjumlahan seluruh nilai TP pada masing-masing kelas, yaitu:

$$275 + 142 + 254 + 246 + 269 = 1186$$

Nilai akurasi keseluruhan dihitung menggunakan persamaan berikut:

$$\text{Akurasi} = \frac{\text{banyak prediksi benar}}{\text{seluruh data}}$$

$$\text{Akurasi} = \frac{1186}{1188} = 0{,}9983$$

Nilai akurasi sebesar 0,9983 merupakan hasil evaluasi pada konfigurasi akhir dengan *random seed* 42, bukan rata-rata dari seluruh pengujian menggunakan lima *random seed* berbeda. Nilai tersebut diperoleh dari 1186 prediksi benar dari total 1188 data uji, sehingga hanya terdapat dua kesalahan klasifikasi pada seluruh data pengujian. Hasil ini menunjukkan bahwa sebagian besar data uji berhasil diklasifikasikan ke dalam kelas yang sesuai.

Perhitungan pada varian Alpha dilakukan sebagai berikut:

$$\text{Presisi} = \frac{TP}{TP + FP} = \frac{275}{275 + 0} = 1$$

$$Recall = \frac{TP}{TP + FN} = \frac{275}{275 + 0} = 1$$

$$F1 - Score = 2 \times \frac{\text{Presisi} \times Recall}{\text{Presisi} + Recall}$$

$$F1 - Score = 2 \times \frac{1 \times 1}{1 + 1}$$

$$F1 - Score = \frac{2}{2} = 1$$

Perhitungan yang sama juga dilakukan pada varian Beta, Delta, Gamma, dan Omicron. Hasil *classification report* secara lengkap disajikan pada Tabel 4.19.

Tabel 4.19 Hasil *Classification Report* untuk Setiap Varian

Varian	**Presisi**	***Recall***	**F1-*score***	***Support***
Alpha	1	1	1	275
Beta	1	1	1	142
Delta	1	0,9922	0,9961	255
Gamma	0,9960	1	0,9980	246
Omicron	0,9963	1	0,9982	270

Berdasarkan Tabel 4.19, model menunjukkan kinerja yang baik pada seluruh varian SARS-CoV-2. Kolom *support* menunjukkan banyak data uji aktual pada setiap varian, yaitu Alpha sebanyak 275 data, Beta sebanyak 142 data, Delta sebanyak 255 data, Gamma sebanyak 246 data, dan Omicron sebanyak 270 data. Varian Alpha dan Beta memperoleh nilai presisi, *recall*, dan F1-*score* sebesar 1 yang menunjukkan bahwa kedua varian berhasil dikenali dengan benar tanpa adanya kesalahan prediksi. Pada varian Delta diperoleh nilai presisi sebesar 1, *recall* sebesar 0,9922, dan F1-*score* sebesar 0,9961. Nilai tersebut menunjukkan bahwa sebagian besar data Delta berhasil diklasifikasikan dengan benar, meskipun masih terdapat sedikit kesalahan prediksi.

Pada varian Gamma diperoleh nilai presisi sebesar 0,9960, *recall* sebesar 1, dan F1-*score* sebesar 0,9980. Nilai *recall* yang sempurna menunjukkan bahwa seluruh data Gamma berhasil dikenali dengan baik oleh model, namun nilai presisi yang sedikit lebih rendah menunjukkan bahwa masih terdapat data dari kelas lain yang salah diprediksi sebagai Gamma. Sementara itu, pada varian Omicron

diperoleh nilai presisi sebesar 0,9963, *recall* sebesar 1, dan F1-*score* sebesar 0,9982. Nilai tersebut menunjukkan bahwa seluruh data aktual Omicron berhasil dikenali dengan benar oleh model, namun nilai presisi sebesar 0,9963 menunjukkan bahwa masih terdapat sedikit kesalahan, yaitu adanya data dari kelas lain yang salah diprediksi sebagai Omicron.

Nilai evaluasi keseluruhan untuk presisi, *recall*, dan *F1-score* dihitung menggunakan *weighted average* berdasarkan Persamaan (2.33), yaitu:

$$Weighted\ Average = \frac{\sum_{i=1}^{5}(M_i \times Support_i)}{\sum_{i=1}^{5} Support_i}$$

$$\textit{Presisi} = \frac{(1 \times 275) + (1 \times 142) + (1 \times 255) + (0{,}9960 \times 246) + (0{,}9963 \times 270)}{1188}$$

$$= \frac{275 + 142 + 255 + 245{,}0160 + 269{,}0010}{1188} = 0{,}9983$$

$$\textit{Recall} = \frac{(1 \times 275) + (1 \times 142) + (0{,}9922 \times 255) + (1 \times 246) + (1 \times 270)}{1188}$$

$$= \frac{275 + 142 + 253{,}0110 + 246 + 1}{1188} = 0{,}9983$$

$$\text{F1-}\textit{Score} = \frac{(1 \times 275) + (1 \times 142) + (0{,}9961 \times 255) + (0{,}9980 \times 246) + (0{,}9982 \times 270)}{1188}$$

$$= \frac{275 + 142 + 254{,}0055 + 245{,}5080 + 269{,}514}{1188} = 0{,}9983$$

Berdasarkan perhitungan tersebut, diperoleh nilai *weighted average* presisi, *recall*, dan *F1-score* masing-masing sebesar 0,9983. Nilai tersebut menunjukkan bahwa model mampu mempertahankan kinerja klasifikasi yang tinggi secara keseluruhan dengan mempertimbangkan jumlah data aktual (*support*) pada setiap varian. Kesamaan nilai *weighted average* presisi, *recall*, dan *F1-score* menunjukkan bahwa model memiliki keseimbangan yang baik antara ketepatan prediksi, kemampuan mengenali data pada setiap kelas, dan konsistensi kinerja klasifikasi pada seluruh varian SARS-CoV-2.

Nilai *weighted average* yang mendekati 1 menunjukkan bahwa kesalahan klasifikasi yang terjadi relatif sangat kecil dibandingkan jumlah data uji yang digunakan. Hasil ini sejalan dengan nilai akurasi sebesar 0,9983 yang diperoleh dari 1186 prediksi benar dari total 1188 data uji. Nilai *weighted average* presisi, *recall*, dan *F1-score* digunakan sebagai nilai evaluasi keseluruhan model karena mempertimbangkan jumlah data aktual (*support*) pada setiap varian. Berbeda dengan nilai presisi, *recall*, dan *F1-score* per varian yang hanya menggambarkan kinerja model pada kelas tertentu, nilai *weighted average* memberikan gambaran kinerja model secara keseluruhan terhadap seluruh data uji. Dengan demikian, model FFNN mampu membedakan karakteristik sekuens genom pada setiap varian SARS-CoV-2 dengan tingkat kesalahan yang rendah.

4.5 Perbandingan dengan Metode Pembanding

Pada penelitian ini, kinerja FFNN dibandingkan dengan beberapa metode lainnya, yaitu *K-Nearest Neighbors* (KNN), *Decision Tree*, dan S*upport Vector Machin*e (SVM). Perbandingan dilakukan untuk mengetahui kemampuan setiap metode dalam mengklasifikasikan varian SARS-CoV-2 berdasarkan nilai akurasi, presisi, *recall*, F1-*score*, waktu pelatihan, dan waktu prediksi. Nilai presisi, *recall*, dan *F1-score* yang digunakan merupakan nilai *weighted average* sehingga dapat menggambarkan kinerja model secara keseluruhan dengan mempertimbangkan jumlah data pada setiap varian.

Seluruh metode menggunakan data latih dan data uji yang sama dari konfigurasi akhir dengan *random seed* 42. Lima kali pengujian dilakukan untuk memperoleh rata-rata waktu komputasi, sedangkan nilai metrik evaluasi tetap sama karena data, konfigurasi, dan *random seed* yang digunakan sama. Nilai pada Tabel

4.20 merupakan rata-rata dari lima kali pengujian proses pada setiap metode dengan konfigurasi dan *random seed* yang sama, sedangkan hasil lengkap masing-masing pengujian disajikan pada Lampiran 7.

Tabel 4.20 Hasil Evaluasi Kinerja Model dengan Metode Pembanding

Model	**Akurasi**	**Presisi**	***Recall***	**F1-*score***	**Waktu *Training* (detik)**	**Waktu Prediksi (detik)**
FFNN	0,9983	0,9983	0,9983	0,9983	8,7880	0,2034
KNN	0,9958	0,9958	0,9958	0,9958	0,0026	0,1870
Decision Tree	0,9874	0,9876	0,9874	0,9874	0,1276	0,0023
SVM	0,9975	0,9975	0,9975	0,9975	1,5860	0,1152

Tabel 4.20 menunjukkan bahwa model FFNN memperoleh nilai metrik evaluasi sebesar 0,9983 dengan waktu pelatihan selama 8,7880 detik dan waktu prediksi selama 0,2034 detik. Sementara itu, metode KNN memperoleh nilai metrik evaluasi sebesar 0,9958 dengan waktu pelatihan paling cepat, yaitu 0,0026 detik dan waktu prediksi selama 0,1870 detik. Metode *Decision Tree* menghasilkan nilai akurasi, *recall*, F1-*score* sebesar 0,9874, nilai presisi sebagai 0,9876 dengan waktu pelatihan selama 0,1276 detik serta waktu prediksi selama 0,0023 detik, sedangkan metode SVM memperoleh nilai metrik evaluasi sebesar 0,9975 dengan waktu pelatihan selama 1,5860 detik dan waktu prediksi selama 0,1152 detik.

Ditinjau dari metrik evaluasi, FFNN memperoleh nilai tertinggi dibandingkan KNN, *Decision Tree*, dan SVM. Selisih akurasi FFNN dengan KNN adalah sebesar 0,0025, dengan *Decision Tree* sebesar 0,0109 dan dengan SVM sebesar 0,0008. Nilai akurasi FFNN sebesar 0,9983 setara dengan 99,83%, sehingga tingkat kesalahan klasifikasi sekitar 0,17%. Berdasarkan kategori akurasi klasifikasi menurut Gorunescu (2011), nilai akurasi pada rentang $90\% - 100\%$ termasuk dalam kategori *excellent classification*. Dengan demikian, FFNN

menunjukkan kinerja klasifikasi yang sangat tinggi berdasarkan nilai akurasi, presisi, *recall*, dan F1-*score*. Semakin tinggi nilai akurasi yang diperoleh, maka semakin kecil tingkat kesalahan prediksi model terhadap data (Géron, 2019). Namun, penilaian ini hanya ditinjau dari metrik evaluasi klasifikasi, bukan dari efisiensi waktu komputasi.

Perbandingan waktu juga perlu diperhatikan. Waktu pelatihan FFNN merupakan yang paling lama dibandingkan metode pembanding lainnya. FFNN membutuhkan waktu pelatihan selama 8,7880 detik, sedangkan KNN hanya 0,0026 detik, *Decision Tree* 0,1276 detik, dan SVM 1,5860 detik. Waktu pelatihan FFNN lebih lama 8,7854 detik dibandingkan KNN, 8,6604 detik dibandingkan *Decision Tree*, dan 7,2020 detik dibandingkan SVM. Hal ini terjadi karena FFNN melakukan proses pembelajaran melalui pembaruan bobot dan bias secara berulang pada setiap *epoch* menggunakan proses *feedforward* dan *backpropagation*. Proses tersebut menyebabkan FFNN membutuhkan komputasi yang lebih besar pada tahap pelatihan. Menurut Haykin (2009), *neural network* bekerja melalui proses penyesuaian bobot untuk mempelajari hubungan antarfitur, sedangkan Géron (2019) menjelaskan bahwa model *neural network* dapat mempelajari pola yang kompleks, tetapi membutuhkan komputasi yang lebih besar selama proses pelatihan.

Pada metode KNN, waktu pelatihan menjadi cepat karena termasuk metode *lazy learning* yang tidak membangun model secara kompleks pada tahap *training*, melainkan hanya menyimpan data latih dan melakukan perhitungan jarak pada tahap prediksi antara data *testing* dan data *training* untuk menentukan kelas berdasarkan tetangga terdekat. Karakteristik tersebut menyebabkan KNN memiliki

waktu pelatihan yang sangat cepat terutama pada *dataset* berukuran kecil. Hal ini terlihat dari waktu pelatihan KNN yang hanya selama 0,0026 detik. Akan tetapi, waktu prediksi KNN selama 0,1870 detik jauh lebih besar dibandingkan waktu pelatihannya karena proses perhitungan jarak dilakukan pada saat prediksi. Konsep KNN sebagai metode *lazy learning* atau *instance-based learning* juga dijelaskan oleh Géron (2019), bahwa proses generalisasi pada metode tersebut dilakukan saat prediksi, bukan pada tahap pelatihan.

Metode *Decision Tree* memiliki waktu pelatihan dan waktu prediksi yang relatif cepat, yaitu waktu pelatihan selama 0,1276 detik dan waktu prediksi selama 0,0023 detik. Berdasarkan Tabel 4.20, *Decision Tree* merupakan metode dengan waktu prediksi paling cepat dibandingkan FFNN, KNN, dan SVM. Hal ini disebabkan proses klasifikasi dilakukan melalui pembentukan struktur pohon berdasarkan aturan pemisahan data pada setiap *node*. Setelah struktur pohon terbentuk, proses prediksi dapat dilakukan dengan cepat karena data hanya mengikuti jalur keputusan dari *root node* hingga *leaf node*. Waktu pelatihan dan prediksi *Decision Tree* yang cepat juga dipengaruhi oleh ukuran *dataset* yang relatif kecil, sehingga proses pembentukan struktur pohon dan penelusuran jalur keputusan pada tahap prediksi dapat dilakukan secara efisien (Géron, 2019).

Metode SVM memperoleh nilai metrik evaluasi sebesar 0,9975 dengan waktu pelatihan 1,5860 detik dan waktu prediksi 0,1152 detik. Menurut Géron (2019), SVM bekerja dengan mencari *hyperplane* optimal yang memisahkan kelas data dan digunakan pada *dataset* berukuran kecil hingga menengah. Pada penelitian ini, waktu pelatihan SVM lebih cepat dibandingkan FFNN karena ukuran *dataset* yang digunakan relatif kecil, sehingga proses optimasi untuk menentukan

hyperplane masih dapat dilakukan secara efisien.

Secara keseluruhan, hasil perbandingan menunjukkan bahwa setiap metode memiliki karakteristik yang berbeda. FFNN memperoleh nilai akurasi, presisi, *recall*, dan F1-*score* tertinggi dibandingkan KNN, *Decision Tree*, dan SVM, sehingga dari metrik evaluasi klasifikasi FFNN memberikan hasil paling tinggi pada penelitian ini. Namun, FFNN bukan metode yang paling cepat dari sisi waktu komputasi karena memiliki waktu pelatihan dan waktu prediksi paling lama dibandingkan metode pembanding. Sebaliknya, KNN memiliki waktu pelatihan paling cepat, *Decision Tree* memiliki waktu prediksi paling cepat, dan SVM memiliki akurasi yang mendekati FFNN. Berdasarkan hasil tersebut, terdapat *trade-off* antara kinerja klasifikasi dan waktu komputasi yang menunjukkan bahwa FFNN memberikan kinerja klasifikasi tertinggi, tetapi menghasilkan waktu komputasi yang lebih besar dibandingkan metode pembanding.

4.6 Perbandingan dengan Penelitian Sebelumnya

Penelitian ini selanjutnya dibandingkan dengan penelitian yang dilakukan oleh Jamhuri dkk. (2025) yang mengembangkan metode optimasi *neural network* berbasis *Gauss-Newton* dengan QR *factorization* untuk klasifikasi varian SARS-CoV-2. Perbedaan utama antara penelitian Jamhuri dkk. (2025) dan penelitian ini terletak pada data, representasi fitur, arsitektur model, dan fokus pengujian. Penelitian Jamhuri dkk. (2025) menggunakan sekuens asam amino *spike* protein SARS-CoV-2 yang direpresentasikan ke dalam bentuk numerik, sedangkan penelitian ini menggunakan sekuens nukleotida yang direpresentasikan menggunakan fitur k-mer. Dari sisi arsitektur, penelitian Jamhuri dkk. (2025) menguji beberapa jenis *neural network*, yaitu SLP, MLP, CNN, dan LSTM,

sedangkan penelitian ini berfokus pada implementasi *Feedforward Neural Network* (FFNN). Dari sisi optimasi, penelitian Jamhuri dkk. (2025) berfokus pada pengembangan *optimizer* QR-GN berbasis *Gauss-Newton* dan QR *factorization*, sedangkan penelitian ini menggunakan optimizer SGD dengan parameter *learning rate* sebesar 0,01.

Berdasarkan hasil penelitian Jamhuri dkk. (2025), model SLP dengan optimizer QR-GN menghasilkan akurasi sebesar 0,9708, sedangkan MLP dengan QR-GN menghasilkan akurasi sebesar 0,9456. Pada arsitektur CNN, QR-GN menghasilkan akurasi sebesar 0,9617, sedangkan pada LSTM akurasi sebesar 0,3038. Hasil tersebut menunjukkan bahwa kinerja QR-GN berbeda pada setiap arsitektur *neural network* yang digunakan. QR-GN memberikan hasil yang cukup tinggi pada SLP, MLP, dan CNN, tetapi kurang optimal pada LSTM. Selain itu, penelitian Jamhuri dkk. (2025) juga menunjukkan bahwa waktu komputasi berbeda pada setiap arsitektur dan *optimizer* yang digunakan.

Pada penelitian ini, model FFNN menghasilkan nilai akurasi sebesar 0,9983 pada konfigurasi terbaik. Jika dibandingkan berdasarkan nilai akurasi, hasil penelitian ini memiliki selisih sebesar 0,0275 terhadap SLP, 0,0527 terhadap MLP, 0,0366 terhadap CNN, dan 0,6945 terhadap LSTM pada penelitian Jamhuri dkk. (2025). Nilai tersebut menunjukkan bahwa model FFNN pada penelitian ini mampu menghasilkan kinerja klasifikasi yang tinggi pada data dan konfigurasi yang digunakan. Namun, perbandingan dengan penelitian Jamhuri dkk. (2025) tidak dapat digunakan untuk menyatakan bahwa salah satu metode lebih unggul, karena *dataset*, bentuk representasi data, arsitektur model, *optimizer*, dan kondisi pengujian yang digunakan.

Jika ditinjau dari metrik evaluasi klasifikasi, hasil penelitian ini menunjukkan nilai akurasi yang lebih baik. Namun, perbedaan tersebut perlu dipahami sebagai perbandingan deskriptif, bukan perbandingan langsung. Penelitian Jamhuri dkk. (2025) menggunakan data *spike* protein dalam bentuk sekuens asam amino dan berfokus pada efisiensi *optimizer* QR-GN, sedangkan penelitian ini menggunakan fitur k-mer dari sekuens nukleotida dan berfokus pada klasifikasi varian SARS-CoV-2 menggunakan FFNN.

Penelitian Jamhuri dkk. (2025) menekankan bahwa proses pelatihan *neural network* dapat membutuhkan sumber daya komputasi yang besar, sehingga diperlukan metode optimasi yang lebih efisien. Dalam penelitian ini, FFNN menghasilkan nilai evaluasi klasifikasi yang tinggi, tetapi tetap membutuhkan waktu pelatihan karena proses pembelajaran dilakukan melalui pembaruan bobot dan bias secara berulang pada setiap *epoch*. Dengan demikian, hasil penelitian ini juga menunjukkan adanya *trade-off* antara kinerja klasifikasi dan waktu komputasi.

Secara keseluruhan, penelitian ini dan penelitian Jamhuri dkk. (2025) sama-sama menunjukkan bahwa pendekatan *neural network* dapat digunakan untuk klasifikasi varian SARS-CoV-2. Perbedaannya, penelitian Jamhuri dkk. (2025) lebih menekankan pengembangan metode optimasi QR-GN untuk meningkatkan efisiensi pembelajaran *neural network*, sedangkan penelitian ini menekankan penggunaan fitur k-mer dan FFNN untuk memperoleh kinerja klasifikasi yang tinggi. Oleh karena itu, hasil penelitian ini dapat dikatakan mendukung penggunaan *neural network* dalam klasifikasi varian SARS-CoV-2, dengan tetap mempertimbangkan bahwa kinerja model perlu dinilai dari metrik evaluasi klasifikasi dan waktu komputasi.

4.7 Pengamalan Ilmu dalam Islam

Hasil penelitian menunjukkan bahwa algoritma *Feedforward Neural Network* (FFNN) menghasilkan kinerja yang baik dalam mengklasifikasikan varian SARS-CoV-2. Hal ini ditunjukkan oleh nilai akurasi, presisi, *recall*, dan F1-*score* sebesar 0,9983. Selain itu, hasil *confusion matrix* menunjukkan bahwa sebagian besar data berhasil diklasifikasikan dengan benar dengan jumlah kesalahan prediksi yang sangat sedikit. Keberhasilan tersebut menunjukkan bahwa metode FFNN mampu bekerja dengan baik dalam mengolah data sekuens genom yang kompleks.

Penggunaan metode yang sesuai sangat menentukan kualitas hasil yang diperoleh. Model FFNN mampu menangkap hubungan nonlinier antar fitur dengan baik, sehingga dapat membedakan karakteristik setiap varian. Selain itu, proses pelatihan yang stabil dan konvergen menunjukkan bahwa model mampu mempelajari pola data secara efektif. Dalam perspektif Islam, ilmu pengetahuan memiliki kedudukan yang tinggi dan harus diimplementasikan untuk memberikan manfaat nyata. Sebagaimana dijelaskan dalam firman Allah SWT pada QS. Al-Mujādilah ayat 11:

دَرَجٰتٍۗ الْعِلْمَ اُوْتُوا وَالَّذِيْنَ مِنْكُمْۙ اٰمَنُوْا الَّذِيْنَ اللّٰهُ

Artinya: "*Allah niscaya akan mengangkat orang-orang yang beriman di antaramu dan orang-orang yang diberi ilmu beberapa derajat*." (QS. al-Mujadalah: 11) (Kementerian Agama RI, 2022)

Menurut Tafsir Tahlili, ayat tersebut menegaskan bahwa Allah SWT akan mengangkat derajat orang-orang yang beriman dan berilmu. Ilmu tidak hanya dimaknai sebagai pengetahuan teoritis, tetapi juga harus diamalkan dan dimanfaatkan untuk memberikan manfaat (Kementerian Agama RI, 2022).

Hal ini juga diperkuat oleh Sayyidina Ali bin Abi Thalib r.a. menyampaikan sebuah ungkapan hikmah (al-Sakhāwī, 1987), yaitu:

ثَمَرٍ بِلَا كَالشَّجَرِ عَمَلٍ بِلَا العِلْمُ

Artinya: *"Ilmu tanpa amal, bagaikan pohon yang tidak berbuah." (al-Sakhāwī dalam al-Maqāsid al-Hasanah fī Bayān Kathīr min al-Ahādīth al-Mushtaharah 'alā al-Alsinah, no. 1030; Dār al-Kutub al-'Ilmiyyah, Beirut, 1987)*

Menurut al-Sakhāwī, ungkapan tersebut merupakan salah satu perkataan masyhur yang dikenal di kalangan ulama. Dalam al-Maqāsid al-Hasanah, menegaskan bahwa ilmu yang tidak diamalkan maka tidak akan bermanfaat. Hasil penelitian ini juga menunjukkan bahwa penggunaan metode yang tepat dan berbasis data yang valid merupakan bagian dari upaya untuk mencapai hasil yang baik dan dapat dipertanggungjawabkan. Hal ini mencerminkan bahwa proses penelitian telah dilakukan secara sistematis sehingga mampu menghasilkan model dengan kinerja yang baik.

Secara keseluruhan, hasil penelitian ini menunjukkan bahwa algoritma FFNN dalam klasifikasi varian SARS-CoV-2 merupakan bentuk penerapan ilmu yang berhasil dan memberikan manfaat nyata. Model yang dihasilkan tidak hanya mampu mencapai tingkat akurasi yang baik, tetapi juga menunjukkan kestabilan dan efisiensi dalam proses pembelajaran. Hal ini membuktikan bahwa ilmu yang dikembangkan melalui penelitian dapat memberikan kontribusi dalam menyelesaikan permasalahan yang kompleks. Selain itu, keberhasilan ini juga menunjukkan bahwa pendekatan komputasional dapat dimanfaatkan secara efektif dalam bidang kesehatan karena berkontribusi dalam analisis data genom. Dengan demikian, penelitian ini tidak hanya berkontribusi pada pengembangan ilmu pengetahuan, tetapi juga mencerminkan pengamalan ilmu.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan seluruh tahapan penelitian yang telah dilakukan diperoleh beberapa kesimpulan sebagai berikut:

1. Hasil penelitian menunjukkan bahwa algoritma *Feedforward Neural Network* (FFNN) dapat diterapkan untuk melakukan klasifikasi varian SARS-CoV-2 berdasarkan data sekuens cDNA nukleotida. Pada data dilakukan ekstraksi fitur menggunakan metode k-mer dan dinormalisasi sebelum digunakan sebagai data masukan. Selain itu, dilakukan pengujian parameter terhadap beberapa nilai k-mer dan banyak neuron pada *hidden layer* untuk memperoleh model terbaik. Hasil menunjukkan bahwa nilai $k = 4$ dan banyak neuron hidden layer $m = 64$ menghasilkan kinerja yang sangat baik. Proses pelatihan dilakukan melalui tahapan *feedforward* dan *backpropagation* untuk memperbarui parameter. Hasil pelatihan menunjukkan hasil stabil yang ditandai dengan penurunan nilai *loss* secara konvergen pada setiap *epoch*. *Output* model berupa probabilitas pada setiap kelas memungkinkan proses klasifikasi multikelas dilakukan secara lebih terstruktur dan terukur dalam menentukan kelas prediksi. Berdasarkan hasil evaluasi pada konfigurasi akhir dengan *random seed* 42, model mampu mengenali pola pada setiap varian SARS-CoV-2 dengan baik, yang ditunjukkan oleh nilai akurasi, presisi, *recall*, dan F1-*score* yang tinggi. Dengan demikian, algoritma FFNN terbukti dapat diterapkan secara efektif untuk klasifikasi varian SARS- CoV-2 berbasis fitur k-mer.

2. Kinerja algoritma *Feedforward Neural Network* (FFNN) dievaluasi berdasarkan akurasi, presisi, *recall*, F1-*score*, *confusion matrix*, dan waktu komputasi. Berdasarkan hasil evaluasi pada data uji dengan konfigurasi akhir dan *random seed* 42, model memperoleh metrik evaluasi sebesar 0,9983 atau 99,83%. Berdasarkan kategori Gorunescu (2011), metrik evaluasi pada rentang $90\% - 100\%$ termasuk dalam kategori *excellent classification*, sehingga hasil 0,9983 menunjukkan bahwa model berada pada kategori klasifikasi sangat baik.

   Nilai presisi, *recall*, dan F1-*score* keseluruhan diperoleh dari *weighted average* yang mempertimbangkan jumlah data aktual (*support*) pada setiap varian dengan hasil evaluasi masing-masing sebesar 0,9983. Sementara itu, kesalahan klasifikasi yang terjadi relatif sangat kecil, terlihat dari rendahnya nilai *false positive* dan *false negative* pada *confusion matrix*. Nilai 1 pada *classification report* hanya diperoleh oleh varian Alpha dan Beta karena kedua varian tersebut tidak memiliki kesalahan prediksi. Pada varian Delta memperoleh presisi sebesar 1, *recall* sebesar 0,9922, dan F1-*score* sebesar 0,9961. Varian Gamma memperoleh presisi sebesar 0,9960, recall sebesar 1, dan F1-*score* sebesar 0,9980. Sementara itu, varian Omicron memperoleh presisi sebesar 0,9963, *recall* sebesar 1, dan F1-*score* sebesar 0,9982.

   Nilai evaluasi yang tinggi pada akurasi, presisi, *recall*, dan F1-*score* menunjukkan bahwa model mampu mengklasifikasikan varian SARS-CoV-2 dengan kesalahan yang rendah. Namun, waktu komputasi FFNN membutuhkan waktu pelatihan dan waktu prediksi yang lebih lama dibandingkan metode pembanding. Berdasarkan hasil perbandingan, FFNN

memperoleh nilai akurasi, presisi, recall, dan F1-score tertinggi, yaitu 0,9983, tetapi membutuhkan waktu pelatihan selama 8,7880 detik dan waktu prediksi selama 0,2034 detik. Oleh karena itu, hasil penelitian ini menunjukkan adanya *trade-off* antara kinerja klasifikasi dan waktu komputasi dengan FFNN memberikan hasil lebih baik dari metrik evaluasi klasifikasi, tetapi kebutuhan waktu komputasi yang lebih besar dibandingkan metode pembanding.

Pengujian tambahan menggunakan lima *random seed* berbeda dilakukan untuk melihat kestabilan kinerja model terhadap variasi parameter acak dengan diperoleh rata-rata akurasi sebesar 0,9463. Hasil ini menunjukkan bahwa kinerja model masih dapat bervariasi ketika menggunakan *random seed* yang berbeda. Oleh karena itu, nilai akurasi 0,9983 merupakan akurasi terbaik atau hasil evaluasi pada konfigurasi akhir dengan *random seed* 42, tetapi kestabilannya terhadap variasi *random seed* masih perlu diperhatikan.

Dataset dalam penelitian ini didominasi oleh data *host* manusia, dengan sebagian kecil data yang berasal dari *host* lain. Oleh karena itu, penelitian ini berfokus pada klasifikasi varian SARS-CoV-2 berdasarkan pola sekuens genom secara umum, bukan pada evaluasi khusus berdasarkan kelompok *host*. Hasil klasifikasi yang tinggi menunjukkan bahwa model FFNN mampu mengenali pola sekuens pada data yang digunakan. Namun, karena data masih bersifat heterogen dan tahap persiapan data belum menerapkan penyaringan *host* secara ketat, hasil klasifikasi tersebut belum sepenuhnya merepresentasikan karakteristik data secara spesifik pada masing-masing *host*. Dengan demikian, hasil penelitian ini menunjukkan kinerja FFNN yang

sangat baik pada data heterogen, tetapi interpretasi terhadap karakteristik biologis spesifik belum dapat dilakukan secara menyeluruh.

5.2 Saran

Berdasarkan hasil penelitian yang telah dilakukan, terdapat beberapa keterbatasan yang dapat menjadi pertimbangan untuk penelitian selanjutnya. Oleh karena itu, terdapat beberapa saran yang dapat dipertimbangkan untuk pengembangan penelitian selanjutnya, yaitu:

1. Penelitian berikutnya disarankan untuk mengembangkan model dengan metode lain, seperti metode *deep learning* atau *ensemble learning* untuk mengetahui apakah kombinasi beberapa model atau arsitektur yang lebih dalam dapat memberikan kinerja baik.
2. Penelitian berikutnya dapat difokuskan menggunakan *dataset* yang lebih besar dan lebih beragam agar model dapat mempelajari karakteristik sekuens SARS-CoV-2 secara lebih luas.
3. Penelitian selanjutnya disarankan untuk melakukan tahap persiapan data dengan proses pengecekan dan penyaringan yang lebih ketat dengan memperhatikan kualitas sekuens, kelengkapan metadata, sumber data, serta keseragaman karakteristik *host*. Misalnya, data sekuens dari manusia saja, sedangkan data sekuens dari hewan dapat dianalisis secara terpisah berdasarkan kelompok *host* yang sesuai. Dengan demikian, data yang digunakan dalam proses klasifikasi dapat bersifat lebih homogen, sehingga hasil klasifikasi yang diperoleh dapat merepresentasikan karakteristik varian SARS-CoV-2 secara lebih spesifik.

DAFTAR RUJUKAN

Al-Haitsami, N. al-D. ‘A. ibn A. B. (2012). *Majma‘ al-Zawā’id wa Manba‘ al-Fawā’id* (Vol. 8). Dār al-Kutub al-‘Ilmiyyah.

Aljaaf, A. J., Mohsin, T. M., Al-Jumeily, D., & Alloghani, M. (2021). A fusion of data science and feed-forward neural network-based modelling of COVID-19 outbreak forecasting in IRAQ. *Journal of Biomedical Informatics*, *118*. https://doi.org/10.1016/j.jbi.2021.103766

Al-Nawawī. (2019). Al-Majmū‘ Syarḥ al-Muhadzdzab, Muqaddimah. Dār al-Fikr. Maktabah al-Shamela.

Al-Nawawī. (2019). *Sharḥ Ṣaḥīḥ Muslim* no. 2722. Dār al-Fikr. Maktabah al-Shamela.

al-Sakhāwī, Muḥammad b. ‘Abd al-Raḥmān. (1987). al-Maqāṣid al-Ḥasanah fī Bayān Kathīr min al-Aḥādīth al-Mushtaharah ‘alā al-Alsinah. Beirut: Dār al-Kutub al-‘Ilmiyyah.

Alshayeji, M. H., Sindhu, S. C. B., & Abed, S. (2023). Viral genome prediction from raw human DNA sequence samples by combining natural language processing and machine learning techniques. *Expert Systems with Applications*, *218*. https://doi.org/10.1016/j.eswa.2023.119641

Al-Ṭabarānī, A. al-Q. S. bin A. (2012). al-Mu‘jam al-Awsaṭ (المعجم الأوسط). Dār al-Kutub al-‘Ilmiyyah (DKI).

Anton, H., & Rorres, C. (2013). *Elementary Linear Algebra, Applications Version-Wiley*.

Choi, J. Y., & Smith, D. M. (2021). SARS-CoV-2 Variants of Concern. Dalam *Yonsei Medical Journal* (Vol. 62, Nomor 11, hlm. 961–968). Yonsei University College of Medicine. https://doi.org/10.3349/ymj.2021.62.11.961

Colas, C., Sigaud, O., & Oudeyer, P.-Y. (2018). *How Many Random Seeds? Statistical Power Analysis in Deep Reinforcement Learning Experiments*. http://arxiv.org/abs/1806.08295

Cornish-Bowden, A. (1985). *Nucleic Acids Research Nomenclature for incompletely specified bases in nucleic acid sequences* (Vol. 13).

Correia, J. P., Silva, L. R. da, & Silva, R. (2025). Multifractal analysis and support vector machine for the classification of coronaviruses and SARS-CoV-2 variants. *Scientific Reports*, *15*. https://doi.org/10.1038/s41598-025-98366-5

Daqiqil Id, I. (2021). *Machine Learning: Teori, Studi Kasus dan Implementasi Menggunakan Python*. UR Press.

Doukim, C. A., Dargham, J. A., & Chekima, A. (2010). *Finding the Number of Hidden Neurons for an MLP Neural Network Using Coarse to Fine Search Technique*. IEEE.

Dullah, A. U., Utami, P., & Unjung, J. (2025). The Asthma Classification Using an Adaptive Boosting Model with SVM-SMOTE Sampling. *Journal of Information System Exploration and Research*, *3*(1), 1–10. https://doi.org/10.52465/joiser.v3i1.486

Gayathri, J. L., Abraham, B., Sujarani, M. S., & Nair, M. S. (2021). A computer-aided diagnosis system for the classification of COVID-19 and non-COVID-19 pneumonia on chest X-ray images by integrating CNN with sparse autoencoder and feed forward neural network. *Computers in Biology and Medicine*, *141*. https://doi.org/10.1016/j.compbiomed.2021.105134

Géron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc.

Gorunescu, F. (2011). Data Mining Concepts, Models and Techniques. *Spinger*.

Han, J., Kamber, M., & Pei, J. (2011). *Data Mining. Concepts and Techniques* (3rd ed.). Morgan Kaufmann.

Haris, Muh. I., Takdir, N., Safitri, A., Habibi, A. R., Fatmawati, A., Pangestika, Y., Jannah, M., Palupi, D., Alfikri, M. R., Fitrianti, V., Burhan, R., Yulianti, M. E. P., & Harsachatri, D. O. (2025). Biologi Dasar. Azzia Karya Bersama.

Haykin, S. S. (2009). *Neural Networks and Learning Machine (Third Edition)*. Prentice Hall/Pearson.

Ibn Mājah, M. (2019). Sunan Ibn Mājah. Kitāb al-Muqaddimah (مقدمة ابن ماجه), no. 250. Dār al-Kutub al-'Ilmiyyah.

Jamhuri, M., Irawan, M. I., Mukhlash, I., Iqbal, M., & Puspaningsih, N. N. T. (2025). Neural networks optimization via Gauss–Newton based QR factorization on SARS-CoV-2 variant classification. *Systems and Soft Computing*, *7*. https://doi.org/10.1016/j.sasc.2025.200195

Jones, N. C., & Pevzner, P. A. (2004). *An Introduction to Bioinformatics Algorithms*.

Kementerian Agama. (2022). *Al-Qur'an dan Terjemahannya. Lajnah Pentashihan Mushaf Al-Qur'an Badan Litbang Kementerian Agama RI*. Diakses pada 06 Mei 2026. https://quran.kemenag.go.id/

Li, M., Lv, F., Li, Z., Zhao, C., Wang, X., Zhu, P., & Zhou, X. (2024). Cross-Species Susceptibility of Emerging Variants of SARS-CoV-2 Spike. *Genes*, *15*(10). https://doi.org/10.3390/genes15101321

Menarianti, I., Wihardjo, E., Hayati, N., Zakarijah, M., Hartanti, N. T., Arfianto, A. Z., Nampira, A. A., Khumaidi, A., Kolyaan, Y., & Rahajeng, E. (2025). *KONSEP DASAR MACHINE LEARNING DI ERA DIGITAL*.

Mwanga, M. J., Obura, H. O., Evans, M., & Awe, O. I. (2023). *Enhanced Deep Convolutional Neural Network for SARS-CoV-2 Variants Classification*. https://doi.org/10.1101/2023.08.09.552643

Özüdoğru, O., Bahçe, Y. G., & Acer, Ö. (2022). SARS CoV-2 reinfection rate is higher in the Omicron variant than in the Alpha and Delta variants. *Irish Journal of Medical Science*, *192*(2), 751–756. https://doi.org/10.1007/s11845-022-03060-4

Parwanto, E. (2021). Virus Corona (SARS-CoV-2) penyebab COVID-19 kini telah bermutasi. *Jurnal Biomedika dan Kesehatan*, *4*(2), 47–49. https://doi.org/10.18051/jbiomedkes.2021.v4.47-49

Putra, J. W. G. (2020). *Pengenalan Konsep Pembelajaran Mesin dan Deep Learning Edisi 1.4*. Self-published.

Ramadhani, N. A., & Rosyid, H. A. (2022). Algoritma - Algoritma Data Mining untuk Klasifikasi Data. *Jurnal Inovasi Teknik dan Edukasi Teknologi*, *2*(12), 550–556.

Salehi-Vaziri, M., Fazlalipour, M., Seyed Khorrami, S. M., Azadmanesh, K., Pouriayevali, M. H., Jalali, T., Shoja, Z., & Maleki, A. (2022). The ins and outs of SARS-CoV-2 variants of concern (VOCs). Dalam *Archives of Virology* (Vol. 167, Nomor 2, hlm. 327–344). Springer. https://doi.org/10.1007/s00705-022-05365-2

Singh, O. P., Vallejo, M., El-Badawy, I. M., Aysha, A., Madhanagopal, J., & Faudzi, A. A. M. (2021). Classification of SARS-CoV-2 and non-SARS-CoV-2 using machine learning algorithms. *Computers in Biology and Medicine*, *136*. https://doi.org/10.1016/j.compbiomed.2021.104650

Susilo, A., Jasirwan, C. O. M., Wafa, S., Maria, S., Rajabto, W., Muradi, A., Fachriza, I., Putri, M. Z., & Gabriella, S. (2022). Mutasi dan Varian Coronavirus Disease 2019 (COVID-19): Tinjauan Literatur Terkini. *Jurnal Penyakit Dalam Indonesia*, *9*(1), 59. https://doi.org/10.7454/jpdi.v9i1.648

Ullah, W., Ullah, A., Malik, K. M., Saudagar, A. K. J., Khan, M. B., Hasanat, M. H. A., AlTameem, A., & AlKhathami, M. (2022). Multi-Stage Temporal Convolution Network for COVID-19 Variant Classification. *Diagnostics*, *12*(11). https://doi.org/10.3390/diagnostics12112736

Vilain, M., & Aris-Brosou, S. (2023). Machine Learning Algorithms Associate Case Numbers with SARS-CoV-2 Variants Rather Than with Impactful Mutations. *Viruses*, *15*(6). https://doi.org/10.3390/v15061226

World Health Organization.. (2025). *WHO COVID-19* dashboard. data.who.int. Diakses pada 06 November 2025. https://data.who.int/dashboards/covid19

Yang, H., & Rao, Z. (2021). Structural biology of SARS-CoV-2 and implications for therapeutic development. Dalam *Nature Reviews Microbiology* (Vol. 19, Nomor 11, hlm. 685–700). Nature Research. https://doi.org/10.1038/s41579-021-00630-8

Yang, Y., Xiao, Z., Ye, K., He, X., Sun, B., Qin, Z., Yu, J., Yao, J., Wu, Q., Bao, Z., & Zhao, W. (2020). SARS-CoV-2: characteristics and current advances in research. Dalam *Virology Journal* (Vol. 17, Nomor 1). BioMed Central. https://doi.org/10.1186/s12985-020-01369-z

Zaki, M. J., & Meira, Jr., W. (2020). *Data Mining and Machine Learning: Fundamental Concepts and Algorithms (2nd Edition)*.

Zhang, Y., Wang, X., & Kang, L. (2011). A k-mer scheme to predict piRNAs and characterize locust piRNAs. *Bioinformatics*, *27*(6), 771–776. https://doi.org/10.1093/bioinformatics/btr016

Zielezinski, A., Vinga, S., Almeida, J., & Karlowski, W. M. (2017). Alignment-free sequence comparison: Benefits, applications, and tools. Dalam *Genome Biology* (Vol. 18, Nomor 1). BioMed Central Ltd. https://doi.org/10.1186/s13059-017-1319-7

LAMPIRAN

Lampiran 1 *Dataset* Penelitian

https://www.kaggle.com/datasets/mutialfi/data-skripsi-mutiara	

Lampiran 2 Kode Program Perhitungan Manual Menggunakan Lima Data Awal

https://www.kaggle.com/code/mutialfi/perhitungan-manual-lima-data-awal	

Lampiran 3 Kode Program Perhitungan Menggunakan Seluruh Data dan Perbandingan dengan Metode

https://www.kaggle.com/code/mutialfi/seluruh-data-dengan-perbandingan-metode	

Lampiran 4 Kode Program Variasi Nilai K-mer

https://www.kaggle.com/code/mutialfi/nilai-k-mer	

Lampiran 5 Kode Program Variasi Banyak Neuron *Hidden* (m)

https://www.kaggle.com/code/mutialfi/variasi-banyak-neuron-hidden-m	

Lampiran 6 Kode Program Percobaan Lima *Random Seed* Berbeda

https://www.kaggle.com/code/mutialfi/variasi-nilai-random-seed	

Lampiran 7 Hasil Evaluasi Perbandingan Model pada Lima Kali Percobaan

Model	Percobaan	Akurasi	Presisi	*Recall*	F1-*score*	Waktu Pelatihan (detik)	Waktu Prediksi (detik)
FFNN	1	0,9983	0,9983	0,9983	0,9983	8,8300	0,1930
	2	0,9983	0,9983	0,9983	0,9983	8,7100	0,1980
	3	0,9983	0,9983	0,9983	0,9983	9,0100	0,2270
	4	0,9983	0,9983	0,9983	0,9983	8,7800	0,2050
	5	0,9983	0,9983	0,9983	0,9983	8,5600	0,1940
KNN	1	0,9958	0,9958	0,9958	0,9958	0,0027	0,2130
	2	0,9958	0,9958	0,9958	0,9958	0,0025	0,1890
	3	0,9958	0,9958	0,9958	0,9958	0,0026	0,1880
	4	0,9958	0,9958	0,9958	0,9958	0,0025	0,1730
	5	0,9958	0,9958	0,9958	0,9958	0,0024	0,1720
Decision Tree	1	0,9874	0,9876	0,9874	0,9874	0,1340	0,0025
	2	0,9874	0,9876	0,9874	0,9874	0,1280	0,0025
	3	0,9874	0,9876	0,9874	0,9874	0,1260	0,0025
	4	0,9874	0,9876	0,9874	0,9874	0,1220	0,0021
	5	0,9874	0,9876	0,9874	0,9874	0,1280	0,0020
SVM	1	0,9975	0,9975	0,9975	0,9975	1,6100	0,1130
	2	0,9975	0,9975	0,9975	0,9975	1,5600	0,1220
	3	0,9975	0,9975	0,9975	0,9975	1,5500	0,1090
	4	0,9975	0,9975	0,9975	0,9975	1,5900	0,1220
	5	0,9975	0,9975	0,9975	0,9975	1,6200	0,1100

Lampiran 8 Hasil Evaluasi Kinerja Model pada Variasi Nilai *K*-mer

Nilai *K*-mer	Percobaan	Akurasi	Presisi	*Recall*	F1-*score*	Waktu Pelatihan (detik)
3	1	0,9091	0,9276	0,9091	0,8949	8,2300
	2	0,9091	0,9276	0,9091	0,8949	8,4100
	3	0,9091	0,9276	0,9091	0,8949	8,2200
	4	0,9091	0,9276	0,9091	0,8949	8,2500
	5	0,9091	0,9276	0,9091	0,8949	8,8800
4	1	0,9983	0,9983	0,9983	0,9983	8,4000
	2	0,9983	0,9983	0,9983	0,9983	8,8300
	3	0,9983	0,9983	0,9983	0,9983	8,7900
	4	0,9983	0,9983	0,9983	0,9983	8,5100
	5	0,9983	0,9983	0,9983	0,9983	8,4900
5	1	0,9983	0,9983	0,9983	0,9983	9,9600
	2	0,9983	0,9983	0,9983	0,9983	10,2000
	3	0,9983	0,9983	0,9983	0,9983	12,7000
	4	0,9983	0,9983	0,9983	0,9983	10,5000
	5	0,9983	0,9983	0,9983	0,9983	10,3000

Lampiran 9 Hasil Evaluasi Kinerja Model pada Variasi Banyak Neuron *Hidden* (m)

Banyak Neuron (m)	**Percobaan**	**Akurasi**	**Presisi**	***Recall***	**F1-*score***	**Waktu Pelatihan (detik)**
32	1	0,9983	0,9983	0,9983	0,9983	8,1300
	2	0,9983	0,9983	0,9983	0,9983	7,6700
	3	0,9983	0,9983	0,9983	0,9983	7,8000
	4	0,9983	0,9983	0,9983	0,9983	7,7300
	5	0,9983	0,9983	0,9983	0,9983	7,7900
64	1	0,9983	0,9983	0,9983	0,9983	7,9500
	2	0,9983	0,9983	0,9983	0,9983	8,2000
	3	0,9983	0,9983	0,9983	0,9983	8,1400
	4	0,9983	0,9983	0,9983	0,9983	7,8000
	5	0,9983	0,9983	0,9983	0,9983	7,9100
128	1	0,9975	0,9975	0,9975	0,9975	8,5100
	2	0,9975	0,9975	0,9975	0,9975	8,5700
	3	0,9975	0,9975	0,9975	0,9975	8,7700
	4	0,9975	0,9975	0,9975	0,9975	8,5600
	5	0,9975	0,9975	0,9975	0,9975	8,4100
256	1	0,9975	0,9975	0,9975	0,9975	8,9400
	2	0,9975	0,9975	0,9975	0,9975	9,0500
	3	0,9975	0,9975	0,9975	0,9975	9,1900
	4	0,9975	0,9975	0,9975	0,9975	8,9800
	5	0,9975	0,9975	0,9975	0,9975	8,8800

Lampiran 10 Hasil Evaluasi Kinerja Model FFNN Menggunakan Lima *Random Seed* Berbeda

Random seed	**Percobaan**	**Akurasi**	**Presisi**	***Recall***	**F1-*score***	**Waktu Pelatihan (detik)**
42	1	0,9983	0,9983	0,9983	0,9983	9,3400
	2	0,9983	0,9983	0,9983	0,9983	9,9200
	3	0,9983	0,9983	0,9983	0,9983	8,8600
	4	0,9983	0,9983	0,9983	0,9983	8,8900
	5	0,9983	0,9983	0,9983	0,9983	9,2700
43	1	0,9975	0,9975	0,9975	0,9975	9,2900
	2	0,9975	0,9975	0,9975	0,9975	10,000
	3	0,9975	0,9975	0,9975	0,9975	8,5800
	4	0,9975	0,9975	0,9975	0,9975	8,8600
	5	0,9975	0,9975	0,9975	0,9975	8,6100
44	1	0,9891	0,9893	0,9891	0,9890	9,1400
	2	0,9891	0,9893	0,9891	0,9890	10,1000
	3	0,9891	0,9893	0,9891	0,9890	8,7900
	4	0,9891	0,9893	0,9891	0,9890	8,8500
	5	0,9891	0,9893	0,9891	0,9890	9,1300
45	1	0,9907	0,9909	0,9907	0,9907	9,4000
	2	0,9907	0,9909	0,9907	0,9907	9,6700
	3	0,9907	0,9909	0,9907	0,9907	8,7200
	4	0,9907	0,9909	0,9907	0,9907	8,9600
	5	0,9907	0,9909	0,9907	0,9907	8,8700
46	1	0,9975	0,9975	0,9975	0,9975	9,5400
	2	0,9975	0,9975	0,9975	0,9975	9,7300
	3	0,9975	0,9975	0,9975	0,9975	8,6100
	4	0,9975	0,9975	0,9975	0,9975	8,9600
	5	0,9975	0,9975	0,9975	0,9975	9,2900

RIWAYAT HIDUP

Mutiara Alfi, lahir di Malang pada 16 Mei 2004. Penulis merupakan anak ketiga dari tiga bersaudara. Anak dari Bapak Moh Nafe' dan Ibu Ninik Fardah. Selama masa pendidikan, penulis menempuh pendidikan mulai dari TK Kartini yang lulus pada tahun 2010, kemudian melanjutkan pendidikan dasar di SDS Taman Siswa Turen dan lulus pada tahun 2016. Kemudian penulis melanjutkan pendidikan menengah pertama di MTsN 1 Malang, lulus pada tahun 2019. Selanjutnya, melanjutkan pendidikan sekolah menengah atas di MAN 1 Malang dan lulus pada tahun 2022. Pada tahun yang sama penulis melanjutkan pendidikan di bangku perkuliahan sebagai mahasiswa program studi Matematika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Selama menempuh pendidikan di perguruan tinggi, penulis pernah mengikuti kepanitiaan SIGMA, yaitu kegiatan orientasi mahasiswa baru Program Studi Matematika pada tahun 2023. Penulis juga beberapa kali mengikuti kegiatan sukarelawan (*volunteer*) sebagai bentuk partisipasi dalam kegiatan sosial.

KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No. 50 Malang 65144 Telp. / Fax. (0341) 558933

BUKTI KONSULTASI SKRIPSI

Nama : Mutiara Alfi
NIM : 220601110071
Fakultas / Jurusan : Sains dan Teknologi / Matematika
Judul Skripsi : Aplikasi *Feedforward Neural Network* untuk Klasifikasi Varian SARS-CoV-2 Berdasarkan Sekuens DNA Nukleotida
Dosen Pembimbing I : Dr. Mohammad Jamhuri, M.Si
Dosen Pembimbing II : Dr. Abdussakir, M.Pd

No	Tanggal	Hal	Tanda Tangan
1.	4 September 2025	Konsultasi Topik dan Data	1.
2.	17 September 2025	Konsultasi Topik dan Data	2.
3.	10 Oktober 2025	Konsultasi Bab I dan II	3.
4.	17 Oktober 2025	Konsultasi Bab I, II, dan III	4.
5.	24 Oktober 2025	Konsultasi Kajian Agama Bab I dan II	5.
6.	31 Oktober 2025	Konsultasi Bab I, II, dan III	6.
7.	6 November 2025	Konsultasi Bab I, II, dan III	7.
8.	19 November 2025	Konsultasi Kajian Agama Bab I dan II	8.
9.	20 November 2025	Konsultasi Bab I, II, dan III	9.
10.	26 November 2025	Konsultasi Kajian Agama Bab I dan II	10.
11.	27 November 2025	Konsultasi Bab I, II, dan III	11.
12.	3 Desember 2025	ACC Bab I, II, dan III	12.
13.	4 Desember 2025	ACC Kajian Agama Bab I dan II	13.
14.	13 Januari 2026	ACC Seminar Proposal	14.
15.	26 Januari 2026	Konsultasi Bab IV dan V	15.
16.	30 Januari 2026	Konsultasi Bab IV dan V	16.

KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No. 50 Malang 65144 Telp. / Fax. (0341) 558933

17.	4 April 2026	Konsultasi Bab IV dan V	17.
18.	28 April 2026	Konsultasi Bab IV dan V	18.
19.	6 Mei 2026	Konsultasi Kajian Agama Bab IV	19.
20.	6 Mei 2026	Konsultasi Bab IV dan V	20.
21.	8 Mei 2026	ACC Kajian Agama Bab IV	21.
22.	11 Mei 2026	ACC Bab IV dan V	22.
23.	20 Mei 2026	ACC Seminar Hasil	23.
24.	25 Mei 2026	Konsultasi Revisi Seminar Hasil	24.
25.	2 Juni 2026	ACC Sidang Skripsi	25.
26.	12 Juni 2026	ACC Keseluruhan	26.

Malang, 12 Juni 2026
Mengetahui,
Ketua Program Studi Matematika

Dr. Fachrur Rozi, M.Si
NIP. 19800527 200801 1 012