

ANALISIS PERBANDINGAN KLASIFIKASI *TRAFFIC NETWORK* APLIKASI *OVER-THE-TOP (OTT)* DENGAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK (CNN)*, *LONG SHORT-TERM MEMORY (LSTM)*, DAN *BI-DIRECTIONAL LONG SHORT-TERM MEMORY (Bi-LSTM)*

SKRIPSI



FARADIAS IZZA AZZAHRA FAISAL

H071191030

**PROGRAM STUDI SISTEM INFORMASI
DEPARTEMEN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS HASANUDDIN
MAKASSAR**

2023

ANALISIS PERBANDINGAN KLASIFIKASI *TRAFFIC NETWORK* APLIKASI *OVER-THE-TOP* (OTT) DENGAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK* (CNN), *LONG SHORT-TERM MEMORY* (LSTM), DAN *BI-DIRECTIONAL LONG SHORT-TERM MEMORY* (Bi-LSTM)

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelas Sarjana Komputer pada Program Studi Sistem Informasi Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin

FARADIAS IZZA AZZAHRA FAISAL

H071191030

**PROGRAM STUDI SISTEM INFORMASI
DEPARTEMEN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS HASANUDDIN
MAKASSAR**

2023

LEMBAR PERNYATAAN KEOTENTIKAN

Yang bertandatangan di bawah ini:

Nama : Faradias Izza Azzahra Faisal

NIM : H071191030

Program Studi : Sistem Informasi

Jenjang : S1

Menyatakan dengan ini bahwa karya tulisan saya berjudul

Analisis Perbandingan Klasifikasi *Traffic Network* Aplikasi *Over-The-Top* (OTT) dengan Algoritma *Convolutional Neural Network* (CNN), *Long Short-Term Memory* (LSTM), dan *Bi-Directional Long Short-Term Memory* (Bi-LSTM)

adalah karya tulisan saya sendiri dan bukan merupakan pengambilan alih tulisan orang lain, dan belum pernah dipublikasikan dalam bentuk apapun.

Makassar, 7 Juli 2023



Faradias Izza Azzahra Faisal

NIM. H071191030

ANALISIS PERBANDINGAN KLASIFIKASI TRAFFIC NETWORK APLIKASI OVER-THE-TOP (OTT) DENGAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (CNN), LONG SHORT-TERM MEMORY (LSTM), DAN BI-DIRECTIONAL LONG SHORT-TERM MEMORY (Bi-LSTM)

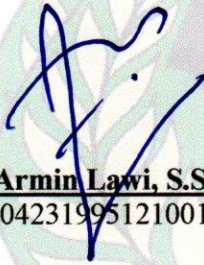
Disusun dan diajukan oleh

FARADIAS IZZA AZZAHRA FAISAL

H071191030

UNIVERSITAS HASANUDDIN
Menyetujui,

Pembimbing Utama



Dr. Eng. Armin Lawi, S.Si., M.Eng.
NIP. 197204231995121001

Pembimbing Pertama



Ir. Eliyah Acantha Manapa
Sampetoding, S.Kom., M.Kom.
NIP. 3273221911910006

Kepala Program Studi



Dr. Hendra, S.Si., M.Kom.
NIP. 197601022002121001



HALAMAN PENGESAHAN

Skripsi ini diajukan oleh:

Nama : Faradias Izza Azzahra Faisal

NIM : H071191030


Program Studi : Sistem Informasi

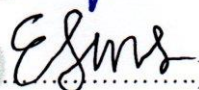
Judul Skripsi : Analisis Perbandingan Klasifikasi *Traffic Network* Aplikasi *Over-The-Top* (OTT) dengan Algoritma *Convolutional Neural Network* (CNN), *Long Short-Term Memory* (LSTM), dan *Bi-Directional Long Short-Term Memory* (Bi-LSTM)

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Komputer pada Program Studi Sistem Informasi Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin.


DEWAN PENGUJI

Tanda Tangan

Ketua : Dr. Eng. Armin Lawi, S.Si., M.Eng. (..........)

Sekretaris : Ir. Eliyah Acantha Manapa Sampetoding, S.Kom., M.Kom. (..........)

Anggota : Rozalina Amran, S.T., M.Eng. (..........)

Anggota : Edy Saputra Rusdi, S.Si., M.Si (..........)

Ditetapkan di : Makassar

Tanggal : 7 Juli 2023



KATA PENGANTAR

Puji syukur penulis panjatkan kehadiran Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya penulis dapat menyelesaikan skripsi ini. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Strata 1. Penulis menyadari telah mendapatkan banyak bantuan serta bimbingan dari berbagai pihak, dari masa perkuliahan hingga pada tahap akhir penyusunan skripsi ini. Oleh karena itu, pada kesempatan ini, dengan segala kerendahan hati penulis menyampaikan terimakasih yang sebesar-besarnya kepada:

1. Keluarga, ayahanda tercinta **Drs. Ir. Faisal Syafar, M.Si., M.InfTech., Ph.D., IPU** dan ibunda tersayang **Dr. Halimah Husain., M.Si** yang telah memberikan dukungan baik moril maupun materiil serta doa yang tiada henti-hentinya kepada penulis. Ucapan terima kasih juga kepada adik-adik penulis, **Fawwas, Fath, dan Fayyad**.
2. Rektor Universitas Hasanuddin, Bapak **Prof. Dr. Ir. Jamaluddin Jompa, M.Sc.**, beserta jajarannya.
3. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin, **Bapak Dr. Eng. Amiruddin, S.Si., M.Si**, beserta staf yang telah membantu dan mengarahkan penulis dalam berbagai hal dalam urusan akademik maupun administrasi.
4. Ketua Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Bapak **Prof. Dr. Nurdin, S.Si., M.Si.**, atas ilmu dan saran-saran yang diberikan.
5. Ketua Program Studi Sistem Informasi, Bapak **Dr. Hendra, S.Si., M.Kom.**, yang senantiasa membantu dan memberikan arahan selama masa studi penulis.
6. Bapak **Dr. Eng. Armin Lawi, S.Si., M.Eng.** selaku dosen pembimbing utama dan Bapak **Ir. Eliyah Acantha Manapa Sampetoding, S.Kom., M.Kom.**, selaku dosen pembimbing pertama untuk segala ilmu dan kesabaran dalam membimbing dan mengarahkan penulis, juga bersedia meluangkan waktunya untuk mendampingi penulis sehingga skripsi ini dapat terselesaikan.

7. Dosen Penguji, Ibu **Rozalina Amran, S.T., M.Eng.**, dan Bapak **Edy Saputra Rusdi, S.Si., M.Si.**, yang telah meluangkan waktunya sejak seminar proposal hingga sidang skripsi untuk memberikan saran dan masukan dalam proses penulisan skripsi penulis.
8. Bapak **A. Muh. Amil Siddik, S.Si., M.Si.**, selaku dosen pembimbing akademik telah memberikan arahan kepada penulis selama masa perkuliahan dan memberikan masukan-masukan dalam penyusunan skripsi.
9. **Dosen Departemen Matematika** khususnya **Bapak dan Ibu Dosen Program Studi Sistem Informasi, Fakultas MIPA Universitas Hasanuddin** atas semua ilmu yang sangat bermanfaat yang telah diajarkan kepada penulis selama menempuh pendidikan.
10. Sahabat **“Nolep menuju Lep”** yaitu Septi, Nisa, Bila, Salsa dan Uly yang saling memberikan informasi, dukungan dan semangat satu sama lain selama masa perkuliahan hingga selesainya penulisan skripsi penulis.
11. Sahabat **“???”** yaitu Diva, Maya, Diah, dan Raihana yang senantiasa memberikan semangat dan dukungan kepada penulis dalam penyelesaian skripsi.
12. Sahabat **KKNT 108 Posko 18 PUPR GOWA**, yaitu Karno, Iyan, Hans, Filah, Annis, Nana, Ayu, dan Tiara, yang senantiasa memberikan dukungan, semangat, dorongan, dan motivasi sejak masa KKN sampai masa penulisan skripsi.
13. Segenap *Executive Board* dan anggota **Unhas Career Women** sebagai wadah dan komunitas yang telah memberikan inspirasi dan tempat berkembang, juga memberi dukungan selama masa perkuliahan sampai penulisan skripsi.
14. Teman - teman **MSIB Huawei Batch 3** yang banyak membantu dan membersamai penulis sejak masa-masa awal penulisan skripsi, serta segenap **tim XL SOC**, atas segala dukungan dan dorongannya, juga memberikan banyak ilmu, motivasi, serta inspirasi dari proses pencarian judul sampai penulisan skripsi.

15. Seluruh teman-teman **Program Studi Sistem Informasi Angkatan 2019** selama empat tahun bersama yang senantiasa memberikan bantuan dan dukungan perkuliahan hingga selesainya penulisan skripsi ini.
16. Komunitas *Stack Overflow, Kaggle, GeeksforGeeks, GitHub, Machine Learning Mastery*, sebagai *website* sumber pembelajaran yang menunjang penyelesaian tugas perkuliahan serta penulisan skripsi ini.

Sebagai manusia biasa penulis menyadari penyusunan skripsi ini jauh dari kata sempurna karena keterbatasan kemampuan dan ilmu pengetahuan yang dimiliki oleh penulis. Oleh karenanya atas kesalahan dan kekurangan dalam penulisan skripsi ini, penulis memohon maaf dan bersedia menerima kritikan yang membangun.

Akhir kata, penulis berharap Tuhan Yang Maha Esa berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga skripsi ini membawa manfaat bagi pengembangan ilmu siapa saja yang membacanya.

Makassar, 7 Juli 2023

Penulis,



Faradias Izza Azzahra Faisal

PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Hasanuddin, saya yang bertanda tangan di bawah ini:

Nama : Faradias Izza Azzahra Faisal
NIM : H071191030
Program Studi : Sistem Informasi
Departemen : Matematika
Fakultas : Matematika dan Ilmu Pengetahuan Alam Jenis
Karya : Skripsi

demikian pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Hasanuddin **Hak Bebas Royalti Noneksklusif (Non-exclusive Royalty-Free Right)** atas karya ilmiah berjudul :

Analisis Perbandingan Klasifikasi Traffic Network Aplikasi Over-The-Top (OTT) dengan Algoritma Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), dan Bi-Directional Long Short-Term Memory (Bi-LSTM)

Berserta perangkat yang ada (jika diperlukan). Terkait dengan hal di atas, maka pihak universitas berhak menyimpan, mengalih-media/format-kan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di Makassar. Pada Tanggal 7 Juli 2023

Yang menyatakan



(Faradias Izza Azzahra Faisal)

ABSTRAK

Pesatnya perkembangan internet dan industri komunikasi berkontribusi pada skala dan kepadatan trafik jaringan yang semakin besar, dinamis, dan kompleks. Sehubungan dengan hal itu, kebutuhan akan klasifikasi dan identifikasi trafik jaringan juga semakin meningkat seiring dengan munculnya berbagai layanan dan aplikasi baru. Pada sisi lain, tugas klasifikasi trafik jaringan yang akurat dipersulit dengan meningkatnya penggunaan protokol terenkripsi yang menjadi praktik standar dalam bidang keamanan jaringan saat ini. Hal ini menyebabkan metode – metode konvensional menjadi tidak efisien dan tidak bisa digunakan lagi. Penelitian ini membandingkan 3 model *deep learning*, yaitu *Convolutional Neural Network* (CNN), *Long Short-Term Memory* (LSTM), dan *Bi-Directional Long Short-Term Memory* (Bi-LSTM) untuk klasifikasi trafik jaringan aplikasi OTT pada dataset yang tidak seimbang. Hasil evaluasi kinerja ketiga model pada tiga jenis *dataset original, oversampling, dan undersampling* secara keseluruhan cukup baik, dengan nilai akurasi pada data *test* berada pada rentang 0,83 – 0,96. Model LSTM memiliki performa yang lebih baik dalam mengklasifikasikan trafik jaringan aplikasi OTT, berdasarkan nilai evaluasi pada data *test* untuk model LSTM dengan akurasi sebesar 0,96 dan f1-score 0,95, untuk model Bi-LSTM dengan akurasi sebesar 0,95 dan f1-score 0,95, dan model CNN dengan akurasi sebesar 0,91 dan f1-score 0,91.

Kata Kunci: Klasifikasi trafik jaringan, Aplikasi OTT, CNN, LSTM, Bi-LSTM

ABSTRACT

The rapid development of the Internet and the communications industry has contributed to network traffic's increasingly large-scale, dynamic, and complex nature. In this regard, the need to classify and identify network traffic is also increasing, along with various new services and applications emerging. On the other hand, the task of accurately classifying network traffic becomes more complicated with the use of encrypted protocols, which have become standard practice in today's network security field. This causes conventional methods to become inefficient and can no longer be used. This study compares 3 deep learning models, namely Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Bi-Directional Long Short-Term Memory (Bi-LSTM) for classifying OTT application network traffic on unbalanced datasets. The performance evaluation results of the three models on the three types of datasets—original, oversampling, and undersampling—are overall quite good, with accuracy values on the test data in the range of 0,83 – 0,96. The LSTM model has better performance in classifying OTT application network traffic, based on the evaluation value upon the test data for the LSTM model with an accuracy of 0,96 and an f1-score of 0,95, for the Bi-LSTM model with an accuracy of 0,95 and an f1-score of 0,95, and for the CNN model with an accuracy of 0,91 and an f1-score of 0,91.

Keywords: Network Traffic Classifications, OTT Applications, CNN, LSTM, Bi-LSTM

DAFTAR ISI

HALAMAN JUDUL.....	i
HALAMAN PERNYATAAN KEOTENTIKAN.....	ii
HALAMAN PERSETUJUAN PEMBIMBING	iii
HALAMAN PENGESAHAN.....	iv
KATA PENGANTAR	v
HALAMAN PERSETUJUAN PUBLIKASI TUGAS AKHIR.....	viii
ABSTRAK	ix
ABSTRACT.....	x
DAFTAR ISI.....	xi
DAFTAR GAMBAR	xiv
DAFTAR TABEL.....	xvi
DAFTAR RUMUS.....	xvii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah	4
1.3 Batasan Masalah.....	4
1.4 Tujuan Penelitian.....	4
BAB II TINJAUAN PUSTAKA	5
2.1 Penelitian Terkait.....	5
2.2 Definisi Analisis	7
2.3 Kerangka Penelitian Sistem Informasi	7
2.4 Layanan OTT.....	9
2.5 Klasifikasi Trafik Jaringan	10
2.6 <i>Deep Learning</i>	15
2.6.1 <i>Convolutional Neural Network</i>	16
2.6.2 <i>Long-Short-Term Memory</i>	17
2.6.3 <i>Bidirectional LSTM</i>	19
2.7 Klasifikasi Multikelas.....	21
2.8 <i>Random Oversampling</i>	22
2.9 <i>Random Undersampling</i>	22
2.10 Normalisasi Data.....	23

2.11	Evaluasi Kinerja Model	24
2.11.1	Kurva AUC - ROC	26
2.12	<i>Kaggle</i>	27
2.13	<i>Jupyter Notebook</i>	27
2.14	<i>Google Colaboratory</i>	27
2.15	<i>Streamlit</i>	28
BAB III	METODE PENELITIAN	29
3.1	Waktu dan Lokasi Penelitian	29
3.2	Instrumen Penelitian	29
3.3	<i>Dataset</i>	30
3.4	Tahap Penelitian	30
3.5	Metode Penelitian	31
3.5.1	Proses <i>Convolutional Neural Network</i>	31
3.5.2	Proses <i>Long Short-Term Memory</i> dan <i>Bi-directional Long Short-Term Memory</i>	33
BAB IV	HASIL DAN PEMBAHASAN	36
4.1	Deskripsi Data	36
4.2	<i>Pre-processing</i> Data	39
4.2.1	<i>Feature Engineering</i>	40
4.2.2	Seleksi Fitur	41
4.2.3	Eliminasi Label Kelas	41
4.2.4	<i>Feature Encoding</i>	43
4.3	Pembagian Data	43
4.4	<i>Resampling</i>	44
4.5	Normalisasi Data	46
4.6	<i>Data Windowing</i>	47
4.7	<i>Modeling</i>	47
4.8	<i>Training Model</i>	49
4.8.1	Kurva <i>Training Model CNN</i>	49
4.8.2	Kurva <i>Training Model LSTM</i>	50
4.8.3	Kurva <i>Training Model Bi-LSTM</i>	52
4.9	Evaluasi Kinerja Model	54
4.9.1	Perbandingan Rata-Rata Akurasi, <i>Precision</i> , <i>Recall</i> , dan <i>F1-Score</i> Model CNN LSTM dan Bi-LSTM	54

4.9.2	<i>Confusion Matrix</i>	55
4.9.3	<i>ROC Curve</i>	59
4.10	<i>Deployment Model</i>	61
BAB V	KESIMPULAN DAN SARAN	65
5.1	Kesimpulan.....	65
5.2	Saran.....	66
	DAFTAR PUSTAKA	67
	LAMPIRAN.....	72

DAFTAR GAMBAR

Gambar 2.1 Desain Organisasi dan Desain Kegiatan Sistem Informasi	8
Gambar 2.2 Kerangka Penelitian Sistem Informasi	8
Gambar 2.3 Ilustrasi Jaringan Saraf	16
Gambar 2.4 Arsitektur CNN.....	16
Gambar 2.5 Arsitektur LSTM	18
Gambar 2.6 Arsitektur Bi-LSTM	20
Gambar 2.7 Proses <i>Random oversampling</i>	22
Gambar 2.8 <i>Confusion matrix</i>	24
Gambar 2.9 Kurva AUC - ROC	26
Gambar 3.1 Diagram alur penelitian	31
Gambar 3.2 Diagram Alur Model CNN	32
Gambar 3.3 Diagram Alur Model LSTM dan Bi-LSTM	35
Gambar 4.1 Diagram Alur <i>Pre-Processing</i> pada Dataset.....	40
Gambar 4.2 Proses Seleksi Fitur.....	41
Gambar 4.3 Daftar Aplikasi Setelah dilakukan Filter Pada Data	42
Gambar 4.4 Data Hasil <i>Resampling</i> Setelah Diurutkan Kembali.....	45
Gambar 4.5 Data Hasil <i>Resampling</i>	46
Gambar 4.6 Gambaran Proses <i>Windowing</i> pada Data <i>Train</i> dan Data <i>Test</i>	47
Gambar 4.7 Arsitektur Model CNN	48
Gambar 4.8 Arsitektur Model LSTM.....	48
Gambar 4.9 Arsitektur Model Bi-LSTM.....	49
Gambar 4.10 Kurva Training Model CNN dengan Dataset <i>Original</i>	49
Gambar 4.11 Kurva Training Model CNN Dengan Dataset <i>Undersampling</i>	50
Gambar 4.12 Kurva Training Model CNN Dengan Dataset <i>Oversampling</i>	50
Gambar 4.13 Kurva Training Model LSTM Dengan Dataset <i>Original</i>	51
Gambar 4.14 Kurva Training Model LSTM Dengan Dataset <i>Undersampling</i> ...	52
Gambar 4.15 Kurva Training Model LSTM Dengan Dataset <i>Oversampling</i>	52
Gambar 4.16 Kurva Training Model Bi-LSTM dengan Dataset <i>Original</i>	53
Gambar 4.17 Kurva Training Model Bi-LSTM dengan Dataset <i>Undersampling</i>	53
Gambar 4.18 Kurva Training Model Bi-LSTM dengan Dataset <i>Oversampling</i> ..	54
Gambar 4.19 <i>Confusion matrix</i> Model CNN	56

Gambar 4.20 <i>Confusion matrix</i> Model LSTM	57
Gambar 4.21 <i>Confusion matrix</i> Model Bi-LSTM	58
Gambar 4. 22 Kurva ROC Model CNN, LSTM, dan Bi-LSTM.....	60
Gambar 4. 23 Diagram proses <i>Deployment Model</i>	62
Gambar 4.24 Hasil Implementasi <i>Deployment Model</i> Dalam Bentuk <i>Website</i> ...	63
Gambar 4.25 Hasil Pengujian <i>Website</i> Untuk Klasifikasi dan Evaluasi Kinerja Model	64

DAFTAR TABEL

Tabel 2.1 Contoh Kategori dan Aplikasi Layanan OTT	10
Tabel 2.2 Format Nomor <i>Port</i> yang ditetapkan oleh IANA.....	11
Tabel 2.3 <i>String</i> bit pada awalan P2P Protokol dari <i>Payload User</i>	12
Tabel 2.4 <i>Dataset Iris Spesies (UCI Machine Learning Repository)</i>	21
Tabel 3.1 Tabel Waktu dan Tahap Penelitian	29
Tabel 4.1 Deskripsi Variabel.....	36
Tabel 4.2 Variabel ' <i>dst_ip</i> ' Sebelum dan Sesudah <i>Feature Engineering</i>	40
Tabel 4.3 Variabel ' <i>flowStart</i> ' Sebelum dan Sesudah <i>Feature Engineering</i>	41
Tabel 4.4 Daftar Aplikasi Untuk Target Klasifikasi	42
Tabel 4.5 Data Sebelum dan Sesudah Dilakukan <i>Feature Encoding</i>	43
Tabel 4.6 Komposisi Pembagian Data	43
Tabel 4.7 Jumlah Sampel Data untuk Tiap Aplikasi.....	44
Tabel 4.8 Data Flow Trafik Sebelum dan Sesudah Normalisasi.....	46
Tabel 4.9 Tabel Perbandingan Akurasi dan <i>Loss</i> Model CNN	50
Tabel 4.10 Tabel Perbandingan Akurasi Model LSTM Terhadap Jumlah <i>Timestep</i>	51
Tabel 4.11 Tabel Perbandingan Akurasi Model Bi-LSTM Terhadap Jumlah <i>Timestep</i>	52
Tabel 4.12 Evaluasi Kinerja Model.....	55
Tabel 4.13 <i>Classification Report</i> Model CNN	56
Tabel 4.14 <i>Classification Report</i> Model LSTM.....	57
Tabel 4.15 <i>Classification Report</i> Model Bi-LSTM.....	59

DAFTAR RUMUS

2.1 CNN Output Width.....	17
2.2 CNN Output Height	17
2.3 CNN Output Depth	17
2.4 Forget Gate	18
2.5 Input Gate.....	18
2.6 Output Gate.....	18
2.7 Candidate Cell State	18
2.8 Cell State	18
2.9 Hidden State.....	18
2.10 Forward Hidden Sequence.....	20
2.11 Backward Hidden Sequence.....	20
2.12 Output Sequence.....	20
2.13 Min-Max Scaling Normalization.....	23
2.14 Akurasi	25
2.15 Recall.....	25
2.16 Presisi	26
2.17 F1-Score.....	26

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pesatnya perkembangan internet dan industri komunikasi saat ini telah berkontribusi pada skala dan kepadatan trafik jaringan yang semakin besar, dinamis, dan kompleks (Fadlullah, dkk., 2017). Sehubungan dengan hal itu, kebutuhan akan klasifikasi dan identifikasi trafik jaringan juga semakin meningkat seiring dengan munculnya berbagai layanan dan aplikasi baru.

Klasifikasi trafik jaringan adalah proses identifikasi setiap *flow* atau aliran pada data trafik jaringan kedalam aplikasi-aplikasi tertentu, berdasarkan fitur yang dimilikinya (Wang, dkk., 2017). Proses ini sangat penting untuk berbagai bidang terutama di bidang telekomunikasi dan keamanan jaringan. Dengan klasifikasi trafik yang akurat, berbagai aktifitas yang berhubungan dengan layanan jaringan seperti pemantauan, kontrol, dan optimalisasi dapat dilakukan dengan tujuan untuk meningkatkan kualitas dan keamanan jaringan (Wang, dkk., 2020).

Evolusi jaringan komunikasi seluler dari generasi pertama (1G) sampai generasi kelima (5G) turut serta mendorong perkembangan layanan media *online*, baik panggilan suara, pemesanan instan, *browsing*, dan lain-lain. Dari era *voice legacy* atau panggilan suara konvensional berbasis sirkuit hingga era VoLTE yang mendukung panggilan suara melalui jaringan LTE, kemajuan dalam bidang telekomunikasi ini memberikan banyak kemudahan bagi pengguna internet, terkhusus dengan lahirnya layanan *over-the-top* (OTT). OTT merupakan layanan media secara general yang disajikan langsung kepada pengguna dengan mengandalkan jaringan internet. Layanan OTT disampaikan melalui infrastruktur atau jaringan milik operator, tetapi tidak secara langsung melibatkan operator. Jenis-jenis layanan yang disediakan secara *over-the-top* ini dapat mencakup berbagai hal, salah satunya layanan VoIP (*Voice Over Internet Protocol*) yang memungkinkan pengguna untuk berkomunikasi dengan metode transmisi suara, pesan, atau media lainnya dengan menggunakan koneksi internet. Beberapa contoh

aplikasi yang menggunakan teknologi VoIP adalah *WhatsApp*, *Telegram*, *LINE*, *Zoom Meeting*, dan lain sebagainya.

Penyedia layanan internet merupakan salah satu industri yang memiliki minat yang mendalam untuk dapat mengidentifikasi trafik jaringan karena beberapa alasan, seperti untuk memantau tren suatu aplikasi dan penerapan kebijakan layanan tertentu yang bergantung pada kategori trafik jaringan, misalnya, memberikan kualitas layanan yang lebih baik untuk kategori VoIP, juga informasi mengenai aplikasi yang teridentifikasi dapat membantu dalam memahami *end-user* dengan lebih baik, sehingga mampu meningkatkan kualitas layanan dan prediksi pemasaran.

Pada sisi lain, tugas klasifikasi trafik jaringan yang akurat dipersulit dengan meningkatnya penggunaan protokol terenkripsi yang menjadi praktik standar dalam bidang keamanan jaringan saat ini. Prosedur enkripsi mengubah data asli menjadi formayant seperti pseudo-acak dengan tujuan untuk membuatnya sulit didekripsi. Akibatnya, data terenkripsi hampir tidak mengandung pola diskriminatif untuk mengidentifikasi trafik jaringan. Hal ini menyebabkan metode – metode konvensional yang telah lama digunakan untuk klasifikasi dan identifikasi trafik menjadi tidak efisien dan hampir tidak bisa digunakan lagi.

Deep Learning mampu melatih model klasifikasi dari data *input* dengan mempelajari representasi fitur secara otomatis (Aceto, dkk., 2018). *Convolutional Neural Network* (CNN) adalah jenis model *deep learning* di mana ekstraksi fitur dari data *input* dilakukan menggunakan lapisan yang terdiri dari operasi konvolusional (Anton, dkk., 2021). Arsitektur CNN terinspirasi oleh struktur visual organisme hidup. Dalam tugas klasifikasi trafik jaringan, CNN dapat menangkap ketergantungan spasial antara *byte* yang berdekatan dalam trafik *flow* yang mengarah pada penemuan pola diskriminatif untuk setiap kelas aplikasi (Lotfollahi, dkk., 2020).

Long Short-Term Memory (LSTM) merupakan tipe populer dari *Recurrent Neural Network* (RNN) mampu memodelkan perilaku temporal yang dinamis dengan ketergantungan jangka panjang (Vinayakumar, dkk., 2017). Unit LSTM bekerja dengan “mengingat” nilai selama interval waktu tertentu yang terdiri dari

cell, *input gate*, *output gate*, dan *forget gate*. Model LSTM memiliki kemampuan menghapus atau menambahkan informasi ke *cell state*, yang diatur oleh *gate*. *Gate* berfungsi menambah atau menghapus informasi sebelumnya, sehingga LSTM bekerja jauh lebih baik untuk sebagian besar tugas klasifikasi (Lim, dkk., 2019).

Bi-Directional Long Short-Term Memory (Bi-LSTM) merupakan varian dari model LSTM. Secara teknis, Bi-LSTM menerima *input* dari dua arah yang berlawanan, satu *input forward* dan satu *input backward* (Nugroho, dkk., 2021). *Output* dari jaringan ini digabungkan, akibatnya, jaringan memiliki informasi *forward* dan *backward* mengenai *sequence* pada setiap *time step*, sehingga mengurangi bias (Wang, dkk., 2017). Kemampuan model BiLSTM untuk menyimpan informasi dari masa depan (*future*) dan masa lalu (*past*) menjadikan model ini dapat mengerti konteks dengan lebih baik, khususnya untuk data yang bersifat kompleks.

Sistem informasi diterapkan dalam suatu organisasi dengan tujuan meningkatkan efektivitas dan efisiensi organisasi tersebut. Penelitian analisis perbandingan klasifikasi trafik jaringan aplikasi OTT dengan menggunakan beberapa algoritma *deep learning* seperti yang disebutkan di atas bertujuan untuk memberikan rekomendasi mengenai model arsitektur yang dapat digunakan untuk mengidentifikasi aplikasi dengan data trafik yang terenkripsi. Hasil analisis dari penelitian ini berguna bagi organisasi atau pemangku kepentingan untuk mengembangkan sistem yang telah ada sehingga tercapai optimalisasi strategi perusahaan khususnya dalam hal peningkatan layanan.

Berdasarkan latar belakang yang telah dijelaskan, peneliti mengajukan penelitian berjudul “**Analisis Perbandingan Klasifikasi Traffic Network Aplikasi Over-the-Top (OTT) dengan Algoritma Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), dan Bi-Directional Long Short-Term Memory (Bi-LSTM)**” dimana dilakukan analisis perbandingan algoritma CNN, LSTM, dan Bi-LSTM untuk mengklasifikasikan trafik jaringan terenkripsi untuk mengidentifikasi aplikasi dari masing-masing *flow* trafik. Model dievaluasi dengan menghitung *accuracy*, *precision*, *re-call*, dan *f1-score*.

1.2 Rumusan Masalah

Rumusan masalah dalam penelitian ini adalah:

1. Bagaimana cara kerja model CNN, LSTM, dan Bi-LSTM dalam mengklasifikasikan trafik jaringan?
2. Bagaimana mengaplikasikan teknik *resampling* pada *dataset* tidak seimbang untuk masalah klasifikasi trafik jaringan?
3. Bagaimana hasil evaluasi model CNN, LSTM, dan Bi-LSTM dalam mengklasifikasikan trafik jaringan?

1.3 Batasan Masalah

Batasan masalah dalam penelitian ini adalah:

1. Metode klasifikasi yang digunakan adalah CNN, LSTM, dan Bi-LSTM.
2. Data yang digunakan oleh peneliti merupakan *dataset* “*Labeled Network Traffic Flows*” dari *Kaggle*.
3. Target klasifikasi adalah aplikasi yang memiliki sampel data lebih dari 10.000 pada *dataset*.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah:

1. Menganalisis kinerja arsitektur model CNN, LSTN, dan Bi-LSTM dalam mengklasifikasikan trafik jaringan.
2. Menyajikan metode *resampling* untuk mengatasi masalah *dataset* tidak seimbang pada klasifikasi trafik jaringan.
3. Hasil analisis diharapkan menjadi masukan bagi pengambil kebijakan dalam menggunakan arsitektur terpilih untuk klasifikasi trafik jaringan.

BAB II

TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Beberapa penelitian telah mengaplikasikan berbagai teknik untuk melakukan klasifikasi pada trafik jaringan.

Penelitian yang dilakukan oleh Mohammad Lotfollahi melakukan pendekatan “*Deep Packet*” berbasis *deep learning* untuk klasifikasi trafik terenkripsi. Pendekatan ini mengintegrasikan fase ekstraksi fitur dan klasifikasi ke dalam satu system yang sama. Skema yang diusulkan dapat menangani karakteristik trafik di mana trafik jaringan dikategorikan ke dalam kelas utama (FTP dan P2P) dan identifikasi aplikasi (BitTorrent dan Skype). *Deep Packet* dapat mengidentifikasi trafik terenkripsi dan membedakan antara trafik jaringan VPN dan non-VPN. Framework ini menggunakan dua model *deep neural network*, yaitu *Stacked Autoencoder* (SAE) dan *Convolutional Neural Network* (CNN). Dalam tahap *pre-processing*, penelitian ini menggunakan teknik *undersampling* untuk menghilangkan kelas mayoritas secara random sampai kelas pada *dataset* relative seimbang. Pada akhir tahap, model CNN memperoleh hasil yang lebih baik untuk tugas identifikasi aplikasi dengan *F1-Score* sebesar 98% dibandingkan dengan SAE yang memperoleh hasil 95%. Sama halnya dengan tugas karakterisasi trafik, CNN mendapatkan *F1-Score* yang lebih baik sebesar 93% sementara SAE 92% (Lotfollahi, dkk., 2020).

Xin Wang, dkk melakukan eksplorasi secara empiris pada metode *deep learning* untuk tugas identifikasi aplikasi seluler dari alat penangkap trafik Android yang khusus dikembangkan untuk membuat *dataset* dengan *ground truth* yang sempurna. Tugas klasifikasi dilakukan dengan membandingkan tiga model, yaitu *Stacked Denoising Autoencoder* (SDAE), *Convolutional Neural Network* (CNN), dan *Bi-directional Long Short-Term Memory* (Bi-LSTM). *Dataset* terdiri dari data trafik telah di *pre-process* untuk menyertakan fitur statistik dari *flow* dan *payload* yang diperlukan oleh data untuk proses *training*. Fitur-fitur ini diambil dari tiga *sequence* pada *packet length* dalam *biflow* (*flow* dengan dua arah). Dari hasil evaluasi

sistematis, model 1D-CNN memiliki kinerja yang paling baik dengan akurasi 93,14% dan *Macro-F1 Score* 84,25% (Wang, dkk., 2020).

Zulu Okonkwo, dkk mengusulkan metode klasifikasi trafik jaringan terenkripsi dengan model CNN. Desain arsitektur CNN digunakan berdasar pada model LeNet-5 yang telah dimodifikasi dan dilatih pada kumpulan *dataset flow* trafik. Model ini terdiri dari sebelas *layer* terdiri dari sebelas lapisan yang terdiri dari empat lapisan konvolusi, dua lapisan *max pooling*, dua lapisan *dropout*, satu lapisan *flatten*, dan dua lapisan *dense*. Hasil ekstraksi fitur dari *packet size* dan *packet arrival time* digunakan untuk menghasilkan gambar untuk kelas-kelas aplikasi yang berbeda berdasarkan *flow* trafik. Gambar-gambar ini kemudian dilatih untuk tujuh masalah klasifikasi yang menggunakan tiga jenis enkripsi, yaitu HTTPS, VPN, dan TOR. Model yang diusulkan pada penelitian ini mencapai akurasi 91%-99% untuk tujuh jenis *dataset* berdasarkan metode enkripsinya (Okonkwo, dkk.m 2022).

Hyun-Kyo Lim dkk, dalam penelitiannya mengusulkan skema klasifikasi trafik menggunakan model *deep learning* untuk *software defined networks* (SDN). Klasifikasi dilakukan menggunakan *flow-based payload dataset* dengan dua model *deep learning* yaitu LSTM multi-layer dan kombinasi CNN dengan *single-layer* LSTM. Untuk menemukan *hyper-parameter* yang optimal dari kedua model ini, dijalankan prosedur *model tuning* dimana metode *grid search* digunakan untuk mencoba setiap kemungkinan kombinasi dari *hyper-parameter* berdasarkan *dataset*. Untuk proses validasi model, metode *k-fold cross validation* digunakan setelahnya. Dari penelitian ini, diperoleh akurasi keseluruhan model LSTM multi-layer lebih tinggi daripada model CNN+LSTM yaitu 99,65% untuk model LSTM dan 98,86% untuk model CNN+LSTM (Lim, dkk., 2019).

Penelitian Wei Wang dkk mengklasifikasikan trafik terenkripsi secara *end-to-end* dengan model CNN satu dimensi (1D-CNN). Metode ini mengintegrasikan proses ekstraksi fitur, seleksi fitur, dan pengklasifikasian ke dalam satu *framework* yang terpadu, sehingga model secara otomatis mempelajari hubungan *non-linear* antara data *input* mentah dan *output* yang diharapkan. Fitur-fitur trafik *flow* dipelajari *layer* demi *layer* oleh model 1D-CNN, dan fitur tingkat tinggi digunakan sebagai *input* dari lapisan softmax. Dari penelitian ini, model 1D-CNN

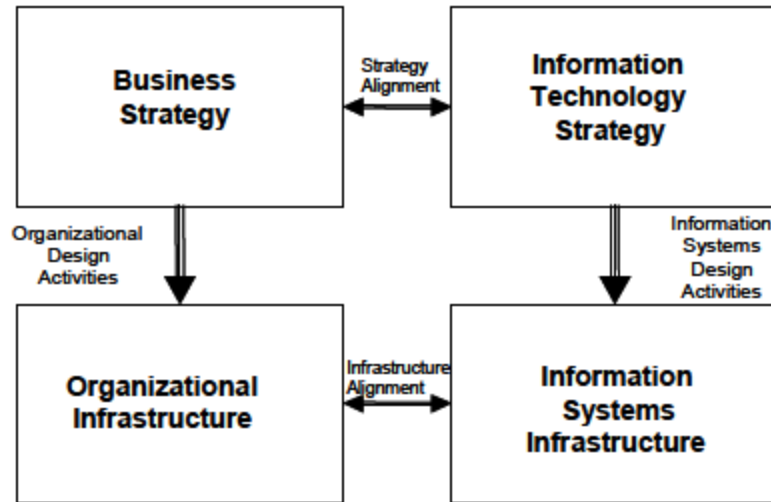
mengungguli model 2D-CNN dan *decision tree* C4.5. Di antara 12 kelas trafik terenkripsi, nilai *precision* 11 kelas 1D-CNN lebih tinggi dari 2D-CNN, dengan rata-rata 3,75% lebih tinggi (Wang, dkk, 2017).

2.2 Definisi Analisis

Analisis merupakan kegiatan menghimpun sejumlah data mentah kemudian mengelompokkan bagian-bagian yang relevan untuk dikaitkan hingga menjawab suatu permasalahan (Sansini, 2020). Menurut Kamus Besar Bahasa Indonesia (KBBI) (2008), analisis adalah penyelidikan terhadap suatu peristiwa, baik itu karangan, perbuatan, dan sebagainya, untuk mengetahui keadaan yang sebenarnya. Lebih lanjut lagi, KBBI mendefinisikan analisis merupakan penelaahan dan penguraian data hingga menghasilkan simpulan.

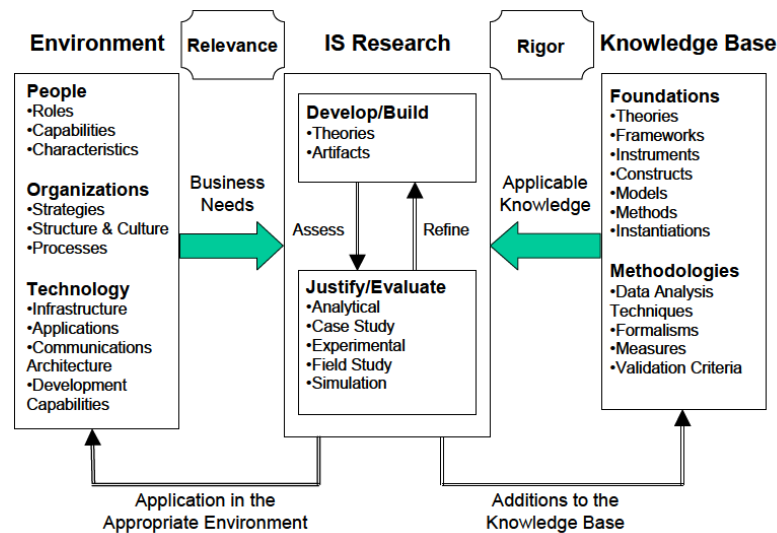
2.3 Kerangka Penelitian Sistem Informasi

Penerapan sistem informasi dalam suatu organisasi bertujuan untuk meningkatkan efektivitas dan efisiensi organisasi tersebut. Kemampuan sistem informasi dan karakteristik organisasi, sistem kerja, orang-orang yang terlibat, serta metodologi pengembangan dan penerapan dapat dijadikan tolak ukur sejauh mana tujuan tersebut tercapai (Silver, dkk., 1995). Gambar 2.1 menunjukkan keselarasan penting antara strategi bisnis dan teknologi informasi, juga antara infrastruktur organisasi dan sistem informasi. Hal-hal ini merupakan kegiatan desain yang saling bergantung dan merupakan kegiatan inti dari penelitian Sistem Informasi. Oleh karena itu, penelitian sistem informasi terdiri dari interaksi antara strategi bisnis, strategi teknologi informasi, infrastruktur organisasi, dan infrastruktur sistem informasi (Hevner, dkk., 2004).



Gambar 2.1 Desain Organisasi dan Desain Kegiatan Sistem Informasi

Gambar 2.2 menyajikan kerangka kerja konseptual untuk proses memahami (*understanding*), melaksanakan (*executing*), dan mengevaluasi (*evaluating*) penelitian sistem informasi yang menggabungkan paradigma *behavioral-science* dan *design-science* (Hevner, dkk., 2004).



Gambar 2.2 Kerangka Penelitian Sistem Informasi

Lingkungan (*environment*) mendefinisikan ruang masalah di mana fenomena tersebut berada. Dalam hal ini, penelitian sistem informasi terdiri dari orang-orang (*people*), bisnis atau organisasi, dan teknologi yang ada atau telah direncanakan (Silver, dkk., 1995). Basis pengetahuan (*knowledge base*) menyediakan bahan baku

dari dan melalui mana penelitian sistem informasi diselesaikan. Tahap basis pengetahuan terdiri dari fondasi dan metodologi. Salah satu elemen dalam fondasi adalah metode. Metode memberikan panduan tentang cara pemecahan masalah, yang dapat mencakup algoritma matematis yang secara eksplisit mendefinisikan proses pencarian atau pemecahan masalah, hingga deskripsi informal dan tekstual yang terdiri dari pendekatan “praktik terbaik”, atau beberapa kombinasi (Hevner, dkk., 2004).

Kontribusi *behavioral-science* dan *design-science* dalam penelitian sistem informasi turut dipertimbangkan karena penerapannya pada kebutuhan bisnis di lingkungan (*environment*) yang sesuai. Kontribusi ini juga bermanfaat untuk penambahan konten dalam basis pengetahuan untuk penelitian dan praktik lebih lanjut.

2.4 Layanan OTT

Over-the-top (OTT) mengacu pada layanan yang mengirimkan audio, video, dan media lainnya melalui internet dengan memanfaatkan infrastruktur yang digunakan oleh operator jaringan tanpa keterlibatan mereka dalam kontrol atau distribusi konten (Rojas, dkk., 2019). Layanan ini dapat berupa audio, video, *network service*, *instant messaging*, *file sharing*, *game*, *streaming*, dan lainnya. Layanan OTT banyak digunakan oleh masyarakat karena memiliki banyak keuntungan. Contohnya layanan komunikasi berbasis OTT seperti *WhatsApp* memungkinkan pengguna untuk melakukan komunikasi tanpa harus mengeluarkan biaya tambahan. Keuntungan ini menyebabkan membanjirnya layanan OTT yang juga berimbas pada terus melesatnya trafik data yang lalu lalang melalui jaringan operator.

Dalam bidang telekomunikasi, layanan OTT diklasifikasikan berdasarkan jenis dan kategori layanan yang ditawarkan. Hal ini bertujuan untuk memudahkan proses analisis dan penarikan kesimpulan dalam rangka optimalisasi kualitas layanan. Salah satu contoh kategori aplikasi adalah video yang khusus menganalisis data trafik aplikasi yang menampilkan konten video atau gambar bergerak. Aplikasi yang termasuk dalam kategori video antara lain *Youtube*, *Netflix*, *Vimeo*, *Twitch*, dan lain-lain. Contoh lainnya adalah kategori aplikasi VoIP (*Voice over Internet*

Protocol) yaitu layanan yang memungkinkan pengguna untuk melakukan panggilan suara menggunakan koneksi internet *broadband* alih-alih menggunakan saluran telpon biasa atau analog. Aplikasi yang mendukung layanan VoIP ini adalah *WhatsApp*, *Skype*, dan *GoogleHangoutDuo*, dan lain-lain. Berikut merupakan tabel contoh aplikasi dan kategori layanan OTT.

Tabel 2.1 Contoh Kategori dan Aplikasi Layanan OTT

No.	Kategori	Aplikasi
1	<i>Cloud</i>	<i>ApplePush, Dropbox, GoogleDrive, MS_OneDrive</i>
2	<i>Video</i>	<i>Youtube, Netflix, AmazonVideo</i>
3	<i>Social Network</i>	<i>Facebook, Instagram, LinkedIn, Twitter</i>
4	<i>Instant Message</i>	<i>Messenger, QQ, Signal, WhatsApp</i>
5	<i>VoIP</i>	<i>GoogleHangoutDuo, Skype, WhatsAppCall</i>
6	<i>Web</i>	<i>Cloudflare, GoogleServices, Wikipedia, Yahoo, Amazon</i>
7	<i>SoftwareUpdate</i>	<i>WindowsUpdate, PlayStore</i>
8	<i>Collaborative</i>	<i>Github, GoogleDocs, Office365, Slack</i>
9	<i>Email</i>	<i>Gmail, IMAPS</i>

2.5 Klasifikasi Trafik Jaringan

Klasifikasi trafik jaringan merupakan tugas yang sangat penting untuk berbagai bidang, utamanya bagi penyedia layanan internet. Tugas ini berguna untuk mengetahui *flow* atau alur aplikasi dalam suatu jaringan. Klasifikasi trafik jaringan adalah langkah pertama untuk menganalisis dan mengidentifikasi berbagai jenis aplikasi. Dengan teknik ini, penyedia layanan internet atau operator jaringan dapat mengatur kinerja dan kualitas jaringan dan layanan internet secara keseluruhan (Shafiq, dkk., 2016). Misalnya, dalam bidang manajemen jaringan, trafik dapat diklasifikasikan berdasarkan prioritas yang berbeda untuk menjamin kualitas layanan jaringan. Sementara di bidang keamanan siber, trafik dapat diklasifikasikan menjadi *benign traffic* atau *malware traffic* untuk tujuan deteksi anomaly jaringan (Wang, dkk., 2017).

Seiring dengan kebutuhan klasifikasi trafik jaringan, meningkatnya permintaan pengguna akan keamanan privasi dan enkripsi data turut serta meningkatkan jumlah trafik terenkripsi pada internet (Velan, dkk., 2015). Tren *encrypted traffic* atau trafik yang terenkripsi menjadi praktik standar dalam bidang keamanan jaringan saat ini. Praktik tersebut melahirkan tantangan baru untuk proses klasifikasi dan identifikasi trafik jaringan konvensional. Terlebih lagi, pesatnya perkembangan internet dan perangkat komunikasi menyebabkan struktur *flow* trafik jaringan yang lebih besar dan lebih rumit (Mohammed, dkk., 2019). Kompleksitas dalam jaringan ini menghasilkan limpahan data trafik yang berjumlah besar dan menimbulkan tantangan baru di sisi manajemen jaringan dan optimasi trafik, termasuk didalamnya klasifikasi trafik jaringan.

Dalam dua dekade terakhir, para peneliti telah mengusulkan beberapa metode untuk klasifikasi trafik jaringan. Diantaranya adalah:

a. *Port-Based Technique*

Port-Based Technique adalah teknik klasifikasi jaringan berbasis nomor *port* yang terdaftar dalam *Internet Assigned Number Authority* (IANA). Tabel 2.2 menampilkan beberapa jenis aplikasi dan nomor *port*nya pada IANA (Shafiq, dkk., 2016).

Tabel 2.2 Format Nomor *Port* yang ditetapkan oleh IANA

Nomor <i>Port</i>	Aplikasi
20	FTTP Data
21	FTTP
22	SSH
23	Telnet
25	SMTP
53	DNS
80	HTTP
110	POP3
123	NTP
161	SNMP

Nomor Port	Aplikasi
3724	WoW

Sayangnya, teknik ini mulai tidak bisa digunakan karena bertambahnya aplikasi *Peer-to-Peer* (P2P), yang menggunakan *port* dinamis. *Port* dinamis ini tidak terdaftar pada IANA. Selain itu, semakin banyak aplikasi cenderung menggunakan teknik pengaburan atau *port obfuscation* agar tidak terdeteksi.

b. *Deep Packet Inspection*

Untuk mengatasi kekurangan pada teknik *port-based*, teknologi *Deep Packet Inspection* dikembangkan. Metode ini tidak terpengaruh oleh *port obfuscation* dan bisa mendapatkan akurasi yang sangat baik dengan menganalisis pola internal dari data *payload* (Liao dkk., 2019).

Menurut Karagiannis dkk (2004), teknik *Deep Packet Inspection* didasarkan pada identifikasi string bit tertentu dalam data *user* pada aplikasi. Teknik ini dilakukan dengan memantau trafik TCP dan UPD menggunakan aplikasi “*tcpdump*”. Tabel 2.3 menampilkan subset dari string untuk beberapa protokol yang dianalisis untuk TCP dan UDP.

Tabel 2.3 String bit pada awalan P2P Protokol dari *Payload User*

Protokol P2P	String	Trans. Protocol	Port Teridentifikasi
<i>eDonkey2000</i>	0xe319010000	TCP/UDP	4661-4665
	0xc53f010000		
<i>Fasttrack</i>	“Get /.hash”	TCP	1214
	0x270000002980	UDP	
<i>BitTorrent</i>	“0x13Bit”	TCP	6881-6889
<i>Gnutella</i>	“GNUT”, “GIV”	TCP	6346-6347
	“GND”	UDP	
MP2P	GO!!, MD5, SIZ0x20	TCP	41170 UDP

Protokol P2P	String	Trans. Protocol	Port Teridentifikasi
<i>Direct Connect</i>	“\$MyN”, “\$Dir”	TCP	411-412
	“\$SR”	UDP	
<i>Ares</i>	“GET hash:”	TCP	-
	“Get sha1:”		

Teknik ini mengklasifikasikan *packet* ke dalam *flow* dengan menggunakan 5 tuple yaitu *source IP*, *destination IP*, protokol, *source port*, dan *destination port*. Terdapat tiga metode yang digunakan untuk mengestimasi trafik P2P, yaitu

- 1) Jika *source* atau *destination port* cocok dengan salah satu nomor *port* teridentifikasi (Tabel 2.3), maka *flow* ditandai sebagai P2P.
- 2) Membandingkan *payload* dari setiap paket dalam *flow* dengan string pada Tabel 2.3. Jika ada kecocokan antara *payload* 16-byte dari sebuah paket dengan salah satu string, maka *flow* ditandai sebagai P2P dengan protokol yang sesuai, misalnya, Fasttrack, eDonkey, dll. Jika tidak ada paket yang cocok, maka *flow* ditandai sebagai non-P2P.
- 3) Jika *flow* ditandai sebagai P2P, maka *source* dan *destination IP address* dari *flow* di-hash ke dalam table. Semua *flow* yang berisi alamat IP *address* dalam table ditandai sebagai “kemungkinan P2P” meski tidak ada kecocokan *payload*.

Namun, meningkatnya penggunaan protokol terenkripsi (TLS) serta terjemahan alamat jaringan (NAT) dan *port* dinamis, membuat teknik DPI semakin tidak relevan untuk digunakan saat ini (Aceto, dkk., 2018). Teknik DPI juga gagal dalam hal *high speed network*, karena jaringan ini merutekan dan meneruskan paket terlalu cepat untuk dapat diperiksa oleh teknik DPI. Teknik ini juga tidak dapat mengenali aplikasi tanpa *signature*, yang artinya teknik ini bukan metode yang “cerdas” (Peng, dkk., 2017). Lebih jauh lagi, klasifikasi trafik dengan teknik DPI

membutuhkan *update* secara berkelanjutan dari pola *signature* aplikasi-aplikasi baru.

c. *Statistical-based classification*

Metode klasifikasi berbasis statistik mengklasifikasikan trafik menggunakan fitur statistik trafik jaringan alih-alih menggunakan data *payload*. Diantara fitur – fitur tersebut adalah nilai minimum, maksimum, dan *mean* dari paket, jumlah paket per detik, dan lain sebagainya. Metode-metode yang berbeda digunakan untuk melakukan tugas klasifikasi pada *dataset* dengan fitur yang berbeda. Pencarian subset fitur terbaik dengan mencoba beberapa metode *feature extraction* dan *feature selection* juga dengan melatih data dengan model-model klasifikasi *machine learning* dapat membantu meningkatkan akurasi metode klasifikasi berbasis statistik.

Kekurangan dari metode ini adalah *feature redundancy* dimana terlalu banyak fitur duplikat atau fitur berkorelasi yang memperlambat proses *training*, mengurangi kemampuan estimasi fitur, dan penafsiran oleh model (Zhao, 2021).

d. *Correlation-based Classification*

Metode klasifikasi berbasis korelasi melakukan agregasi paket ke dalam *flow* dan mengklasifikasikannya sesuai dengan korelasi antara *flow* dalam jaringan (Zhang, dkk., 2012). *Flow* yang dimaksud pada metode ini adalah kumpulan paket data yang memiliki 5 tuple yang sama, yaitu *source IP*, *source port*, *destination IP*, *destination port*, dan protokol yang sama. *Flow-flow* tersebut digabungkan ke dalam *Bag-of-Flow* (BoF). Metode ini mengatasi masalah *feature redundancy* yang dimiliki oleh metode berbasis statistik, tetapi masih memiliki beban komputasi yang sangat tinggi saat proses *feature matching*.

e. *Behavioral-based Classification*

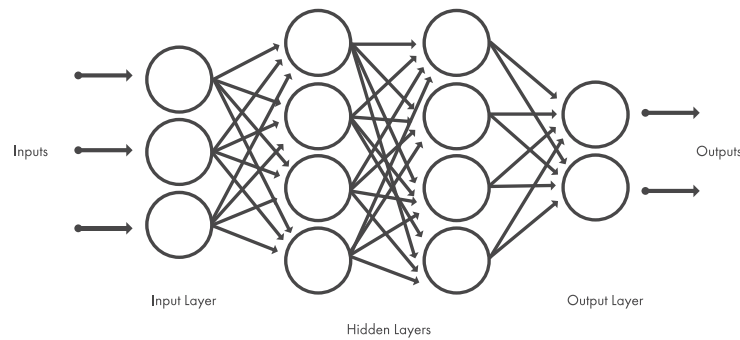
Metode berbasis perilaku melakukan klasifikasi dengan memeriksa dan menghitung perilaku *host*, seperti, *IP address* mana yang berkomunikasi dengan *host*, protokol apa yang digunakan, dan *port* mana yang saling berinteraksi (Iliofotou, dkk., 2009). Karena memerhatikan interaksi antar

host, flow yang digunakan oleh metode ini adalah *flow* dengan dua arah, kecuali untuk menganalisis trafik *attack* yang tidak normal karena jenis analisis ini umumnya menggunakan *flow* satu arah. Metode klasifikasi berbasis perilaku menggunakan konsep grafik untuk merepresentasikan pola perilaku antar host. Kekurangan dari metode ini adalah karena hanya mengandalkan informasi dari *packet header*, maka perincian klasifikasi tidak cukup baik. Hal tersebut menyebabkan klasifikasi dengan metode ini hanya mampu mengklasifikasikan trafik pada kategori *layer* level 4 (TCP, UDP, dan ICMP).

2.6 Deep Learning

Machine Learning merupakan suatu bidang penelitian yang menggabungkan statistik, kecerdasan buatan, dan ilmu komputer, dan juga dikenal sebagai proses analisis prediktif atau pembelajaran statistik (Muller & Guido, 2017). *Deep learning* merupakan salah satu jenis *machine learning* yang merupakan bagian dari kecerdasan buatan. *Deep learning* didefinisikan sebagai penggunaan *deep network* yang saling berhubungan dengan tujuan untuk menghitung algoritma yang secara bergantian menggunakan beberapa *layer* untuk menghasilkan suatu *output* (Wang & Dong, 2016). Dalam proses ini, jaringan dengan tingkat berikutnya menggunakan hasil pemrosesan dari fase sebelum sebagai *input* agar menghasilkan *output*.

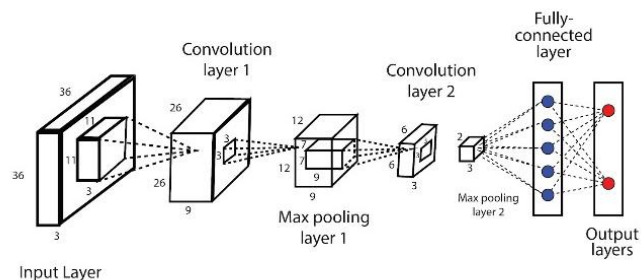
Sebagian besar model *deep learning* menggunakan arsitektur jaringan saraf tiruan. Oleh karena itu model-model ini sering disebut *deep neural network*. Istilah “*deep*” umumnya mengacu pada jumlah lapisan tersembunyi yang terdapat dalam jaringan saraf, Jaringan saraf tradisional dapat berisi sekitar 2-3 lapisan tersembunyi, sedangkan *deep network* dapat memiliki sebanyak 100-200 lapisan tersembunyi. Gambar 2.3 mengilustrasikan jaringan saraf yang tersusun dalam lapisan-lapisan (*layers*) yang terdiri dari sekumpulan *node* yang saling berhubungan. Model *deep learning* terinspirasi dari kedalaman struktur otak manusia yang belajar dari konsep level yang rendah ke level yang lebih tinggi (Sarker, 2021).



Gambar 2.3 Ilustrasi Jaringan Saraf

2.6.1 Convolutional Neural Network

Convolutional Neural Network atau jaringan saraf konvolusional adalah jenis jaringan saraf khusus untuk memproses data yang memiliki topologi jala atau grid-like topology (Goodfellow, dkk., 2019). CNN mengekstraksi fitur dari *input* data menggunakan lapisan yang terdiri dari operasi konvolusional (Lotfollahi, dkk., 2020). Arsitektur CNN terinspirasi dari korteks visual binatang, yaitu bagian pada otak yang bertugas untuk memroses informasi dalam bentuk visual. Arsitektur CNN ditunjukkan pada Gambar 2.4



Gambar 2.4 Arsitektur CNN

CNN terdiri dari beberapa lapisan konvolusi, lapisan *pooling*, dan *fully connected layer*. Dalam lapisan konvolusi, satu set kernel dengan ukuran kecil dengan sejumlah parameter digunakan untuk menangkap pola *output* dari *layer* sebelumnya. Untuk menghasilkan *output*, lapisan konvolusi menggunakan *set* kernel yang sama pada seluruh *input*. Dengan menggunakan *set* kernel yang sama dalam satu lapisan, jumlah parameter berkurang secara drastis. Penggunaan kernel ini pada seluruh *layer input* membantu model untuk menangkap fitur *shift invarian*

dengan lebih mudah. Lapisan lain yang biasanya digunakan dalam model CNN adalah lapisan *pooling* yang berfungsi untuk subsampling. Pada akhir set konvolusi dan *pooling layer*, satu set *fully connected layer* biasanya digunakan untuk menangkap fitur *input* tingkat tinggi (Rezaei & Liu, 2020).

Cara kerja dari lapisan konvolusi adalah dengan menerima *input* berupa ukuran *volume* $W_1 \times H_1 \times D_1$, dimana parameter yang diperlukan adalah jumlah filter K , tingkat spasial F , *stride* S , dan jumlah *zero padding* P . Selanjutnya, lapisan konvolusi menghasilkan *volume* dengan ukuran $W_2 \times H_2 \times D_2$, dimana:

$$W_2 = \frac{W_1 - F + 2P}{S + 1} \quad (2.1)$$

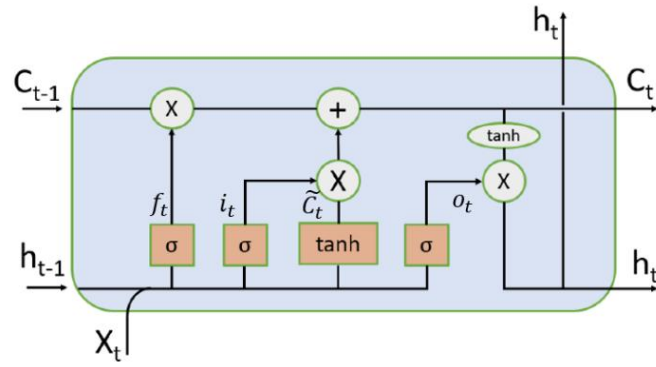
$$H_2 = \frac{H_1 - F + 2P}{S + 1} \quad (2.2)$$

$$D_2 = K \quad (2.3)$$

Dengan *parameter sharing*, lapisan konvolusi menghasilkan $F \times F \times D_1$ bobot per filter untuk total $(F \times F \times D_1) \times K$ bobot dan K bias. Pada *volume output*, irisan dengan kedalaman ke- d (dengan ukuran $W_2 \times H_2$) adalah hasil dari melakukan konvolusi yang valid dari filter ke- d pada *volume input* dengan *stride* S kemudian diimbangi dengan bias ke- d .

2.6.2 Long-Short-Term Memory

Long-Short-Term Memory merupakan salah satu jenis *Recurrent Neural Network* (RNN) yang umumnya digunakan untuk mempelajari, memproses, dan mengklasifikasikan data berurutan karena dapat mempelajari ketergantungan jangka panjang antar data. Model ini pertama kali dikembangkan oleh Hochreiter dan Schmidhuber pada tahun 1997, dimana setiap *node* berulang digantikan oleh *memory cell*. Setiap *memory cell* berisi *internal state*, yaitu sebuah *node* dengan *recurrent edge* yang terhubung sendiri dengan bobot tetap 1, untuk memastikan bahwa gradien dapat melewati banyak *timesteps* tanpa menghilang (Zhang, dkk., 2021).



Gambar 2.5 Arsitektur LSTM

Arsitektur LSTM dapat dilihat pada Gambar 2.5. Cara kerja algoritma ini adalah sebagai berikut. Pada langkah t , terdapat *hidden state* h_t dan *cell state* c_t . Baik h_t dan c_t adalah vektor berukuran n . LSTM dapat membaca, menghapus, dan menulis informasi ke dan dari sel c_t . Cara LSTM mengubah sel c_t adalah melalui tiga gerbang khusus, yaitu *input gate* i , *forget gate* f , dan *output gate* o . Nilai gerbang ini bervariasi dari tertutup (0) hingga terbuka (1). Semua gerbang i , f , dan o adalah vektor berukuran n .

Pada setiap *timestep* terdapat vektor masukan x_t , *hidden state* sebelumnya h_{t-1} , *state* sebelumnya c_{t-1} , dan LSTM yang menghitung *hidden state* berikutnya h_t , dan *cell state* berikutnya c_t pada *timestep* t :

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (2.4)$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (2.5)$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (2.6)$$

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (2.7)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (2.8)$$

$$h_t = o_t \odot \tanh(c_t) \quad (2.9)$$

dimana \odot adalah produk Hadamard *element-wise*. g_t pada rumus di atas adalah *cache* perhitungan perantara yang nantinya digunakan dengan *gate* o .

Karena semua nilai vektor pada gerbang \mathbf{i} , \mathbf{f} , dan \mathbf{o} berkisar dari 0 hingga 1 oleh fungsi sigmoid σ , ketika dikalikan dengan elemen, dapat dilihat bahwa:

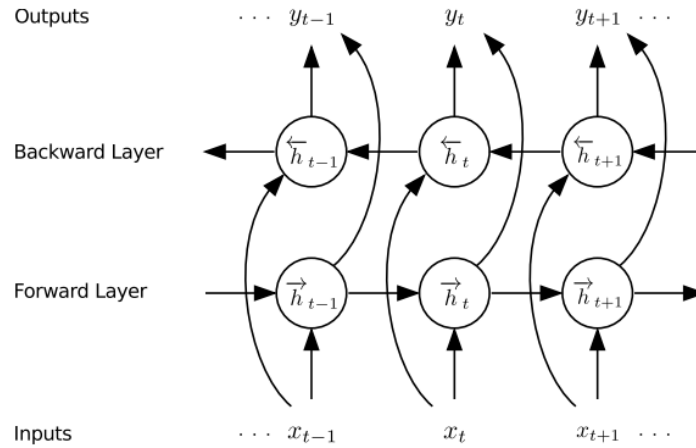
- a. *Forget gate* f_t pada *timestep* t mengontrol berapa banyak informasi yang perlu dihapus dari *cell state* sebelumnya c_{t-1} . *Forget gate* ini belajar untuk menghapus *hidden state* dari *timestep* sebelumnya, yang menyebabkan LSTM memiliki dua *hidden state* h_t dan *cell state* c_t . c_t akan disebarkan dari waktu ke waktu dan mempelajari apakah akan melupakan *cell state* sebelumnya atau tidak.
- b. *Input gate* pada *timestep* t mengontrol berapa banyak informasi yang perlu ditambahkan ke *cell state* berikutnya c_t dari *hidden state* sebelumnya h_{t-1} dan *input* x_t . *Input gate* i memiliki fungsi sigmoid yang mengubah data *input* menjadi nilai antara nol dan satu. *Input gate* ini memutuskan apakah akan mengambil *output* RNN yang dihasilkan oleh *gate* g dan mengalikan *output* dengan *input gate* i .
- c. *Output gate* o_t pada *timestep* t mengontrol berapa banyak informasi yang perlu ditampilkan sebagai *output* dalam *hidden state* h_t saat ini (Karpathy, 2016).

Model LSTM dapat memproses data sekuensial dengan melintasi semua elemen urutan pada data menggunakan *timesteps* dan melintasi informasi terkait konten data untuk memberbarui memori, dimana fitur yang terakhir diletakkan di atas fitur yang sebelumnya. Hubungan sekuensial temporal ini digunakan untuk menangkap karakteristik urutan fitur dengan lebih baik. Keadaan atau *state* dari *recurrent neural network* ini akan di-*reset* saat pemrosesan antara dua *independent sequence* yang berbeda. Dibandingkan dengan RNN, LSTM melangkah lebih jauh dengan memecahkan masalah hilangnya gradien pada RNN dan dapat membawa sebagian informasi untuk ditransmisikan ke *timestep* selanjutnya dan digunakan langsung saat dibutuhkan (Guo, dkk., 2021).

2.6.3 Bidirectional LSTM

Bidirectional Long Short – Term Memory (Bi-LSTM) adalah tipe RNN yang dirancang khusus untuk mengatasi keterbatasan yang terdapat pada model RNN dan

merupakan perkembangan dari LSTM. Neural *state* pada Bi-LSTM dibagi menjadi dua, yaitu *forward state* dan *backward state*, yang menghasilkan dua *forward RNN* dan *backward RNN* yang berbeda (Ojha, dkk., 2021). Dengan menggabungkan *output* dari dua RNN yang menyampaikan informasi dari arah yang berlawanan, dimungkinkan untuk menangkap konteks dari kedua ujung *sequence* tersebut (Cornegruta, dkk., 2016).



Gambar 2.6 Arsitektur Bi-LSTM

Seperti yang diilustrasikan pada Gambar 2.6, Bi-LSTM menghitung *forward hidden sequence* \vec{h} , *backward hidden sequence* \overleftarrow{h} , dan *output sequence* y dengan melakukan iterasi pada *backward layer* dari $t = T$ ke 1, *forward layer* dari $t = 1$ ke T lalu memperbarui lapisan *output* (Graves, dkk., 2013).

$$\vec{h}_t = \mathcal{H}(W_{x\vec{h}}x_t + W_{\vec{h}\vec{h}}\vec{h}_{t-1} + b_{\vec{h}}) \quad (2.10)$$

$$\overleftarrow{h}_t = \mathcal{H}(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t-1} + b_{\overleftarrow{h}}) \quad (2.11)$$

$$y_t = W_{\vec{h}y}\vec{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t + b_y \quad (2.12)$$

Pada dasarnya, Bi-LSTM mengikuti proses di mana neuron dari RNN normal dipecah menjadi dua arah. Satu untuk *backward state* atau arah waktu negatif, dan satu lagi untuk *forward state* atau arah waktu positif. *Input* dari *reverse direction state* tidak terhubung ke hasil kedua *state* ini. Dengan memanfaatkan dua arah waktu, *input* data dari masa lalu dan masa depan dari time frame saat ini dapat

digunakan. Hal ini berbeda dengan RNN standar yang membutuhkan delay untuk memasukkan data masa depan (Sharfuddin, dkk., 2018).

2.7 Klasifikasi Multikelas

Klasifikasi dalam *machine learning* adalah masalah pemodelan prediktif di mana label kelas diprediksi dari data *input* tertentu. Klasifikasi multikelas merupakan tugas klasifikasi yang melibatkan lebih dari dua kelas *output*. Klasifikasi multikelas membuat asumsi bahwa setiap sampel ditugaskan ke satu dan hanya satu label, misalnya buah dapat berupa apel atau pir tetapi tidak keduanya sekaligus. Klasifikasi multikelas berbeda dengan klasifikasi multi-label, di mana terdapat banyak label yang diprediksi untuk setiap data *input* (Brownlee, 2021).

Data *input* untuk tugas klasifikasi merupakan koleksi dari *record*. Tiap data *input* ini disebut sebagai *instance* yang ditentukan oleh *tuple* (x, y) , dimana x adalah atribut dari data dan y adalah kelas label. Tabel 2.4 menampilkan contoh *data set* yang digunakan untuk klasifikasi multikelas pada *dataset* Iris.

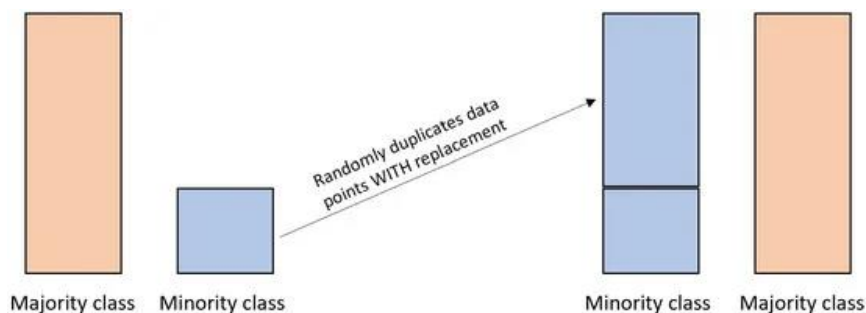
Tabel 2.4 *Dataset* Iris Spesies (UCI Machine Learning Repository)

<i>Sepal Length</i>	<i>Sepal Width</i>	<i>Petal Length</i>	<i>Petal Width</i>	<i>Spesies</i>
5.0	3.6	1.4	0.2	<i>Iris-setosa</i>
5.8	2.7	5.1	1.9	<i>Iris-virginica</i>
4.8	3.0	1.4	0.1	<i>Iris-setosa</i>
5.7	2.9	4.3	1.3	<i>Iris-versicolor</i>
5.8	2.6	4.0	1.2	<i>Iris-versicolor</i>
6.7	3.3	5.7	2.1	<i>Iris-virginica</i>
5.0	3.0	1.6	0.2	<i>Iris-setosa</i>
6.1	2.6	5.6	1.4	<i>Iris-virginica</i>
6.0	3.0	4.8	1.8	<i>Iris-virginica</i>
6.0	2.7	5.1	1.6	<i>Iris-versicolor</i>
4.9	3.1	1.5	0.1	<i>Iris-setosa</i>
5.5	2.4	3.7	1.0	<i>Iris-versicolor</i>
4.5	2.3	1.3	0.3	<i>Iris-setosa</i>

2.8 Random Oversampling

Dataset tidak seimbang merupakan suatu permasalahan umum pada tugas klasifikasi *machine learning*. Hal ini membuat hasil akurasi yang kurang efektif karena penyebaran data yang tidak merata. *Random Over Sampling* (ROS) merupakan salah satu metode untuk mengatasi *dataset* yang tidak seimbang dengan melakukan *oversampling*, dimana kelas minoritas pada data latih diduplikasi hingga menghasilkan sampel data baru.

Random Oversampling menghasilkan sampel data baru dengan cara menduplikasi secara acak titik data dari kelas minoritas dengan penggantian (*replacement*) hingga proporsi kedua kelas tersebut seimbang (Wongvorachan, dkk., 2023). Dalam hal klasifikasi trafik jaringan, algoritma *random oversampling* bekerja dengan mereplikasi data aplikasi yang termasuk dalam kelas minoritas secara acak sampai jumlahnya sesuai dengan data aplikasi kelas mayoritas. Ilustrasi dari algoritma *random oversampling* dapat dilihat pada Gambar 2.7.



Gambar 2.7 Proses *Random oversampling*

2.9 Random Undersampling

Undersampling adalah proses mengurangi jumlah contoh atau sampel target kelas mayoritas pada *dataset* tidak seimbang (Mohammed, dkk., 2020). Teknik ini menyebabkan jumlah data yang ada pada kelas mayoritas menjadi berkurang karena disesuaikan dengan sebaran jumlah kelas minoritas. *Undersampling* adalah metode yang cukup cepat ketika digunakan untuk data yang tidak seimbang karena banyak sampel kelas mayoritas yang diabaikan. Kelemahan dari metode ini adalah

beberapa informasi berguna bisa jadi berada di dalam sampel yang diabaikan tersebut (Liu, dkk., 2008).

Salah satu metode yang banyak digunakan untuk teknik *undersampling* adalah *Random Under Sampling*. Cara kerja dari metode ini adalah dengan menghilangkan sampel dari kelas mayoritas secara acak sehingga meratakan jumlah sampel dengan kelas minoritas (Ma & He, 2013).

Teknik ini dimulai dengan menentukan *input* $(x_i, y_i), \dots, (x_n, y_n)$ dimana $y_i \in \{-1, +1\}$. Algoritma *Random Undersampling* pertama-tama bekerja dengan mengidentifikasi kelas negatif / kelas mayoritas ($y_i = -1$) dan kelas positif / kelas minoritas ($y_i = 1$). Lalu algoritma menghitung jumlah member dari setiap kelas ($y_i = -1$) dan kelas ($y_i = 1$). Jika panjang ($y_i = -1$) lebih dari panjang dari ($y_i = 1$), maka member dari kelas x akan dihilangkan secara acak. *Output* dari algoritma ini adalah: jumlah kelas ($y_i = -1$) sama dengan kelas ($y_i = 1$) dan panjang ($y_i = -1$) sama dengan ($y_i = 1$) (Rahmi, 2020).

2.10 Normalisasi Data

Normalisasi adalah proses mengubah skala nilai pada variabel dalam *dataset* berada pada nilai rentang yang sama. Tujuan dari normalisasi data adalah untuk menghasilkan data baru dengan nilai yang lebih kecil, mewalikili data asli tanpa kehilangan karakteristiknya (Aziz, 2022). Normalisasi data berguna untuk mempercepat proses *training*. Metode normalisasi data yang digunakan dalam penelitian ini adalah *Min-Max Scaling Normalization*. Persamaan umum yang digunakan untuk normalisasi data dalam rentang $[0, 1]$ dapat dilihat pada persamaan (2.13).

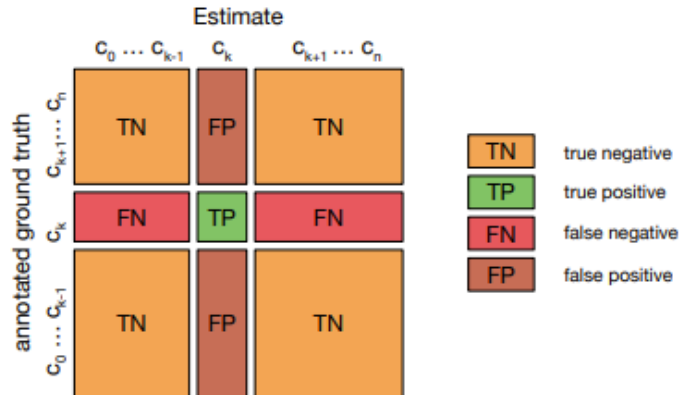
$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (2.13)$$

dengan:

- x = Nilai data aktual
- x' = Nilai normalisasi
- x_{max} = Nilai maksimum data aktual
- x_{min} = Nilai minimum data aktual

2.11 Evaluasi Kinerja Model

Dalam pengukuran model klasifikasi, digunakan konsep *confusion matrix* untuk menghasilkan skor dari hasil prediksi model terhadap data uji. Gambar 2.8 menampilkan *confusion matrix* untuk klasifikasi multikelas.



Gambar 2.8 *Confusion matrix*

Confusion matrix menampilkan jumlah klasifikasi untuk setiap kelas. Poin estimasi dikumpulkan ke dalam *confusion matrix* $C := (c_{ij})$, dimana c_{ij} adalah jumlah *timestep* di mana kelas sebenarnya adalah i dan kelas j telah diestimasi. *Confusion matrix* memberikan empat jenis hasil klasifikasi sehubungan dengan satu target klasifikasi k .

- True Positive* (TP), jumlah data positif yang diestimasi dengan benar (c_{kk}).
- True Negative* (TN), jumlah data negative yang diestimasi dengan benar ($\sum_{i,j \in N \setminus \{k\}} c_{ij}$).
- False Positive* (FP), jumlah data negative yang diestimasi sebagai data positif ($\sum_{i,j \in N \setminus \{k\}} c_{ik}$).
- False Negative* (FN), jumlah data positif yang diestimasi sebagai data negatif ($\sum_{i,j \in N \setminus \{k\}} c_{ki}$).

Dua hasil pertama mewakili klasifikasi yang benar, sedangkan dua terakhir mewakili klasifikasi yang salah.

Beberapa ukuran evaluasi kinerja model dapat dihitung dari *confusion matrix* yang masing-masing memberikan informasi mengenai berbagai aspek klasifikasi. Dalam hal klasifikasi multikelas, ukuran ini dihitung sebagai bobot rata-rata berdasarkan kelas. Dengan mempertimbangkan N aktifitas trafik yang harus diklasifikasikan ke dalam kelas M , maka:

a. Akurasi

Akurasi memberikan jumlah aktifitas trafik yang diklasifikasikan dengan benar dengan menghubungkan jumlah aktifitas trafik yang diklasifikasikan dengan benar dengan jumlah aktifitas keseluruhan. Rumus untuk akurasi terdapat pada persamaan 2.15

$$\text{Accuracy} := \frac{\sum_{i=0}^N c_{ii}}{\sum_{i=0}^N \sum_{j=0}^N c_{ij}} \quad (2.14)$$

b. *Recall*

Recall adalah kemampuan *classifier* untuk mengidentifikasi kelas tertentu dengan benar. Pengukuran ini diperoleh dari jumlah aktifitas trafik prediksi yang benar (TP) dikaitkan dengan jumlah aktifitas trafik di mana kelas diestimasikan dengan benar ($TP + FN$). Rumus *recall* ditampilkan pada persamaan 2.16.

$$\text{Recall}_{class} := \frac{TP_{class}}{TP_{class} + FN_{class}} \quad (2.15)$$

$$\text{Recall} := \frac{\sum_{i=0}^M \text{Recall}_i * (TP_i + FN_i)}{\sum_{i=0}^N \sum_{j=0}^N c_{ij}}$$

c. Presisi

Presisi merupakan keyakinan *classifier* untuk memprediksi kelas tertentu dengan benar. Presisi menghubungkan jumlah aktifitas trafik yang diprediksi positif (TP) dengan jumlah aktifitas trafik di mana kelas tertentu diprediksi ($TP + FP$). Presisi ditampilkan pada persamaan 2.18

$$Precision_{class} := \frac{TP_{class}}{TP_{class} + FP_{class}} \quad (2.16)$$

$$Precision := \frac{\sum_{i=0}^M Precision_i * (TP_i + FP_i)}{\sum_{i=0}^N \sum_{j=0}^N c_{ij}}$$

d. *F1-Score*

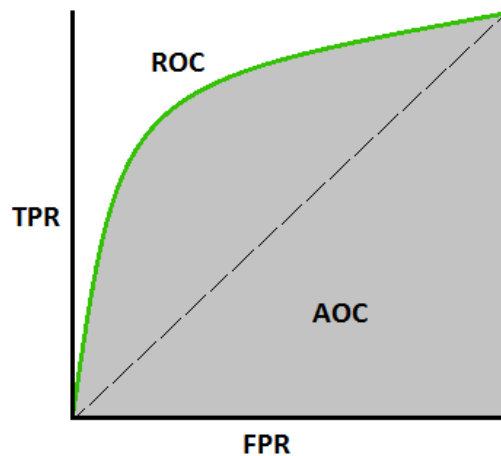
F1-Score adalah kemampuan *classifier* untuk memprediksi kelas tertentu. *F1-score* ditentukan dengan mempertimbangkan presisi dan *recall*. Persamaan 2.20 memberikan rumus untuk *F1-score*.

$$F1 - score_{class} := \frac{2TP_{class}}{2TP_{class} + FN_{class} + FP_{class}} \quad (2.17)$$

$$F1 - score := \frac{\sum_{i=0}^M F1 - score_i * (TP_i + FN_i)}{\sum_{i=0}^N \sum_{j=0}^N c_{ij}}$$

2.11.1 Kurva AUC - ROC

Kurva AUC-ROC merupakan teknik pengukuran kinerja model klasifikasi di berbagai pengaturan ambang batas. ROC (*Receiver Operating Characteristic*) merupakan kurva probabilitas sedangkan AUC (*Area Under Curve*) mengukur seluruh area dua dimensi dibawah seluruh kurva ROC. Kurva AUC – ROC bertujuan menghitung kemampuan model untuk membedakan antarkelas. Nilai



Gambar 2.9 Kurva AUC - ROC

AUC berkisar dari 0 hingga 1. Model yang memiliki AUC 1 mampu mengklasifikasikan pengamatan ke dalam kelas dengan sempurna.

2.12 *Kaggle*

Kaggle merupakan sebuah komunitas *online* yang dibentuk oleh Anthony Goldbloom sebagai CEO dan Ben Hamner sebagai CTO di tahun 2010. Platform ini khusus didirikan untuk masalah *data science* dan *machine learning*. *Kaggle* menyediakan kompetisi untuk para penggunanya dalam menyelesaikan masalah yang berhubungan dengan *data science* serta memungkinkan pengguna untuk berkolaborasi dalam menyelesaikan masalah. Selain itu, *Kaggle* juga menyediakan *dataset* dari berbagai bidang untuk dapat diolah dan digunakan oleh penggunanya. Pengguna dapat secara langsung mengolah dan menganalisis data pada *Kaggle*, juga mengunggah hasil analisis untuk dinilai oleh pengguna lain. Salah satu *dataset* yang tersedia pada *Kaggle* adalah *Labeled Network Traffic Flows – 141 Applications* yang berisikan data trafik jaringan yang telah diberi label berdasarkan aplikasi yang diakses.

2.13 *Jupyter Notebook*

Jupyter Notebook adalah salah satu aplikasi antarmuka buatan *Jupyter* yang digunakan untuk membuat dokumen yang berisikan persamaan matematika, analisis dan visualisasi data, paragraph, dan tautan. *Output* pada *Jupyter Notebook* berada pada satu *cell* dengan kode *input* sehingga mudah untuk dibaca dan dimengerti. Kemudahan untuk menulis teks dan kode dengan *Jupyter Notebook* menjadikan aplikasi ini banyak digunakan oleh para ilmuwan untuk tugas dan proyek kolaborasi (Haryanto, dkk., 2023).

2.14 *Google Colaboratory*

Google Colaboratory merupakan sebuah *tools* dari *Google*. Layanan yang disediakan oleh *Google Colaboratory* hampir sama dengan *Jupyter Notebook* karena dibuat atas *environment Jupyter*. Berbeda dengan *Jupyter Notebook*, *Google Colaboratory* berjalan pada sistem yang berbasis *cloud* dan menggunakan *Google Drive* sebagai media penyimpanan. Untuk mengeksekusi kode, *Google Colaboratory* menyediakan layanan GPU secara gratis untuk memaksimalkan proses komputasi (Maulida N., 2022).

2.15 *Streamlit*

Streamlit merupakan *open-source framework* dengan bahasa pemrograman Python yang banyak digunakan dalam bidang *data science* dan *machine learning* untuk pengembangan model dalam bentuk aplikasi web. *Streamlit* memudahkan *user* untuk memvisualisasikan data yang telah diolah dengan *interface* yang interaktif karena didukung dengan fitur *open sharing* sehingga mudah untuk dibagikan (Maula, M. 2022).