

**ANALISIS SENTIMEN KEPUASAN PELANGGAN PADA JASA  
ESKPEDISI MENGGUNAKAN BILSTM DAN BIGRU**



Disusun Oleh:

N a m a : Salsabila Zahirah Pranida  
NIM : 18523066

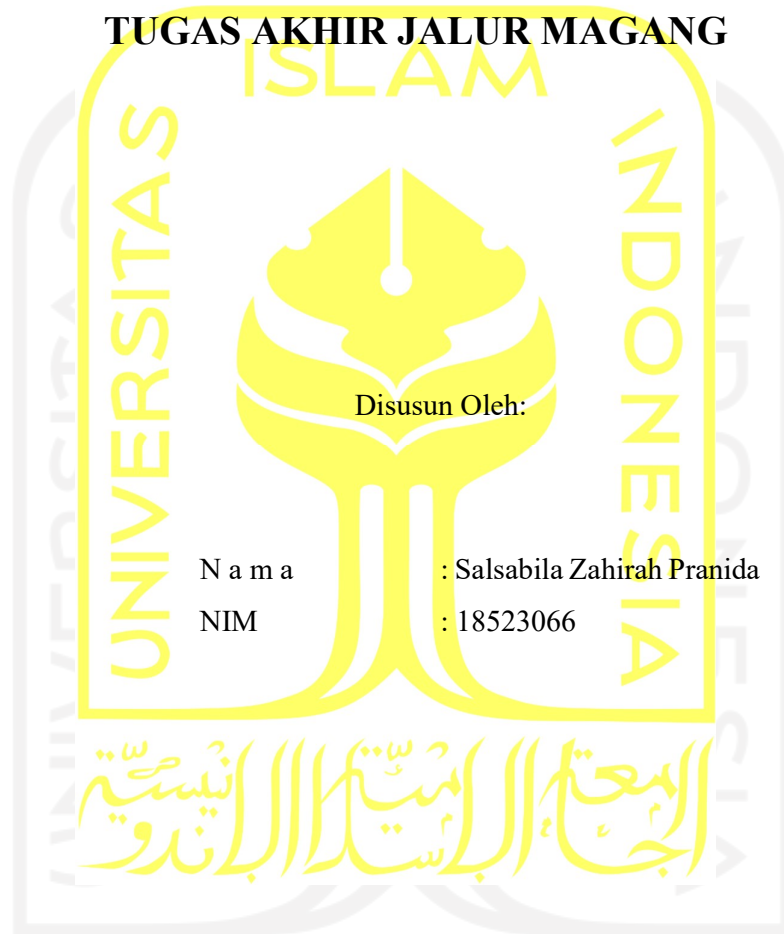
**PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA  
FAKULTAS TEKNOLOGI INDUSTRI  
UNIVERSITAS ISLAM INDONESIA**

**2022**

HALAMAN PENGESAHAN DOSEN PEMBIMBING

**ANALISIS SENTIMEN KEPUASAN PELANGGAN PADA JASA  
ESKPEDISI MENGGUNAKAN BILSTM DAN BIGRU**

**TUGAS AKHIR JALUR MAGANG**



Disusun Oleh:  
N a m a : Salsabila Zaherah Pranida  
NIM : 18523066

Yogyakarta, 21 Juni 2022

Pembimbing,

  
( Arrie Kurniawardhani, S.Si., M.Kom )

HALAMAN PENGESAHAN DOSEN PENGUJI

**ANALISIS SENTIMEN KEPUASAN PELANGGAN PADA JASA  
ESKPEDISI MENGGUNAKAN BILSTM DAN BIGRU**

**TUGAS AKHIR JALUR MAGANG**

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia  
Yogyakarta, 17 Agustus 2022

Tim Penguji

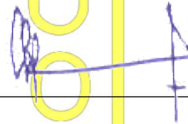
**Ketua Penguji**

Arrie Kurniawardhani, S.Si., M.Kom



**Anggota 1**

Chanifah Indah Ratnasari, S.Kom.,  
M.Kom.



**Anggota 2**

Sheila Nurul Huda, S.Kom., M.Cs



Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia



( Dr. Raden Teduh Dirgahayu, S.T., M.Sc. )

**HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR**

Yang bertanda tangan di bawah ini:

Nama : Salsabila Zahirah Pranida

NIM : 18523066

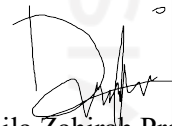
Tugas akhir dengan judul:

**ANALISIS SENTIMEN KEPUASAN PELANGGAN PADA JASA  
ESKPEDISI MENGGUNAKAN BILSTM DAN BIGRU**

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apa pun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

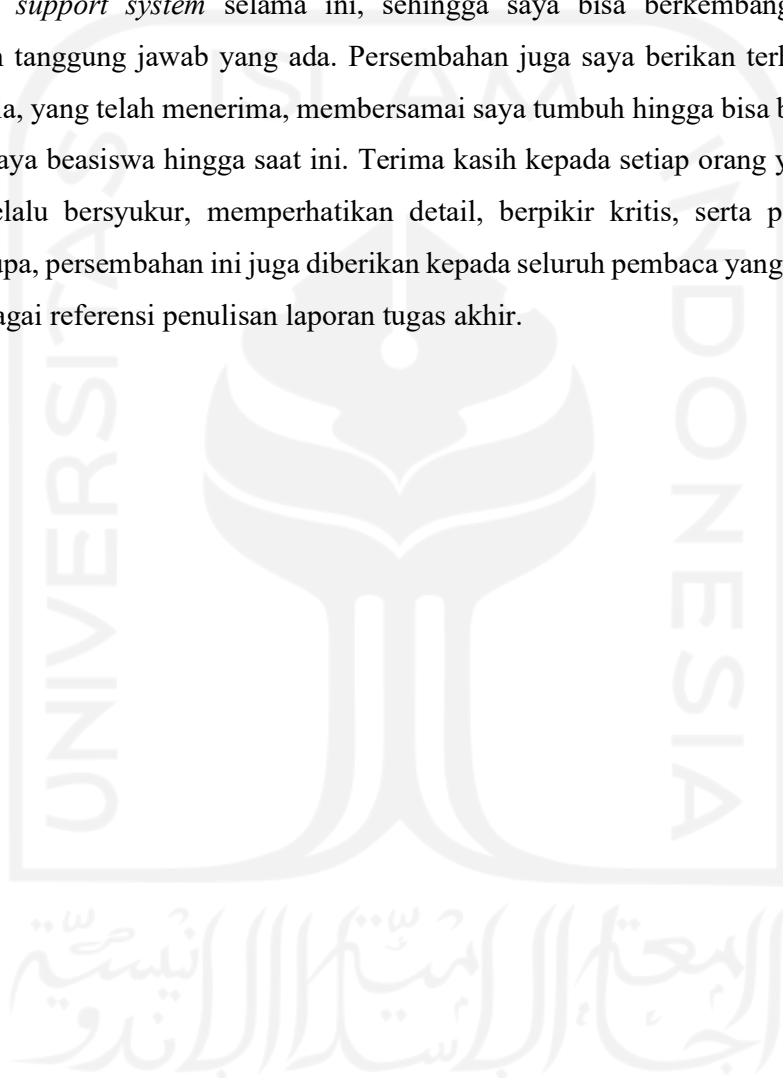
Yogyakarta, 21 Juni 2022



( Salsabila Zahirah Pranida )

## HALAMAN PERSEMBAHAN

Puji syukur atas rahmat Allah Yang Mahakuasa, yang telah memberikan saya kesempatan untuk menyelesaikan laporan tugas akhir. Laporan ini merupakan persembahan istimewa yang saya berikan untuk siapa pun yang telah mendukung saya, baik dalam bentuk materiel maupun moral. Persembahan besar saya berikan kepada orang tua, sahabat, dan teman-teman saya yang telah menjadi *support system* selama ini, sehingga saya bisa berkembang sejauh ini dan menyelesaikan tanggung jawab yang ada. Persembahan juga saya berikan terhadap Universitas Islam Indonesia, yang telah menerima, membersamai saya tumbuh hingga bisa belajar di sini, dan memberikan saya beasiswa hingga saat ini. Terima kasih kepada setiap orang yang mengajarkan saya untuk selalu bersyukur, memperhatikan detail, berpikir kritis, serta pelajaran-pelajaran lainnya. Tak lupa, persembahan ini juga diberikan kepada seluruh pembaca yang ingin menjadikan tulisan ini sebagai referensi penulisan laporan tugas akhir.



## HALAMAN MOTO

*“Life is about accepting the challenges along the way, choosing to keep moving forward,  
and savoring the journey.”*

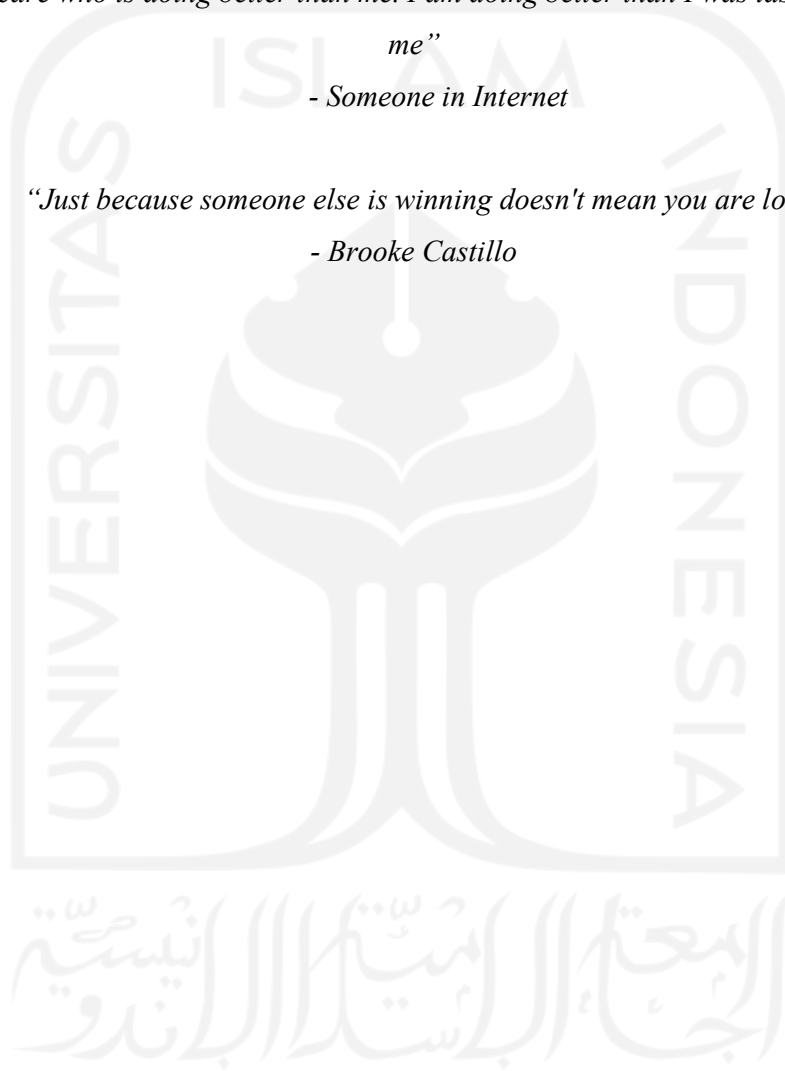
*- Roy T. Bennett in The Light in The Heart*

*“I do not care who is doing better than me. I am doing better than I was last year. It is me vs  
me”*

*- Someone in Internet*

*“Just because someone else is winning doesn't mean you are losing”*

*- Brooke Castillo*



## KATA PENGANTAR

*Assalamu'alaikum Warahmatullahi Wabarakatuh...*

*Alhamdulillah*, segala puji syukur atas kehadiran Allah *subhanahu wata'ala* yang telah memberikan rahmat, hidayah, serta inayahnya sehingga penulisan Laporan Tugas Akhir yang berjudul “ANALISIS SENTIMEN KEPUASAN PELANGGAN PADA JASA ESKPEDISI MENGGUNAKAN BILSTM DAN BIGRU” dapat diselesaikan tepat waktu. Selawat serta salam senantiasa tercurahkan kepada Rasul dan kekasih-Nya, Muhammad saw, yang telah membawa kita dari zaman jahiliah ke zaman yang terang benderang penuh dengan ilmu.

Adapun latar belakang dari selesainya Laporan Tugas Akhir ini ialah sebagai persyaratan Tugas Akhir Jalur Magang dan selesainya studi yang dilakukan di Program Studi Informatika - Program Sarjana, Fakultas Teknologi Industri, Universitas Islam Indonesia.

Dalam menjalani seluruh aktivitas, baik magang dan penyusunan Laporan Tugas Akhir, terdapat banyak pihak yang terlibat. Penulis ingin menyampaikan rasa terima kasih kepada seluruh pihak yang terlibat dalam penulisan ini. Beberapa di antaranya adalah sebagai berikut:

1. Kedua orang tua saya, yang selalu memberikan semangat, dukungan, dan memahami penulis selama ini.
2. Bapak Dr. Hendrik, S.T., M.Eng., selaku Ketua Jurusan Informatika Fakultas Teknologi Industri Universitas Islam Indonesia.
3. Bapak Dr. Raden Teduh Dirgahayu, S.T., M.Sc., selaku Ketua Program Studi Informatika Program Sarjana Fakultas Teknologi Industri Universitas Islam Indonesia
4. Ibu Arrie Kurniawardhani, S.Kom., M.Kom., selaku dosen pembimbing yang mengayomi dan membimbing penulis selama magang hingga penyusunan laporan dapat diselesaikan.
5. Bapak Ferry Pranolo, selaku *supervisor* yang telah banyak membantu penulis dengan berbagai kesempatan, pengalaman, dan ilmu selama magang dilakukan.
6. Mbak Zakya Rayhana dan Mbak Intan Hartri, selaku karyawan yang sudah menerima dan membantu penulis selama magang.
7. Rayhan Mahardhika Wijaya, yang telah memberikan semangat, nasihat, dan dukungan selama perkuliahan dilakukan.
8. Seluruh teman kontrakan Wisma Adisty, J&M - Alumni Bangkit, PT NJA, Bangkit 2021, dan DAI Microsoft, yang telah menjadi bagian dari penyemangat selama penulisan berlangsung.

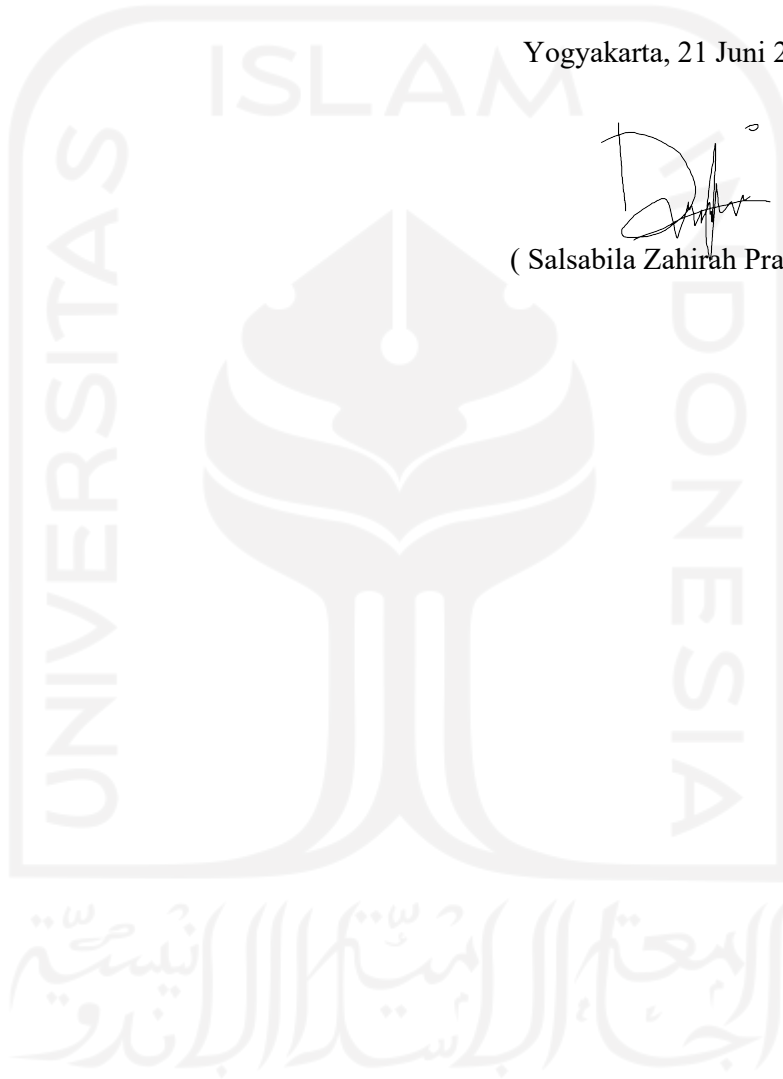
Atas bantuan berbagai pihak, penulis akhirnya dapat menyelesaikan laporan tugas akhir dengan sebaik mungkin. Namun, penulis menyadari bahwa laporan tugas akhir yang dibuat masih jauh dari kata sempurna, sehingga penulis sangat membutuhkan kritik dan saran untuk membuat laporan tugas akhir menjadi lebih baik. Akhir kata, semoga laporan ini dapat bermanfaat bagi kita semua.

*Wassalamu'alaikum Warrahmatullahi Wabarakatuh...*

Yogyakarta, 21 Juni 2022



( Salsabila Zahirah Pranida )





## SARI

SOLUSI247 merupakan sebuah perusahaan berbasis *Information Communication Technology* (ICT) yang menyediakan jasa dalam pemrosesan data berskala besar, seperti YAVA yang merupakan *Big Data Platform*. MedAn247 (*Media Analytics 247*) merupakan proyek yang dikerjakan selama pemagangan berlangsung sejak Agustus 2021. MedAn247 secara garis besar merupakan produk yang digunakan untuk menganalisis media untuk dapat menarik kesimpulan serta keputusan secara tepat dan akurat. Salah satu tugas yang diberikan adalah dalam melakukan analisis sentimen mengenai kepuasan pelanggan terhadap beberapa ekspedisi antara lain JNE, JNT, Sicepat, dan Anteraja. Analisis sentimen digunakan untuk mengetahui sentimen yang diberikan oleh pengguna ekspedisi melalui platform Twitter, dengan kata lain untuk mengetahui sikap atau perasaan pengguna. Dilakukan beberapa tahapan untuk mengetahui sentimen kepuasan pelanggan, di antaranya adalah *crawling*, *preprocessing*, *data labeling*, *modeling*, dan *evaluation*. Pada tahap *modeling*, digunakan algoritma BiGRU dan BiLSTM dalam melakukan klasifikasi sentimen. Hasil yang diperoleh pada tahap *evaluation* adalah algoritma BiGRU menghasilkan nilai akurasi sebesar 70.1% dan algoritma BiLSTM menghasilkan nilai akurasi sebesar 71.9%. Sebagai Data Scientist, pengalaman kerja berupa ketelitian dan kecermatan selama melakukan *data preprocessing* sangatlah diperlukan untuk membuahkan hasil model yang baik dan tepat.

Kata kunci: Analisis Sentimen, BiGRU, BiLSTM, Kepuasan Pelanggan Ekspedisi

## GLOSARIUM

<i>DataFrame</i>	tabel atau data tabular dengan array dua dimensi yang juga merupakan salah satu fungsi dari <i>library Pandas</i> di Python.
<i>API call</i>	<i>Application Programming Interface</i> ; penghubung antar aplikasi yang berbeda.
<i>RegEx</i>	merupakan <i>library</i> yang memuat <i>regular expression</i> dan digunakan untuk mendeskripsikan pola yang ingin dicari.
<i>NLTK</i>	merupakan <i>library</i> yang digunakan untuk membantu pekerjaan dalam analisis teks.
<i>Sastrawi</i>	merupakan <i>library</i> yang digunakan untuk mengubah kata berimbuhan dalam Bahasa Indonesia ke dalam bentuk dasar.
<i>keyword</i>	merupakan kata kunci yang dimasukkan ketika melakukan pencarian data.
<i>features</i>	merupakan data yang dipilih untuk dilatih.
<i>training</i>	merupakan tahap pelatihan yang dilakukan terhadap data yang bertujuan untuk memprediksi.
<i>RMSprop</i>	algoritma yang digunakan untuk mengature <i>learning rate</i> berdasarkan besaran nilai rata-rata dari <i>weight</i> , dan menggunakan nilai pertama dari gradien untuk menentukan rerata nilai <i>weight</i> .

## DAFTAR ISI

HALAMAN JUDUL .....	i
HALAMAN PENGESAHAN DOSEN PEMBIMBING .....	ii
HALAMAN PENGESAHAN DOSEN PENGUJI .....	iii
HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR .....	iv
HALAMAN PERSEMBAHAN .....	v
HALAMAN MOTO .....	vi
KATA PENGANTAR .....	vii
SARI .....	ix
GLOSARIUM .....	x
DAFTAR ISI .....	xi
DAFTAR TABEL .....	xiii
DAFTAR GAMBAR .....	xiv
BAB I PENDAHULUAN .....	1
1.1 Latarbelakang .....	1
1.2 Ruang Lingkup Magang .....	2
1.3 Tujuan .....	4
1.4 Manfaat .....	4
1.5 Sistematika Penulisan .....	4
BAB II DASAR TEORI .....	6
2.1 Analisis Sentimen .....	6
2.2 Crawling .....	6
2.3 Data Labeling .....	7
2.4 Preprocessing .....	7
2.5 Bidirectional Long-short Term Memory .....	8
2.6 Bidirectional Gated Recurrent Networks .....	10
2.7 Matriks Evaluasi .....	12
2.8 Tinjauan Pustaka .....	13
BAB III PELAKSANAAN MAGANG .....	16
3.1 Manajemen Proyek .....	16
3.2 Metodologi .....	16
3.3 <i>Crawling</i> .....	18
3.4 Preprocessing .....	24
3.4.1 <i>Cleaning</i> .....	24
3.4.2 <i>Normalization</i> .....	26
3.4.3 <i>Stopword Removal</i> .....	28
3.5 <i>Data Labeling</i> .....	31
3.6 <i>Modeling</i> .....	33
3.6.1 <i>Bidirectional GRU</i> .....	39
3.6.2 <i>Bidirectional LSTM</i> .....	45
3.7 <i>Hyperparamter Tuning</i> .....	51
3.7.1 <i>Imbalance BiGRU</i> .....	52
3.7.2 <i>Imbalance BiLSTM</i> .....	53
3.7.3 <i>Balance BiGRU</i> .....	54
3.7.4 <i>Balance BiLSTM</i> .....	55
3.8 <i>Evaluasi</i> .....	56
3.8.1 <i>Bidirectional GRU</i> .....	58
3.8.2 <i>Bidirectional LSTM</i> .....	59

BAB IV REFLEKSI PELAKSANAAN MAGANG .....	62
4.1 Relevansi Akademik.....	62
4.1.1 Pengalaman dalam <i>Preprocessing</i> .....	62
4.1.2 Pengalaman dalam <i>Data Labeling</i> .....	62
4.2 Pembelajaran Magang.....	63
4.2.1 Manfaat Magang.....	63
4.2.2 Kendala dan Hambatan Magang .....	64
4.2.3 Tantangan Magang.....	64
BAB V KESIMPULAN DAN SARAN.....	65
5.1 Kesimpulan.....	65
5.2 Saran.....	65
DAFTAR PUSTAKA.....	67
LAMPIRAN .....	71



## DAFTAR TABEL

Tabel 1.1 Ruang Lingkup Magang .....	3
Tabel 2.1 Tinjauan Pustaka.....	15
Tabel 3.1 Dataset hasil <i>crawling</i> .....	23
Tabel 3.2 Distribusi Hasil <i>Crawling</i> .....	23
Tabel 3.3 Karakter punctuation.....	25
Tabel 3.4 Hasil <i>Cleaning</i> .....	26
Tabel 3.5 Contoh dari kamus normalisasi.....	27
Tabel 3.6 Hasil Normalisasi.....	28
Tabel 3.7 Contoh <i>Stopword</i> .....	29
Tabel 3.8 Hasil <i>Stopword Removal</i> .....	30
Tabel 3.9 Hasil <i>Cleaning</i> Duplikasi Hasil <i>Stopword Removal</i> .....	31
Tabel 3.10 Data Labeled .....	32
Tabel 3.11 Anomali Data Label.....	32
Tabel 3.12 Distribusi Label.....	33
Tabel 3.13 <i>DataFrame</i> Hasil Konversi.....	35
Tabel 3.14 Distribusi <i>splitting data</i> .....	39
Tabel 3.15 <i>Initial Parameter</i> .....	39
Tabel 3.16 Skenario <i>Hyperparameter Tuning</i> .....	51
Tabel 3.17 <i>Summary of Hyperparameter Tuning</i> .....	52
Tabel 3.18 <i>Imbalance BiGRU Hyperparameter Tuning</i> .....	53
Tabel 3.19 Hasil <i>Hyperparameter Tuning Imbalance BiGRU</i> .....	53
Tabel 3.20 <i>Imbalance BiLSTM Hyperparameter Tuning</i> .....	54
Tabel 3.21 Hasil <i>Hyperparameter Tuning Imbalance BiLSTM</i> .....	54
Tabel 3.22 <i>Balance BiGRU Hyperparameter Tuning</i> .....	55
Tabel 3.23 Hasil <i>Hyperparameter Tuning Balance BiGRU</i> .....	55
Tabel 3.24 <i>Balance BiLSTM Hyperparameter Tuning</i> .....	56
Tabel 3.25 Hasil <i>Hyperparameter Tuning Balance BiLSTM</i> .....	56
Tabel 3.27 Confusion Matrix BiGRU.....	58
Tabel 3.27 Hasil data baru BiGRU .....	59
Tabel 3.28 Confusion Matrix BiLSTM .....	60
Tabel 3.30 Hasil dengan data baru BiLSTM .....	61

## DAFTAR GAMBAR

Gambar 2.1 Ilustrasi Analisis Sentimen.....	6
Gambar 2.2 Alur Crawling Twitter.....	7
Gambar 2.3 Arsitektur dari Bidirectional LSTM.....	10
Gambar 2.4 Arsitektur dari Bidirectional GRU.....	12
Gambar 3.1 Tahapan analisis sentimen .....	17
Gambar 3.2 Tahapan <i>preprocessing</i> .....	17
Gambar 3.3 Pengecekan koneksi dengan server Twitter.....	19
Gambar 3.4 Memasukkan <i>keyword</i> dan melakukan API Call .....	19
Gambar 3.5 Hasil <i>crawling</i> dari Twitter .....	21
Gambar 3.6 Hasil <i>full_text</i> .....	21
Gambar 3.7 Simpan JSON dan konversi ke dalam CSV .....	22
Gambar 3.8 Visualisasi distribusi hasil <i>crawling</i> .....	23
Gambar 3.9 Kode program <i>preprocessing</i> .....	24
Gambar 3.10 Kode program cleaning.....	26
Gambar 3.11 Kode program normalisasi.....	27
Gambar 3.12 Kode program <i>stopword removal</i> .....	30
Gambar 3.13 Kode program menghapus data duplikasi.....	30
Gambar 3.14 <i>Chart</i> distribusi dari label .....	33
Gambar 3.15 Konversi <i>string</i> ke dalam kategorikal.....	34
Gambar 3.16 Kode program dari <i>tokenization</i> dan <i>maxseqLen</i> .....	35
Gambar 3.17 Kode program <i>balancing dataset</i> .....	36
Gambar 3.18 Kode program mengenai data .....	36
Gambar 3.19 Hasil dari word indexing.....	37
Gambar 3.20 Kode program vocab.....	38
Gambar 3.21 Kode program <i>splitting</i> data.....	38
Gambar 3.22 Kode program <i>modeling</i> BiGRU .....	40
Gambar 3.23 Kode program model <i>training</i> .....	41
Gambar 3.24 Hasil dari <i>Model Summary Imbalance</i> BiGRU.....	41
Gambar 3.25 Hasil dari <i>training model imbalance</i> BiGRU .....	43
Gambar 3.26 Hasil dari <i>Model Summary Balance</i> BiGRU .....	44
Gambar 3.27 Hasil dari <i>training model balance</i> BiGRU .....	45
Gambar 3.28 Kode program <i>modeling</i> BiLSTM.....	46

Gambar 3.29 Hasil dari <i>Model Summary Imbalance</i> BiLSTM .....	47
Gambar 3.30 Hasil dari <i>training model imbalance</i> BiLSTM .....	48
Gambar 3.31 Hasil dari <i>Model Summary Balance</i> BiLSTM .....	49
Gambar 3.32 Hasil dari <i>training model balance</i> BiGRU .....	50
Gambar 3.33 Kode program fungsi evaluasi .....	57
Gambar 3.34 Kode program untuk mengetahui akurasi .....	57
Gambar 3.35 Kode program confusion matrix dan classification report.....	57
Gambar 3.36 Hasil classification report model BiGRU .....	58
Gambar 3.37 Hasil classification model BiLSTM.....	60



## BAB I PENDAHULUAN

### 1.1 Latarbelakang

PT Dua Empat Tujuh, SOLUSI247 merupakan suatu perusahaan yang bergerak dalam pelayanan jasa pemrosesan data berskala besar dan berbasis *Information Communication Technology* (ICT) (SOLUSI247, 2020). Salah satu produk yang dimiliki oleh perusahaan adalah YAVA-247. YAVA-247 atau dikenal juga dengan YAVA adalah sebuah *platform open-source* yang menyediakan *environment* untuk manajemen Big Data serta pemantauan *cluster* Hadoop (YAVA-247, n.d.).

Pelaksanaan magang dimulai sejak Agustus 2021 hingga Februari 2022. Saat magang, penulis ditempatkan ke dalam proyek MedAn-247 (*Media Analytics-247*) untuk melakukan riset mengenai analisis sentimen. MedAn-247 bertujuan untuk menganalisis media agar dapat menarik kesimpulan dan keputusan secara tepat dan akurat (MedAn, 2020). MedAn-247 memiliki beberapa sumber data, yakni Twitter dan media berita daring. Analisis sentimen merupakan salah satu fitur yang dimiliki oleh MedAn-247 yang berfungsi untuk memonitoring informasi mengenai prediksi sentimen terhadap suatu konten di sosial media ataupun media kabar daring dalam tiga sudut pandang, yakni positif, netral, dan negatif (MedAn, 2020). Analisis sentimen yang dilakukan mencakup analisis sentimen media berita daring, analisis sentimen Twitter, serta analisis sentimen Twitter dengan studi kasus kepuasan pelanggan ekspedisi.

Analisis sentimen dengan studi kasus kepuasan pelanggan akan dikaji lebih dalam pada tugas akhir. Analisis sentimen merupakan sebuah proses dalam memahami, mengekstrak, serta mengolah data untuk mendapatkan informasi mengenai informasi yang terkandung di dalamnya (Liesnawan, 2019). Data yang akan digunakan adalah data spesifik yang diambil melalui *crawling* dari Twitter. Twitter dipilih karena memiliki pengguna sebanyak 18,5 juta pengguna Indonesia (Kemp, 2022). Hal tersebut juga didukung dengan banyaknya opini dari pengguna yang spesifik terhadap ekspedisi yang akan digunakan.

Pengguna Twitter Indonesia kerap mengunggah opini mereka mengenai keluhan, apresiasi, dan pertanyaan kepada *customer* servis ekspedisi yang digunakan. Beberapa keluhan umumnya mengenai pembobolan atau pencurian barang oleh kurir ekspedisi, paket yang hilang, dan paket yang tidak sampai sesuai estimasi. Apresiasi juga tidak jarang diberikan oleh pengguna apabila ekspedisi sesuai dengan ekspektasi mereka, yakni paket yang dipesan sampai dengan cepat dan barang atau paket datang dalam kondisi aman dan baik. Dengan adanya berbagai opini yang



terunggah melalui Twitter, dimungkinkan untuk melakukan analisis yang bertujuan sebagai riset mengenai kepuasan pelanggan, pada studi kasus ekspedisi. Salah satu fitur yang dimiliki oleh Twitter, yang umumnya digunakan untuk menganalisis data Twitter adalah Twitter API (Twitter, 2022).

Pengerjaan analisis sentimen kepuasan pelanggan pada studi kasus ekspedisi dilakukan dengan mengambil *tweet* atau cuitan pengguna secara spesifik yang mengandung *keywords* seperti JNE, JNT, Sicepat, dan Anteraja. Kemudian, cuitan yang telah didapat akan dibersihkan dan dimodelkan menggunakan algoritma *deep learning* untuk mendapatkan model. Penggunaan metode *bidirectional* pada BiGRU dan BiLSTM dilakukan karena memiliki keunggulan dalam menangkap informasi semantik dua arah, yaitu *forward* dan *backward*. Sedangkan penggunaan GRU memiliki keunggulan dalam menangkap data dengan urutan panjang yang diperlukan pada pembelajaran NLP (Zulqarnain et al., 2019), dan LSTM dibentuk dengan perilaku *default* mempelajari dependensi informasi secara jangka panjang (Xu et al., 2019). Hasil akhir dari kedua metode yang digunakan diharapkan untuk dapat mengetahui label sentimen yang diberikan secara otomatis.

## 1.2 Ruang Lingkup Magang

Program magang dilaksanakan dalam waktu enam bulan sebagai *data scientist* di SOLUSI247. SOLUSI247 merupakan suatu perusahaan yang bergerak dalam pelayanan jasa pemrosesan data berskala besar dan berbasis ICT yang telah berdiri sejak tahun 2000 (SOLUSI247, 2020). SOLUSI247 memiliki kantor pusat yang beralamat di Segitiga Emas Business Park, Jalan Professor Dr. Satrio No. Kav. 6 Unit 4 & 5, Karet Pedurenan, Kota Jakarta Selatan, Daerah Ibukota Jakarta 12940. Penugasan yang diberikan selama magang seluruhnya bersinggungan dengan MedAn247. Website resmi perusahaan dapat diakses pada laman <https://www.solusi247.com/>.

Seluruh penugasan yang diberikan merupakan proyek riset yang digunakan untuk kebutuhan produk MedAn247. Penugasan secara garis besar antara lain mencakup analisis sentimen media berita daring, analisis sentimen Twitter, dan analisis sentimen Twitter dengan studi kasus kepuasan pelanggan ekspedisi. Selama magang dilaksanakan, komunikasi dilakukan menggunakan Telegram dan Google Meet, apabila diperlukan. Rincian durasi pengerjaan tugas, metode yang digunakan, aktivitas, dan hasil tugas dapat dilihat pada Tabel 1.1.

Tugas yang dilakukan selama durasi magang sejak Agustus hingga September 2021 adalah analisis sentimen yang dilakukan terhadap media berita daring. Analisis sentimen ini

menggunakan metode BiLSTM, GRU, dan FastText. Beberapa aktivitas yang dilakukan antara lain adalah *preprocessing*, *modeling*, dan evaluasi. Evaluasi yang dilakukan menunjukkan bahwa penggunaan metode FastText memperoleh performa akurasi tertinggi sebesar 54%. Sedangkan, metode BiLSTM dan GRU memperoleh performa akurasi sebesar 32%, yang mana masih di bawah ekspektasi.

Tugas kedua yang dilakukan selama durasi magang sejak Oktober hingga November 2021 adalah analisis sentimen terhadap media Twitter. Analisis sentimen yang dilakukan menggunakan metode BiLSTM dan GRU. Beberapa aktivitas yang dilakukan antara lain *crawling*, *preprocessing*, *data labeling*, dan evaluasi. Evaluasi yang dilakukan menunjukkan bahwa penggunaan metode BiLSTM dan GRU menghasilkan performa akurasi yang sama, yakni 73%.

Tugas terakhir yang dilakukan selama durasi magang sejak November 2021 hingga Februari 2022 adalah analisis sentimen menggunakan media Twitter dengan studi kasus kepuasan pelanggan ekspedisi. Analisis sentimen dilakukan menggunakan metode BiLSTM dan BiGRU. Beberapa aktivitas yang dilakukan antara lain adalah *crawling*, *preprocessing*, *data labeling*, *modeling*, dan evaluasi. Hasil evaluasi menunjukkan performa akurasi model BiLSTM sebesar 70,9% dan model BiGRU sebesar 69,1%.

Tabel 1.1 Ruang Lingkup Magang

Tugas	Durasi	Metode	Aktivitas	Hasil
Analisis sentimen media berita daring	Agustus - September 2021	Metode yang digunakan selama tugas berlangsung adalah BiLSTM, GRU, FastText	Aktivitas yang dilakukan selama tugas berlangsung adalah <i>preprocessing</i> , <i>modeling</i> , dan evaluasi	Evaluasi model menunjukkan akurasi performa model yang di bawah ekpektasi, dengan perolehan performa tertinggi oleh model FastText sebesar 54%. Sedangkan performa akurasi metode BiLSTM dan GRU hanya sebesar 32%.
Analisis sentimen Twitter	Oktober - November 2021	Metode yang digunakan selama tugas berlangsung adalah BiLSTM dan GRU	Aktivitas yang dilakukan selama tugas berlangsung adalah <i>crawling</i> , <i>preprocessing</i> , <i>data labeling</i> , <i>modeling</i> , dan evaluasi.	Evaluasi menunjukkan akurasi performa model menghasilkan nilai yang sama, yakni 73%.
Analisis sentimen Twitter studi kasus kepuasan pelanggan ekspedisi	November 2021 - Februari 2022	Metode yang digunakan selama tugas berlangsung adalah BiLSTM dan BiGRU	Aktivitas yang dilakukan selama tugas berlangsung adalah <i>crawling</i> , <i>preprocessing</i> , <i>data labeling</i> , <i>modeling</i> , dan evaluasi.	Evaluasi menunjukkan akurasi performa model BiLSTM sebesar 70,9% dan model BiGRU sebesar 69,1%.

Penugasan terakhir juga merupakan topik yang diambil sebagai tugas akhir dikarenakan memiliki peluang keberlanjutan analisis sentimen dengan berbagai topik lainnya. Penjelasan detail akan dijabarkan pada bab-bab selanjutnya.

### 1.3 Tujuan

Berdasarkan latar belakang yang dijabarkan, maka laporan akhir ini ditulis untuk mencapai tujuan sebagai berikut:

- a. Melakukan analisis sentimen berdasarkan data *tweets* mengenai ekspedisi sehingga dapat mengetahui model terbaik untuk mengenali label sentimen terhadap data secara otomatis.
- b. Melihat performa metode BiGRU dan BiLSTM pada data ekspedisi dan performanya melalui evaluasi matriks.

### 1.4 Manfaat

Berdasarkan manfaat yang telah dijabarkan, maka laporan akhir ini ditulis untuk mencapai tujuan sebagai berikut:

- a. Menghasilkan dokumentasi mengenai panduan *data labeling* yang dapat menjadi referensi pada tahap *labeling* dengan topik yang berbeda.
- b. Hasil dapat dijadikan referensi dalam melakukan analisis sentimen dengan topik yang berbeda selanjutnya.

### 1.5 Sistematika Penulisan

Sistematika penulisan merupakan sebuah urutan rangkaian penulisan dalam suatu dokumen. Adapun penulisan dalam tugas akhir ini disusun dengan sistematika penulisan sebagai berikut:

- a. BAB I – PENDAHULUAN  
Bab I akan berfokus pada pendahuluan yang meliputi latar belakang, ruang lingkup, tujuan, dan manfaat dari riset yang dilakukan pada *sentiment analysis* dengan topik kepuasan pelanggan ekspedisi.
- b. BAB II - DASAR TEORI  
Bab II akan membahas secara detail mengenai teori-teori yang berkaitan dengan proyek yang dikerjakan.

c. **BAB III - PELAKSANAAN MAGANG**

BAB III akan menjelaskan mengenai pelaksanaan magang, mulai dari tahap persiapan pemagangan hingga implementasi dari berbagai teori yang telah dituliskan pada bab sebelumnya.

d. **BAB IV - REFREKSI PELAKSANAAN MAGANG**

BAB IV akan berfokus pada refleksi kegiatan yang dilakukan selama magang, dilihat dari dua sudut pandang, yakni sudut pandang teknis dan non teknis.

e. **BAB V - KESIMPULAN DAN SARAN**

BAB V akan membahas mengenai kesimpulan yang didapatkan dari hasil pengerjaan. Selain hal tersebut, bab ini juga akan berisi saran dari penulis sehingga dapat dijadikan pembelajaran bagi pembaca.

f. **DAFTAR PUSTAKA**

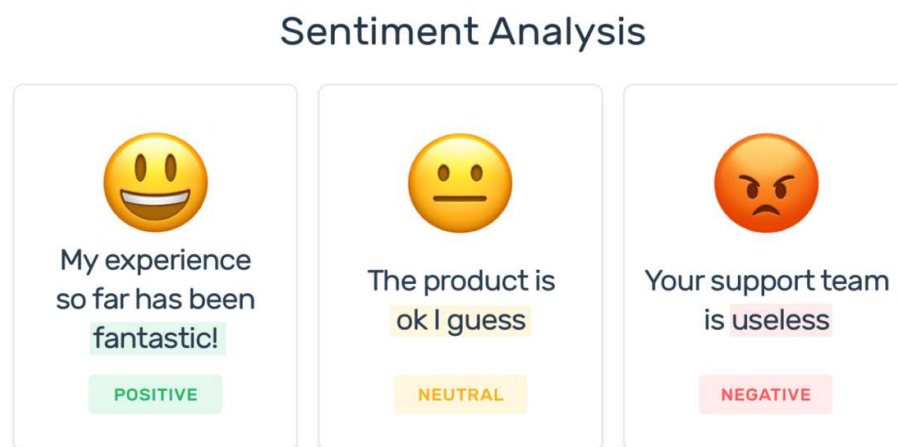
Daftar pustaka akan berisi tentang sumber-sumber yang dijadikan penulis sebagai referensi dari penulisan laporan tugas akhir.

## BAB II

### DASAR TEORI

#### 2.1 Analisis Sentimen

Analisis sentimen mengacu pada bidang yang luas dari NLP, *linguistic computational*, dan *text mining*, yang memiliki tujuan menganalisis pendapat, sentimen, evaluasi, sikap, penilaian, hingga emosi seseorang dalam suatu topik, produk, layanan, organisasi, individu, ataupun suatu kegiatan tertentu (Liu, 2012). Analisis sentimen juga merupakan suatu proses yang berfokus pada pendapat, kemudian pengelompokan polaritas teks dalam kalimat sehingga didapatkan suatu kategori yang dapat ditentukan sebagai sentimen *positive*, *negative*, atau *neutral* (Samsir et al., 2021). Tujuan dari adanya analisis sentimen adalah untuk mengelompokkan atau mengkategorikan teks ke dalam kelas yang diberikan. Alustrasi dari analisis sentimen diperlihatkan pada Gambar 2.1.



Gambar 2.1 Ilustrasi Analisis Sentimen

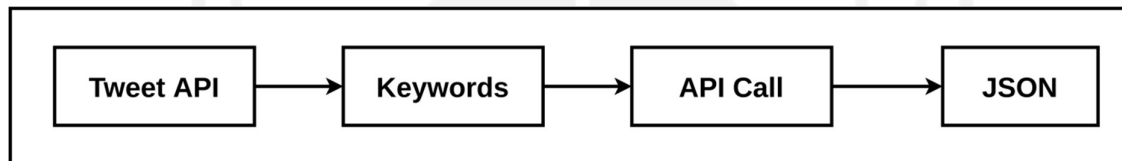
Sumber: (Wolff, 2022)

#### 2.2 Crawling

*Crawling* data merupakan tahapan awal dalam penelitian yang bertujuan untuk mengumpulkan atau mengunduh data dari suatu *database* (Dikiyanti et al., 2021). *Crawling* oleh (Peng et al., 2018) dapat disebut juga sebagai *web crawler* atau *web spider* yang merupakan salah satu konsep inti dari *Internet of Things*, yang mana sebuah program yang dapat otomatis

mengunduh informasi dari sebuah *website* dengan logika dan aturan yang telah ditentukan. *Crawling* juga merupakan suatu proses *data mining* yang dilakukan melalui *Application Programming Interface* (API) suatu media sosial (Idris et al., 2020) seperti portal berita, media sosial, ataupun *web* Internet lainnya.

Sebagai contoh media sosial yang digunakan adalah Twitter yang alurnya diilustrasikan pada Gambar 2.2, digunakan bantuan dari Twitter API untuk melakukan *crawling*. Data yang akan diambil saat *crawling* dilakukan hanyalah data yang mengandung *keyword* spesifik. Setelah *keyword* dimasukkan, dilakukan pengambilan data menggunakan API Call. Proses *crawling* tidak hanya mengambil data teksnya saja, tetapi juga mengambil seluruh atribut informasi yang menyertainya, seperti waktu pengunggahan, pengguna yang mengunggah, dan lainnya (Harywanto et al., 2021). Data *crawling* kemudian akan disimpan ke dalam file berformat JSON.



Gambar 2.2 Alur Crawling Twitter

### 2.3 Data Labeling

*Data labeling* oleh (Harywanto et al., 2021) didefinisikan dalam konteks *machine learning* sebagai sebuah proses guna mendeteksi dan memberikan sebuah tanda atau label sambil memberikan sebuah makna dan/atau sebuah konteks ke dalam data. *Labeling* diperlukan dalam beberapa kasus seperti *computer vision*, *speech recognition*, dan *natural language preprocessing*. Pada studi kasus yang dilakukan, data teks diambil dari platform Twitter, yang mana menurut (Pristiyono et al., 2021), *labeling* juga merupakan sebuah proses yang bertujuan untuk menentukan sebuah kelas dari data *tweets*, contohnya adalah menentukan data ke dalam kelas positif atau negatif.

### 2.4 Preprocessing

*Preprocessing* merupakan sebuah tahapan awal dalam pengolahan teks dan menjadi salah satu faktor yang dapat meningkatkan akurasi dari sebuah model dan efisiensi waktu pemrosesan (Chandrasekar & Qian, 2016). Tahap *preprocessing* mengubah data yang ada menjadi terstruktur sesuai kebutuhan dan agar dapat dengan mudah diolah lebih lanjut pada proses selanjutnya. Beberapa tahap *preprocessing* biasanya yang dilakukan adalah:

- a. *Converting* dilakukan dalam mengubah format data yang sebelumnya menjadi CSV, untuk memfasilitasi tahap *preprocessing* yang akan dilakukan selanjutnya (Rianto et al., 2021).
- b. *Case folding* merupakan sebuah proses untuk mengubah karakter dokumen ke dalam bentuk *uppercase* atau *lowercase* (Rianto et al., 2021).
- c. *Tokenizing* dilakukan untuk mengubah kalimat ke dalam token (Rianto et al., 2021).
- d. *Filtering* dilakukan untuk membersihkan data. Beberapa aktivitas yang dilakukan antara lain adalah *stopword removal*, membersihkan token dari angka, dan membersihkan tanda baca. Tahapan ini dilakukan untuk mengurangi *noise* dan meningkatkan efisiensi (Khader et al., 2019).
- e. *Stemming* adalah proses yang bertujuan untuk mengurangi ukuran kosa kata dengan memetakan varian berdasarkan akar kata (Singh & Gupta, 2019).
- f. *Normalizing* merupakan proses yang dilakukan untuk menormalisasi kalimat menggunakan aturan Bahasa Indonesia, hal ini berkaitan dengan singkatan dan kata pinjaman (Rianto et al., 2021)
- g. *Stopword Removal* merupakan proses yang bertujuan untuk menghilangkan kata-kata yang sangat umum dan tidak memberikan informasi penting sehingga keadannya dapat diabaikan. Terdapat beberapa cara yang dapat digunakan dalam menghilangkan *stopword*, umumnya adalah menggunakan *library* yang telah tersedia seperti NLTK (NLTK, 2022) dan Sastrawi (Sastrawi, 2018).

### **2.5 Bidirectional Long-short Term Memory**

*Long Short Term Memory* (LSTM) dibuat untuk melengkapi Jaringan Syaraf Tiruan tradisional yang memiliki kekurangan tidak dapat menangkap informasi semantik yang terdapat pada kalimat. LSTM juga dimaksudkan untuk menyelesaikan permasalahan gradien yang hilang. LSTM dibentuk dengan perilaku *default* mempelajari dependensi informasi secara jangka panjang (Xu et al., 2019). Hal ini kemudian menyebabkan dilengkapi dengan adanya *input gate*, *output gate*, *forget gate* dan *memory cell* (Raza et al., 2021). Komponen dan perhitungan dasar yang digunakan pada LSTM dijabarkan pada persamaan (1) hingga persamaan (7).

$$f_t = \sigma_g(W_f X_t + U_f h_{t-1} + V_f C_{t-1} + b_f) \quad (1)$$

*Forget gate* atau yang disimbolkan dengan  $f_t$  pada persamaan (1) merupakan sebuah gerbang yang menentukan informasi yang harus dilupakan oleh *memory cell*. Terdapat dua input dari *forget gate*, yakni  $X_t$  dan  $h_{t-1}$ .  $W_f$  merupakan bobot penghubung dari  $X_t$  dan *forget gate*.  $U_f$  merupakan bobot penghubung antara  $h_{t-1}$  dan *forget gate*.  $C_{t-1}$  adalah keadaan pada *memory cell* terakhir.  $V_f$  merupakan bobot penghubung dari  $C_{t-1}$  dan *forget gate*.  $b_f$  merupakan bias.  $\sigma_g$  merupakan fungsi aktivasi sigmoid.

$$i_t = \sigma_g(W_i X_t + U_i h_{t-1} + V_i C_{t-1} + b_i) \quad (2)$$

*Input gate* atau yang disimbolkan dengan  $i_t$  pada persamaan (2) merupakan sebuah gerbang yang menentukan informasi yang harus diperbarui oleh *memory cell*.  $W_i$  merupakan bobot penghubung dari  $X_t$  dan *input gate*.  $U_i$  merupakan bobot penghubung antara  $h_{t-1}$  dan *input gate*.  $V_i$  merupakan bobot penghubung dari  $C_{t-1}$  dan *input gate*.  $b_i$  merupakan bias.

$$C'_t = \sigma_c(W_c X_t + U_c h_{t-1} + V_c C_{t-1} + b_c) \quad (3)$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot C'_t \quad (4)$$

*Memory cell candidate gate* atau yang disimbolkan dengan  $C'_t$  pada persamaan (3) merupakan sebuah gerbang yang berisi kandidat informasi *memory cell*.  $W_c$  merupakan bobot penghubung dari  $X_t$  dan *memory cell candidate gate*.  $U_c$  merupakan bobot penghubung antara  $h_{t-1}$  dan *memory cell candidate gate*.  $V_c$  merupakan bobot penghubung dari  $C_{t-1}$  dan *input gate*.  $f_t$  dan  $i_t$  mengacu pada bobot dari  $C_{t-1}$  dan  $C'_t$ .  $C_t$  merupakan *memory cell gate* yang dijabarkan pada persamaan (4).  $b_c$  merupakan bias.  $\sigma_c$  merupakan fungsi aktivasi *tanh*.

$$o_t = \sigma_g(W_o X_t + U_o h_{t-1} + V_o C_{t-1} + b_o) \quad (6)$$

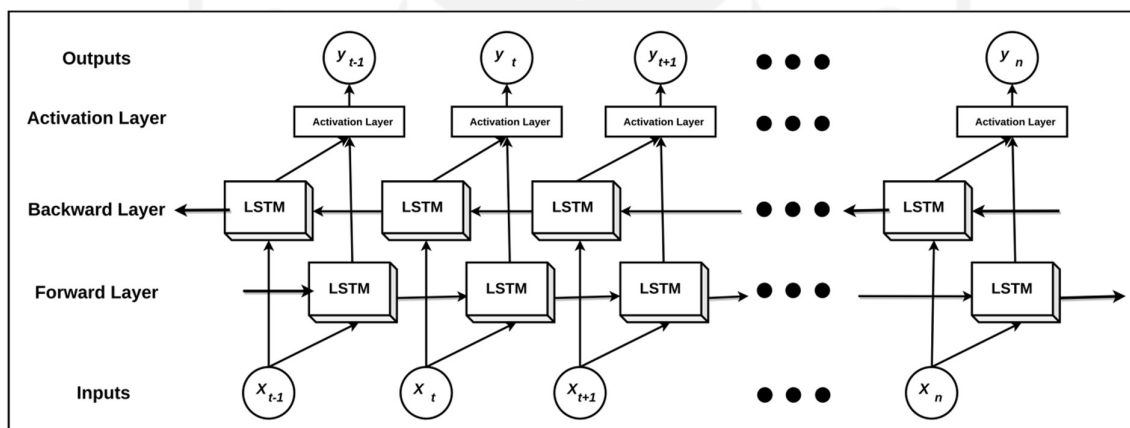
$$h_t = o_t \cdot \sigma_c(C_t) \quad (7)$$

*Output gate* atau yang disimbolkan dengan  $o_t$  pada persamaan (6) merupakan sebuah gerbang yang mengeluarkan nilai dari LSTM.  $W_o$  merupakan bobot penghubung dari  $X_t$  dan *output gate*.  $U_o$  merupakan bobot penghubung antara  $h_{t-1}$  dan *output gate*.  $V_o$  merupakan bobot penghubung dari  $C_{t-1}$  dan *output gate*.  $b_o$  merupakan bias.  $h_t$  pada persamaan (7) merupakan *hidden state*.



Sayangnya model arsitektur LSTM hanya menangkap informasi secara satu arah, secara *forward* atau mengambil informasi selanjutnya. Maka, diusulkan penggunaan *bidirectional neural networks* agar model LSTM dapat menangkap informasi secara dua arah, secara *forward* dan *backward*.

Bidirectional LSTM (BiLSTM) juga disebut sebagai tumpukan LSTM, sebuah improvisasi dari LSTM. *Forward layer* yang terdapat pada arsitektur LSTM dimanfaatkan untuk menyimpan konteks informasi setelahnya. Sedangkan, *backward layer* dimanfaatkan untuk menyimpan konteks informasi sebelumnya (Lample et al., n.d.). Ilustrasi arsitektur dari BiLSTM dapat dilihat pada Gambar 2.3, yang memperlihatkan setiap *hidden layer* komponen yang terdapat di atas dan di bawah bergabung sehingga membentuk arsitektur yang lebih panjang dibandingkan LSTM. BiLSTM yang memiliki kerangka arsitektur lebih panjang juga menyebabkan informasi yang diproses dipelajari lebih mendetail.



Gambar 2.3 Arsitektur dari Bidirectional LSTM

## 2.6 Bidirectional Gated Recurrent Networks

*Gated Recurrent Unit* (GRU) adalah sebuah metode yang digunakan untuk metode yang juga digunakan untuk menangani *short-term memory* dengan dua buah gerbang; *reset gate* dan *update gate* (Raza et al., 2021). Kedua gerbang yang ada pada GRU digunakan untuk mengendalikan permasalahan dalam gradien yang hilang. GRU tidak mempertahankan *status cell* seperti LSTM, tetapi mempertahankan *hidden cell*, yang kemudian mengakibatkan model GRU lebih mudah dan cepat untuk melatih set data kecil. Komponen dan perhitungan dasar yang digunakan pada GRU dijabarkan pada persamaan (8), (9), (10), dan (11).

$$Z_t = \sigma(W_z \cdot [h_{t-1}, X_t]) \quad (8)$$

*Update gate* dilambangkan dengan  $Z_t$  yang ditunjukkan pada persamaan (8) bertujuan untuk menghilangkan masalah gradien yang hilang. Masalah tersebut dapat dihilangkan karena model mempelajari berapa banyak informasi masa lalu yang diteruskan ke masa depan. Pada *update gate*, variabel  $X_t$  sebagai input dikalikan dengan bobot  $W_z$ . Sama halnya dengan  $X_t$ , variabel  $h_{t-1}$  yang menyimpan informasi dari unit-unit sebelumnya dikalikan dengan bobot pada  $W_z$ . Kemudian,  $\sigma$  yang berupa fungsi aktivasi sigmoid ditambahkan untuk mengetahui output yang berupa nilai diantara 0 dan 1.

$$R_t = \sigma(W_r \cdot [h_{t-1}, X_t]) \quad (9)$$

*Reset gate* yang dilambangkan dengan  $R_t$  yang pada persamaan (9) memiliki fungsi yang berlawanan dengan *update gate* karena berfungsi untuk memutuskan seberapa banyak informasi masa lalu yang harus dilupakan. Rumus *reset gate* sangat mirip dengan *update gate*, yang berbeda hanya bobot dan fungsinya.

$$\hat{h}_t = \tanh(W \cdot [r_t * h_{t-1}, X_t]) \quad (10)$$

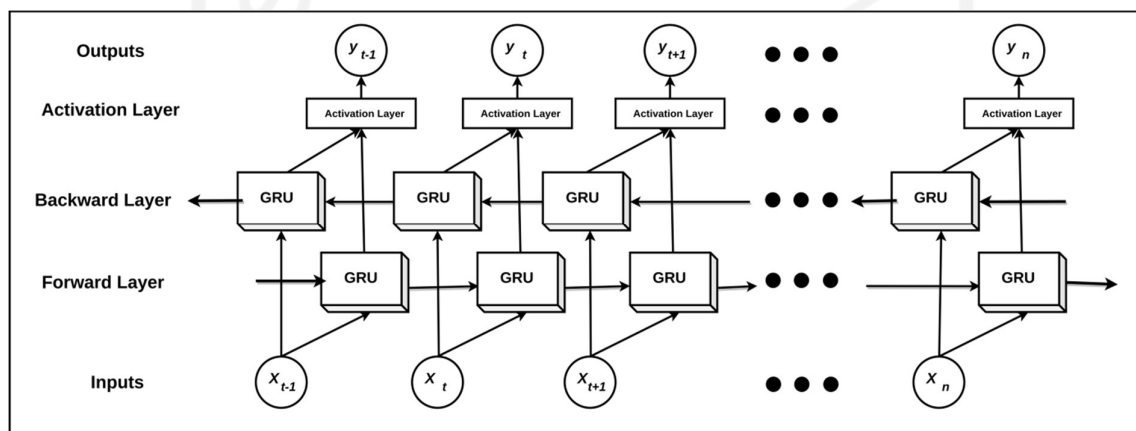
*Memory cell* dilambangkan dengan  $\hat{h}_t$  pada persamaan (10). *Memory cell* memiliki input berupa  $X_t$ , yang kemudian dikalikan ke dalam bobot  $W$ . Penerapan *element-wise multiplication* pada *reset gate* dan output  $h_{t-1}$  sebelumnya memungkinkan untuk melewati informasi masa lalu yang relevan. Keduanya ditambahkan bersama dengan penerapan fungsi aktivasi tanh.

$$h_t = (1 - Z_t) * h_{t-1} + Z_t * \hat{h}_t \quad (11)$$

*Final memory cell* dilambangkan sebagai  $h_t$  pada persamaan (11). Vektor  $h_t$  menyimpan banyak informasi pada unit dan akan diteruskan lebih lanjut ke dalam *network*. Penerapan *element-wise multiplication* ke dalam *update gate* dan *memory cell*. Penerapan *element-wise multiplication* juga diterapkan pada  $1 - Z_t$ . Hasil dari penerapan keduanya kemudian ditambahkan bersama-sama.

GRU dinilai cocok pada data sekuensial terutama dengan data dalam jumlah banyak. GRU juga dapat menangkap data dengan urutan panjang yang diperlukan pada pembelajaran NLP

(Zulqarnain et al., 2019). Untuk memaksimalkan kinerja GRU dalam pengambilan informasi semantik dua arah, digunakanlah metode *bidirectional*. Penggunaan metode *bidirectional* yang arsitekturnya ditunjukkan pada Gambar 2.4, memainkan peranan penting dalam pengambilan informasi semantik yang terjadi secara dua arah dalam peningkatan kinerja klasifikasi. Metode *bidirectional* menangkap data secara sekuens dengan dua arah, yaitu *forward* dan *backward*. *Forward layer* pada BiGRU menangkap data sekuensial yang berisi informasi pada urutan selanjutnya. Sedangkan, *backward layer* pada BiGRU menangkap data sekuensial pada urutan sebelumnya.



Gambar 2.4 Arsitektur dari Bidirectional GRU

## 2.7 Matriks Evaluasi

Evaluasi dilakukan menggunakan beberapa matriks, yaitu matriks akurasi, matriks *precision*, dan matriks *recall* atau sensitivitas. Penambahan matriks *precision* dan *recall* dilakukan agar dapat menambahkan analisis mengenai hasil akurasi, apabila hasil berada di bawah ekspektasi. Matriks akurasi tidak selalu menjadi *proper method* untuk melakukan evaluasi performa dari klasifikasi, karena tidak memberikan hasil detailnya. Karena hal tersebut, maka matriks *precision*, *recall* dan *f1-score* dapat digunakan (Krishna, 2020).

Evaluasi umumnya menggunakan Confusion Matriks yang terdiri dari empat variabel, di antaranya merupakan *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN). TP merupakan informasi *positive* yang benar diprediksi oleh model. TN merupakan informasi *negative* yang benar diprediksi oleh model. FP adalah informasi *negative* yang diprediksi sebagai *positive* oleh model. FN adalah informasi *positive* yang diprediksi sebagai *negative* oleh model.

Matriks akurasi didefinisikan sebagai rasio jumlah prediksi benar dibandingkan dengan keseluruhan data prediksi. Umumnya, matriks akurasi didefinisikan pada persamaan (11).

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (11)$$

Matriks *precision* didefinisikan sebagai rasio jumlah prediksi benar yang bernilai *positive* dibandingkan dengan keseluruhan data hasil prediksi yang diprediksi *positive*. Umumnya, matriks *precision* didefinisikan pada persamaan (12).

$$Precision = \frac{TP}{TP+FP} \quad (12)$$

Matriks *recall* atau sensitivitas didefinisikan sebagai rasio jumlah prediksi benar *positive* dibandingkan dengan keseluruhan data prediksi yang benar *positive*. Umumnya, matriks *recall* didefinisikan pada persamaan (13).

$$Recall = \frac{TP}{TP+FN} \quad (13)$$

Matriks *f1-score* didefinisikan sebagai *mean* dari *precision* dan *recall*. Nilai terbaik dari *f1-score* adalah 1.0 dan terburuknya adalah 0. Umumnya, matriks *f1-score* didefinisikan pada persamaan (14).

$$F1\ score = \frac{2 \times Precision \times Recall}{Precision+Recall} \quad (14)$$

## 2.8 Tinjauan Pustaka

Beberapa penelitian sebelumnya sudah pernah dilakukan mengenai *Bidirectional GRU* dan *Bidirectional LSTM*. Penelitian ini kemudian diringkas dan ke dalam Tabel 2.1.

Penelitian sebelumnya pernah dilakukan oleh (Ali et al., 2021) dalam penggunaan metode BiGRU yang membahas mengenai analisis sentimen pada level aspek SIoT (*Social Internet of Things*) dengan tiga kelas dataset berbeda, *dataset Laptop*, *dataset Twitter*, dan *dataset Restaurant*, yang bertujuan untuk meningkatkan fungsionalitas media sosial. Lapisan arsitektur yang terdapat pada BiGRU dimanfaatkan untuk mendapatkan vektor dari sebuah input, dengan memanfaatkan informasi yang diperoleh setiap kata dengan sebaik-baiknya. Hasil dari penggunaannya ialah

peningkatan performa model yang meningkat secara signifikan pada seluruh dataset yang digunakan. Hasil dari performa akurasi dan *macro-f1* pada *dataset Laptop* menunjukkan 77.11% dan 73.28%. Sedangkan, hasil dari performa akurasi dan *macro-f1* pada *dataset Twitter* menunjukkan 82.02% dan 72.53%. Kemudian, hasil dari performa akurasi dan *macro-f1* pada *dataset Restaurant* menunjukkan 74,56% dan 73.52%

Penelitian serupa mengenai metode BiGRU juga dilakukan oleh (Han et al., 2020) yang digunakan pada dataset SentiDrugs untuk melakukan analisis sentimen pada level aspek pada ulasan penggunaan obat. Penggunaan metode BiGRU diharapkan mampu memberikan hasil yang lebih baik, karena memainkan peran penting dalam pengambilan informasi semantik dua arah dan peningkatan kinerja klasifikasi. Hasil dari eksperimen yang dilakukan terhadap model BiGRU pada matriks akurasi dan *macro-f1* menunjukkan sebesar 71,35% dan 69,88%.

Sedangkan penelitian mengenai metode BiLSTM telah diusulkan dan diterapkan pada analisis sentimen komentar pada dataset yang berbasis ulasan hotel yang dilakukan oleh (Xu et al., 2019) Metode BiLSTM menggunakan *word representation* sebagai input, sehingga model dapat mempelajari informasi yang terkandung. BiLSTM juga dinilai dapat menangkap informasi semantik dari sebuah konteks dengan lebih efisien dan efektif karena mempertimbangkan informasi secara *forward* dan *backward*. Hal ini dibuktikan dengan hasil matriks *precision* sebesar 91,54%, matriks *recall* sebesar 92,82%, dan matriks *f1-score* sebesar 92,18%.

Analisis sentimen yang dilakukan oleh (Hameed & Garcia-Zapirain, 2020) menggunakan model BiLSTM dengan *single-layered* pada data mengenai ulasan film, yakni data MR dan IMDb. Model yang dibuat dapat secara efektif memprediksi sentimen dengan menggunakan metode komputasi yang berbiaya rendah. Hasil yang dicapai menungguli beberapa metode baru dalam hal akurasi. Didapatkan performa akurasi dan *f1-score* terhadap data MR sebesar 80.5% dan 80,49%. Sedangkan performa akurasi dan *f1-score* terhadap data IMDb sebesar 90.58% dan 90,58%.

Tabel 2.1 Tinjauan Pustaka

No	Nama Peneliti	Judul	Metode	Dataset	Hasil
1	Ali, Waqar Yang, Yuwang Qiu, Xiulin Ke, Yaqi Wang, Yinyin	Aspect-Level Sentiment Analysis Based on Bidirectional- GRU in SIoT	BiGRU	Laptop, Twitter, Restaurant	Hasil dari eksperimen yang dilakukan pada dataset <i>Laptop</i> , performa akurasi dan <i>macro-f1</i> menunjukkan 77.11% dan 73.28%. Kemudian, pada dataset <i>Twitter</i> , performa akurasi dan <i>macro-f1</i> menunjukkan 82.02% dan 72.53%. Sedangkan, pada dataset <i>Restaurant</i> , performa akurasi dan <i>macro-f1</i> menunjukkan 74.56% dan 73.52%.
2	Han, Yue Liu, Meiling Jing, Weipeng	Aspect-Level Drug Reviews Sentiment Analysis Based on Double BiGRU and Knowledge Transfer	BiGRU	SentiDrugs	Hasil dari eksperimen yang dilakukan terhadap model BiGRU pada matriks akurasi dan <i>macro-f1</i> menunjukkan sebesar 71,35% dan 69,88%.
3	Xu, Guixian Meng, Yueting Qiu, Xiaoyu Yu, Ziheng Wu, Xu	Sentiment analysis of comment texts based on BiLSTM	BiLSTM	Ulasan Hotel	Hasil metode yang digunakan menghasilkan matriks <i>precision</i> sebesar 91,54%, matriks <i>recall</i> sebesar 92,82%, dan matriks <i>f1-score</i> sebesar 92,18%.
4	Hameed, Zabit Garcia-Zapirain, Begonya	Sentiment Classification Using a Single- Layered BiLSTM Model	BiLSTM	MR, IMDb	Hasil dari eksperimen yang dilakukan dengan metode tersebut memberikan performa akurasi dan <i>f1-score</i> terhadap data MR sebesar 80.5% dan 80,49%. Sedangkan performa akurasi dan <i>f1-score</i> terhadap data IMDb sebesar 90.58% dan 90,58%

## BAB III PELAKSANAAN MAGANG

### 3.1 Manajemen Proyek

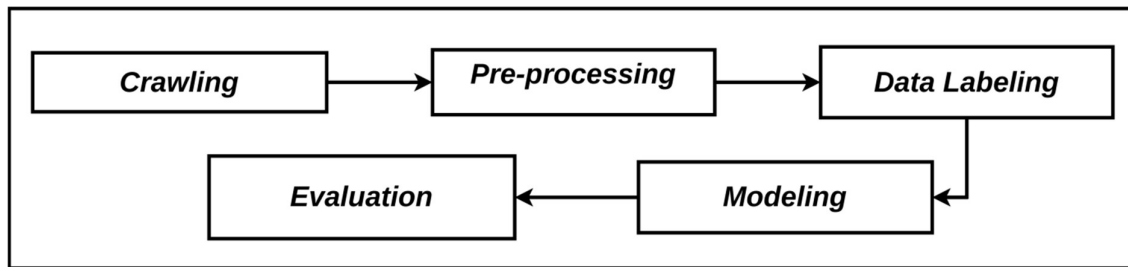
Selama magang dilaksanakan selama enam bulan hingga bulan Februari 2022 sebagai Data Scientist di SOLUSI247, penulis berada di tim MedAn (*Media Analytics*) pada divisi SDO (*Skill Development Office*). Magang dilakukan secara *Work from Home* atau *Remote*, karena adanya pandemi *coronavirus*. Jadwal pelaksanaan magang adalah setiap hari Senin - Jum'at atau lima hari dalam satu minggu.

Dalam hal berkomunikasi, seluruh komunikasi dilakukan menggunakan platform Telegram, baik komunikasi dengan supervisor ataupun anggota tim MedAn lainnya. Komunikasi dilakukan setiap pagi hari dengan mengirimkan informasi mengenai kegiatan yang akan dilakukan pada hari itu. Begitu pula dengan pelaporan kemajuan tugas yang diberikan dan hasil akhir tugas juga diinformasikan melalui Telegram secara *private*. Sedangkan, apabila ada hambatan yang memerlukan penjelasan panjang atau diskusi, komunikasi akan dilangsungkan lewat Google Meet.

Tugas-tugas selama magang berlangsung diberikan langsung oleh Mas Ferry, selaku supervisor. Seluruh tugas dilakukan secara individu, karena magang dilaksanakan secara mandiri sehingga komunikasi mengenai informasi tugas dan pelaporannya hanya dilakukan dengan supervisor saja. Seluruh pengerjaan yang berkaitan dengan data menggunakan Google Colaboratory, Visual Studio Code, dan Google Spreadsheet. Sedangkan, seluruh *resource* data didapatkan dari hasil *crawling* yang dilakukan secara mandiri dan data milik perusahaan.

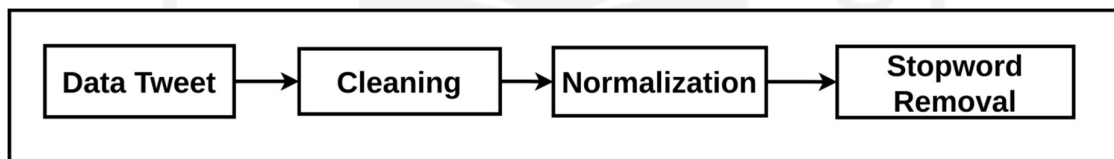
### 3.2 Metodologi

Terdapat beberapa tahapan yang digunakan dalam proses analisis sentimen dengan topik kepuasan pelanggan. Tahapan yang dilakukan divisualisasikan pada Gambar 3.1 yang menunjukkan lima tahapan antara lain *crawling*, *preprocessing*, *data labeling*, *modeling*, dan *evaluation*.



Gambar 3.1 Tahapan analisis sentimen

Pada tahap pertama dalam melakukan analisis sentimen dengan studi kasus kepuasan pelanggan ekspedisi dilakukan tahapan *crawling*. Tahapan *crawling* dilakukan menggunakan alur Gambar 2.2 untuk mendapatkan data yang akan digunakan, yakni data Twitter. Aktivitas *crawling* yang dilakukan menggunakan Twitter API dilakukan untuk mendapatkan data spesifik berdasarkan empat *keywords* ekspedisi, yakni ‘jne’, ‘jnt’, ‘sicepat’, dan ‘anteraja’. Setelah data tersebut diambil, kemudian pembersihan data pada tahap *preprocessing*.



Gambar 3.2 Tahapan *preprocessing*

Tahap *preprocessing* yang ditunjukkan pada Gambar 3.2 merupakan alur *preprocessing* yang akan digunakan. Data *tweet* yang mengandung *keyword* spesifik kemudian dibersihkan dengan menghilangkan *username* (@), URLs, re-tweet (RT), *digit*, dan tanda baca. Setelah data dibersihkan kemudian dilakukan tahapan normalisasi yang bertujuan untuk mengubah kata *slang* ataupun tidak baku menjadi kata baku. Terakhir, data akan dilakukan penghilangan *stopword* yang bertujuan untuk menghilangkan kata-kata umum yang tidak memiliki arti penting dalam kalimat. Tidak lupa pada tahap *preprocessing* dilakukan penghilangan duplikat untuk mempermudah pada tahap *data labeling*.

Data-data bersih setelah tahap *preprocessing* kemudian diberikan label ke dalam tiga kelas berbeda, yakni *positive*, *negative*, dan *neutral*. Pemberian label ini dilakukan oleh penulis secara manual. Pemberian label terhadap data dilakukan berdasarkan asumsi penulis ke dalam tiga kelas tersebut.



Setelah data-data telah usai diberikan label ke dalam kelasnya, maka akan dilakukan tahap *modeling*. Pada tahap ini, data *tweet* akan dibagi menjadi data *training*, data *validation*, dan data *testing* dengan rasio 85:15:15. Pada tahap ini dibangun arsitektur model menggunakan metode *Bidirectional GRU* dan *Bidirectional LSTM*. Kedua arsitektur menggunakan *optimizer RMSprop* dan *loss function categorical\_crossentropy*. Setelah arsitektur dibuat, dilakukan *fitting* untuk melakukan *training*.

Tahap evaluasi dilakukan untuk mengukur performa dari model yang telah dilatih dengan melakukan *testing* menggunakan data *testing*. Performa model diukur dengan menggunakan matriks akurasi, matriks *precision*, matriks *recall*, dan matriks *f1-score*. Selain menggunakan ketiga matriks tersebut, model juga dites menggunakan beberapa data yang belum berlabel, yang mana bertujuan untuk mengetahui hasil dari analisis sentimen yang dilakukan oleh model secara otomatis.

### 3.3 Crawling

*Crawling* yang juga disebut sebagai salah satu cara untuk mengumpulkan data. Pada analisis sentimen dengan studi kasus kepuasan pelanggan ekspedisi, data yang digunakan berupa data Twitter, disebut juga dengan data *tweets*. Data *Tweets* yang diambil hanyalah *tweets* yang mengandung *keywords* spesifik yang telah ditentukan. Alur kerja yang akan dilakukan *crawling* divisualisasikan pada Gambar 2.2 dengan empat tahapan. Tahapan yang akan dilakukan antara lain adalah konektivitas dengan server Twitter menggunakan Twitter API, memasukkan *keyword* spesifik, melakukan API Call dan menyimpan data final ke dalam JSON.

*Crawling* dilakukan menggunakan bahasa pemrograman Python dan menggunakan koneksi dengan *server* Twitter dengan Twitter API *keys* yang dapat dilihat pada Gambar 3.3. Konektivitas yang dilakukan menggunakan Twitter API *keys* memerlukan empat *keys* yang telah didapatkan yakni *consumer\_key* (Ck), *consumer\_secret* (Cs), *access\_token* (At), dan *access\_secret* (As). Dengan adanya Twitter API *keys*, maka dapat dilakukan koneksi terhadap server Twitter menggunakan fungsi *twitter\_connect(Ck, Cs, At, As, verbose = 0)*.

```
# Contoh API Keys (Sesuaikan dengan API keys masing-masing)
Ck = 'key' # consumer_key
Cs = 'key' # consumer_secret
At = 'key' # access_token
As = 'key' # access_secret

def twitter_connect(Ck, Cs, At, As, verbose = 0):
    try:
        auth = tweepy.OAuthHandler(Ck, Cs)
```

```

    auth.set_access_token(At, As)
    twitter = tweepy.API(auth, timeout=120)
    if verbose != 0:
        user = twitter.verify_credentials()
        print('Welcome "%s" you are now connected to twitter server'
%user.name)
    return twitter
except:
    print("Connection failed, please check your API keys or connection")
    return None

# checking connection to Twitter
twitter = twitter_connect(Ck, Cs, At, As, verbose = 1)

# Hasil dari pengecekan koneksi
# Welcome "User" you are now connected to twitter server

```

Gambar 3.3 Pengecekan koneksi dengan server Twitter

Setelah konektivitas dengan *server* Twitter tersambung, dapat dilakukan *API call* dengan memasukkan *keywords* spesifik. *Keywords* yang akan digunakan pada topik kepuasan pelanggan di antaranya adalah JNE, JNT, Sicepat, dan Anteraja. Penulisan *keyword* dilakukan pada variabel *topic* pada Gambar 3.4 secara bergantian. Pengambilan *tweets* berdasarkan *keyword* dilakukan dengan *API Call*, yang mana hanya akan mengambil 100 *tweets* pada setiap *API Call* yang dilakukan.

```

# Max 100 tweet per "API call"
topic = 'JNE'
N = 100 # jumlah tweet yang ingin diambil
bahasa = 'id'
T = twitter.search(q=topic, lang=bahasa, count=N, tweet_mode = 'extended')
tweets = [t._json for t in T]
print(' Berhasil mendapatkan {} tweets'.format(len(tweets)))

# Hasil
# Berhasil mendapatkan 100 tweets

```

Gambar 3.4 Memasukkan *keyword* dan melakukan *API Call*

*Crawling* yang dilakukan tidak hanya mengambil data teks saja, tetapi juga mengambil seluruh atribut penyertanya, seperti waktu pengunggahan *tweet*, isi dari *tweet* yang diunggah, informasi pengguna, geolokasi, jumlah *like*, jumlah *retweet* dan informasi detail mengenai *tweet* yang diunggah. Keseluruhan atribut yang dihasilkan dari tahapan *crawling* dengan melakukan *API Call* pada Gambar 3.4 dan hasilnya dapat dilihat pada Gambar 3.5. Gambar 3.5 menunjukkan hasil *crawling* dan seluruh atributnya terhadap satu *tweet* secara mendetail.

```

{"created_at": "Thu Dec 23 06:19:43 +0000 2021",

```

```

"id": 1473901110981128192,
"id_str": "1473901110981128192",
"full_text": "@sicepat_ekspres \nResi 002842651058\nNgga dikirim2 dan sdh lewat
batas estimasi\nSejak tgl 15 udh smpai di kota tujuan tp kurir ga antar2 ke
alamat. Pdhl juga di kota ga dipelosok\nSmpe capek saya tiap hari tlp customer
service ga ada hasil. Brg saya hilang atau gimana sih ?????",
"truncated": false,
"display_text_range": [0, 275],
"entities": {
  "hashtags": [],
  "symbols": [],
  "user_mentions": [
    {
      "screen_name": "sicepat_ekspres",
      "name": "SiCepat Ekspres",
      "id": 4417582514,
      "id_str": "4417582514",
      "indices": [0, 16]}],
  "urls": []},
"metadata": {
  "iso_language_code": "in",
  "result_type": "recent"},
"source": "<a href=\"http://twitter.com/download/android\"
rel=\"nofollow\">Twitter for Android</a>",
"in_reply_to_status_id": null,
"in_reply_to_status_id_str": null,
"in_reply_to_user_id": 4417582514,
"in_reply_to_user_id_str": "4417582514",
"in_reply_to_screen_name": "sicepat_ekspres",
"user": {
  "id": 1355723793205915648,
  "id_str": "1355723793205915648",
  "name": "KatinaVol",
  "screen_name": "KatinaVol",
  "location": "",
  "description": "Tak ada Hasil yang meghianati usaha :)",
  "url": null,
  "entities": {
    "description": {"urls": []}},
  "protected": false,
  "followers_count": 3,
  "friends_count": 49,
  "listed_count": 0,
  "created_at": "Sun Jan 31 03:45:24 +0000 2021",
  "favourites_count": 16,
  "utc_offset": null,
  "time_zone": null,
  "geo_enabled": false,
  "verified": false,
  "statuses_count": 95,
  "lang": null,
  "contributors_enabled": false,
  "is_translator": false,
  "is_translation_enabled": false,
  "profile_background_color": "F5F8FA",
  "profile_background_image_url": null,
  "profile_background_image_url_https": null,
  "profile_background_tile": false,
  "profile_image_url":
"http://pbs.twimg.com/profile_images/1355724050576773125/QbbNq2Ox_normal.jpg"
,

```

```

    "profile_image_url_https":
"https://pbs.twimg.com/profile_images/1355724050576773125/QbbNq2Ox_normal.jpg
",
    "profile_link_color": "1DA1F2",
    "profile_sidebar_border_color": "C0DEED",
    "profile_sidebar_fill_color": "DDEEF6",
    "profile_text_color": "333333",
    "profile_use_background_image": true,
    "has_extended_profile": true,
    "default_profile": true,
    "default_profile_image": false,
    "following": false,
    "follow_request_sent": false,
    "notifications": false,
    "translator_type": "none",
    "withheld_in_countries": []},
"geo": null,
"coordinates": null,
"place": null,
"contributors": null,
"is_quote_status": false,
"retweet_count": 0,
"favorite_count": 0,
"favorited": false,
"retweeted": false,
"lang": "in"}

```

Gambar 3.5 Hasil *crawling* dari Twitter

Namun, tidak semua atribut yang ada akan digunakan, hanya atribut *full\_text* yang akan digunakan sebagai dataset. Berdasarkan hasil *crawling*, dilakukan penyederhanaan sehingga dapat diketahui nilai dari atribut *full\_text* yang akan digunakan. Hasil dari nilai *full\_text* pada Gambar 3.6 mengacu pada Gambar 3.5.

```

print('tweet pertama oleh "{}" :
"{}".format(T2[0]['user']['screen_name'],T2[0]['full_text']))

# Hasil
tweet pertama oleh "KatinaVol" : "@sicepat_ekspres
Resi 002842651058
Ngga dikirim2 dan sdh lewat batas estimasi
Sejak tgl 15 udh smpai di kota tujuan tp kurir ga antar2 ke alamat. Pdhl juga
di kota ga dipelosok
Smpe capek saya tiap hari tlp customer service ga ada hasil. Brg saya hilang
atau gimana sih ????"

```

Gambar 3.6 Hasil *full\_text*

Setelah mendapatkan data-data melalui *crawling*, maka data akan disimpan menggunakan JSON yang mana hanya akan menyimpan nilai dari *full\_text* saja menggunakan fungsi *saveTweets(tweets, file='Tweets.json')*. Kemudian, untuk memudahkan dalam membaca dataset,

data akan dikonversikan ke dalam format CSV yang diperlihatkan pada Gambar 3.7. Tampilan contoh data yang digunakan sebagai dataset berdasarkan *keyword* spesifik ditampilkan pada Tabel 3.1.

```

## Menyimpan ke dalam JSON
def saveTweets(tweets, file='Tweets.json'): #in Json Format
    with open(file, 'w') as f:
        for t in tweets:
            try:
                f.write(json.dumps(t)+'\n')
            except:
                pass

# Menyimpan hasil crawling twitter
fileName = '/content/tweets_jne.json'
saveTweets(tweets, file=fileName)
print('Saved to '+fileName)

# Hasil
Saved to /content/tweets_jne.json

# Menkonversikan ke dalam format CSV
header = ['text']
with open('tweets_jne.csv', 'w') as f:
    writers = csv.writer(f, lineterminator = '\n')
    writers.writerow(header)
    for i in D: # D adalah variabel yang menyimpan data seluruh full_text dari
JSON file
        writers.writerow([i])

```

Gambar 3.7 Simpan JSON dan konversi ke dalam CSV

Dapat dilihat pada Tabel 3.1 yang merupakan hasil dari *crawling* dataset yang ditampilkan dalam tabel. Terdapat enam contoh data hasil *crawling*, yang dua di antaranya merupakan data yang terduplikasi atau berisi data yang sama. Data yang terduplikasi pada Tabel 3.1 dikarenakan karena adanya pengguna yang melakukan *retweet* (RT) terhadap suatu *tweets*, sehingga data tersebut ikut terambil saat *crawling* dilakukan. Kemudian, seluruh data akan dibersihkan dalam tahap selanjutnya, tahap *preprocessing*. Data yang terduplikasi juga akan dihilangkan pada tahap tersebut.

Tabel 3.1 Dataset hasil *crawling*

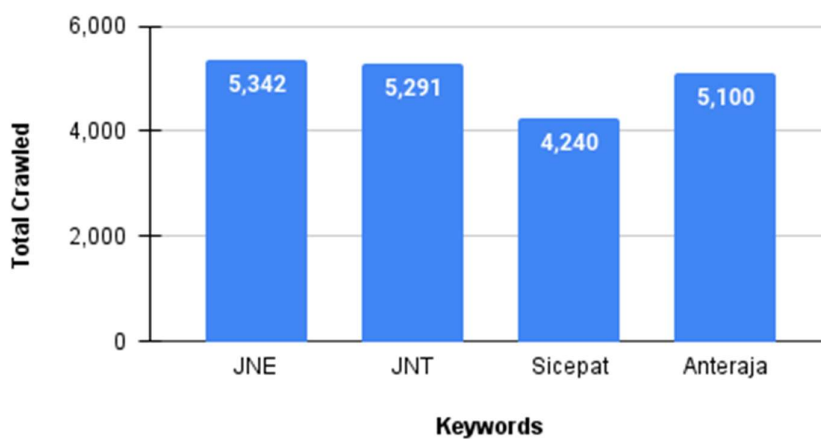
full_text
@sicepat_ekspres \nResi 002842651058\nNngga dikirim2 dan sdh lewat batas estimasi\nSejak tgl 15 udh sampai di kota tujuan tp kurir ga antar2 ke alamat. Pdhl juga di kota ga dipelosok\nSmpe capek saya tiap hari tlp customer service ga ada hasil. Brg saya hilang atau gimana sih ????
@sicepat_ekspres weee nyari kerjaan aja luuu bisa2nya salah nempel resi jelas2 itu angkanya beda heyyy akhirnya paket org nyasar ke gue kannnn?????? https://t.co/neBjei4LHK
Makanya kurir si cepat kalo pickup jangan marah marah, jadi salah kan kerja nya @sicepat_ekspres \nUdah di tanyain paket kemana, di suruh cari ngga mau, malah ngotot sendiri
Halo @sicepat_ekspres mohon direspon
RT @jawanwithpocky: Kalau ada paket sicepat yang ketuker, tolong dm ya. Terutama nanti yang terima yurugao inumaki. Ini abang sicepat nya nempel resinya salah salah
RT @jawanwithpocky: Kalau ada paket sicepat yang ketuker, tolong dm ya. Terutama nanti yang terima yurugao inumaki. Ini abang sicepat nya nempel resinya salah salah

Total data yang diambil pada tahap *crawling* adalah sebanyak 19,973 data yang mengandung *keyword* yang telah ditentukan. Distribusi dari hasil *crawling* dapat dilihat pada Tabel 3.2 dan divisualisasikan pada Gambar 3.8.

Tabel 3.2 Distribusi Hasil *Crawling*

Keywords	Total
JNE	5,342
JNT	5,291
Sicepat	4,240
Anteraja	5,100
Total	19,973

Total Crawled Data by Keywords

Gambar 3.8 Visualisasi distribusi hasil *crawling*

### 3.4 Preprocessing

Pada tahap *preprocessing* dilakukan proses pengubahan data hasil *crawling*. Proses ini dimaksudkan untuk membersihkan data agar sesuai dengan kebutuhan dan dengan mudah untuk diolah. Adapun tahapan yang akan dilakukan pada *preprocessing* telah divisualisasikan pada Gambar 3.9 dan akan dijelaskan secara lebih detail pada anak sub-bab.

Tahap *preprocessing* menggunakan fungsi *preprocessing(str)* yang ditampilkan pada Gambar 3.9 yang berisi kumpulan dari fungsi-fungsi *preprocessing* yang lebih mendetail, seperti *cleaning*, *normalization*, dan *stopword removal*. Tahap *preprocessing* secara keseluruhan dapat dilihat pada Gambar 3.2.

```
#preprocessing function
def preprocessing(str):
    str = cleaning(str)
    str = normalizing(str)
    str = removeStopword(str)
    return str

df = pd.read_csv('/content/dataset.csv') # read data result of crawling
df['preprocessed'] = df['full_text'].apply(preprocessing) #apply preprocessing
process
df = df['preprocessed']
```

Gambar 3.9 Kode program *preprocessing*

#### 3.4.1 *Cleaning*

*Cleaning* pada dasarnya dilakukan untuk membersihkan data yang kotor dan agar dapat meningkatkan efisiensi dan kemudahan pada tahap selanjutnya. Dalam melakukan pembersihan data, terdapat beberapa tahapan yang akan dilakukan. Pembersihan akan melibatkan hilangnya beberapa karakter dari dalam data yang telah dikumpulkan pada tahapan sebelumnya.

Tahapan awal dari *cleaning* dilakukan dengan menormalisasikan karakter non-ASCII, yakni karakter-karakter yang tidak dikodekan ke dalam ASCII, seperti Unicode dengan contoh huruf latin à, á, â, yang mana pada ASCII dikodekan sebagai huruf a. Normalisasi pada karakter non-ASCII dimaksudkan agar karakter-karakter yang tidak standar diubah menjadi standar. Pada implementasinya, karakter-karakter non-ASCII yang ada akan dikodekan ke dalam UTF-8, yang mana mendukung pengkodean standar Unicode yang mendukung karakter non-ASCII.

Pembersihan selanjutnya akan menghilangkan *username* (@), *retweet* (RT), angka, dan URLs yang kerap muncul pada data *tweets*. Penghapusan hal-hal tersebut dilakukan karena ketiganya tidak mengekspresikan atau menginformasikan sebuah sentimen dan tidak memberikan efek

terhadap klasifikasi yang akan dilakukan (Khader et al., 2019). Hal ini dilakukan untuk mendapatkan data yang lebih bersih yakni data inti dari *tweets* yang siap diolah untuk tahap selanjutnya.

Kemudian, pada tahap selanjutnya adalah menghilangkan karakter tanda baca atau *punctuation*. Hal ini dilakukan untuk mengurangi *noise* yang muncul pada data *tweets*. Tanda baca atau *punctuation* dianggap sebagai *noise* karena dapat menurunkan performa dari proses klasifikasi analisis sentimen yang dilakukan (Khader et al., 2019). Tanda baca yang akan dihilangkan umumnya ada sedikitnya sebanyak 16 karakter yang dapat dilihat pada Tabel 3.3. Setelah setiap tahap *cleaning* usai dilakukan, maka data *tweets* akan diubah menjadi *lowercase*. Perubahan data menjadi *lowercase* dilakukan untuk menyeragamkan seluruh data yang digunakan (Rianto et al., 2021).

Tabel 3.3 Karakter punctuation

Character	Name
.	period / full stop
?	question mark
!	exclamation point
,	comma
;	semicolon
:	colon
_	underline
-	hyphen
[]	square brackets
{}	brace
' '	apostrophe
“ ”	quotation
/	slash
...	ellipsis
*	asterisk
@	at sign

*Cleaning* dilakukan dengan membuat sebuah fungsi dengan nama *cleaning* yang memiliki parameter input sebuah *string*. Fungsi yang dibuat memanfaatkan penggunaan *library RegEx* yang membantu dalam menghilangkan berbagai karakter. Pada Gambar 3.10 disajikan fungsi *cleaning* yang digunakan oleh penulis.

```
# cleaning
def cleaning(str):
    #remove non-ascii
    str = unicodedata.normalize('NFKD', str).encode('ascii',
'ignore').decode('utf-8', 'ignore')
```



```

str = re.sub("b'|b\\", '', str)
#remove RT
str = re.sub ('RT', '', str)
#remove @username
str = re.sub('@[^\s]+', '', str)
#remove hashtag
str = re.sub('#[^\s]+', '', str)
#remove username inside brackets
str = re.sub('SENSITIVE-NO', ' ', str)
#remove URLs
str = re.sub(r'(?:i)\b(?:https?://|www\d{0,3}[.][a-z0-9.\-]+[.][a-z]{2,4}/) (?:[^\s()<>]+|\\((([^\s()<>]+|\\([^\s()<>]+\\)))*\\))+(?:\\((([^\s()<>]+|\\([^\s()<>]+\\)))*\\)|[^\s`!()\\[\]{};:\'"., <>?«»“”‘’]))', '', str)
#remove punctuations
str = re.sub(r'^\w\d#\s|_|_', ' ', str)
#remove digit from string
str = re.sub("\S*\d\S*", "", str).strip()
#remove digit or numbers
str = re.sub(r"\b\d+\b", " ", str)
#Remove additional white spaces
str = re.sub('\s\s+', ' ', str)
#to lowercase
str = str.lower()

return str

```

Gambar 3.10 Kode program cleaning

Hasil penggunaan fungsi *cleaning* ditampilkan pada Tabel 3.4. Hasil menunjukkan tidak lagi terdapat karakter non-ASCII, *username*, RT, URLs, karakter *punctuation*, *digit*, dan seluruh data telah menjadi *lowercase*. Data yang telah bersih kemudian akan masuk ke dalam tahap normalisasi.

Tabel 3.4 Hasil *Cleaning*

clean text
resi ngga dan sdh lewat batas estimasi
sejak tgl udh smpai di kota tujuan tp kurir ga ke alamat pdhl juga di kota ga dipelosok smpe capek saya tiap hari tlp customer service ga ada hasil brg saya hilang atau gimana sih weee nyari kerjaan aja luuu salah nempel resi itu angkanya beda heyyy akhirnya paket org nyasar ke gue kannnn
makanya kurir si cepat kalo pickup jangan marah marah jadi salah kan kerja nya udah di tanyain paket kemana di suruh cari ngga mau malah ngotot sendiri
halo mohon direspon
kalau ada paket sicepat yang ketuker tolong dm ya terutama nanti yang terima yurugao inumaki ini abang sicepat nya nempel resinya salah salah
kalau ada paket sicepat yang ketuker tolong dm ya terutama nanti yang terima yurugao inumaki ini abang sicepat nya nempel resinya salah salah

### 3.4.2 Normalization

Tahap normalisasi adalah sebuah tahapan untuk menormalisasikan kata-kata yang tidak baku, termasuk kata gaul atau *slang words*, menjadi kata yang baku. Hal ini dilakukan agar data *tweets*

lebih mudah dibaca pada tahap *labeling* nantinya. Normalisasi dilakukan dengan menggunakan kamus yang telah ada dibuat oleh (Ramaprakoso, 2019) dan diperkaya dengan penambahan manual secara berkala. Sampai saat ini terdapat sedikitnya 15,977 kata yang dinormalisasikan dan terus bertambah. Seluruh data mengenai kamus normalisasi disimpan ke dalam sebuah file CSV dengan nama *kamus.csv*.

Beberapa contoh normalisasi yang akan dilakukan dapat dilihat pada Tabel 3.5. Kolom *original* pada Tabel memberikan contoh-contoh kata tidak baku dan kata *slang* yang kerap digunakan di Twitter. Kata-kata tersebut kemudian mengalami perubahan ke dalam kolom *replacement*, yang mana menjadi kata baku. Tahap normalisasi juga mengubah kata-kata singkatan seperti ‘mager’ ke dalam bentuk sebenarnya, yaitu ‘malas gerak’.

Tabel 3.5 Contoh dari kamus normalisasi

Original	Replacement
mengsedih	sedih
lutuna	lucunya
menelfon	menelepon
pantengin	perhatikan
pait	pahit
becanda	bercanda
mendingan	lebih baik
lngsung	langsung
maacih	terima kasih
mager	malas gerak

Normalisasi dilakukan menggunakan fungsi yang bernama *normalizing* dan memiliki parameter input *text*. Normalisasi dilakukan dengan mengubah kata yang terdeteksi dalam kolom *original* ke dalam kolom *replacement*. Fungsi normalisasi dapat dilihat pada Gambar 3.11.

```
#normalization of words
normalisasi = pd.read_csv('kamus.csv', encoding = 'latin-1')
normalisasi_map = dict(zip(normalisasi['original'],
normalisasi['replacement']))
def normalizing(text):
    return ' '.join([normalisasi_map[word] if word in normalisasi_map else word
for word in text.split(' ')])
```

Gambar 3.11 Kode program normalisasi

Hasil dari fungsi normalisasi ditunjukkan oleh Tabel 3.6, yang mana kata-kata tidak baku, singkatan, hingga *slang words* sudah diminimalisir atau sudah tidak ada lagi. Beberapa kata yang masih belum dapat dinormalisasi diakibatkan karena adanya kata yang memiliki arti yang sama

dan akan menyebabkan ambiguitas. Keadaan ambiguitas akan menyebabkan adanya kebingungan dalam pelabelan data pada tahap selanjutnya.

Pada Tabel 3.6 juga terdapat beberapa kata yang tidak ternormalisasikan dengan baik seperti ‘heyyy’, ‘weee’, dan ‘kannnn’. Hal ini terjadi karena adanya banyaknya versi yang berbeda-beda pada setiap pengguna Twitter dan belum terdapat pada kamus karena tidak pernah melihat adanya penggunaan kata tersebut. Misalkan pada kata ‘heyyy’, seseorang dapat menulis hanya dengan satu huruf ‘y’ ataupun dapat ditambahkan dengan puluhan huruf ‘y’. Maka dari itu, kata-kata yang tidak ternormalisasikan akan ditambahkan secara manual dengan memperkaya kamus normalisasi, sehingga seluruh kata pada kalimat nantinya dapat dinormalisasikan seluruhnya dengan baik.

Tabel 3.6 Hasil Normalisasi

normalized text
resi tidak dan sudah lewat batas estimasi sejak tanggal sudah sampai di kota tujuan tetapi kurir tidak ke alamat padahal juga di kota tidak dipelosok sampai capek saya tiap hari telepon customer service tidak ada hasil barang saya hilang atau bagaimana sih
weee mencari pekerjaan saja kamu salah menempel resi itu angkanya beda heyyy akhirnya paket orang menyasar ke saya kannnn
makanya kurir sicepat kalau pickup jangan marah marah jadi salah kan kerja nya sudah di tanyakan paket kemana di suruh cari tidak mau bahkan mengotot sendiri
halo mohon direspon
kalau ada paket sicepat yang tertukar tolong direct message iya terutama nanti yang menerima yurugao inumaki ini abang sicepat nya menempel resinya salah salah
kalau ada paket sicepat yang tertukar tolong direct message iya terutama nanti yang menerima yurugao inumaki ini abang sicepat nya menempel resinya salah salah

### 3.4.3 Stopword Removal

*Stopword* pada dasarnya merupakan kata-kata yang sangat umum dan tidak memberikan informasi penting sehingga dapat diabaikan atau dibuang saja. Pada proses ini, *stopword* yang terdapat pada data *tweets* akan dihilangkan, sehingga data hanya akan menyampaikan informasi penting saja (Pradana & Hayaty, 2019). Keseluruhan *stopword* dalam bahasa Indonesia. Tahapan *stopword removal* menggunakan bantuan dari *library* Sastrawi dan menggunakan dokumen daftar *stopword* yang dimiliki MedAn (MedAn, 2020) sebanyak 973 *stopword*.

Daftar *stopword* yang berbeda kemudian digabungkan untuk memperkaya daftar *stopword*. Beberapa contoh dari *stopword* yang digunakan dapat dilihat pada Tabel 3.7. Kata-kata yang termasuk dalam daftar *stopword* nantinya akan dihilangkan di data *tweets*.

Tabel 3.7 Contoh *Stopword*

Stopwords
akhirnya
amatlah
berlebihan
dipastikan
hanyalah
kini
yang

Menghilangkan *stopword* pada tahap ini juga menggunakan sebuah fungsi *removeStopword(str)*. Fungsi ini menggunakan gabungan *stopword* yang telah ada pada Sastrawi dan milik MedAn. Guna menghilangkan *stopword* dibutuhkan *string* sebagai input pada fungsi. Fungsi *removeStopword* dapat dilihat pada Gambar 3.12. Penggunaan kamus list *stopword* dari *library* Sastrawi digunakan dengan memanggil fungsi *StopWordRemoverFactory()* dan *create\_stop\_word\_removal()* untuk membuat *stopword remover*. Sedangkan pada *stopword* yang dimiliki oleh MedAn disimpan pada file *stopwords-id.txt*, dan digunakan fungsi *get\_stopwords()* untuk menjadikan seluruh *stopword* yang ada menjadi list.

Kemudian, seluruh *stopword* yang dimiliki oleh *library* Sastrawi disimpan pada variabel *stop\_factory*, sedangkan seluruh *stopword* yang dimiliki oleh MedAn disimpan pada variabel *stop\_words*. Keduanya kemudian digabungkan dan disimpan pada variabel *data*. Selanjutnya, pada fungsi *removeStopword* dilakukan tokenisasi untuk setiap data menggunakan *word\_tokenize* milik *library* NLTK.

```

from nltk.tokenize import word_tokenize
from Sastrawi.StopWordRemover.StopWordRemoverFactory import
StopWordRemoverFactory

factoryStopword = StopWordRemoverFactory()
stopword = factoryStopword.create_stop_word_removal()
stop_factory = StopWordRemoverFactory().get_stop_words()

def get_stopwords():
    docs = pd.read_csv('/content/stopwords-id.txt', header = None, names=
['stopword'])
    docs = docs['stopword'].tolist()
    return docs

stop_words = get_stopwords()

# Merge stopword
data = set(stop_factory + stop_words)

#remove stopword
def removeStopword(str):

```

```
word_tokens = word_tokenize(str)
filtered_sentence = [w for w in word_tokens if not w in data]
return ' '.join(filtered_sentence)
```

Gambar 3.12 Kode program *stopword removal*

Hasil dari proses *stopword removal* diperlihatkan pada Tabel 3.8. Sangat berbeda dengan hasil-hasil dari tabel sebelumnya yang masih kotor dan belum dinormalisasikan, Tabel 3.8 hanya mengandung informasi pentingnya saja. Hal tersebut kemudian berakibat banyaknya kata yang dibuang karena terdeteksi sebagai *stopword*.

Tabel 3.8 Hasil *Stopword Removal*

<b>stopword removed text</b>
resi batas estimasi tanggal kota tujuan kurir alamat kota dipelosok sampai capek hari telepon customer service barang hilang sih
weee mencari pekerjaan kamu menempel resi angkanya beda heyyy paket orang menysasar kannnn
kurir sih pickup marah marah kerja tanyakan paket suruh cari mengotot sendiri halo mohon direspon
paket sicepat tertukar direct message iya menerima yurugao inumaki abang sicepat menempel resinya
paket sicepat tertukar direct message iya menerima yurugao inumaki abang sicepat menempel resinya

Usai keseluruhan tahap selesai, tidak jarang terjadi data-data yang terduplikasi akibat dari adanya normalisasi dan *stopword removal*. Data-data yang terduplikasi harus dihapuskan agar ketika *labeling* dilakukan, tidak muncul dua label yang berbeda dan kemudian menyebabkan *noise*. Pada Tabel 3.1 yang menyajikan data hasil *crawling*, terdapat dua data *tweets* yang memiliki nilai yang sama atau data duplikasi. Data duplikasi kemudian dihilangkan dengan memanggil fungsi *drop\_duplicates()*. Hal ini akan mengakibatkan hilangnya data-data yang terduplikasi dari keseluruhan data.

```
#checking duplicate data
df.duplicated().sum()

#delete duplicated data
df = df.drop_duplicates()
df.duplicated().sum() # expected result = 0
```

Gambar 3.13 Kode program menghapus data duplikasi

Pada Tabel 3.9 juga diperlihatkan tidak ada lagi data yang terduplikasi seperti tabel sebelumnya. Cuplikan kode dari menghilangkan duplikasi data dapat dilihat pada Gambar 3.13. Total data bersih yang didapatkan adalah sebanyak 8,107 data *tweets*.

Tabel 3.9 Hasil *Cleaning* Duplikasi Hasil *Stopword Removal*

cleaner text
resi batas estimasi tanggal kota tujuan kurir alamat kota dipelosok sampai capek hari telepon customer service barang hilang sih
weee mencari pekerjaan kamu menempel resi angkanya beda heyyy paket orang menyasar kannnn
kurir sih pickup marah marah kerja tanyakan paket suruh cari mengotot sendiri
halo mohon direspon
paket sicepat tertukar direct message iya menerima yurugao inumaki abang sicepat menempel resinya

### 3.5 Data Labeling

Setelah tahap *preprocessing* telah selesai, maka dilanjutkan dengan tahap *labeling* yang dilakukan oleh penulis secara manual. Tahap *labeling* dilakukan setelah tahap *preprocessing* karena data yang akan diberi label telah bersih. *Labeling* dilakukan untuk mengkategorikan atau mengklasifikasikan data yang ada ke dalam suatu kelas berdasarkan polaritasnya. Data-data yang telah melewati proses *preprocessing* akan dimasukkan ke dalam salah satu kelas dari tiga kelas yang ada, yakni *positive*, *negative*, atau *neutral*. Sayangnya, *labeling* yang dilakukan secara manual memakan banyak waktu, karena *labeling* merupakan kegiatan yang dilakukan secara iteratif atau berulang. Aktivitas *labeling* juga dilakukan berdasarkan asumsi penulis ke dalam tiga kelas label yang ada. Contoh-contoh data yang diberikan label ke dalam kelas tersebut disajikan pada Tabel 3.10.

Kalimat *positive* dalam data ekspedisi ini biasanya memberikan atau melibatkan pujian seperti paket yang datang tepat waktu, pujian kepada ekspedisi terkait, dan refleksi emosi yang sifatnya *positive*. Refleksi ini misalnya adalah kegembiraan, kesenangan, serta kebahagiaan yang dirasakan oleh pengguna terhadap ekspedisi.

Kemudian, kalimat *negative* dalam data ekspedisi biasanya berkaitan dengan keluhan, sindiran, kritik terhadap ekspedisi karena tidak sesuai ekpektasi yang diharapkan, dan refleksi atas emosi *negative*. Refleksi ini biasanya berkaitan dengan frustrasi, ketidakpuasaan, hingga kekecewaan mengenai ekspedisi yang digunakan oleh pengguna.

Terakhir, kalimat yang dianggap sebagai polaritas *neutral* pada data ekspedisi biasanya berkaitan dengan pertanyaan pengguna terharap *customer service* atau admin mengenai paket yang

dikirimkan, promosi, iklan, saran dan masukan, penawaran gratis atau *giveaway*, dan hal-hal lainnya yang tidak memiliki atau memberikan refleksi emosi yang *positive* atau *negative* terhadap ekspedisi yang digunakan.

Tabel 3.10 Data Labeled

Text	Label
aku chat wasap ke jnt kak	neutral
ayo ikutan segera jnewsonline jne	neutral
pakai sicepat sangat rekomen	positive
iya pakai anteraja cepat murah lagi kurinya sat set banget banget	positive
kalau belum bisa melayani dengan baik tutup saja jne	negative
sicepat tumben lambat paketku masih di depok sejak Kamis	negative

Selama proses *labeling* berlangsung, terjadi beberapa kebingungan. Beberapa kata yang biasanya diyakini sebagai kata yang bermakna *negative* tetapi kemudian tidak menjadikan label sebagai *negative*, hal ini dikarenakan adanya refleksi *positive* pada kalimat yang memiliki kata *negative* tersebut. Kata-kata seperti ‘anjing’ dan ‘gila’ umumnya akan dilabelkan sebagai *negative* karena memiliki konotasi yang *negative*. Tetapi, dalam beberapa kasus yang terjadi pada saat pelabelan dilakukan, kedua kata tersebut tidak masuk ke dalam kategori kelas *negative* pada beberapa data *tweets*. Kata-kata tersebut terkadang mengekspresikan atau merefleksikan emosi yang sifatnya *positive*. Sebagai contoh dari data yang mengandung kata tersebut dapat dilihat pada Tabel 3.11.

Tabel 3.11 Anomali Data Label

Text	Label
gila bro sicepat cepat banget	positive
anjing beruntung pakai anteraja cepat	positive
tumben gila jnt sampainya cepat	positive

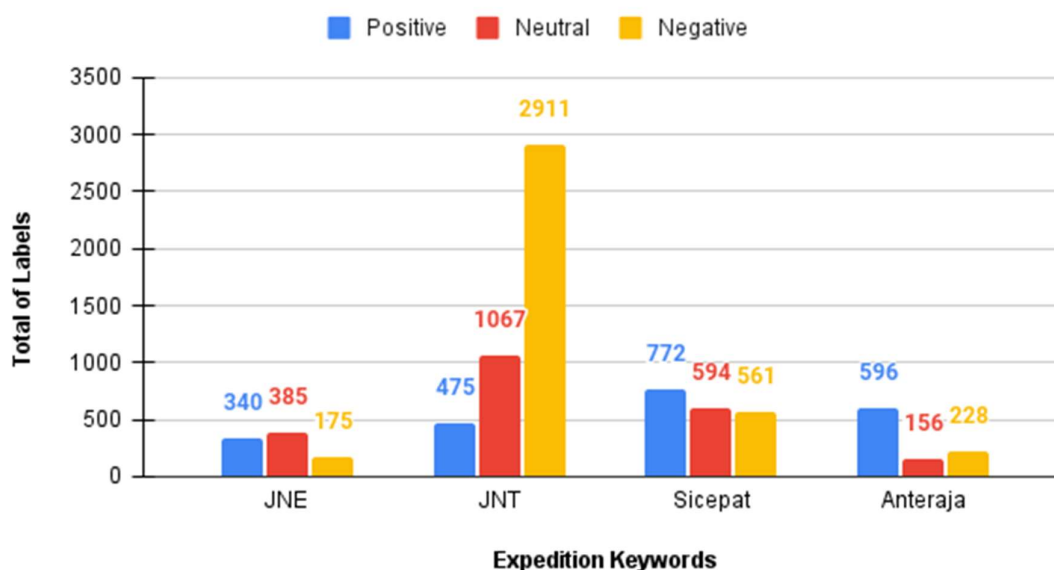
Terdapat 8,107 data *tweets* yang perlu diberikan label seperti pada Tabel 3.10. Setelah keseluruhan data diberikan label yang sesuai, maka tahapan akan dilanjutkan ke dalam tahap *modeling*. Distribusi dari data yang telah diberikan label dapat dilihat pada Tabel 3.12 dan Gambar 3.14 yang mendistribusikan gambaran label secara garis besar. Berdasarkan distribusinya, *keyword* JNT memiliki data dengan kelas *negative* dan *neutral* paling banyak dan *keyword* Sicepat memiliki data dengan kelas *positive* paling banyak. Data *negative* kerap diisi dengan kekecewaan pengguna jasa ekspedisi karena adanya paket yang *delay*, paket yang tidak bergerak atau *stuck* di suatu

daerah, dan paket yang tidak aman atau dibobol. Sedangkan data dengan kelas *positive* umumnya diisi dengan kurir ekspedisi yang ramah, pengiriman paket yang cepat, serta paket yang tiba dengan aman. Kemudian, data *neutral* umumnya diisi dengan tanya-jawab antara pengguna dengan *customer service* atau data tanpa konteks.

Tabel 3.12 Distribusi Label

Keywords	Positive	Neutral	Negative
JNE	340	385	175
JNT	475	1067	2911
Sicepat	772	594	561
Anteraja	596	156	228
Total	2145	2136	3826

### Distribution of Labels by Keywords



Gambar 3.14 Chart distribusi dari label

### 3.6 Modeling

Setelah data *tweets* bersih berhasil didapatkan lewat proses *preprocessing*, maka kemudian data *tweets* yang berisi data ekspedisi akan masuk ke tahapan selanjutnya, yakni tahap *modeling*. Tahap *modeling* dilakukan dengan menggunakan dua metode, yaitu *Bidirectional GRU* dan *Bidirectional LSTM*. Keduanya menggunakan *bidirectional neural networks* yang menyebabkan



adanya *forward* dan *backward layer*, yang dinilai lebih unggul dalam menyimpan informasi pada semantik dua arah.

*Modeling* mulanya dilakukan dengan mengkonversi label yang awalnya bertipe *string* ke dalam kategorikal. Konversi ini dapat dilihat pada Gambar 3.15. Ketiga label yang telah ada, yakni *negative*, *positive*, dan *neutral*, yang masih berbentuk *string* diubah menjadi bentuk kategorikal untuk memudahkan di saat *modeling*. Hasil yang didapatkan ialah label *negative* menjadi kategori 0, label *neutral* menjadi kategori 1, dan label *positive* menjadi kategori 2.

Pada Gambar 3.15, konversi label ke dalam kategorikal menggunakan `LabelEncoder()` dari *library* `sklearn`. *Encoding* dilakukan pada kolom *label* saja, untuk mentransformasikan label menjadi kategorikal. Hasil transformasi kemudian disimpan pada kolom *label\_id*. *Values* dari *label* dan *label\_id* kemudian disimpan pada variabel *label\_dict* sebagai *dictionary* yang menghasilkan `{0: 'neg', 1: 'neu', 2: 'pos'}`.

```
#convert label string to categorical
# Import label encoder
from sklearn import preprocessing

# label_encoder object knows how to understand word labels.
label_encoder = preprocessing.LabelEncoder()

# Encode labels in column label.
df['label_id']= label_encoder.fit_transform(df['label'])

label_dict = dict(df[['label_id', 'label']].values)
label_dict

{0: 'neg', 1: 'neu', 2: 'pos'}
```

Gambar 3.15 Konversi *string* ke dalam kategorikal

Perubahan yang terjadi pada *DataFrame* dataset ditampilkan pada Tabel 3.13. Sesuai dengan perubahan label yang mulanya *string* menjadi kategorikal yang tertampil pada kolom *label\_id*. Kemudian, kolom label tidak akan digunakan lagi pada *modeling*. Angka 0, 1, dan 2 akan menggantikan label.

Tabel 3.13 *DataFrame* Hasil Konversi

text	label	label_id
aku chat wasap ke jnt kak	neutral	1
ayo ikutan segera jnewsonline jne	neutral	1
kalaupun belum bisa melayani dengan baik tutup saja jne	negative	0
sicepat tumben lambat paketku masih di depok sejak kamis	negative	0
pakai sicepat sangat rekomen	positive	2
iya pakai anteraja cepat murah lagi kurinya sat set banget banget	positive	2

Terdapat *maxseqLen* yang merupakan maksimal dari jumlah kata yang dapat digunakan sebagai *input\_length* pada *embedding layer*. Perbedaan mendasar dari *input\_dim* dan *input\_length* adalah *input\_length* merupakan panjang dari sebuah *sequence*, sedangkan *input\_dim* merupakan angka dari *input features* dalam data. Maksimal dari *maxseqLen* dihitung dari data dengan kalimat terpanjang dari data *tweets*. Hasil dari perhitungan yang dilakukan menghasilkan nilai 70, yang merupakan maksimal panjang dari *maxseqLen*. Seluruh hal tersebut dapat dilihat pada Gambar 3.16.

```
#tokenizer
tokenizer = Tokenizer()
tokenizer.fit_on_texts(text)

#max sequential per word in NN
maxseqLen = max([len(i.split()) for i in text])
print(maxseqLen) # 70
```

Gambar 3.16 Kode program dari *tokenization* dan *maxseqLen*

Sayangnya, data yang dimiliki banyaknya tidak seimbang sehingga dilakukan *balancing dataset*. Akan ada empat luaran hasil dari sebaran dataset, yakni model BiGRU dan BiLSTM terhadap data yang tidak seimbang, dan terhadap data yang seimbang. Penyeimbangan dataset dilakukan dengan membuat jumlah setiap label menjadi sebanyak jumlah label terkecil. Berdasarkan Tabel 3.12, label *neutral* memiliki jumlah total data paling sedikit dibandingkan label *positive* dan *negative*. Maka, pada *balancing dataset*, label *positive* dan *negative* akan dikurangi hingga jumlahnya sama seperti label *neutral*.

```
#balancing dataset
label_neg, label_pos, label_neu = df['label_id'].value_counts()

dataset_label_neg = df[df['label_id'] == 0]
dataset_label_neu = df[df['label_id'] == 1]
dataset_label_pos = df[df['label_id'] == 2]
```

```

#making dataset as much as label 2
dataset_label_neg = dataset_label_neg.sample(label_neu)
dataset_label_neu = dataset_label_neu.sample(label_neu)
dataset_label_pos = dataset_label_pos.sample(label_neu)

df = pd.concat([dataset_label_neg, dataset_label_neu, dataset_label_pos],
axis=0)

print("Total sampling: ")
print(df.label.value_counts())

```

Gambar 3.17 Kode program *balancing dataset*

*Balancing dataset* diperlihatkan pada Gambar 3.17. Hasil dari *balancing dataset* yang dilakukan adalah seimbangny jumlah total dataset setiap label. Total dari *balancing dataset* adalah 6,408 data, yang setiap data labelnya memiliki sebanyak 2,136 data.

Pada Gambar 3.18 disuguhkan kode program mengenai bentuk data yang akan menjadi data *training* dan data *testing*. Variabel X menyimpan hasil dari tokenisasi dan memiliki panjang data tensor maksimal sebanyak *maxseqlen*. Kemudian, variabel y mengkategorikan label yang sebelumnya menjadi *binary* dengan tiga kelas, sebagai contoh label *neutral* yang memiliki kategori kelas 1, diubah ke dalam *binary* dengan tiga kelas menjadi [0, 1, 0]. Hal ini juga berlaku pada label *negative* dan *positive*, yang mana apabila diubah ke dalam *binary* menjadi [1, 0, 0] untuk *negative*, dan [0, 0, 1] untuk *positive*. Hasil dari dimensi data untuk ['text'] adalah (8107, 70), yang mana 8107 merupakan total data *tweets* yang digunakan, dan 70 merupakan hasil *maxseqlen*. Sedangkan pada dimensi data untuk ['label'] adalah (8107, 3), yang mana 3 merupakan kelas yang dikategorikan. Sedangkan, apabila data yang masuk adalah data yang seimbang, maka dimensi yang dihasilkan adalah (6408, 70) dan (6408, 3).

```

from keras.preprocessing.sequence import pad_sequence

X = tokenizer.texts_to_sequences(text)
X = pad_sequences(X, maxlen = maxseqlen)
y = to_categorical(label, num_classes = 3)

print("Shape of data ['text']:", X.shape) # (8107, 70)
print("Shape of data ['label']:", y.shape) # (8107, 3)

```

Gambar 3.18 Kode program mengenai data

Kemudian pada Gambar 3.19 ditampilkan kode program yang digunakan untuk memberikan *index* ke dalam setiap kata yang sebelumnya telah ditekonesiasi. Pemberian *index* ke dalam kata ditunjukkan juga hasilnya pada Gambar 3.19.

```
#tokening word index
idx_token = tokenizer.word_index

# result of idx_token
{'jnt': 1,
 'tidak': 2,
 'sudah': 3,
 'sicepat': 4,
 'di': 5,
 'ini': 6,
 'yang': 7,
 'saya': 8,
 'banget': 9,
 'pakai': 10,
 'iya': 11,
 'paket': 12,
 'sampai': 13,
 'aku': 14,
 'kalau': 15,
 'ada': 16,
 'saja': 17,
 'jne': 18,
 'anteraja': 19,
 'ke': 20,
 'sama': 21,
 'sih': 22,
 'juga': 23,
 'dari': 24,
 'lagi': 25,
 'dan': 26,
 'itu': 27,
 'nya': 28,
 'hari': 29,
 'cepat': 30,
 'lama': 31,
 'kurir': 32,
 'bisa': 33,
 ...}
```

Gambar 3.19 Hasil dari word indexing

Setelah dilakukan pemberian *index* ke dalam kata, maka dilakukanlah perhitungan dari maksimal kata yang dimiliki menggunakan hasil dari variabel *idx\_token*. Hasil dari perhitungan disimpan dalam variabel *vocab* pada Gambar 3.20 dengan maksimal 10738 kata ter-index dari 8107 data. Sedangkan, apabila data yang digunakan seimbang, maka akan menghasilkan *vocab* sebanyak 9,417.

```
#determine vocab data used
vocab = len(tokenizer.word_index) + 1
vocab
```

Gambar 3.20 Kode program vocab

Selanjutnya terdapat aktivitas *splitting* atau pembagian dataset. Pembagian dilakukan ke dalam empat variabel yang berbeda-beda dengan bantuan *train\_test\_split*, yakni *Xtrain*, *Xtest*, *ytrain*, dan *ytest*. *Splitting* dataset membagi antara data *training* dengan data *testing*. Variabel X yang sebelumnya merupakan jumlah total dari data, dibagi menjadi data *training* dan data *testing* dengan rasio 85:15, yang mana data *training* selanjutnya akan dibagi menjadi data *validation*, sehingga memerlukan rasio data yang lebih banyak (Lee et al., 2022; Piyaphakdeesakun et al., 2019). Hal yang sama juga terjadi pada variabel y yang menyimpan data label.

Hasil yang didapatkan dari adanya pembagian dataset menjadi enam bagian adalah variabel *Xtrain*, *Xtest*, *Xval*, *ytrain*, *ytest*, dan *yval* ditunjukkan pada Gambar 3.21. Variabel *Xtrain* dan *ytrain* merupakan jumlah dari data *training* sebanyak 85% dari total data. Variabel *Xtest* dan *ytest* merupakan jumlah dari data *testing* sebanyak 15% dari total data. Sedangkan *Xval* dan *yval* merupakan jumlah dari data *validation* sebanyak 15% dari total data *training*.

```
from sklearn.model_selection import train_test_split

#split dataset
XXtrain, Xtest, ytrain, ytest = train_test_split(X, y, test_size = 0.15) #
training 85%, testing 15%
Xval, Xtrain, yval, ytrain = train_test_split(Xtrain, ytrain, test_size = 0.85)
# training 70%, validation 15%

print(Xtrain.shape, ytrain.shape) #(5857, 70) (5857, 3)
print(Xtest.shape, ytest.shape) #(1217, 70) (1217, 3)
print(Xval.shape, yval.shape) #(1033, 70) (1033, 3)
```

Gambar 3.21 Kode program *splitting* data

Hasil yang didapatkan dari adanya pembagian dataset dengan data *imbalance* dan *balance* ditunjukkan pada Tabel 3.14. Data *training* yang dimiliki oleh data *imbalance* adalah 5.857 dan data *balance* sebanyak 4930. Kemudian, data *testing* yang dimiliki oleh data *imbalance* adalah 1.217 dan data *balance* sebanyak 962. Sedangkan, data *validation* yang dimiliki oleh data *imbalance* adalah 1.033, dan data *balance* sebanyak 816. Maka, total data yang dimiliki oleh data *imbalance* adalah 8.107 dan data *balance* adalah 6.048

Tabel 3.14 Distribusi *splitting data*

Variable	Imbalance	Balance
Xtrain, ytrain	5857	4930
Xtest, ytest	1217	962
Xval, yval	1033	816
Total	8107	6048

Untuk melakukan *modeling*, maka digunakan beberapa parameter inisial. Parameter inilah yang akan diubah apabila *hyperparameter* dilakukan, yakni *units*, *dropout*, *learning rate*, *batch size*, dan *epoch*. Parameter inisial dapat dilihat pada Tabel 3.15. Pemilihan *hyperparameter initial* dan *tuning* berdasarkan beberapa penelitian mengenai *units*, *dropout*, *learning rate*, *batch size*, dan *epoch*.

Tabel 3.15 *Initial Parameter*

Keywords	Total
<i>Units</i>	512
<i>Dropout</i>	0.5
<i>Learning rate</i>	0.001
<i>Batch size</i>	64
<i>Epoch</i>	25

Untuk melakukan *modeling*, maka digunakan beberapa parameter inisial. Parameter inilah yang akan diubah apabila *hyperparameter* dilakukan, yakni *units*, *dropout*, *learning rate*, *batch size*, dan *epoch*.

### 3.6.1 *Bidirectional GRU*

Setelah seluruh variable pada Tabel 3.15 yang dibutuhkan telah diinisiasi, maka *modeling* dapat dilakukan. Model yang dibangun merupakan sebuah *model sequential* dengan *bidirectional neural networks* GRU. Pada embedding layer, diisi dengan variabel *vocab* sebagai *input\_dim*, 128 sebagai *output\_dim*, variabel *maxseqlen* sebagai *input\_length*. Lalu, pada *bidirectional neural network* yang menggunakan GRU, digunakan 512 *unit* GRU. Setelahnya digunakan beberapa *hidden* dan *dropout layer* seperti yang tertera pada Gambar 3.22.

```
#architecture model
model = Sequential()
model.add(Embedding(input_dim = vocab, output_dim = 128, input_length =
maxseqlen))
model.add(Bidirectional(GRU(512, return_sequences = True)))
model.add(GlobalMaxPool1D())
model.add(Dense(128, activation='relu')) # dense 1
```

```

model.add(Dropout(0.5))
model.add(Dense(64, activation='relu')) # dense 2
model.add(Dropout(0.5))
model.add(Dense(3, activation='softmax')) # output dense

opt = tensorflow.keras.optimizers.RMSprop(learning_rate=0.001)
model.compile(optimizer=opt, loss='categorical_crossentropy',
metrics=['acc'])
model.summary()

```

Gambar 3.22 Kode program *modeling* BiGRU

Setelah arsitektur dari model dibangun, selanjutnya akan dilakukan konfigurasi model untuk *training* dengan *model.compile()*. Konfigurasi yang dilakukan beberapa parameter seperti *optimizer*, *loss*, dan *metrics*. *Optimizer* merupakan *instance* dari optimasi, yang mana pada model yang telah dibangun mengimplementasikan RMSprop. RMSprop dipilih karena terbukti memiliki kinerja yang baik, terutama cocok untuk menangani NLP seperti klasifikasi sentimen (Yu et al., 2019). Kemudian, *loss function* menggunakan *categorical\_crossentropy* yang diperuntukkan untuk klasifikasi *multi-class* (Kowsher et al., 2022) dan bertujuan untuk menghitung *crossentropy loss* yang terjadi diantara label dan prediksinya. Sedangkan, *metrics function* merupakan daftar *metrics* yang akan dievaluasi oleh model selama *training* berlangsung. Untuk mengetahui ringkasan dari arsitektur yang dibuat, maka digunakan *model.summary()* yang hasilnya dapat dilihat pada Gambar 3.24, Gambar 3.26, Gambar 3.29, dan Gambar 3.31.

*Modeling* akan dilakukan ke dalam dua tipe dataset, yakni dengan data yang tidak seimbang atau *imbalance* dan dengan data yang seimbang atau *balance*. Pembagian data ditunjukkan pada Tabel 3.14. Sebelum *training* dimulai, terdapat variable *checkpoint* yang akan menyimpan model terbaik dan memonitor *val\_acc* dengan fungsi *ModelCheckpoint*. *Callbacks checkpoint* akan menyimpan model terbaik ke dalam direktori '*update*'. *Training* dilakukan menggunakan fungsi *model.fit()*. Data *training* yang telah dibagi sebelumnya akan menjadi salah satu parameter dari *training*. Terdapat validasi data sebanyak 0.15 yang telah disimpan pada variable *Xval* dan *yval* untuk memvalidasi dataset saat *training* dilakukan. Kode *training* dapat dilihat pada Gambar 3.23.

```

# callbacks checkpoint to save only best model and monitor val_acc score
checkpoint = ModelCheckpoint(
    'update/merged-imbalance-bigru-v1.h5',
    monitor='val_acc',
    save_best_only=True,
    verbose=0
)

# training model
start_time = datetime.now()
history = model.fit(Xtrain, ytrain,

```

```

        batch_size = 64, epochs = 25, shuffle = True,
        validation_data = (Xval, yval), verbose = 1,
        callbacks=[checkpoint])
end_time = datetime.now()
print("Time out: {}".format(end_time - start_time))

```

Gambar 3.23 Kode program model *training*

#### a. *Bidirectional GRU Imbalance*

Pada *modeling* yang menggunakan data yang tidak seimbang sesuai dengan distribusinya pada Tabel 3.14 dan dijalankan sesuai dengan Gambar 3.22, maka digunakanlah *model.summary()* untuk menghasilkan *summary* dari arsitektur model. Arsitektur model dapat dilihat pada Gambar 3.24. Jumlah unit yang digunakan dalam GRU adalah sebanyak 512, tetapi pada hasil *summary* ditampilkan penggunaan sebanyak 1024. Hal ini disebabkan oleh penggunaan *bidirectional*, yang mana melakukan penangkapan informasi secara *forward* dan *backward*.

Layer (type)	Output Shape	Param #
embedding_6 (Embedding)	(None, 70, 128)	1374464
bidirectional_6 (Bidirectional)	(None, 70, 1024)	1972224
global_max_pooling1d_6 (GlobalMaxPooling1D)	(None, 1024)	0
dense_18 (Dense)	(None, 128)	131200
dropout_12 (Dropout)	(None, 128)	0
dense_19 (Dense)	(None, 64)	8256
dropout_13 (Dropout)	(None, 64)	0
dense_20 (Dense)	(None, 3)	195
=====		
Total params: 3,486,339		
Trainable params: 3,486,339		
Non-trainable params: 0		

Gambar 3.24 Hasil dari *Model Summary Imbalance BiGRU*

Kemudian, dilakukan *training* dengan menggunakan kode program dari Gambar 3.23. Hasil dari *training* yang dilakukan diperlihatkan pada Gambar 3.25. *Metric* yang digunakan adalah *loss*, *acc*, *val\_loss*, dan *val\_acc*. Kedua *metric* validasi merupakan hasil dari validasi data sebanyak 0.15



yang dijelaskan sebelumnya dan diperlihatkan jumlahnya pada Tabel 3.14. *Training* dilakukan sebanyak 25 *epochs* dengan total waktu 1 menit 13 detik.

```

Epoch 1/25
Epoch 1/25
92/92 [=====] - 7s 38ms/step - loss: 1.1613 - acc:
0.4393 - val_loss: 1.0294 - val_acc: 0.4811
Epoch 2/25
92/92 [=====] - 3s 30ms/step - loss: 0.9256 - acc:
0.5575 - val_loss: 0.8378 - val_acc: 0.5915
Epoch 3/25
92/92 [=====] - 3s 30ms/step - loss: 0.7929 - acc:
0.6111 - val_loss: 0.7735 - val_acc: 0.6418
Epoch 4/25
92/92 [=====] - 3s 29ms/step - loss: 0.7267 - acc:
0.6356 - val_loss: 0.8514 - val_acc: 0.5663
Epoch 5/25
92/92 [=====] - 3s 29ms/step - loss: 0.6705 - acc:
0.6541 - val_loss: 0.7686 - val_acc: 0.6321
Epoch 6/25
92/92 [=====] - 3s 29ms/step - loss: 0.6363 - acc:
0.6788 - val_loss: 0.8112 - val_acc: 0.6254
Epoch 7/25
92/92 [=====] - 3s 31ms/step - loss: 0.6003 - acc:
0.7127 - val_loss: 0.7701 - val_acc: 0.6854
Epoch 8/25
92/92 [=====] - 3s 30ms/step - loss: 0.5532 - acc:
0.7577 - val_loss: 0.7204 - val_acc: 0.6970
Epoch 9/25
92/92 [=====] - 3s 30ms/step - loss: 0.4999 - acc:
0.7893 - val_loss: 0.7408 - val_acc: 0.7270
Epoch 10/25
92/92 [=====] - 3s 29ms/step - loss: 0.4581 - acc:
0.8163 - val_loss: 0.7742 - val_acc: 0.7047
Epoch 11/25
92/92 [=====] - 3s 29ms/step - loss: 0.4169 - acc:
0.8405 - val_loss: 0.7941 - val_acc: 0.7260
Epoch 12/25
92/92 [=====] - 3s 29ms/step - loss: 0.3720 - acc:
0.8603 - val_loss: 0.7475 - val_acc: 0.7260
Epoch 13/25
92/92 [=====] - 3s 31ms/step - loss: 0.3422 - acc:
0.8767 - val_loss: 0.8154 - val_acc: 0.7183
Epoch 14/25
92/92 [=====] - 3s 32ms/step - loss: 0.3104 - acc:
0.8871 - val_loss: 0.9575 - val_acc: 0.7183
Epoch 15/25
92/92 [=====] - 3s 30ms/step - loss: 0.2723 - acc:
0.9008 - val_loss: 1.0278 - val_acc: 0.7212
Epoch 16/25
92/92 [=====] - 3s 30ms/step - loss: 0.2525 - acc:
0.9093 - val_loss: 1.0688 - val_acc: 0.7222
Epoch 17/25
92/92 [=====] - 3s 30ms/step - loss: 0.2215 - acc:
0.9192 - val_loss: 1.2181 - val_acc: 0.7212
Epoch 18/25

```

```

92/92 [=====] - 3s 30ms/step - loss: 0.2046 - acc:
0.9271 - val_loss: 1.4479 - val_acc: 0.7260
Epoch 19/25
92/92 [=====] - 3s 29ms/step - loss: 0.1878 - acc:
0.9312 - val_loss: 1.3805 - val_acc: 0.7260
Epoch 20/25
92/92 [=====] - 3s 30ms/step - loss: 0.1679 - acc:
0.9361 - val_loss: 2.3120 - val_acc: 0.7009
Epoch 21/25
92/92 [=====] - 3s 29ms/step - loss: 0.1639 - acc:
0.9401 - val_loss: 1.4243 - val_acc: 0.7115
Epoch 22/25
92/92 [=====] - 3s 29ms/step - loss: 0.1416 - acc:
0.9474 - val_loss: 1.8063 - val_acc: 0.7144
Epoch 23/25
92/92 [=====] - 3s 29ms/step - loss: 0.1326 - acc:
0.9503 - val_loss: 1.9154 - val_acc: 0.7018
Epoch 24/25
92/92 [=====] - 4s 40ms/step - loss: 0.1345 - acc:
0.9501 - val_loss: 1.9182 - val_acc: 0.7338
Epoch 25/25
92/92 [=====] - 3s 29ms/step - loss: 0.1188 - acc:
0.9549 - val_loss: 2.1994 - val_acc: 0.7086
Time out: 0:01:13.610384

```

Gambar 3.25 Hasil dari *training model imbalance BiGRU*

#### b. Bidirectional GRU Balance

Selanjutnya, dilakukan *training* terhadap data yang tidak seimbang atau *balance* sesuai dengan distribusi data pada Tabel 3.14 dan sesuai dengan kode program Gambar 3.22. Arsitektur model BiGRU untuk data yang seimbang diperlihatkan pada Gambar 3.26.

Model: "sequential\_2"

Layer (type)	Output Shape	Param #
dense_0 (Embedding)	(None, 70, 128)	1374464
dense_1 (Bidirectional)	(None, 70, 1024)	1972224
dense_2 (GlobalMaxPooling1D)	(None, 1024)	0
dense_3 (Dense)	(None, 128)	131200
dense_4 (Dropout)	(None, 128)	0
dense_5 (Dense)	(None, 64)	8256
dense_6 (Dropout)	(None, 64)	0
dense_7 (Dense)	(None, 8)	520
dense_8 (Dropout)	(None, 8)	0
dense_9 (Dense)	(None, 3)	27

```

=====
Total params: 3,486,691
Trainable params: 3,486,691
Non-trainable params: 0

```

Gambar 3.26 Hasil dari *Model Summary Balance BiGRU*

Hasil dari *training* yang dilakukan diperlihatkan pada Gambar 3.27. *Metric* yang digunakan adalah *loss*, *acc*, *val\_loss*, dan *val\_acc*. Kedua *metric* validasi merupakan hasil dari validasi data sebanyak 0.15 yang dijelaskan sebelumnya dan diperlihatkan jumlahnya pada Tabel 3.14. *Training* dilakukan sebanyak 25 *epochs* dengan total waktu 1 menit 5 detik.

```

Epoch 1/25
73/73 [=====] - 13s 53ms/step - loss: 0.9999 - acc:
0.4903 - val_loss: 0.7983 - val_acc: 0.6581
Epoch 2/25
73/73 [=====] - 2s 33ms/step - loss: 0.7358 - acc:
0.7017 - val_loss: 0.7414 - val_acc: 0.6569
Epoch 3/25
73/73 [=====] - 3s 38ms/step - loss: 0.5949 - acc:
0.7717 - val_loss: 0.6771 - val_acc: 0.7096
Epoch 4/25
73/73 [=====] - 2s 29ms/step - loss: 0.4914 - acc:
0.8173 - val_loss: 0.9389 - val_acc: 0.5821
Epoch 5/25
73/73 [=====] - 2s 31ms/step - loss: 0.4083 - acc:
0.8531 - val_loss: 1.0300 - val_acc: 0.6556
Epoch 6/25
73/73 [=====] - 2s 29ms/step - loss: 0.3410 - acc:
0.8752 - val_loss: 0.7862 - val_acc: 0.7010
Epoch 7/25
73/73 [=====] - 2s 29ms/step - loss: 0.2881 - acc:
0.8998 - val_loss: 0.8205 - val_acc: 0.6850
Epoch 8/25
73/73 [=====] - 2s 30ms/step - loss: 0.2606 - acc:
0.9065 - val_loss: 0.9937 - val_acc: 0.6985
Epoch 9/25
73/73 [=====] - 2s 30ms/step - loss: 0.2116 - acc:
0.9253 - val_loss: 1.0462 - val_acc: 0.6777
Epoch 10/25
73/73 [=====] - 2s 29ms/step - loss: 0.1732 - acc:
0.9410 - val_loss: 1.2435 - val_acc: 0.6863
Epoch 11/25
73/73 [=====] - 2s 30ms/step - loss: 0.1565 - acc:
0.9428 - val_loss: 1.2277 - val_acc: 0.6863
Epoch 12/25
73/73 [=====] - 2s 29ms/step - loss: 0.1317 - acc:
0.9505 - val_loss: 1.4430 - val_acc: 0.6605
Epoch 13/25
73/73 [=====] - 2s 30ms/step - loss: 0.1230 - acc:
0.9566 - val_loss: 1.4337 - val_acc: 0.6936
Epoch 14/25

```

```

73/73 [=====] - 2s 30ms/step - loss: 0.1031 - acc:
0.9603 - val_loss: 1.6770 - val_acc: 0.6887
Epoch 15/25
73/73 [=====] - 2s 30ms/step - loss: 0.0904 - acc:
0.9639 - val_loss: 1.7954 - val_acc: 0.6973
Epoch 16/25
73/73 [=====] - 2s 28ms/step - loss: 0.0915 - acc:
0.9646 - val_loss: 1.2620 - val_acc: 0.6654
Epoch 17/25
73/73 [=====] - 2s 28ms/step - loss: 0.0860 - acc:
0.9650 - val_loss: 2.1055 - val_acc: 0.6728
Epoch 18/25
73/73 [=====] - 2s 28ms/step - loss: 0.0774 - acc:
0.9721 - val_loss: 2.0119 - val_acc: 0.6752
Epoch 19/25
73/73 [=====] - 2s 28ms/step - loss: 0.0654 - acc:
0.9695 - val_loss: 2.3083 - val_acc: 0.6777
Epoch 20/25
73/73 [=====] - 2s 28ms/step - loss: 0.0751 - acc:
0.9711 - val_loss: 1.9410 - val_acc: 0.6801
Epoch 21/25
73/73 [=====] - 2s 28ms/step - loss: 0.0701 - acc:
0.9702 - val_loss: 2.7910 - val_acc: 0.6814
Epoch 22/25
73/73 [=====] - 2s 28ms/step - loss: 0.0673 - acc:
0.9728 - val_loss: 2.1504 - val_acc: 0.6973
Epoch 23/25
73/73 [=====] - 2s 28ms/step - loss: 0.0548 - acc:
0.9747 - val_loss: 3.2133 - val_acc: 0.6667
Epoch 24/25
73/73 [=====] - 2s 28ms/step - loss: 0.0600 - acc:
0.9741 - val_loss: 2.0300 - val_acc: 0.6801
Epoch 25/25
73/73 [=====] - 2s 29ms/step - loss: 0.0609 - acc:
0.9745 - val_loss: 2.8798 - val_acc: 0.6716
Time out: 0:01:05.426132

```

Gambar 3.27 Hasil dari *training model balance BiGRU*

Setelah *training* selesai dilakukan, maka selanjutnya akan dilakukan evaluasi untuk mengetahui performa. Evaluasi akan dilakukan menggunakan beberapa matriks, seperti *precision*, *recall*, *f1-score*, dan *accuracy* yang akan dijelaskan pada sub-bab selanjutnya.

### 3.6.2 Bidirectional LSTM

Pada pembangunan arsitektur model dengan Bidirectional LSTM, hal yang sama juga dilakukan seperti pada arsitektur model BiGRU sebelumnya. Arsitektur model yang ditunjukkan pada Gambar 3.22 memiliki beberapa variabel dan parameter yang juga sama. *Optimizer* yang digunakan tidak berbeda seperti sebelumnya. Perbedaan hanya terletak pada *bidirectional layer*, karena menggunakan metode yang berbeda, yakni metode LSTM. Kode program yang digunakan untuk *modeling* diperlihatkan pada Gambar 3.28.

```

#architecture model
model = Sequential()
model.add(Embedding(input_dim = vocab, output_dim = 128, input_length =
maxseqlen))
model.add(Bidirectional(GRU(512, return_sequences = True)))
model.add(GlobalMaxPool1D())
model.add(Dense(128, activation='relu')) # dense 1
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu')) # dense 2
model.add(Dropout(0.5))
model.add(Dense(3, activation='softmax')) # output dense

opt = tensorflow.keras.optimizers.RMSprop(learning_rate=0.001)
model.compile(optimizer=opt, loss='categorical_crossentropy',
metrics=['acc'])
model.summary()

```

Gambar 3.28 Kode program *modeling* BiLSTM

#### a. *Bidirectional LSTM Imbalance*

Pada *modeling* yang menggunakan data yang tidak seimbang sesuai dengan distribusinya pada Tabel 3.14 dan dijalankan sesuai dengan Gambar 3.28, maka digunakanlah *model.summary()* untuk menghasilkan *summary* dari arsitektur model. Arsitektur model dapat dilihat pada Gambar 3.29. Jumlah unit yang digunakan dalam LSTM adalah sebanyak 512, tetapi pada hasil *summary* ditampilkan penggunaan sebanyak 1024. Hal ini disebabkan oleh penggunaan *bidirectional*, yang mana melakukan penangkapan informasi secara *forward* dan *backward*.

Model: "sequential\_4"

Layer (type)	Output Shape	Param #
embedding_4 (Embedding)	(None, 70, 128)	1374464
bidirectional_4 (Bidirectional)	(None, 70, 1024)	2625536
global_max_pooling1d_4 (GlobalMaxPooling1D)	(None, 1024)	0
dense_12 (Dense)	(None, 128)	131200
dropout_8 (Dropout)	(None, 128)	0
dense_13 (Dense)	(None, 64)	8256
dropout_9 (Dropout)	(None, 64)	0
dense_14 (Dense)	(None, 3)	195

```

=====
Total params: 4,139,651
Trainable params: 4,139,651
Non-trainable params: 0

```

Gambar 3.29 Hasil dari *Model Summary Imbalance BiLSTM*

Kemudian, dilakukan *training* dengan menggunakan kode program dari Gambar 3.23. Hasil dari *training* yang dilakukan diperlihatkan pada Gambar 3.30. *Metric* yang digunakan adalah *loss*, *acc*, *val\_loss*, dan *val\_acc*. Kedua *metric* validasi merupakan hasil dari validasi data sebanyak 0.15 yang dijelaskan sebelumnya dan diperlihatkan jumlahnya pada Tabel 3.14. *Training* dilakukan sebanyak 25 *epochs* dengan total waktu 2 menit 21 detik.

```

Epoch 1/25
92/92 [=====] - 4s 47ms/step - loss: 0.6096 - acc:
0.7902 - val_loss: 0.6430 - val_acc: 0.7318
Epoch 2/25
92/92 [=====] - 4s 42ms/step - loss: 0.4490 - acc:
0.8327 - val_loss: 0.6947 - val_acc: 0.7396
Epoch 3/25
92/92 [=====] - 4s 43ms/step - loss: 0.3947 - acc:
0.8579 - val_loss: 0.7642 - val_acc: 0.7309
Epoch 4/25
92/92 [=====] - 4s 41ms/step - loss: 0.3302 - acc:
0.8813 - val_loss: 0.8368 - val_acc: 0.7270
Epoch 5/25
92/92 [=====] - 3s 38ms/step - loss: 0.2985 - acc:
0.8974 - val_loss: 0.9188 - val_acc: 0.6738
Epoch 6/25
92/92 [=====] - 3s 38ms/step - loss: 0.2531 - acc:
0.9146 - val_loss: 0.8967 - val_acc: 0.7222
Epoch 7/25
92/92 [=====] - 3s 36ms/step - loss: 0.2087 - acc:
0.9264 - val_loss: 1.1224 - val_acc: 0.6893
Epoch 8/25
92/92 [=====] - 3s 35ms/step - loss: 0.1889 - acc:
0.9382 - val_loss: 1.3377 - val_acc: 0.6757
Epoch 9/25
92/92 [=====] - 3s 35ms/step - loss: 0.1609 - acc:
0.9496 - val_loss: 1.1623 - val_acc: 0.7280
Epoch 10/25
92/92 [=====] - 3s 35ms/step - loss: 0.1425 - acc:
0.9522 - val_loss: 1.3756 - val_acc: 0.7125
Epoch 11/25
92/92 [=====] - 3s 35ms/step - loss: 0.1278 - acc:
0.9585 - val_loss: 1.3923 - val_acc: 0.7193
Epoch 12/25
92/92 [=====] - 3s 35ms/step - loss: 0.8459 - acc:
0.9375 - val_loss: 1.3873 - val_acc: 0.7164
Epoch 13/25
92/92 [=====] - 3s 35ms/step - loss: 0.0997 - acc:
0.9659 - val_loss: 1.7272 - val_acc: 0.7096
Epoch 14/25

```

```

92/92 [=====] - 3s 35ms/step - loss: 0.1086 - acc:
0.9655 - val_loss: 1.6149 - val_acc: 0.7096
Epoch 15/25
92/92 [=====] - 3s 34ms/step - loss: 0.0865 - acc:
0.9694 - val_loss: 2.0039 - val_acc: 0.6931
Epoch 16/25
92/92 [=====] - 3s 34ms/step - loss: 0.0882 - acc:
0.9682 - val_loss: 2.1865 - val_acc: 0.7067
Epoch 17/25
92/92 [=====] - 3s 34ms/step - loss: 0.0911 - acc:
0.9701 - val_loss: 1.9568 - val_acc: 0.6980
Epoch 18/25
92/92 [=====] - 3s 34ms/step - loss: 0.0721 - acc:
0.9725 - val_loss: 2.3682 - val_acc: 0.6999
Epoch 19/25
92/92 [=====] - 3s 34ms/step - loss: 0.0688 - acc:
0.9723 - val_loss: 1.7873 - val_acc: 0.7086
Epoch 20/25
92/92 [=====] - 3s 34ms/step - loss: 0.0630 - acc:
0.9749 - val_loss: 2.4221 - val_acc: 0.6960
Epoch 21/25
92/92 [=====] - 4s 39ms/step - loss: 0.0630 - acc:
0.9761 - val_loss: 2.8420 - val_acc: 0.6931
Epoch 22/25
92/92 [=====] - 3s 34ms/step - loss: 0.0565 - acc:
0.9781 - val_loss: 2.8560 - val_acc: 0.7009
Epoch 23/25
92/92 [=====] - 3s 34ms/step - loss: 0.0491 - acc:
0.9790 - val_loss: 2.9062 - val_acc: 0.6989
Epoch 24/25
92/92 [=====] - 3s 34ms/step - loss: 0.0555 - acc:
0.9770 - val_loss: 2.9497 - val_acc: 0.7076
Epoch 25/25
92/92 [=====] - 3s 34ms/step - loss: 0.0493 - acc:
0.9790 - val_loss: 2.5834 - val_acc: 0.7076
Time out: 0:02:21.962275

```

Gambar 3.30 Hasil dari *training model imbalance* BiLSTM

#### b. Bidirectional LSTM Balance

Selanjutnya, dilakukan *training* terhadap data yang tidak seimbang atau *balance* sesuai dengan distribusi data pada Tabel 3.14 dan sesuai dengan kode program Gambar 3.28. Arsitektur model BiLSTM untuk data yang seimbang diperlihatkan pada Gambar 3.31.

Model: "sequential\_3"

Layer (type)	Output Shape	Param #
embedding_3 (Embedding)	(None, 70, 128)	1189376
bidirectional_3 (Bidirectional)	(None, 70, 1024)	2625536
global_max_pooling1d_3 (GlobalMaxPooling1D)	(None, 1024)	0

dense_9 (Dense)	(None, 128)	131200
dropout_6 (Dropout)	(None, 128)	0
dense_10 (Dense)	(None, 64)	8256
dropout_7 (Dropout)	(None, 64)	0
dense_11 (Dense)	(None, 3)	195
=====		
Total params: 3,954,563		
Trainable params: 3,954,563		
Non-trainable params: 0		

Gambar 3.31 Hasil dari *Model Summary Balance BiLSTM*

Hasil dari *training* yang dilakukan diperlihatkan pada Gambar 3.32. *Metric* yang digunakan adalah *loss*, *acc*, *val\_loss*, dan *val\_acc*. Kedua *metric* validasi merupakan hasil dari validasi data sebanyak 0.15 yang dijelaskan sebelumnya dan diperlihatkan jumlahnya pada Tabel 3.14. *Training* dilakukan sebanyak 25 *epochs* dengan total waktu 1 menit 13 detik.

Epoch 1/25	73/73 [=====] - 11s 69ms/step - loss: 1.0886 - acc: 0.4259 - val_loss: 0.8518 - val_acc: 0.5968
Epoch 2/25	73/73 [=====] - 3s 37ms/step - loss: 0.8216 - acc: 0.6382 - val_loss: 0.7002 - val_acc: 0.6936
Epoch 3/25	73/73 [=====] - 3s 38ms/step - loss: 0.6541 - acc: 0.7428 - val_loss: 0.6596 - val_acc: 0.7120
Epoch 4/25	73/73 [=====] - 3s 38ms/step - loss: 0.5461 - acc: 0.7940 - val_loss: 0.6785 - val_acc: 0.7279
Epoch 5/25	73/73 [=====] - 3s 35ms/step - loss: 0.4564 - acc: 0.8270 - val_loss: 0.7699 - val_acc: 0.6789
Epoch 6/25	73/73 [=====] - 2s 34ms/step - loss: 0.3918 - acc: 0.8637 - val_loss: 0.9687 - val_acc: 0.6409
Epoch 7/25	73/73 [=====] - 2s 34ms/step - loss: 0.3328 - acc: 0.8743 - val_loss: 0.8630 - val_acc: 0.6912
Epoch 8/25	73/73 [=====] - 2s 33ms/step - loss: 0.2892 - acc: 0.9004 - val_loss: 1.0980 - val_acc: 0.6838
Epoch 9/25	73/73 [=====] - 2s 34ms/step - loss: 0.2530 - acc: 0.9123 - val_loss: 0.8837 - val_acc: 0.6814
Epoch 10/25	73/73 [=====] - 2s 34ms/step - loss: 0.2139 - acc: 0.9238 - val_loss: 1.1789 - val_acc: 0.6875
Epoch 11/25	



```

73/73 [=====] - 2s 34ms/step - loss: 0.1895 - acc:
0.9341 - val_loss: 1.2966 - val_acc: 0.6789
Epoch 12/25
73/73 [=====] - 2s 34ms/step - loss: 0.1910 - acc:
0.9335 - val_loss: 1.4783 - val_acc: 0.6471
Epoch 13/25
73/73 [=====] - 2s 34ms/step - loss: 0.1565 - acc:
0.9438 - val_loss: 1.6053 - val_acc: 0.6703
Epoch 14/25
73/73 [=====] - 2s 34ms/step - loss: 0.1424 - acc:
0.9479 - val_loss: 1.5677 - val_acc: 0.6703
Epoch 15/25
73/73 [=====] - 3s 37ms/step - loss: 0.1260 - acc:
0.9540 - val_loss: 1.6115 - val_acc: 0.6838
Epoch 16/25
73/73 [=====] - 3s 40ms/step - loss: 0.1075 - acc:
0.9607 - val_loss: 1.8696 - val_acc: 0.6581
Epoch 17/25
73/73 [=====] - 2s 34ms/step - loss: 0.1071 - acc:
0.9598 - val_loss: 1.9404 - val_acc: 0.6544
Epoch 18/25
73/73 [=====] - 2s 34ms/step - loss: 0.0927 - acc:
0.9613 - val_loss: 2.3994 - val_acc: 0.6630
Epoch 19/25
73/73 [=====] - 2s 34ms/step - loss: 0.0823 - acc:
0.9670 - val_loss: 2.2898 - val_acc: 0.6495
Epoch 20/25
73/73 [=====] - 2s 34ms/step - loss: 0.0871 - acc:
0.9639 - val_loss: 1.2896 - val_acc: 0.6765
Epoch 21/25
73/73 [=====] - 2s 34ms/step - loss: 0.0813 - acc:
0.9663 - val_loss: 2.1354 - val_acc: 0.6630
Epoch 22/25
73/73 [=====] - 3s 35ms/step - loss: 0.0736 - acc:
0.9695 - val_loss: 2.4124 - val_acc: 0.6471
Epoch 23/25
73/73 [=====] - 3s 34ms/step - loss: 0.0700 - acc:
0.9708 - val_loss: 2.7561 - val_acc: 0.6642
Epoch 24/25
73/73 [=====] - 2s 34ms/step - loss: 0.0618 - acc:
0.9708 - val_loss: 3.3096 - val_acc: 0.6691
Epoch 25/25
73/73 [=====] - 2s 34ms/step - loss: 0.0811 - acc:
0.9700 - val_loss: 2.6792 - val_acc: 0.6801
Time out: 0:01:12.451712

```

Gambar 3.32 Hasil dari *training model balance BiGRU*

Setelah *training* selesai dilakukan, maka akan dilakukan evaluasi untuk mengukur performa menggunakan beberapa matriks seperti *metric* akurasi, *precision*, *f1-score* dan *recall*. Evaluasi akan dijelaskan pada sub-bab selanjutnya.

### 3.7 Hyperparameter Tuning

*Hyperparameter* dilakukan dengan beberapa skenario, antara lain dengan mengubah nilai *units*, *dropout*, *learning rate*, *batch size*, dan *epoch*. Skenario yang dilakukan diperlihatkan pada Tabel 3.16. Skenario *hyperparameter tuning* dilakukan setelah mendapatkan *base model*; yang dibangun menggunakan *initial hyperparameter* dengan kinerja terbaik terhadap empat model, yakni *imbalance BiGRU*, *imbalance BiLSTM*, *balance BiGRU*, dan *balance BiLSTM*.

Tabel 3.16 Skenario *Hyperparameter Tuning*

Skenario	Sub Skenario	Hyperparameter Settings
A	A.1	units = 256
	A.2	units = 512
	A.3	units = 1024
B	B.1	dropout = 0.2
	B.2	dropout = 0.5
	B.3	dropout = 0.8
C	C.1	learning rate = 0.01
	C.2	learning rate = 0.001
	C.3	learning rate = 0.0001
D	D.1	batch size = 64
	D.2	batch size = 128
	D.3	batch size = 256
E	E.1	epoch = 25
	E.2	epoch = 35
	E.3	epoch = 45

Beberapa *hyperparameter* yang akan diubah adalah *units*, *dropout*, *learning rate*, *batch size*, dan *epoch*. Pada skenario A digunakan *initial hyperparameter setting*, hanya dilakukan perubahan nilai *units*. Pada skenario ini akan dicari nilai *units* yang paling optimal untuk digunakan pada skenario selanjutnya. *Units* adalah *hyperparameter* yang menjelaskan mengenai jumlah unit yang masuk. *Hyperparameter* terhadap *units* akan dilakukan pada beberapa konfigurasi, yakni 256 *units*, 512 *units*, dan 1024 *units* (Corallo et al., 2022; Ndikumana et al., 2018; A. H. Uddin et al., 2019).

Pada skenario B digunakan *hyperparameter setting* optimal yang dihasilkan dari skenario A. Pada skenario ini akan dicari nilai *dropout* yang paling optimal untuk digunakan pada skenario selanjutnya. *Dropout* adalah *hyperparameter* yang merupakan sebuah teknik yang digunakan untuk mengabaikan *neuron* secara acak pada saat training. *Hyperparameter* tuning akan dilakukan dengan tiga konfigurasi, yakni 0.2, 0.5, dan 0.8 (ByungSoo et al., 2017; Srivastava, 2013).

Kemudian, pada skenario C digunakan *hyperparameter setting* optimal yang dihasilkan dari skenario B. Pada skenario ini akan dicari nilai *learning rate* yang paling optimal. *Learning rate*

adalah *hyperparameter* yang dikonfigurasi yang memiliki nilai positif yang kecil. Eksperimen dilakukan terhadap tiga konfigurasi *learning rate*, antara lain adalah 0.01, 0.001, dan 0.0001 (Li et al., 2021; M. J. Uddin et al., 2022).

Selanjutnya, pada skenario D digunakan *hyperparameter setting* optimal yang dihasilkan dari skenario C. Pada skenario ini akan dicari nilai *batch size* yang paling optimal. *Batch size* adalah *hyperparameter* yang digunakan setiap kali melakukan *forward* dan *backward pass*. Eksperimen terhadap *batch size* akan dilakukan dengan tiga konfigurasi, yakni 64, 128, dan 256 (Ay Karakuş et al., 2018; Shalini et al., 2018).

Terakhir, *epoch* adalah *hyperparameter* yang akan dilakukan dengan menggunakan tiga konfigurasi, yakni 25, 35, dan 45. Dapat dilihat pada Tabel 3.15 dan hasilnya secara detail dijelaskan pada anak sub-bab selanjutnya.

Hasil keseluruhan dari *hyperparameter tuning* yang dilakukan diperlihatkan pada Tabel 3.17. Diperlihatkan bahwa *units* 1024 memberikan kinerja terbaik. Kemudian, *learning rate* sebesar 0.0001, dan *epoch* sebesar 35 juga memberikan kinerja terbaik. Sedangkan pada *dropout*, kinerja terbaik diberikan pada *dropout* sebesar 0.5 dan 0.8. Dan, *batch size* terbaik diberikan oleh *batch size* sebesar 64 dan 128. Berdasarkan hasil dari matriks evaluasi yang digunakan selama *hyperparameter tuning*, didapatkan bahwa model dengan data yang seimbang atau *balance* memiliki nilai matriks evaluasi yang lebih baik dibandingkan dengan model yang memiliki data yang tidak seimbang.

Tabel 3.17 Summary of Hyperparameter Tuning

Hyperparameter	Imbalance BiGRU	Imbalance BiLSTM	Balance BiGRU	Balance BiLSTM
<i>Units</i>	1024	1024	1024	1024
<i>Dropout</i>	0.8	0.5	0.8	0.8
<i>Learning rate</i>	0.0001	0.0001	0.0001	0.0001
<i>Batch size</i>	64	128	64	64
<i>Epoch</i>	35	35	35	35

### 3.7.1 Imbalance BiGRU

*Hyperparameter tuning* yang dilakukan terhadap data *imbalance* menggunakan model BiGRU diperlihatkan pada Tabel 3.18. Pada tabel tersebut juga diperlihatkan hasil dari matriks akurasi, *f1-score*, *precision*, dan *recall* terhadap setiap *hyperparameter* yang dilakukan.

Tabel 3.18 *Imbalance BiGRU Hyperparameter Tuning*

Kode	Sub Skenario	Hyperparameter Settings	Accuracy	F1-score	Precision	Recall	Training time per epoch
A	A.1	units = 256	0.684	0.687	0.695	0.684	1s 16ms/step
	A.2	units = 512	0.727	0.723	0.723	0.727	3s 32ms/step
	<b>A.3</b>	<b>units = 1024</b>	<b>0.698</b>	<b>0.703</b>	<b>0.721</b>	<b>0.698</b>	<b>7s 73ms/step</b>
B	B.1	dropout = 0.2	0.678	0.676	0.687	0.678	7s 75ms/step
	B.2	dropout = 0.5	0.684	0.687	0.695	0.684	1s 16ms/step
	<b>B.3</b>	<b>dropout = 0.8</b>	<b>0.722</b>	<b>0.728</b>	<b>0.753</b>	<b>0.722</b>	<b>7s 75ms/step</b>
C	C.1	learning rate = 0.01	0.468	0.298	0.219	0.468	6s 67ms/step
	C.2	learning rate = 0.001	0.708	0.704	0.703	0.708	7s 73ms/step
	<b>C.3</b>	<b>learning rate = 0.0001</b>	<b>0.718</b>	<b>0.714</b>	<b>0.716</b>	<b>0.718</b>	<b>7s 75ms/step</b>
D	<b>D.1</b>	<b>batch size = 64</b>	<b>0.718</b>	<b>0.714</b>	<b>0.716</b>	<b>0.718</b>	<b>7s 75ms/step</b>
	D.2	batch size = 128	0.681	0.680	0.692	0.681	6s 129ms/step
	D.3	batch size = 256	0.624	0.587	0.627	0.636	6s 241ms/step
E	E.1	epoch = 25	0.624	0.587	0.627	0.636	6s 241ms/step
	<b>E.2</b>	<b>epoch = 35</b>	<b>0.694</b>	<b>0.699</b>	<b>0.729</b>	<b>0.694</b>	<b>7s 76ms/step</b>
	E.3	epoch = 45	0.690	0.696	0.711	0.690	7s 77ms/step

Model kinerja terbaik dihasilkan oleh model BiGRU dengan data *imbalance* dengan *hyperparameter setting* sesuai Tabel 3.18 dengan akurasi 69.4%, *f1-score* 69.9%, *recall* 72.9%, dan *precision* 69.4%. Hasil *hyperparameter tuning* dengan parameter data *imbalance* terbaik ditunjukkan pada Tabel 3.19, dengan nilai *units* 1024, *dropout* 0.8, *learning rate* 0.0001, *batch size* 64, dan *epoch* 35.

Tabel 3.19 Hasil *Hyperparameter Tuning Imbalance BiGRU*

Hyperparameter	Nilai
<i>Units</i>	1024
<i>Dropout</i>	0.8
<i>Learning rate</i>	0.0001
<i>Batch size</i>	64
<i>Epoch</i>	35

### 3.7.2 Imbalance BiLSTM

*Hyperparameter tuning* yang dilakukan terhadap data *imbalance* menggunakan model BiLSTM diperlihatkan pada Tabel 3.20. Pada tabel tersebut juga diperlihatkan hasil dari matriks akurasi, *f1-score*, *precision*, dan *recall* terhadap setiap *hyperparameter* yang dilakukan.

Tabel 3.20 *Imbalance BiLSTM Hyperparameter Tuning*

Kode	Sub Skenario	Hyperparameter Settings	Accuracy	F1-score	Precision	Recall	Training time per epoch
A	A.1	units = 256	0.653	0.653	0.654	0.653	1s 15ms/step
	A.2	units = 512	0.674	0.671	0.671	0.674	2s 29ms/step
	<b>A.3</b>	<b>units = 1024</b>	<b>0.676</b>	<b>0.678</b>	<b>0.686</b>	<b>0.676</b>	<b>6s 76ms/step</b>
B	B.1	dropout = 0.2	0.672	0.670	0.675	0.672	6s 75ms/step
	B.2	dropout = 0.5	0.674	0.671	0.671	0.674	2s 29ms/step
	<b>B.3</b>	<b>dropout = 0.8</b>	<b>0.688</b>	<b>0.689</b>	<b>0.693</b>	<b>0.688</b>	<b>5s 75ms/step</b>
C	C.1	learning rate = 0.01	0.318	0.154	0.101	0.318	5s 69ms/step
	C.2	learning rate = 0.001	0.674	0.671	0.671	0.674	2s 29ms/step
	<b>C.3</b>	<b>learning rate = 0.0001</b>	<b>0.693</b>	<b>0.692</b>	<b>0.692</b>	<b>0.693</b>	<b>6s 78ms/step</b>
D	<b>D.1</b>	<b>batch size = 64</b>	<b>0.691</b>	<b>0.690</b>	<b>0.704</b>	<b>0.612</b>	<b>6s 77ms/step</b>
	D.2	batch size = 128	0.634	0.631	0.634	0.634	6s 152ms/step
	D.3	batch size = 256	0.599	0.558	0.658	0.599	6s 294ms/step
E	E.1	epoch = 25	0.691	0.690	0.704	0.612	6s 77ms/step
	<b>E.2</b>	<b>epoch = 35</b>	<b>0.707</b>	<b>0.705</b>	<b>0.710</b>	<b>0.707</b>	<b>6s 76ms/step</b>
	E.3	epoch = 45	0.680	0.680	0.688	0.680	6s 76ms/step

Model kinerja terbaik dihasilkan oleh model BiLSTM dengan data *imbalance* dengan *hyperparameter setting* sesuai Tabel 3.20 dengan akurasi 70.7%, *f1-score* 70.5%, *recall* 71.0%, dan *precision* 70.7%. Hasil *hyperparameter tuning* dengan parameter data *imbalance* terbaik ditunjukkan pada Tabel 3.21, dengan nilai *units* 1024, *dropout* 0.8, *learning rate* 0.0001, *batch size* 64, dan *epoch* 35.

Tabel 3.21 Hasil *Hyperparameter Tuning Imbalance BiLSTM*

Hyperparameter	Nilai
<i>Units</i>	1024
<i>Dropout</i>	0.8
<i>Learning rate</i>	0.0001
<i>Batch size</i>	64
<i>Epoch</i>	35

### 3.7.3 Balance BiGRU

*Hyperparameter tuning* yang dilakukan terhadap data *balance* menggunakan model BiGRU diperlihatkan pada Tabel 3.22. Pada tabel tersebut juga diperlihatkan hasil dari matriks akurasi, *f1-score*, *precision*, dan *recall* terhadap setiap *hyperparameter* yang dilakukan.

Tabel 3.22 Balance BiGRU Hyperparameter Tuning

Kode	Sub Skenario	Hyperparameter Settings	Accuracy	F1-score	Precision	Recall	Training time per epoch
A	A.1	units = 256	0.660	0.659	0.665	0.660	1s 19step/ms
	A.2	units = 512	0.726	0.726	0.729	0.726	3s 36ms/step
	<b>A.3</b>	<b>units = 1024</b>	<b>0.729</b>	<b>0.726</b>	<b>0.725</b>	<b>0.729</b>	<b>7s 79ms/step</b>
B	B.1	dropout = 0.2	0.699	0.702	0.750	0.699	7s 78ms/step
	B.2	dropout = 0.5	<b>0.729</b>	<b>0.726</b>	<b>0.725</b>	<b>0.729</b>	<b>7s 79ms/step</b>
	<b>B.3</b>	<b>dropout = 0.8</b>	0.694	0.695	0.716	0.694	7s 78ms/step
C	C.1	learning rate = 0.01	0.457	0.287	0.209	0.457	7s 73ms/step
	C.2	learning rate = 0.001	0.694	0.695	0.716	0.694	7s 78ms/step
	<b>C.3</b>	<b>learning rate = 0.0001</b>	<b>0.703</b>	<b>0.704</b>	<b>0.704</b>	<b>0.703</b>	<b>7s 80ms/step</b>
D	<b>D.1</b>	<b>batch size = 64</b>	0.703	0.704	0.704	0.703	7s 80ms/step
	D.2	batch size = 128	0.701	0.706	0.715	0.701	7s 144ms/step
	D.3	batch size = 256	0.698	0.678	0.679	0.680	7s 293ms/step
E	E.1	epoch = 25	0.698	0.678	0.679	0.680	7s 293ms/step
	<b>E.2</b>	<b>epoch = 35</b>	<b>0.701</b>	<b>0.702</b>	<b>0.704</b>	<b>0.701</b>	<b>6s 193ms/step</b>
	E.3	epoch = 45	0.688	0.687	0.688	0.688	6s 139ms/step

Model kinerja terbaik dihasilkan oleh model BiGRU dengan data *imbalance* dengan *hyperparameter setting* sesuai Tabel 3.22 dengan akurasi 70.1%, *f1-score* 70.2%, *recall* 70.4%, dan *precision* 70.1%. Hasil *hyperparameter tuning* dengan parameter data *imbalance* terbaik ditunjukkan pada Tabel 3.23, dengan nilai *units* 1024, *dropout* 0.5, *learning rate* 0.001, *batch size* 128, dan *epoch* 35.

Tabel 3.23 Hasil Hyperparameter Tuning Balance BiGRU

Hyperparamter	Nilai
<i>Units</i>	1024
<i>Dropout</i>	0.5
<i>Learning rate</i>	0.001
<i>Batch size</i>	128
<i>Epoch</i>	35

### 3.7.4 Balance BiLSTM

*Hyperparameter tuning* yang dilakukan terhadap data *balance* menggunakan model BiLSTM diperlihatkan pada Tabel 3.24. Pada tabel tersebut juga diperlihatkan hasil dari matriks akurasi, *f1-score*, *precision*, dan *recall* terhadap setiap *hyperparameter* yang dilakukan.

Tabel 3.24 *Balance BiLSTM Hyperparameter Tuning*

Kode	Sub Skenario	Hyperparameter Settings	Accuracy	F1-score	Precision	Recall	Training time per epoch
A	A.1	units = 256	0.661	0.662	0.669	0.661	7s 91ms/step
	A.2	units = 512	0.670	0.662	0.682	0.670	2s 32ms/step
	<b>A.3</b>	<b>units = 1024</b>	<b>0.667</b>	<b>0.666</b>	<b>0.666</b>	<b>0.667</b>	<b>7s 91ms/step</b>
B	B.1	dropout = 0.2	0.655	0.654	0.654	0.655	7s 91ms/step
	B.2	dropout = 0.5	0.667	0.666	0.666	0.667	7s 91ms/step
	<b>B.3</b>	<b>dropout = 0.8</b>	<b>0.660</b>	<b>0.662</b>	<b>0.700</b>	<b>0.660</b>	<b>7s 92ms/step</b>
C	C.1	learning rate = 0.01	0.350	0.182	0.123	0.350	7s 80ms/step
	C.2	learning rate = 0.001	0.660	0.662	0.700	0.660	7s 92ms/step
	<b>C.3</b>	<b>learning rate = 0.0001</b>	<b>0.652</b>	<b>0.648</b>	<b>0.652</b>	<b>0.652</b>	<b>7s 92ms/step</b>
D	<b>D.1</b>	<b>batch size = 64</b>	<b>0.652</b>	<b>0.648</b>	<b>0.652</b>	<b>0.652</b>	<b>7s 92ms/step</b>
	D.2	batch size = 128	0.603	0.604	0.663	0.603	6s 163ms/step
	D.3	batch size = 256	0.575	0.519	0.563	0.575	6s 299ms/step
E	E.1	epoch = 25	0.575	0.519	0.563	0.575	6s 299ms/step
	<b>E.2</b>	<b>epoch = 35</b>	<b>0.719</b>	<b>0.721</b>	<b>0.724</b>	<b>0.719</b>	<b>6s 80ms/step</b>
	E.3	epoch = 45	0.660	0.655	0.696	0.660	6s 79ms/step

Model kinerja terbaik dihasilkan oleh model BiLSTM dengan data *imbalance* dengan *hyperparameter setting* sesuai Tabel 3.24 dengan akurasi 71.9%, *f1-score* 72.1%, *recall* 72.4%, dan *precision* 71.9%. Hasil *hyperparameter tuning* dengan parameter data *imbalance* terbaik ditunjukkan pada Tabel 3.25, dengan nilai *units* 1024, *dropout* 0.8, *learning rate* 0.0001, *batch size* 64, dan *epoch* 35.

Tabel 3.25 Hasil *Hyperparameter Tuning Balance BiLSTM*

Hyperparameter	Nilai
<i>Units</i>	1024
<i>Dropout</i>	0.8
<i>Learning rate</i>	0.0001
<i>Batch size</i>	64
<i>Epoch</i>	35

### 3.8 Evaluasi

Perhitungan akurasi dari performa model dilakukan dengan menggunakan fungsi *evaluation()* yang dapat dilihat pada Gambar 3.33. Fungsi *evaluation* memiliki tiga parameter, yakni model, X, dan y. X dan y merupakan variabel yang menyimpan data teks dan data label. Sedangkan model menyimpan hasil *training* yang telah dilakukan oleh model sebelumnya.

```
#build eveluation function
def evaluation(model, X, Y):
    global Y_pred, Y_act
    Y_pred = model.predict(X)
```

```

Y_pred = np.argmax(Y_pred, axis=1)
Y_act = np.argmax(Y, axis=1)

accuracy = accuracy_score(Y_act, Y_pred)
return accuracy

```

Gambar 3.33 Kode program fungsi evaluasi

Untuk mengetahui nilai akurasi dapat menggunakan kode program yang berada pada Gambar 3.34, yang mana menunjukkan penggunaan fungsi *evaluation()*. Data *testing* yang berupa *Xtest* dan *ytest* menjadi input parameter dari fungsi *evaluation()* yang diperoleh dari Gambar 3.21 dan diperlihatkan distribusinya pada Tabel 3.14 Distribusi *splitting data* Tabel 3.14. Sedangkan, parameter *model* pada fungsi *evaluation()* didapatkan dari hasil pembuatan model BiGRU pada Gambar 3.22 dan model BiLSTM pada Gambar 3.28. Hasil dari akurasi akan ditampilkan dalam desimal tiga angka dalam persen.

```

accuracy = evaluation(model, Xtest, ytest)
print('accuracy: %.3f' % (accuracy * 100), '%')

```

Gambar 3.34 Kode program untuk mengetahui akurasi

Matriks *precision*, *recall*, dan *f1-score* juga digunakan untuk mengetahui performa dari model yang ada. Digunakan *confusion\_matrix* untuk mengetahui hasil dari *confusion matrix* dan *classification\_report* untuk mengetahui nilai dari beberapa matriks lain seperti *f1-score*, *recall*, dan *precision*. Gambar 3.35 menampilkan kode program terkait *confusion matrix* dan *classification report*.

```

from sklearn.metrics import classification_report, confusion_matrix
target = ['neu', 'neg', 'pos']
print(confusion_matrix(Y_act, Y_pred))
print(classification_report(Y_act, Y_pred, target_names = target))

```

Gambar 3.35 Kode program confusion matrix dan classification report

Karena menggunakan dua metode yang berbeda, yakni BiGRU dan BiLSTM, maka terdapat pula hasil yang berbeda. Hasil yang akan dibahas pada sub-bab evaluasi hanya hasil *modeling* yang memberikan hasil paling baik, yakni hasil dengan data yang seimbang pada kedua metode, *balance BiGRU* dan *balance BiLSTM*.



### 3.8.1 Bidirectional GRU

Menggunakan kode program yang tertera pada Gambar 3.34 dan Gambar 3.35, maka didapatkan hasil akurasi pada penggunaan model BiGRU sebesar 70,1%. Kemudian, hasil *confusion matrix*, *precision*, dan *recall* diperlihatkan pada Tabel 3.26. Berdasarkan hasil yang didapat menunjukkan sebagai berikut:

- Model dapat memprediksi bahwa data *testing* dengan label *neutral* dan memprediksi dengan benar sebagai label *neutral* sebanyak 249 data. Prediksi label *neutral* sebagai *negative* sebanyak 24 data, dan prediksi label *neutral* sebagai *positive* sebanyak 40 data. *Precision* pada label *neutral* sebanyak 0.67. *Recall* pada label *neutral* sebanyak 0.80. *F1-score* pada label *neutral* sebanyak 0.73
- Model dapat memprediksi bahwa data *testing* dengan label *negative* dan memprediksi dengan benar sebagai label *negative* sebanyak 184 data. Prediksi label *negative* sebagai *neutral* sebanyak 72 data, dan prediksi label *negative* sebagai *positive* sebanyak 87. *Precision* pada label *negative* sebanyak 0.70. *Recall* pada label *negative* sebanyak 0.54. *F1-score* pada label *negative* sebanyak 0.61.
- Model dapat memprediksi bahwa data *testing* dengan label *positive* dan memprediksi dengan benar sebagai label *positive* sebanyak 202 data. Prediksi label *positive* sebagai *neutral* sebanyak 48, dan prediksi label *positive* sebagai *neutral* sebanyak 56. *Precision* pada label *positive* sebanyak 0.61. *Recall* pada label *positive* sebanyak 0.66. *F1-score* pada label *positive* sebanyak 0.64.

Tabel 3.26 Confusion Matrix BiGRU

Actual	Prediction			Precision	Recall	F1-score
	neu	neg	pos			
neu	249	24	40	0.67	0.80	0.73
neg	72	184	87	0.70	0.54	0.61
pos	48	56	202	0.61	0.66	0.64

	precision	recall	f1-score	support
neg	0.67	0.80	0.73	313
neu	0.70	0.54	0.61	343
pos	0.61	0.66	0.64	306
accuracy			0.66	962
macro avg	0.66	0.66	0.66	962
weighted avg	0.66	0.66	0.66	962

Gambar 3.36 Hasil classification report model BiGRU

Kemudian untuk hasil *classification report* yang mengandung matriks *precision*, *recall*, *f1-score*, serta *accuracy* dapat dilihat pada Gambar 3.36. Variabel *support* menjelaskan jumlah dataset pada spesifik kategori yang digunakan sebagai *testing*. Penjabaran dari hasil adalah sebagai berikut:

- Pada label *neutral*, dihasilkan nilai matriks *precision* sebesar 0.67, nilai matriks *recall* sebesar 0.80, *f1-score* sebesar 0.73, dan jumlah data *support* sebesar 313.
- Pada label *negative*, dihasilkan nilai matriks *precision* sebesar 0.70, nilai matriks *recall* sebesar 0.54, *f1-score* sebesar 0.66, dan jumlah data *support* sebesar 343.
- Pada label *positive*, dihasilkan nilai matriks *precision* sebesar 0.61, nilai matriks *recall* sebesar 0.66, *f1-score* sebesar 0.64, dan jumlah data *support* sebesar 306.

Model yang dibuat kemudian dicoba ke dalam data baru yang tertampil pada Tabel 3.27. Diberikan enam data baru yang memuat keseluruhan label. Dari data yang diberikan, model BiGRU mampu memprediksi seluruh label dengan benar.

Tabel 3.27 Hasil data baru BiGRU

text	true label	pred label
Gilakk tumben dah sampenya lama banget	negative	negative
Gilakkk tumben dah sampe	positive	positive
Ongkir Surabaya Jogja berapa min	neutral	neutral
Aduh ekspedisinya ruwet banget, gak sampe-sampe	negative	negative
Alhamdulillah kurir di tempatku Amanah sih	positive	positive
Halo minnn, tolong cek dm ku	neutral	neutral

### 3.8.2 Bidirectional LSTM

Menggunakan kode program yang tertera pada Gambar 3.34 dan Gambar 3.35, maka didapatkan hasil akurasi pada penggunaan model BiLSTM sebesar 71,9%. Kemudian, hasil *confusion matrix*, *precision*, *f1-score*, dan *recall* diperlihatkan pada Tabel 3.28. Berdasarkan hasil yang didapat menunjukkan sebagai berikut:

- Model dapat memprediksi bahwa data *testing* dengan label *neutral* dan memprediksi dengan benar dengan label *neutral* sebanyak 224 data. Prediksi label *neutral* sebagai *negative* sebanyak 44 data, dan prediksi label *neutral* sebagai *positive* sebanyak 55 data. *Precision* pada label *neutral* sebanyak 0.74. *Recall* pada label *neutral* sebanyak 0.69. *F1-score* pada label *neutral* sebanyak 0.72.

- b. Model dapat memprediksi bahwa data *testing* dengan label *negative* dan memprediksi dengan benar dengan label *negative* sebanyak 193 data. Prediksi label *negative* sebagai *neutral* sebanyak 39 data, dan prediksi label *negative* sebagai *positive* sebanyak 92. *Precision* pada label *negative* sebanyak 0.66. *Recall* pada label *negative* sebanyak 0.60. *F1-score* pada label *negative* sebanyak 0.63.
- c. Model dapat memprediksi bahwa data *testing* dengan label *positive* dan memprediksi dengan benar dengan label *positive* sebanyak 222 data. Prediksi label *positive* sebagai *neutral* sebanyak 39, dan prediksi label *positive* sebagai *neutral* sebanyak 54. *Precision* pada label *positive* sebanyak 0.60. *Recall* pada label *positive* sebanyak 0.70. *F1-score* pada label *positive* sebanyak 0.65.

Tabel 3.28 Confusion Matrix BiLSTM

Actual	Prediction			Precision	Recall	F1-score
	neu	neg	pos			
neu	224	44	55	0.74	0.69	0.72
neg	39	193	92	0.66	0.60	0.63
pos	39	54	222	0.60	0.70	0.65

	precision	recall	f1-score	support
neg	0.74	0.69	0.72	323
neu	0.66	0.60	0.63	324
pos	0.60	0.70	0.65	315
accuracy			0.66	962
macro avg	0.67	0.66	0.66	962
weighted avg	0.67	0.66	0.66	962

Gambar 3.37 Hasil classification model BiLSTM

Kemudian untuk hasil *classification report* yang mengandung matriks *precision*, *recall*, *f1-score*, serta *accuracy* dapat dilihat pada Gambar 3.37. Variabel *support* menjelaskan jumlah dataset pada spesifik kategori yang digunakan sebagai testing. Penjabaran dari hasil adalah sebagai berikut:

- Pada label *neutral*, dihasilkan nilai matriks *precision* sebesar 0.74, nilai matriks *recall* sebesar 0.69, *f1-score* sebesar 0.72, dan jumlah data *support* sebesar 323.
- Pada label *negative*, dihasilkan nilai matriks *precision* sebesar 0.66, nilai matriks *recall* sebesar 0.60, *f1-score* sebesar 0.63, dan jumlah data *support* sebesar 324.
- Pada label *positive*, dihasilkan nilai matriks *precision* sebesar 0.60, nilai matriks *recall* sebesar 0.70, *f1-score* sebesar 0.65, dan jumlah data *support* sebesar 315.

Model yang dibuat kemudian dicoba ke dalam data baru yang tertampil pada Tabel 3.29. Diberikan enam data baru yang memuat keseluruhan label. Dari data yang diberikan, model BiLSTM mampu memprediksi lima label dengan benar.

Tabel 3.29 Hasil dengan data baru BiLSTM

<b>text</b>	<b>true label</b>	<b>pred label</b>
Gilakk tumben dah sampenya lama banget	negative	negative
Gilakkk tumben dah sampe	positive	negative
Ongkir Surabaya Jogja berapa min	neutral	neutral
Aduh ekspedisinya ruwet banget, gak sampe-sampe	negative	negative
Alhamdulillah kurir di tempatku Amanah sih	positive	positive
Halo minnn, tolong cek dm ku	neutral	neutral

## BAB IV REFLEKSI PELAKSANAAN MAGANG

### 4.1 Relevansi Akademik

Selama menjalani periode magang, penulis mendapatkan banyak pembelajaran. Terdapat juga beberapa hal yang dapat direfleksikan, seperti pengalaman dalam *preprocessing* dan pengalaman dalam data *labeling*.

#### 4.1.1 Pengalaman dalam *Preprocessing*

Tidak dapat dipungkiri bahwa tahap *preprocessing* merupakan tahap yang esensial dalam melakukan analisis, terlebih jika dataset yang digunakan belum bersih. Data yang bersih dapat memberikan akurasi dan performa yang lebih baik saat *modeling* dilakukan. Penulis dipercaya dalam melakukan *preprocessing* terhadap setiap data yang diberikan dan digunakan untuk analisis berbagai topik lainnya. Akibatnya terjadi perkembangan karakter penulis dalam ketelitian dan menjadi lebih detail. Hal ini disebabkan adanya berbagai sub-tahap yang terjadi dalam *preprocessing*, khususnya pada proses *normalization* dan proses *stopwords*.

Proses penyusunan kamus normalisasi akan lebih mudah dilakukan dengan cara menyimpan seluruh kata dari hasil *idx\_token* ke dalam *DataFrame*. Hal ini dilakukan untuk menghemat waktu dalam pembuatan dan memperkaya kamus normalisasi, dibandingkan dengan berulang kali melakukan pengecekan terhadap hasil yang belum ternormalisasi. Proses *normalization* yang terjadi juga memberikan sumbangsih dalam ketelitian. Hal ini karena adanya banyak singkatan atau *slang words* yang diadopsi oleh masyarakat, sehingga kamus normalisasi perlu untuk selalu *up-to-date*.

Sedangkan, pada proses *stopwords removal*, beberapa dataset terkadang tidak memerlukan proses *stopwords removal*. Yang mana proses tersebut, berdasarkan dataset ekspedisi, beberapa kata yang penting pada refleksi emosi *negative* menjadi hilang, seperti ‘tidak’, ‘salah’, dan ‘jangan’. Penghilangan proses *stopwords removal* juga sering dilakukan karena memudahkan proses *data labeling*, hal ini disebabkan karena label diberikan kepada data yang lengkap.

#### 4.1.2 Pengalaman dalam *Data Labeling*

*Labeling* merupakan kegiatan untuk memberikan kategori terhadap data. Pada proses *labeling*, penulis mendapatkan perkembangan dalam kesabaran dan ketelitian. Pelabelan data juga

merupakan proses yang sangat repetitif. Dalam proses *labeling*, penulis kerap membandingkan dari beberapa hasil *preprocessing* yang berbeda-beda. Salah satunya adalah hasil *preprocessing* tanpa menggunakan *stopwords*, yang mana hasilnya dapat lebih mudah diberikan label dibandingkan hasil *preprocessing* yang menggunakan *stopwords*. Hal ini disebabkan karena hilangnya kata hubung atau kata yang sering muncul tanpa informasi penting di dalamnya. Beberapa data kemudian kehilangan penjelasannya sehingga penulis kesusahan dalam memberikan label.

Aktivitas *labeling* yang dilakukan setelah tahap *preprocessing* bertujuan untuk meringankan pekerjaan penulis, karena hanya perlu untuk memberikan label terhadap 8,107 data bersih hasil *preprocessing*. Sedangkan, apabila proses *labeling* dilakukan sebelum adanya proses *preprocessing*, penulis perlu memberikan label terhadap 19,973 data.

## **4.2 Pembelajaran Magang**

Selain mendapatkan refleksi mengenai relevansi akademis mengenai *sentiment analysis* yang dilakukan selama periode pemagangan, penulis juga mendapatkan beberapa pengalaman pembelajaran magang yang dapat direfleksikan seperti manfaat adanya magang, kendala, hambatan dan tantangan yang dijumpai selama pemagangan dilakukan.

### **4.2.1 Manfaat Magang**

Selama pemagangan dilakukan, penulis mendapatkan banyak manfaat. Beberapa di antaranya adalah bertambahnya wawasan mengenai topik NLP hingga perkembangan karakter individu penulis. Wawasan yang bertambah tidak hanya mengenai *sentiment analysis* saja, tetapi juga ke dalam beberapa topik riset lainnya, seperti topik ambiguitas yang kerap dijumpai dalam *preprocessing* dan *data labeling*. Adanya kejadian ambiguitas pada *sentiment analysis* dapat ditangani dengan *Part-of-Speech Tagging* (POS Tag).

Karakter yang berkembang bukanlah hal yang dapat dihindari ketika belajar. Karakter penulis mengalami perkembangan yang baik, terutama pada ketelitian dan kesabaran. Ketelitian terus berkembang dan menjadi salah satu keharusan saat berurusan dengan data. Mulai dari pembersihannya, pengkategorian data, hingga melatih data. Ketelitian juga selalu berjalan beriringan dengan kesabaran, hal ini dimaksudkan bahwa proses-proses yang terjadi selama *sentiment analysis*, seluruh tahapannya memerlukan kesabaran yang besar. Kesabaran yang besar juga merupakan akibat karena proyek yang dilakukan juga merupakan riset yang dilakukan oleh MedAn, sehingga banyak *trials* dan *errors* yang terjadi.

#### 4.2.2 Kendala dan Hambatan Magang

Selain adanya manfaat yang didapatkan oleh penulis, terhadap kendala dan hambatan yang dihadapi oleh penulis selama magang dilaksanakan. Kendala yang dihadapi selama magang adalah keterbatasan komunikasi, dan gangguan koneksi. Sedangkan, hambatan dialami pada saat tahap *labeling*.

Magang yang dilaksanakan dengan konsep *work from home* atau secara *remote*, menyebabkan adanya keterbatasan dalam komunikasi. Keterbatasan komunikasi disebabkan karena adanya informasi yang tidak dapat diakses secara langsung, seperti permasalahan dalam melakukan *data labeling*, atau saat adanya permasalahan pada *modeling*.

Kemudian, kendala dalam gangguan koneksi menyebabkan *modeling* yang dilakukan terganggu, karena tahapan tersebut dilaksanakan menggunakan Google Colaboratory, yang mana membutuhkan koneksi Internet. Hal ini, juga memberikan efek dalam keterbatasan dalam komunikasi. Selain menyebabkan *modeling* terganggu, kendala dalam gangguan koneksi juga mengganggu komunikasi penulis apabila terdapat diskusi menggunakan *platform* yang memerlukan koneksi seperti Google Meet, atau Telegram.

Hambatan yang terjadi selama periode magang terjadi pada pelabelan data. Hal ini disebabkan ada banyaknya data yang perlu diberikan label. Sebagian besar waktu dihabiskan pada tahapan tersebut, hal ini disebabkan karena perilaku pengerjaan yang iteratif. Pelabelan juga menjadi hambatan karena dilakukan berdasarkan asumsi penulis saja.

#### 4.2.3 Tantangan Magang

Selama pemagangan dilakukan, penulis juga mendapatkan tantangan. Tantangan yang dihadapi adalah saat penulis harus menaikkan performa akurasi dari model BiGRU dan BiLSTM. Untuk meningkatkan performa model, dilakukan *hyper-parameter tuning* atau mengubah parameter-parameter yang ada pada model BiGRU dan BiLSTM. Parameter yang biasanya diubah, adalah parameter unit setiap *layers*, *learning\_rate* dari optimizer RMSprop, dan *batch\_size*.

Selanjutnya, tantangan lain yang dihadapi adalah manajemen waktu. Kesibukan untuk harus tetap mengikuti beberapa mata perkuliahan yang dilaksanakan selama jam kerja menjadi suatu tantangan. Penulis mempelajari manajemen waktu untuk dapat mencapai target produktivitas untuk tetap menyelesaikan kedua kesibukan yang ada, yakni untuk tetap dapat mengikuti perkuliahan dengan baik dan menyelesaikan tugas magang yang diberikan.

## BAB V

### KESIMPULAN DAN SARAN

#### 5.1 Kesimpulan

Dari pengerjaan yang dilakukan mengenai *sentiment analysis* terhadap data ekspedisi menggunakan metode BiGRU dan BiLSTM, dapat diambil kesimpulan sebagai berikut:

- a. Performa dari penggunaan metode BiGRU dan BiLSTM dalam tahap evaluasi pada data ekspedisi ditunjukkan oleh *confusion matrix*, *classification report*, serta pengujian terhadap data yang baru. Pada metode BiGRU, diperoleh hasil akurasi sebesar 70.1%, yang mana dibuktikan juga dengan *confusion matrix* pada Tabel 3.26 serta hasil *classification report* yang ditunjukkan pada Gambar 3.36. Sedangkan, pada metode BiLSTM, diperoleh hasil akurasi sebesar 71.9%, yang mana dibuktikan juga dengan *confusion matrix* pada Tabel 3.28 dan hasil *classification report* yang ditunjukkan pada Gambar 3.37.
- b. Berdasarkan hasil yang didapatkan pada tahap evaluasi, dapat dilakukan analisis sentimen berdasarkan data *tweets* mengenai ekspedisi. Analisis sentimen berhasil dilakukan ke dalam tiga kelas yang ditentukan, yakni *positive*, *negative*, dan *neutral*.
- c. Pada proses *preprocessing*, tahap *stopwords removal* dapat dipertimbangkan untuk tidak digunakan berdasarkan kesulitan yang terjadi dalam *labeling*, karena hilangnya kata penghubung dan kebingungan dalam menentukan label kalimat.

#### 5.2 Saran

Adapun saran yang dapat diberikan oleh penulis terhadap peneliti Analisis Sentimen adalah sebagai berikut:

- a. Perlu adanya dokumentasi terhadap setiap pekerjaan riset yang dilakukan secara teratur dan rapi. Hal ini dimaksudkan untuk mempermudah apabila diperlukan riset ulang terhadap suatu topik.
- b. Baik kamus normalisasi maupun panduan *labeling* yang digunakan dapat berbeda-beda tergantung pada topik. Hal ini disebabkan karena setiap topik terkadang tidak memiliki keterkaitan dan parameter yang berbeda, seperti topik politik dan kepuasan pelanggan.
- c. Dibutuhkan adanya pembelajaran mengenai *clean code* atau hal-hal terkait dalam perbaikan kualitas perkodingan. Hal ini dilakukan karena beberapa *repository* yang



terdokumentasikan membingungkan pembaca, dengan pembelajaran tersebut pula dapat dimanfaatkan untuk mempermudah pengembang ataupun periset lain untuk mempelajari *repository* yang ada.



## DAFTAR PUSTAKA

- Ali, W., Yang, Y., Qiu, X., Ke, Y., & Wang, Y. (2021). Aspect-Level Sentiment Analysis Based on Bidirectional-GRU in SIoT. *IEEE Access*, 9, 69938–69950. <https://doi.org/10.1109/ACCESS.2021.3078114>
- Ay Karakuş, B., Talo, M., Hallaç, İ. R., & Aydin, G. (2018). Evaluating deep learning models for sentiment classification. *Concurrency and Computation: Practice and Experience*, 30(21). <https://doi.org/10.1002/cpe.4783>
- ByungSoo, K., Han-Gyu, K., Kyo-Joong, O., & Ho-Jin, C. (2017). *Controlled Dropout: a Different Approach to Using Dropout on Deep Neural Network*.
- Chandrasekar, P., & Qian, K. (2016). The Impact of Data Preprocessing on the Performance of a Naïve Bayes Classifier. *Proceedings - International Computer Software and Applications Conference*, 2, 618–619. <https://doi.org/10.1109/COMPSAC.2016.205>
- Corallo, L., Li, G., Reagan, K., Saxena, A., Varde, A. S., & Wilde, B. (2022). *A Framework for German-English Machine Translation with GRU RNN*. <http://ceur-ws.org>
- Dikiyanti, T. D., Rukmi, A. M., & Irawan, M. I. (2021). Sentiment analysis and topic modeling of BPJS Kesehatan based on twitter crawling data using Indonesian Sentiment Lexicon and Latent Dirichlet Allocation algorithm. *Journal of Physics: Conference Series*, 1821(1). <https://doi.org/10.1088/1742-6596/1821/1/012054>
- Hameed, Z., & Garcia-Zapirain, B. (2020). Sentiment Classification Using a Single-Layered BiLSTM Model. *IEEE Access*, 8, 73992–74001. <https://doi.org/10.1109/ACCESS.2020.2988550>
- Han, Y., Liu, M., & Jing, W. (2020). Aspect-Level Drug Reviews Sentiment Analysis Based on Double BiGRU and Knowledge Transfer. *IEEE Access*, 8, 21314–21325. <https://doi.org/10.1109/ACCESS.2020.2969473>
- Harywanto, G. N., Veron, J. S., & Suhartono, D. (2021). An annotated dataset for identifying behaviour change based on five doors theory under coral bleaching phenomenon on Twitter. *Data in Brief*, 39, 107617. <https://doi.org/10.1016/j.dib.2021.107617>
- Idris, Utami, E., & Hartanto, A. D. (2020). Klasifikasi Kepribadian dengan Metode DISC pada Twitter Menggunakan Algoritma Artificial Neural Network. *Tecnoscienza*, 5(1), 1–20.
- Kemp, S. (2022, February 15). *Digital 2022: Indonesia — DataReportal – Global Digital Insights*. <https://datareportal.com/reports/digital-2022-indonesia>

- Khader, M., Awajan, A., & Al-Naymat, G. (2019). The impact of natural language preprocessing on big data sentiment analysis. *International Arab Journal of Information Technology*, 16(3ASpecial Issue).
- Kowsher, M., Sobuj, M. S. I., Shahriar, M. F., Prottasha, N. J., Arefin, M. S., Dhar, P. K., & Koshiba, T. (2022). An Enhanced Neural Word Embedding Model for Transfer Learning. *Applied Sciences (Switzerland)*, 12(6). <https://doi.org/10.3390/app12062848>
- Krishna, R. (2020, July 27). *ML Classification-Why accuracy is not a best measure for assessing??* [https://medium.com/@KrishnaRaj\\_Parthasarathy/ml-classification-why-accuracy-is-not-a-best-measure-for-assessing-ceed964ae47c](https://medium.com/@KrishnaRaj_Parthasarathy/ml-classification-why-accuracy-is-not-a-best-measure-for-assessing-ceed964ae47c)
- Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (n.d.). *Neural Architectures for Named Entity Recognition*.
- Lee, E., Rustam, F., Ashraf, I., Washington, P. B., Narra, M., & Shafique, R. (2022). Inquest of Current Situation in Afghanistan Under Taliban Rule Using Sentiment Analysis and Volume Analysis. *IEEE Access*, 10, 10333–10348. <https://doi.org/10.1109/ACCESS.2022.3144659>
- Li, Y., Ren, X., Zhao, F., & Yang, S. (2021). A zeroth-order adaptive learning rate method to reduce cost of hyperparameter tuning for deep learning. *Applied Sciences (Switzerland)*, 11(21). <https://doi.org/10.3390/app112110184>
- Liesnawan, K. I. (2019). *UNIKOM\_KHAIRUL IMAM LIESNAWAN\_BAB 2*. Universitas Komputer Indonesia.
- Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1). <https://doi.org/10.2200/S00416ED1V01Y201204HLT016>
- MedAn. (2020). *Media Analytics User Guide. 1*, 7–8.
- Ndikumana, E., Minh, D. H. T., Baghdadi, N., Courault, D., & Hossard, L. (2018). Deep recurrent neural network for agricultural classification using multitemporal SAR Sentinel-1 for Camargue, France. *Remote Sensing*, 10(8). <https://doi.org/10.3390/rs10081217>
- NLTK. (2022). *NLTK Documentation*. <https://www.nltk.org/>
- Peng, D., Li, T., Wang, Y., & Philip Chen, C. L. (2018). Research on information collection method of shipping job hunting based on web crawler. *8th International Conference on Information Science and Technology, ICIST 2018*, 57–62. <https://doi.org/10.1109/ICIST.2018.8426183>
- Piyaphakdeesakun, C., Facundes, N., & Polvichai, J. (2019). Thai Comments Sentiment Analysis on Social Networks with Deep Learning Approach. *34th International Technical Conference*

- on *Circuits/Systems, Computers and Communications, ITC-CSCC 2019*.  
<https://doi.org/10.1109/ITC-CSCC.2019.8793324>
- Pradana, A. W., & Hayaty, M. (2019). The Effect of Stemming and Removal of Stopwords on the Accuracy of Sentiment Analysis on Indonesian-language Texts. *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control*, 4(3), 375–380. <https://doi.org/10.22219/kinetik.v4i4.912>
- Pristiyono, Ritonga, M., Ihsan, M. A. Al, Anjar, A., & Rambe, F. H. (2021). Sentiment analysis of COVID-19 vaccine in Indonesia using Naïve Bayes Algorithm. *IOP Conference Series: Materials Science and Engineering*, 1088(1). <https://doi.org/10.1088/1757-899x/1088/1/012045>
- Ramaprakoso. (2019). *Analisis Sentimen*. <https://github.com/ramaprakoso/analisis-sentimen>
- Raza, M. R., Hussain, W., & Merigo, J. M. (2021). Cloud Sentiment Accuracy Comparison using RNN, LSTM and GRU. *Proceedings - 2021 Innovations in Intelligent Systems and Applications Conference, ASYU 2021*. <https://doi.org/10.1109/ASYU52992.2021.9599044>
- Rianto, Mutiara, A. B., Wibowo, E. P., & Santosa, P. I. (2021). Improving the accuracy of text classification using stemming method, a case of non-formal Indonesian conversation. *Journal of Big Data*, 8(1). <https://doi.org/10.1186/s40537-021-00413-1>
- Samsir, S., Ambiyar, A., Verawardina, U., Edi, F., & Watrianthos, R. (2021). Analisis Sentimen Pembelajaran Daring Pada Twitter di Masa Pandemi COVID-19 Menggunakan Metode Naïve Bayes. *Stmik-Budidarma.Ac.Id*, 5(1), 157–163. <https://doi.org/10.30865/mib.v5i1.2604>
- Sastrawi. (2018, September 24). *har07/PySastrawi: Indonesian stemmer. Python port of PHP Sastrawi project*. <https://github.com/har07/PySastrawi>
- Shalini, K., Barathi Ganesh, H., Anand Kumar, M., & Soman, K. P. (2018). *Sentiment Analysis for Code-Mixed Indian SocialMedia Text With Distributed Representation*. 1126–1131.
- Singh, J., & Gupta, V. (2019). A novel unsupervised corpus-based stemming technique using lexicon and corpus statistics. *Knowledge-Based Systems*, 180(xxxx), 147–162. <https://doi.org/10.1016/j.knosys.2019.05.025>
- SOLUSI247. (2020). *Chanthel SOLUSI 247 - Be Smart Office*. <https://chanthel.solusi247.com/index.php/s/XPFBXrtqqgLxRSy#pdfviewer>
- Srivastava, N. (2013). *Improving Neural Networks with Dropout*.
- Twitter. (2022). *Twitter API Documentation*. <https://developer.twitter.com/en/docs/twitter-api>

- Uddin, A. H., Bapery, D., & Mohammad Arif, A. S. (2019, May 1). Depression Analysis of Bangla Social Media Data using Gated Recurrent Neural Network. *1st International Conference on Advances in Science, Engineering and Robotics Technology 2019, ICASERT 2019*. <https://doi.org/10.1109/ICASERT.2019.8934455>
- Uddin, M. J., Li, Y., Sattar, M. A., Nasrin, Z. M., & Lu, C. (2022). Effects of Learning Rates and Optimization Algorithms on Forecasting Accuracy of Hourly Typhoon Rainfall: Experiments With Convolutional Neural Network. *Earth and Space Science*, 9(3). <https://doi.org/10.1029/2021EA002168>
- Wolff, R. (2022). *Quick Introduction to Sentiment Analysis*. Medium. <https://towardsdatascience.com/quick-introduction-to-sentiment-analysis-74bd3dfb536c>
- Xu, G., Meng, Y., Qiu, X., Yu, Z., & Wu, X. (2019). Sentiment analysis of comment texts based on BiLSTM. *IEEE Access*, 7, 51522–51532. <https://doi.org/10.1109/ACCESS.2019.2909919>
- YAVA-247. (n.d.). *YAVA - BIG DATA SOLUTION WITHIN YOUR REACH*. <https://yava.labs247.id/>
- Yu, Q., Zhao, H., & Wang, Z. (2019). Attention-based bidirectional gated recurrent unit neural networks for sentiment analysis. *ACM International Conference Proceeding Series*, 391, 116–119. <https://doi.org/10.1145/3357254.3357262>
- Zulqarnain, M., Ghazali, R., Ghouse, M. G., & Mushtaq, M. F. (2019). Efficient processing of GRU based on word embedding for text classification. *International Journal on Informatics Visualization*, 3(4), 377–383. <https://doi.org/10.30630/JOIV.3.4.289>

LAMPIRAN

