



TUGAS AKHIR - SS234862

**REPRESENTASI SENTIMEN X (TWITTER) PILKADA
JAWA TIMUR DENGAN HASIL KPU MENGGUNAKAN
METODE TRANSFER LEARNING INDOBERTWEET**

ABDILLAH ILHAM

NRP 5003211069

Dosen Pembimbing

Adatul Mukarromah, S.Si., M.Si.

NIP 19800418 200312 2 001

Program Studi Sarjana Statistika

Departemen Statistika

Fakultas Sains dan Analitika Data

Institut Teknologi Sepuluh Nopember

Surabaya

2025



TUGAS AKHIR - SS234862

**REPRESENTASI SENTIMEN X (TWITTER) PILKADA
JAWA TIMUR DENGAN HASIL KPU MENGGUNAKAN
METODE *TRANSFER LEARNING* INDOBERTWEET**

ABDILLAH ILHAM

NRP 5003211069

Dosen Pembimbing

Adatul Mukarromah, S.Si., M.Si.

NIP 19800418 200312 2 001

Program Studi Sarjana Statistika

Departemen Statistika

Fakultas Sains dan Analitika Data

Institut Teknologi Sepuluh Nopember

Surabaya

2025



FINAL PROJECT - SS234862

REPRESENTATIONS OF X (TWITTER) SENTIMENT ON THE EAST JAVA REGIONAL HEAD ELECTION IN RELATION TO KPU RECAPITULATION USING INDOBERTWEET TRANSFER LEARNING METHOD

ABDILLAH ILHAM

NRP 5003211069

Advisor

Adatul Mukarromah, S.Si., M.Si.

NIP 19800418 200312 2 001

Bachelor Program of Statistics

Department of Statistics

Faculty of Science and Data Analytics

Institut Teknologi Sepuluh Nopember

Surabaya

2025

LEMBAR PENGESAHAN

REPRESENTASI SENTIMEN X (TWITTER) PILKADA JAWA TIMUR DENGAN HASIL REKAPITULASI KPU MENGGUNAKAN METODE TRANSFER LEARNING INDOBERTWEET

TUGAS AKHIR

Diajukan untuk memenuhi salah satu syarat
memperoleh gelar Sarjana Statistika pada
Program Studi Sarjana Statistika
Departemen Statistika
Fakultas Sains dan Analitika Data
Institut Teknologi Sepuluh Nopember

Oleh: **ABDILLAH ILHAM**

NRP 5003211069

Tanggal Ujian: 08 Juli 2025
Periode Wisuda: September 2025

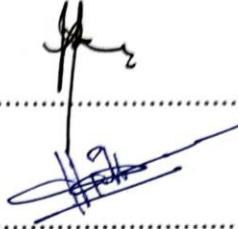
Disetujui oleh:
Pembimbing:

1. Adatul Mukarromah, S.Si., M.Si.
NIP 19800418 200312 2 001

.....


Pengaji:

2. Dr. Dra. Kartika Fithriasari, M.Si.
NIP 19691212 199303 2 002
3. Prof. Drs. Nur Iriawan, M.Ikomp., Ph.D.
NIP 19621015 198803 1 002

.....


Kepala Departemen Statistika

Fakultas Sains dan Analitika Data



Dr.rer.pol. Dedy Dwi Prastyo, S.Si., M.Si. 

NIP 19831204 200812 1 002

(Halaman ini sengaja dikosongkan)

APPROVAL SHEET

REPRESENTATIONS OF X (TWITTER) SENTIMENT ON THE EAST JAVA REGIONAL HEAD ELECTION IN RELATION TO KPU RECAPITULATION USING INDOBERTWEET TRANSFER LEARNING METHOD

FINAL PROJECT

Submitted to fulfill one of the requirements
For obtaining a degree in Bachelor of Statistics at
Bachelor Program of Statistics
Department of Statistics
Faculty of Science and Data Analytics
Institut Teknologi Sepuluh Nopember

By: **ABDILLAH ILHAM**

NRP 5003211069

Exam Date: 08 July 2025
Graduation Period: September 2025

Approved by:

Advisor:

1. Adatul Mukarromah, S.Si., M.Si.
NIP 19800418 200312 2 001


.....

Examiners:

2. Dr. Dra. Kartika Fithriasari, M.Si.
NIP 19691212 199303 2 002


.....

3. Prof. Drs. Nur Iriawan, M.Ikomp, Ph.D.
NIP 19621015 198803 1 002


.....

Head of Statistics Department

Faculty of Science and Data Analytics



Dr.rer.pol. Dedy Dwi Prastyo, S.Si., M.Si. 

NIP 19831204 200812 1 002

(Halaman ini sengaja dikosongkan)

PERNYATAAN ORIGINALITAS

Yang bertanda tangan di bawah ini:

Nama mahasiswa / NRP : Abdillah Ilham / 5003211069

Departemen : Statistika

Dosen Pembimbing / NIP : Adatul Mukarromah, S.Si., M.Si. / 19800418 200312 2 001

dengan ini menyatakan bahwa Tugas Akhir dengan judul **“Representasi Sentimen X (Twitter) Pilkada Jawa Timur Dengan Hasil Rekapitulasi KPU Menggunakan Metode Transfer Learning IndoBERTweet”** adalah hasil karya sendiri, bersifat orisinal, dan ditulis dengan mengikuti kaidah penulisan ilmiah.

Bilamana di kemudian hari ditemukan ketidaksesuaian dengan pernyataan ini, maka saya bersedia menerima sanksi sesuai dengan ketentuan yang berlaku di Institut Teknologi Sepuluh Nopember.

Surabaya, 28 Juli 2025

Mengetahui
Dosen Pembimbing


(Adatul Mukarromah, S.Si., M.Si.)
NIP 19800418 200312 2 001

Mahasiswa


(Abdillah Ilham)
NRP 5003211069

(Halaman ini sengaja dikosongkan)

STATEMENT OF ORIGINALITY

The undersigned below:

Name of student / NRP : Abdillah Ilham / 5003211069

Department : Statistics

Advisor / NIP : Adatul Mukarromah, S.Si., M.Si. / 19800418 200312 2 001

hereby declare that the Final Project with the title of "**Representations of X (Twitter) Sentiment on the East Java Regional Head Election in Relation to KPU Recapitulation Using IndoBERTweet Transfer Learning Method**" is the result of my own work, is original, and is written by following the rules of scientific writing.

If in the future there is a discrepancy with this statement, then I am willing to accept sanctions in accordance with the provisions that apply at Institut Teknologi Sepuluh Nopember.

Surabaya, 28 July 2025

Acknowledged
Advisor

(Adatul Mukarromah, S.Si., M.Si.)
NIP 19800418 200312 2 001

Student

(Abdillah Ilham)
NRP 5003211069

(Halaman ini sengaja dikosongkan)

ABSTRAK

REPRESENTASI SENTIMEN X (TWITTER) PILKADA JAWA TIMUR DENGAN HASIL REKAPITULASI KPU MENGGUNAKAN METODE *TRANSFER LEARNING* INDOBERTWEET

Nama Mahasiswa / NRP : Abdillah Ilham / 5003211069
Prodi, Departemen : Sarjana Statistika, Departemen Statistika FSAD ITS
Dosen Pembimbing : Adatul Mukarromah, S.Si., M.Si.

Abstrak

Pemilihan Kepala Daerah (Pilkada) merupakan elemen penting dalam demokrasi di Indonesia, termasuk di Jawa Timur. Selama masa kampanye, media sosial, khususnya X (dulu dikenal sebagai Twitter), menjadi *platform* utama untuk menyuarakan opini politik. Analisis sentimen terhadap cuitan pengguna X dapat memberikan wawasan yang mendalam mengenai sentimen publik terhadap pasangan calon. Dalam penelitian ini, data cuitan dari X dikumpulkan melalui mekanisme crawling selama masa kampanye, yaitu dari 25 September 2024 hingga 23 November 2024. Pendekatan berbasis *Transfer Learning* menggunakan IndoBERTweet dipilih karena kemampuannya dalam memahami konteks bahasa Indonesia terlebih karena model ini dilatih pada korpus bahasa X. Untuk meningkatkan performa klasifikasi sentimen, IndoBERTweet dikombinasikan dengan arsitektur CNN, LSTM, dan Hybrid CNN-LSTM dengan penambahan *class weight*, augmentasi data *back translation*, dan regularisasi L2. Pemodelan terbaik didapatkan pada model IndoBERTweet Bi-LSTM yang mendapatkan akurasi dan *weighted F1 Score* pada data *test* sebesar 83% dan 81% tanpa indikasi *overfitting* tinggi. Hasil analisis sentimen, kemudian dianalisis menggunakan korelasi rank Kendall's Tau dengan hasil rekapitulasi Komisi Pemilihan Umum (KPU). Hasilnya, persentase sentimen positif memiliki hubungan linear positif terhadap hasil rekapitulasi KPU. Hal ini berkebalikan dengan sentimen netral dan negatif yang memiliki hubungan linear negatif. Hasil sentimen juga dibandingkan dengan hasil survei dari waktu ke waktu dan ditemukan hubungan yang sedang–kuat, namun terkadang tidak konsisten dari waktu ke waktu dan untuk beberapa pasangan calon.

Kata kunci: *Analisis Sentimen, CNN, IndoBERTweet, Jatim, LSTM, Pilkada*

(Halaman ini sengaja dikosongkan)

ABSTRACT

REPRESENTATIONS OF X (TWITTER) SENTIMENT ON THE EAST JAVA REGIONAL HEAD ELECTION IN RELATION TO KPU RECAPITULATION USING INDOBERTTWEET TRANSFER LEARNING METHOD

Student Name / NRP	: Abdillah Ilham / 5003211069
Program, department	: Bachelor Program of Statictics, Department of Statistics F-Scientics ITS
Advisor	: Adatul Mukarromah, S.Si., M.Si.

Abstract

The Regional Head Election (Pilkada) is a crucial element of democracy in Indonesia, including in East Java. During the campaign period, social media, particularly X (formerly known as Twitter) emerged as a primary platform for expressing political opinions. Sentiment analysis of user tweets on X can offer deep insights into public sentiment toward the candidates. In this study, tweet data from X was collected through a crawling mechanism during the campaign period, from September 25, 2024, to November 23, 2024. A Transfer Learning approach using IndoBERTweet was selected due to its capability to understand the Indonesian language context, especially as the model was trained on X's language corpus. To enhance sentiment classification performance, IndoBERTweet was combined with CNN, LSTM, and Hybrid CNN-LSTM architectures, incorporating class weights, back-translation data augmentation, and L2 regularization. The best model, IndoBERTweet Bi-LSTM achieved a test accuracy and weighted F1 score of 83% and 81%, respectively. Subsequently, the sentiment analysis results were examined using Kendall's Tau rank correlation. The results indicate that the percentage of positive sentiment has a linear positive correlation with the KPU recapitulation results, while negative and neutral sentiments shows a linear negative relation. The sentiment result is also compared to survey result overtime and it is showed that it has a moderate to high positive relationship, although this relationship is not consistent overtime and for some candidate.

Keywords: *CNN, IndoBERTweet, East Java, LSTM, Pilkada, Sentiment Analysis*

(Halaman ini sengaja dikosongkan)

KATA PENGANTAR

Puji syukur Alhamdulillah penulis panjatkan ke hadirat Allah SWT karena atas rahmat, karunia, dan inayah-Nya, laporan Tugas Akhir yang berjudul “Representasi Sentimen X (Twitter) Pilkada Jawa Timur Dengan Hasil KPU Menggunakan Metode *Transfer Learning IndoBERTweet*” dapat terselesaikan dengan baik. Selawat serta salam semoga senantiasa tercurahkan kepada Nabi Muhammad SAW, suri teladan umat, beserta keluarga, sahabat, dan pengikutnya hingga akhir zaman. Laporan ini tidak akan terselesaikan tanpa bimbingan, bantuan, dan dukungan dari berbagai pihak. Oleh karena itu, penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada semua pihak yang telah memberikan dukungan dan kontribusinya dalam penyusunan tugas akhir ini.

1. Keluarga penulis, khususnya kedua orang tua penulis, serta kakak-kakak yang telah lebih dulu menyelesaikan pendidikan tinggi dan telah membimbing penulis dalam hal non-akademik.
2. Bapak Dr.rer.pol. Dedy Dwi Prastyo, S.Si., M.Si., selaku Kepala Departemen Statistika ITS, Ibu Dr. Wibawati, S.Si., M.Si., selaku Sekretaris Departemen Statistika ITS, dan Ibu Shofi Andari, S.Stat., M.Si., Ph.D., selaku Kepala Program Studi Sarjana Statistika ITS, serta jajaran manajemen Departemen Statistika ITS.
3. Ibu Adatul Mukarromah, S.Si., M.Si., selaku dosen pembimbing yang telah memberikan arahan dan motivasi selama pelaksanaan tugas akhir dari awal hingga akhir.
4. Ibu Dr. Dra. Kartika Fithriasari, M.Si., selaku dosen penguji I yang telah banyak memberikan saran yang membangun terkait kebermanfaatan dalam tugas akhir ini.
5. Bapak Prof. Drs. Nur Iriawan, M.Ikomp., Ph.D., selaku dosen penguji II yang telah banyak memberikan saran yang membangun terkait landasan teori dan penulisan akademik dalam tugas akhir ini.
6. Bapak Prof. Dr. Bambang Widjanarko Otok, S.Si., M.Si., selaku dosen wali penulis yang telah memberikan banyak wejangan dan semangat untuk terus mendorong penulis ke zona yang berkembang.
7. Seluruh dosen Departemen Statistika ITS yang selama ini telah membagikan ilmunya kepada penulis dan terus mendukung penulis, serta segenap karyawan/tenaga kependidikan Departemen Statistika FSAD ITS yang telah memberikan dukungan administrasi.
8. Seluruh kawan angkatan 2021 Bimasakti yang telah memberikan dukungan emosional dan material selama masa perkuliahan.
9. Semua pihak yang berkontribusi secara langsung atau tidak yang penulis tidak dapat sebutkan satu per satu.

Penulis menyadari masih terdapat kekurangan dalam Tugas Akhir ini dan terbuka terhadap saran yang membangun. Semoga laporan ini bermanfaat bagi yang membutuhkannya.

Surabaya, 7 Juni 2025

Penulis

(Halaman ini sengaja dikosongkan)

DAFTAR ISI

LEMBAR PENGESAHAN.....	i
APPROVAL SHEET	iii
PERNYATAAN ORIGINALITAS.....	v
STATEMENT OF ORIGINALITY	vii
ABSTRAK	ix
ABSTRACT	xi
KATA PENGANTAR.....	xiii
DAFTAR ISI.....	xv
DAFTAR GAMBAR.....	xix
DAFTAR TABEL.....	xxi
DAFTAR LAMPIRAN	xxiii
DAFTAR NOTASI.....	xxv
BAB I PENDAHULUAN	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah	3
1.3 Batasan Masalah.....	3
1.4 Tujuan.....	3
1.5 Manfaat.....	3
BAB II TINJAUAN PUSTAKA.....	5
2.1 Hasil Penelitian Terdahulu	5
2.2 Dasar Teori	6
2.2.1 Peran X dalam Pilkada.....	6
2.2.2 <i>Text Mining</i>	6
2.2.3 Transfer Learning.....	7
2.2.4 <i>Backpropagation</i>	11
2.2.5 <i>Bidirectional Encoder Representation from Transformer</i> (BERT)	13
2.2.6 IndoBERTweet	17
2.2.7 Penanganan Ketidakseimbangan Kelas dan <i>Overfitting</i>	17
2.2.8 Ukuran Kebaikan Klasifikasi	19
2.2.9 Kendall's Tau.....	21
BAB III METODOLOGI	23
3.1 Sumber Data	23
3.2 Variabel Penelitian	23

(Halaman ini sengaja dikosongkan)

3.3	Struktur Data	23
3.4	Tahapan Penelitian	25
3.5	Diagram Alir Penelitian.....	29
BAB IV	ANALISIS DAN PEMBAHASAN	31
4.1	Karakteristik Data Umum.....	31
4.2	Karakteristik Cuitan Umum	32
4.3	Karakteristik Cuitan Per Pasangan Calon.....	33
4.4	Pemrosesan Teks Untuk Pemodelan.....	34
4.5	Pemodelan	36
4.5.1	Skenario Pemodelan IndoBERTweet-CNN.....	36
4.5.2	Skenario Pemodelan IndoBERTweet Bi-LSTM.....	38
4.5.3	Skenario Pemodelan IndoBERTweet CNN Bi-LSTM	39
4.6	Mengatasi Ketidakseimbangan Data dan <i>Overfitting</i>	40
4.6.1	Teknik Augmentasi <i>Backtranslation</i>	40
4.6.2	Penggunaan <i>Class Weight</i> dan Regularisasi L2	44
4.7	Penetapan Model Terbaik	47
4.8	Karakteristik Sentimen Setiap Pasangan Calon.....	47
4.8.1	Karakteristik Sentimen Pasangan Calon Pertama.....	47
4.8.2	Karakteristik Sentimen Pasangan Calon Kedua	47
4.8.3	Karakteristik Sentimen Pasangan Calon Ketiga	48
4.9	Analisis Hubungan Sentimen Paslon dengan Hasil Pilkada.....	48
BAB V	KESIMPULAN DAN SARAN	53
5.1	Kesimpulan.....	53
5.2	Saran	53
DAFTAR PUSTAKA	55	
LAMPIRAN	59	

(Halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

Gambar 2.1 Grafik Fungsi Aktivasi <i>Sigmoid</i>	8
Gambar 2.2 Arsitektur LSTM.....	10
Gambar 2.3 Ilustrasi <i>Neural Network</i>	11
Gambar 2.4 Arsitektur <i>Encoder</i> dalam <i>Transformer</i>	13
Gambar 2.5 Ilustrasi Kedekatan Nilai <i>Cosine Similarity</i>	15
Gambar 2.6 Ilustrasi Proses <i>Back Translation</i>	18
Gambar 3.1 Arsitektur Utama Model	26
Gambar 3.2 Diagram Alir Penelitian	29
Gambar 4.1 a) <i>Word Cloud</i> Umum, b) Frekuensi Kata Terbanyak Umum	33
Gambar 4.2 a) <i>Word Cloud</i> Cuitan Paslon 1, b) Frekuensi Kata Terbanyak Cuitan Paslon 1	33
Gambar 4.3 a) <i>Word Cloud</i> Cuitan Paslon 2, b) Frekuensi Kata Terbanyak Cuitan Paslon 2	34
Gambar 4.4 a) <i>Word Cloud</i> Cuitan Paslon 3, b) Frekuensi Kata Terbanyak Cuitan Paslon 3	34
Gambar 4.5 Grafik Skenario CNN per <i>Epochs</i> : a) <i>Loss</i> , b) Akurasi	37
Gambar 4.6 Grafik Skenario Bi-LSTM Setiap <i>Epochs</i> : a) <i>Loss</i> , b) Akurasi	38
Gambar 4.7 Grafik Skenario CNN Bi-LSTM Setiap <i>Epochs</i> : a) <i>Loss</i> , b) Akurasi.....	39
Gambar 4.8 Grafik skenario BT Lima Bahasa CNN Bi-LSTM Setiap <i>Epoch</i> : a) <i>Loss</i> , b) Akurasi	43
Gambar 4.9 Grafik skenario BT Lima Bahasa CNN+ L2+Class Weight Setiap <i>Epoch</i> : a) <i>Loss</i> , b) Akurasi	46
Gambar 4.10 <i>Word cloud</i> Paslon 1: a) Sentimen Negatif, b) Sentimen Netral, c) Sentimen Positif	47
Gambar 4.11 <i>Word cloud</i> Paslon 2: a) Sentimen Negatif, b) Sentimen Netral, c) Sentimen Positif	48
Gambar 4.12 <i>Word cloud</i> Paslon 3: a) Sentimen Negatif, b) Sentimen Netral, c) Sentimen Positif	48
Gambar 4.13 Visualisasi Sentimen dan Rekapitulasi KPU	49
Gambar 4.14 Hubungan Setiap Sentimen dengan Rekapitulasi KPU	50
Gambar 4.15 Hubungan Sentimen Positif dengan Hasil Survei Waktu ke Waktu: a) Paslon 1, b) Paslon 2, c) Paslon 3	51

(Halaman ini sengaja dikosongkan)

DAFTAR TABEL

Tabel 2.1 Penelitian Terdahulu	5
Tabel 2.2 Perbandingan Model BERT untuk Tugas Analisis Sentimen.....	17
Tabel 2.3 <i>Confusion Matrix</i> Multi Kelas	19
Tabel 2.4 <i>Confusion Matrix</i> yang Difokuskan Untuk Kelas C2	19
Tabel 3.1 Variabel Penelitian Analisis Sentimen	23
Tabel 3.2 Variabel Penelitian Analisis Korelasi <i>Rank Kendall Tau</i>	23
Tabel 3.3 Struktur Data Analisis Sentimen	24
Tabel 3.4 Struktur Data <i>Input Embedding</i>	24
Tabel 3.5 Struktur Data Analisis Korelasi <i>Rank Kendall</i>	24
Tabel 3.6 Skenario Pemodelan	26
Tabel 4.1 Data Hasil <i>Crawling</i> per Paslon	31
Tabel 4.2 Hasil <i>Preprocessing</i> Cuitan Iklan dan Duplikat	31
Tabel 4.3 Jumlah Data Setelah <i>Preprocessing</i>	31
Tabel 4.4 Ilustrasi Tahapan Pemrosesan Teks Untuk <i>Word Cloud</i>	32
Tabel 4.5 Ilustrasi Tahapan Pemrosesan Teks Untuk Pemodelan	35
Tabel 4.6 Ilustrasi Tahapan Pemrosesan Teks Untuk Pelabelan Vader	35
Tabel 4.7 Ilustrasi Tahapan Tokenasi BERT	36
Tabel 4.8 Akurasi <i>Data Test</i> Skenario Pemodelan CNN.....	37
Tabel 4.9 Akurasi <i>Data Test</i> Skenario Pemodelan Bi-LSTM	38
Tabel 4.10 Akurasi <i>Data Test</i> Skenario Pemodelan CNN Bi-LSTM.....	40
Tabel 4.11 Ilustrasi Tahapan <i>Back Translate</i> Lima Bahasa	41
Tabel 4.12 Ilustrasi Tahapan <i>Back Translate</i> Empat Bahasa	41
Tabel 4.13 Distribusi Kelas <i>Data Train</i> Setelah Augmentasi	42
Tabel 4.14 Metrik Kebaikan Model Klasifikasi <i>Back Translation</i> Pada <i>Data Test</i>	42
Tabel 4.15 Metrik Setiap Kelas Pemodelan BT Lima Bahasa Bi-LSTM.....	43
Tabel 4.16 Kebaikan Model Dengan Penambahan Regularisasi L2 dan <i>Class Weight</i>	45
Tabel 4.17 Akurasi <i>Data Test</i> Skenario BT Lima Bahasa CNN Dengan Regularisasi L2 dan <i>Class Weight</i>	46
Tabel 4.18 Hasil Tabulasi Prediksi per Pasangan Calon dan Jenis Sentimen	49
Tabel 4.19 Hubungan Antara Sentimen dengan Rekapitulasi KPU	50
Tabel 4.20 Perbandingan Survei Selama Masa Kampanye dengan Sentimen Positif	51

(Halaman ini sengaja dikosongkan)

DAFTAR LAMPIRAN

Lampiran 1. Surat Pernyataan Data Sekunder	59
Lampiran 2. Ukuran Sampel Berdistribusi Multinomial	60
Lampiran 3. Kode <i>Crawling</i> Data X.....	61
Lampiran 4. Kode untuk Pembersihan Data untuk Pemodelan	62
Lampiran 5. Kode untuk Pembersihan Data untuk Statistika Deskriptif.....	64
Lampiran 6. Kode untuk Statistika Deskriptif (<i>Wordcloud</i>).....	66
Lampiran 7. Kode untuk <i>Back Translation</i> (Contoh dengan bahasa Jawa)	68
Lampiran 8. Kode untuk <i>Split</i> dan Tokenisasi BERT	69
Lampiran 9. Kode untuk Latih Skema CNN	70
Lampiran 10. Kode untuk Latih Skema Bi-LSTM	72
Lampiran 11. Kode untuk Latih Skema CNN Bi-LSTM	74
Lampiran 12. Kode Pemodelan CNN Dengan <i>Class Weight</i>	76
Lampiran 13. Kode Pemodelan CNN Dengan Regularisasi L2	78
Lampiran 14. Kode Pemodelan CNN Dengan Regularisasi L2 dan <i>Class Weight</i>	80
Lampiran 15. Kode Mengeluarkan Classification Report dan Grafik Akurasi dan <i>Loss</i> setiap <i>Epoch</i>	82

(Halaman ini sengaja dikosongkan)

DAFTAR NOTASI

z	: <i>Output</i> kombinasi linear
w	: Vektor pembobot fungsi aktivasi
x	: Vektor <i>input</i> fungsi aktivasi
b	: Bias
y_{ReLU}	: <i>Output</i> fungsi aktivasi ReLU
y_σ	: <i>Output</i> fungsi aktivasi <i>sigmoid</i>
X	: Matriks <i>input embedding</i>
$\mathbf{x}_{i:i+h-1}$: Vektor embedding untuk kata ke- i hingga $i + h - 1$
$c_{i,j}$: hasil konvolusi untuk baris ke- i dan kolom ke- j
$W_{c(j)}^T$: Kernel untuk konvolusi filter ke- j berukuran $d_{model} \times \text{window size}$
C	: Matriks hasil kovolusi berukuran $(n_{words} - h + 1) \times \text{filter size}$
$b_{c(j)}$: Bias konvolusi untuk filter ke- j
h	: Window size untuk konvolusi
$p_{i,j}$: Hasil pooling elemen baris ke- i dan kolom ke- j
$c_{[(i:k_{pool}+r),j]}$: Hasil konvolusi baris ke- j pada baris ke i hingga $k_{pool} + r$
k_{pool}	: Ukuran pooling
\mathbf{p}_j	: Vektor hasil pooling kolom ke- j
f_t	: Keluaran <i>forget gate</i> LSTM untuk sekuens ke- t
i_t	: Keluaran <i>input gate</i> LSTM untuk sekuens ke- t
\tilde{S}_t	: Kandidat <i>cell state</i> untuk sekuens ke- t
C_t	: <i>Cell state</i> baru untuk sekuens ke- t
O_t	: Keluaran <i>output gate</i> LSTM untuk sekuens ke- t
h_t	: <i>hidden state</i> baru untuk sekuens ke- t
W_f	: Matriks pembobot <i>forget gate</i> LSTM
W_i	: Matriks pembobot <i>input gate</i> LSTM
W_s	: Matriks pembobot kandidat <i>cell state</i>
W_o	: Matriks pembobot <i>ouput gate</i> LSTM
\mathbf{x}_t	: Vektor <i>embedding</i> pada sekuens ke- t
h_{t-1}	: <i>Hidden state</i> pada sekuens $t - 1$
b_f	: Bias <i>forget gate</i> LSTM
b_i	: Bias <i>input gate</i> LSTM
b_s	: Bias kandidat <i>cell state</i> LSTM

(Halaman ini sengaja dikosongkan)

\odot	: Produk Hadamard
S_{t-1}	: <i>Cell state</i> untuk sekuens $t - 1$
b_o	: Bias <i>output gate</i> LSTM
\widehat{y}_k	: Keluaran <i>softmax</i> neuron ke- k
y_k	: Nilai aktual kelas ke- k
L	: <i>Loss</i> untuk <i>backpropagation</i>
y_k	: y aktual
\widehat{y}_k	: Nilai keluaran fungsi <i>softmax</i> pada neuron ke- k
$\frac{\partial L}{\partial \widehat{w}_k^l}$: Vektor turunan <i>loss</i> terhadap pembobot pada <i>layer</i> ke- l dan neuron ke- k
$\frac{\partial L}{\partial z_k^l}$: Turunan <i>loss</i> terhadap kombinasi linear pada <i>layer</i> l dan neuron k
$\frac{\partial z_k^l}{\partial w_k^l}$: Vektor turunan kombinasi linear terhadap pembobot <i>layer</i> l dan neuron k
$\widehat{w}_k^{l(t)}$: Vektor pembobot (estimasi) pada <i>layer</i> l , neuron k , dan iterasi t
η	: <i>Learning rate</i>
$m_{w^l}^t$: Momen pertama untuk Adam pada t dan untuk w_l
$v_{w^l}^t$: Momen kedua untuk Adam pada t dan untuk w_l
β_1	: Parameter <i>exponential moving average</i> momen pertama
β_2	: Parameter <i>exponential moving average</i> momen kedua
ϵ	: Error sebagai <i>noise</i> dalam Adam
X_{token}	: Matriks <i>token embedding</i> BERT
e_i	: Vektor <i>embedding</i> pada kata ke- i
e_j	: Vektor <i>embedding</i> pada kata ke- j
pos	: Posisi kata dalam kalimat
n_{words}	: Jumlah kata dalam kalimat
d_{model}	: Dimensi model/ <i>embedding</i>
Q	: Matriks <i>query</i> berukuran $n_{words} \times d_{model}$
K	: Matriks <i>key</i> berukuran $n_{words} \times d_{model}$
V	: Matriks <i>value</i> berukuran $n_{words} \times d_{model}$
W^Q	: Matriks pembobot <i>query</i> berukuran $d_{model} \times d_{model}$
W^K	: Matriks pembobot <i>key</i> berukuran $d_{model} \times d_{model}$
W^V	: Matriks pembobot <i>value</i> berukuran $d_{model} \times d_{model}$
d_{head}	: Ukuran setiap <i>head</i> dalam <i>multihead attention</i> , $d_{head} = d_{model}/head$
Q_i	: Matriks <i>query</i> untuk <i>head</i> ke- i berukuran $n_{words} \times d_{head}$
K_i	: Matriks <i>key</i> untuk <i>head</i> ke- i berukuran $n_{words} \times d_{head}$

(Halaman ini sengaja dikosongkan)

V_i	: Matriks <i>value</i> untuk <i>head</i> ke- i berukuran $n_{words} \times d_{head}$
$head_i$: Matriks hasil operasi <i>attention</i>
k	: Indeks kelas
TP_k	: <i>True positive rate</i> dengan acuan kelas k
TN_k	: <i>True negative rate</i> dengan acuan kelas k
FP_k	: <i>False positive rate</i> dengan acuan kelas k
FN_k	: <i>False negative rate</i> dengan acuan kelas k
$\hat{\tau}$: Estimasi korelasi <i>rank</i> Kendall
n_c	: Jumlah pasangan data berurutan wajar
n_d	: Jumlah pasangan data berurutan tidak wajar
K_τ	: Jumlah pasangan data <i>concordant</i> dikurangi <i>discordant</i>
n	: Jumlah data korelasi <i>rank</i> Kendall
W^o	: Matriks pembobot integrasi <i>Multi Head Attention</i>
$\xi(x_i, x_j, y_i, y_j)$: Fungsi untuk menghitung jumlah pasangan data <i>discordant</i> atau <i>concordant</i>
x_i	: Data x baris ke- i
x_j	: Data x baris ke- j
y_i	: Data y baris ke- i
y_j	: Data y baris ke- j
$p_{i,hasil}$: Persentase hasil tabulasi pasangan calon ke- i terhadap hasil sentimen atau KPU
$n_{max\ length}$: Jumlah maksimum kata setelah dilakukan <i>padding</i>
n_{cuitan}	: Jumlah cuitan
n_1	: Jumlah cuitan pasangan calon pertama
n_2	: Jumlah cuitan pasangan calon kedua
n_3	: Jumlah cuitan pasangan calon ketiga
L_ω	: <i>Loss</i> setelah diterapkan <i>class weight</i>
ω_k	: Bobot kelas ke- k
λ	: Parameter regularisasi
L_λ	: <i>Loss</i> setelah dilakukan regularisasi
K	: Jumlah kelas
$C_{i,j}$: Tabulasi data dari kelas i diklasifikasikan kelas j

(Halaman ini sengaja dikosongkan)

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pemilihan Kepala Daerah (Pilkada) merupakan salah satu elemen penting dalam demokrasi di Indonesia, termasuk di provinsi Jawa Timur yang merupakan salah satu provinsi terbesar dan paling berpengaruh. Pemilihan umum adalah mekanisme demokrasi di mana warga negara memiliki hak untuk memilih wakil-wakil mereka dalam pemerintahan. Pemilu merupakan instrumen utama yang memungkinkan partisipasi politik serta pengambilan keputusan oleh rakyat (Reginantis, Priyambodo, & Jamal, 2024). Setiap warga negara yang sesuai dengan syarat, berhak mengikuti pemilu seperti pemilihan gubernur dan wakil gubernur, bupati dan wakil bupati, serta walikota dan wakil walikota dan dijamin oleh undang-undang (KPU RI, 2022). Jawa Timur akan menjadi salah satu provinsi yang melakukan pemilihan umum kepala daerah (gubernur dan wakil gubernur) pada tahun 2024.

Pada pemilihan Gubernur dan Wakil Gubernur Provinsi Jawa Timur tahun 2024, terdapat 3 pasangan calon Gubernur, yakni Khofifah Indar Parawansa dan Emil Dardak yang juga sebagai petahana jabatan Gubernur dan Wakil Gubernur Jatim, Tri Rismaharini dan Zarul Azhar dari partai pengusung PDI-P, dan Luluk Nur Hamidah dan Lukmanul Khakim dari partai pengusung PKB (Kominfo Jatim, 2024). Prediksi pengamat politik, ketiga pasangan calon ini akan bertarung secara sengit, dilihat dari kelebihan dan kekurangan masing-masing pasangan calon (Rinanda, 2024). Oleh karena itu, strategi diperlukan dalam menggaet elektabilitas pasangan calon, terutama pada media sosial sebagai salah satu perantara penting dalam kampanye politik pada era modern (Conway, Kenski, & Wang, 2014).

X adalah salah satu media sosial merupakan salah satu *platform* dimana setiap orang di dunia memiliki akses untuk mengeluarkan pendapatnya melalui *tweet* atau cuitan dan dapat dilihat oleh setiap pengguna lainnya (Fitriyyah, Safriadi, & Pratama, 2019). Melalui fitur tagar, pengguna X dapat melihat topik yang dibahas secara *realtime*. Bahkan, beberapa berita saat ini memanfaatkan twitter sebagai sumber berita mereka, yang mana pembahasan tersebut telah tersedia di X (Arsi & Waluyo, 2021). Dengan adanya pemilihan umum, X digunakan oleh masyarakat atau institusi politik sebagai tempat untuk menyuarakan pendapat politiknya, terutama pada masa kampanye (Zuhdi, Utami, & Raharjo, 2019). Sehingga, media sosial (terutama X) memainkan peranan penting sebagai media kampanye pemilihan umum (Lestari, Perdana, & Fauzi, 2017). Diyakini pula bahwa dengan menginterpretasikan data X, akan menjadikannya sumber data yang berpotensi dan efisien dalam menggambarkan kondisi pemilihan umum (Zuhdi, Utami, & Raharjo, 2019). Sehingga, dibutuhkan cara untuk mengekstrak informasi mengenai cuitan-cuitan yang berkaitan dengan pemilihan umum.

Analisis sentimen adalah proses yang bertujuan untuk mengekstrak, mengolah, dan memahami teks tidak terstruktur guna memperoleh informasi terkait sentimen yang terkandung dalam sebuah pendapat atau opini (Brahimi, Touahria, & Tari, 2021). Proses ini digunakan untuk mengevaluasi opini dan kecenderungan dari sebuah pernyataan, apakah bersifat positif atau negatif dimana penerapannya termasuk pada berbagai bidang, termasuk ekonomi, politik, sosial, dan hukum (Rozi, Pramono, & Dahlan, 2012). Dalam konteks pemilihan umum, hal ini dapat diterapkan sebagai salah satu cara dalam memahami perilaku media sosial terhadap

kondisi pemilihan umum, terutama pada masing-masing pasangan calon. Analisis sentimen tentu merupakan salah satu cara dalam memahami pendapat dalam cuitan X, namun analisis sentimen memiliki beberapa permasalahan. Jumlah cuitan yang sangat banyak memerlukan waktu yang cukup lama untuk menyelesaikan proses klasifikasi. Oleh karena itu, diperlukan sebuah metode yang efektif agar klasifikasi data X dapat dilakukan dengan lebih cepat dan hasilnya bisa dikelompokkan ke dalam kategori yang relevan. Analisis sentimen merupakan cabang penelitian lanjutan yang mencakup beberapa bidang seperti *Natural Language Processing* (NLP), *Data Mining*, dan *Machine Learning*, yang berfokus pada proses ekstraksi sentimen dari sebuah kalimat berdasarkan konten yang terkandung di dalamnya (Al-Ayyoub, Khamaiseh, Jararweh, & Al-Kabi, 2019). Beberapa model yang dapat digunakan untuk melakukan pemrosesan bahasa tersebut seperti *Random Forest*, *Support Vector Machine* (SVM), dan *Deep Learning*. *Deep Learning* merupakan salah satu teknik dalam *machine learning* yang juga dapat digunakan untuk melakukan analisis sentimen. Meskipun metode ini memerlukan waktu lebih lama dalam proses pelatihan, keunggulannya adalah kemampuannya untuk memahami konteks kata secara lebih mendalam melalui *transfer learning*. Salah satu model pembelajaran yang populer dalam kalangan penelitian analisis sentimen di X adalah penggunaan model *pretrained* BERT (*Bidirectional Encoder Representation from Transformer*). Terdapat berbagai macam model BERT, namun model yang dilatih secara spesifik pada korpus bahasa X atau Twitter pada jumlah yang besar dan topik yang bermacam-macam adalah IndoBERTweet (Koto, Lau, & Baldwin, 2021).

Pada penelitian-penelitian sebelumnya, Setiawan dkk (2023) melakukan analisis sentimen terhadap pengguna Twitter (sekarang dikenal sebagai X) terkait ulasan aplikasi TikTok dengan menggunakan metode *fine tuning* IndoBERTweet dan *Long Short-Term Memory* (LSTM). Hasil penelitian ini menunjukkan bahwa LSTM menghasilkan skor F1 sebesar 78%, sedangkan *fine tuning* IndoBERTweet menghasilkan skor yang lebih tinggi, yaitu sebesar 80% (Setiawan, Lhaksamana, & Bunyamin, 2023). Selain itu, penelitian lain oleh Maulana dan Lhaksamana (2023) menganalisis sentimen pengguna X terhadap tragedi Kanjuruhan, di mana model *fine tuning* IndoBERTweet memberikan akurasi sebesar 88%, jauh lebih tinggi dibandingkan Naïve Bayes yang hanya menghasilkan akurasi 62% (Maulana & Lhaksamana, 2023). Penelitian serupa mengenai model BERT yang dilakukan oleh Anggraeni et al (2024) dalam konteks klasifikasi cuitan X mengenai penyakit dengue dengan model *transfer learning* IndoBERT CNN-LSTM mampu mencapai skor F1 terbaik sebesar 90%, mengungguli skenario pemodelan CNN atau LSTM saja (Anggraeni, et al., 2024). Penelitian lain menggunakan IndoBERT untuk melakukan klasifikasi dengan AUC data *testing* sebesar 93,72% dan dilanjutkan dengan analisis diagram kendali *p* seiring berjalannya waktu (Nurfaizah & Ahsan, 2024).

Dalam penelitian ini, akan digunakan data cuitan dari pengguna X mengenai topik Pemilihan Gubernur Jawa Timur 2024. Data tersebut akan diproses menggunakan metode *transfer learning* dengan memanfaatkan model *pretrained* IndoBERTweet untuk melakukan analisis sentimen. Tujuan dari penelitian ini adalah untuk memahami kecenderungan sentimen warganet terkait Pemilihan Gubernur Jawa Timur 2024 dan menyimpulkan hubungan yang dimilikinya dengan hasil rekapitulasi KPU. Selain itu, penelitian ini juga bertujuan untuk mengukur kinerja model IndoBERTweet dalam konteks analisis sentimen terkait Pemilihan Gubernur dan Wakil Gubernur Jawa Timur. Dengan menggunakan pendekatan ini, diharapkan dapat memberikan hasil yang akurat dan relevan dalam memahami sentimen publik terhadap proses politik di Jawa

Timur dan hubungannya terhadap hasil pemilihan umum, serta mengevaluasi keunggulan model IndoBERTweet dibandingkan metode lainnya dalam tugas serupa.

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, diidentifikasi beberapa permasalahan sebagai berikut.

1. Bagaimana kecenderungan sentimen warganet X mengenai Pilkada Gubernur dan Wakil Gubernur Provinsi Jawa Timur?
2. Bagaimana performa dan tingkat kebaikan model *transfer learning* IndoBERTweet dalam melakukan analisis sentimen pada data cuitan X?
3. Bagaimana menganalisis hubungan antara sentimen X dengan hasil rekapitulasi KPU?

1.3 Batasan Masalah

Berdasarkan rumusan masalah, batasan masalah pada penelitian ini adalah sebagai berikut.

1. Data yang digunakan bersumber dari data cuitan warganet X pada masa kampanye Pilkada Gubernur dan Wakil Gubernur Provinsi Jawa Timur dengan topik *pilgub jatim*.
2. Model *pretrained* yang digunakan adalah IndoBERTweet *base uncased* yang tersedia pada *platform* HuggingFace.
3. Penelitian ini terbatas pada kasus Pemilihan Gubernur Jawa Timur 2024.

1.4 Tujuan

Tujuan dari penelitian ini adalah sebagai berikut.

1. Mengetahui kecenderungan sentimen warganet X mengenai Pilkada Gubernur dan Wakil Gubernur Provinsi Jawa Timur.
2. Mengetahui performa dan tingkat kebaikan model *transfer learning* IndoBERTweet dalam melakukan analisis sentimen pada data cuitan X.
3. Mengetahui hubungan antara hasil analisis sentimen X dengan hasil rekapitulasi KPU.

1.5 Manfaat

Adapun manfaat dari penelitian ini adalah sebagai berikut.

1. Hasil penelitian ini dapat dimanfaatkan oleh tim kampanye pasangan calon sebagai salah satu cara untuk menetukan elektabilitas pasangan calon Gubernur dan Wakil Gubernur Provinsi Jawa Timur.
2. Hasil penelitian ini juga dapat digunakan oleh pembaca untuk mengetahui performa dan kebaikan model dari *transfer learning* IndoBERTweet dalam melaksanakan analisis sentimen pada data cuitan X.

(Halaman ini sengaja dikosongkan)

BAB II

TINJAUAN PUSTAKA

2.1 Hasil Penelitian Terdahulu

Berbagai penelitian mengenai analisis sentimen baik di pilkada maupun kejadian publik lainnya dengan metode serupa atau metode lain telah banyak dilakukan. Berikut merupakan tabulasi mengenai penelitian-penelitian tersebut beserta temuannya.

Tabel 2.1 Penelitian Terdahulu

Referensi	Studi Kasus	Temuan
(Lestari, Perdana, & Fauzi, 2017)	Analisis Sentimen Opini Publik pada Pilkada DKI Jakarta 2017	Data yang digunakan berasal dari X dan dilakukan pembobotan non-teksual menggunakan pembobotan emoji dan Naïve Bayes untuk model <i>classifier</i> , mendapatkan akurasi tertinggi 74,81%.
(Fitriyyah, Safriadi, & Pratama, 2019)	Analisis Sentimen Calon Presiden Indonesia 2019	Data yang digunakan berasal dari X dan dilakukan analisis sentimen menggunakan Naïve Bayes untuk <i>classifier</i> didapatkan akurasi untuk kedua kelas berturut-turut sebagai berikut 77,7% dan 88%.
(Mozafari, Farahbakhsh, & Crespi, 2019)	Deteksi <i>Hate Speech</i> menggunakan BERT	Data yang digunakan adalah data UGC dan dilakukan prediksi dengan <i>BERT_{BASE}</i> , didapatkan metrik terbaik skor F1 92% menggunakan kombinasi BERT+CNN.
(Fithriasari, Jannah, & Reyhana, 2020)	Analisis Sentimen Twitter (X) Performa Pemerintah	Data yang digunakan berasal dari X dan dilakukan analisis sentimen menggunakan metode BNN dan CNN. Didapatkan model terbaik CNN dengan AUC 81%.
(Pratama & Romadhony, 2020)	Identifikasi Komentar Toksik dengan BERT	Data yang digunakan adalah data <i>User Generated Content</i> (UGC) dilakukan klasifikasi dengan <i>fine tuning BERT_{BASE}</i> , didapatkan metrik terbaik skor F1 66,22% tanpa perlakuan lemmatisasi dan penghapusan <i>stop word</i> .
(Anggraeni, et al., 2024)	Klasifikasi Cuitan Dengue Indonesia menggunakan <i>Hybrid-CNN-LSTM</i>	Data berasal dari X dan dilakukan analisis sentimen tanpa penghapusan <i>stopwords</i> dan lemmatisasi didapatkan metrik terbaik skor F1 90% menggunakan kombinasi <i>Hybrid-CNN-LSTM</i> pada kasus data <i>imbalanced</i> .
(Nurfaizah & Ahsan, 2024)	Klasifikasi Sentimen menggunakan BERT dilanjutkan dengan monitoring diagram kendali <i>p</i>	Data berasal dari kolom komentar Google Maps dilakukan pemrosesan dengan tambahan normalisasi kata. Didapatkan hasil klasifikasi dengan AUC data <i>train</i> sebesar 99,95% dan data <i>test</i> 93,72%.

Tidak banyak penelitian mengenai hubungan langsung antara hasil analisis sentimen X dengan hasil pemilu atau hasil survei, mayoritas penelitian hanya membahas mengenai analisis deskriptif pengaruh/hubungan X terhadap politik. Adapun ditemukan sebuah penelitian terdahulu oleh (Alvi, Ali, Ahmed, & Khan, 2023) yang mengungkapkan bahwa tidak serta merta hasil analisis sentimen mengungkapkan tingkat elektabilitas dalam hasil pemilu. Adapun hasil penelitian tersebut mengungkapkan bahwa hasil klasifikasi dapat dimaksimalkan melalui eksplorasi berbagai aspek dari teks seperti sarkasme, subjektivitas, dan emosi untuk mendapatkan analisis dengan akurasi tinggi.

2.2 Dasar Teori

2.2.1 Peran X dalam Pilkada

Sejak lama media sosial menjadi alat penting dalam partisipasi politik modern, dimana media sosial memberikan ruang bagi publik untuk berinteraksi, berdiskusi, dan berpartisipasi dalam proses politik secara terbuka. X (dulu bernama Twitter) menjadi salah satu tempat untuk berpartisipasi politik. Kelebihan X sebagai *platform* media sosial adalah setiap penggunanya memiliki kesempatan yang sangat luas untuk menyuarakan pendapat mereka dengan cara menulis *tweet* atau cuitan yang dapat diakses dan dibaca oleh pengguna lainnya tanpa batasan geografis. Hal ini mengubah dinamika komunikasi global, di mana pendapat seseorang, terlepas dari asal atau status sosialnya, dapat langsung menjangkau jutaan pengguna di seluruh dunia. Hal tersebut memberikan akses yang hampir merata bagi setiap orang untuk mengekspresikan pandangan mereka dan turut berkontribusi dalam berbagai diskusi sosial dan politik secara bebas (Fitriyyah, Safriadi, & Pratama, 2019).

Salah satu fitur unggulan di platform X adalah penggunaan tagar, atau lebih dikenal dengan istilah *hashtag*. Melalui fitur ini, pengguna dapat dengan mudah menemukan dan mengikuti topik-topik yang sedang menjadi tren atau dibahas secara realtime oleh masyarakat luas. *Hashtag* berfungsi sebagai alat agregasi yang memudahkan diskusi terbuka, memungkinkan pengguna X dari berbagai belahan dunia untuk saling berinteraksi dalam satu ruang diskusi virtual yang sama. Beberapa media berita telah mulai memanfaatkan X sebagai sumber informasi, mengingat cepatnya arus informasi yang dibahas oleh para pengguna melalui cuitan mereka (Arsi & Waluyo, 2021). Hal ini menunjukkan bahwa data yang ada pada X dapat digunakan sebagai salah satu indikator pengambilan keputusan yang baik.

Dalam tahun politik, X semakin menonjol sebagai alat komunikasi politik. Institusi politik, partai, maupun kandidat pemilihan menggunakan X sebagai salah satu strategi utama dalam kampanye mereka. Masyarakat juga menggunakan platform ini untuk menyampaikan opini politik, baik dalam bentuk dukungan, kritik, maupun diskusi mengenai kandidat atau kebijakan yang sedang dibicarakan. Dengan memanfaatkan data mengenai cuitan tersebut, seseorang dapat mengambil kesimpulan dengan melakukan *text mining* untuk mengetahui kecenderungan opini publik, misalkan dalam mengetahui elektabilitas pasangan calon.

2.2.2 Text Mining

Text mining adalah proses penerapan *data mining* dalam bentuk data teks. Sumber data biasanya berasal dari suatu dokumen. Tujuannya adalah untuk menemukan istilah-istilah yang relevan yang dapat mewakili isi dokumen, sehingga memungkinkan analisis keterkaitan antar dokumen (Ernawati & Wati, 2018). *Text mining* sering digunakan untuk menyelesaikan masalah seperti klasifikasi, pengelompokan, ekstraksi informasi, dan pencarian informasi.

Proses *text mining* melibatkan pencarian informasi dari kata-kata yang mewakili konten dokumen dengan terlebih dahulu melakukan kategorisasi teks, ekstraksi teks, dan ekstraksi kata, sehingga pola-pola menarik dapat ditemukan (Feldman & Sanger, 2007).

Klasifikasi merupakan cara pengelompokan benda berdasarkan ciri-ciri yang dimiliki oleh objek klasifikasi. Dalam prosesnya, klasifikasi dapat dilakukan dengan berbagai cara baik secara manual ataupun dengan bantuan teknologi. Klasifikasi yang dilakukan secara manual yakni klasifikasi yang dilakukan oleh manusia tanpa adanya bantuan dari algoritma cerdas komputer. Sedangkan klasifikasi yang dilakukan dengan bantuan teknologi, memiliki beberapa algoritma, diantaranya *Naive Bayes*, *Support Vector Machine*, *Decision Tree*, *Fuzzy*, Jaringan Saraf Tiruan (JST), dan lain sebagainya (Wibawa, Purnama, Akbar, & Dwiyanto, 2018).

2.2.3 Transfer Learning

Transfer learning merupakan metode yang memanfaatkan kesamaan data, distribusi, model, tugas, dsb untuk diterapkan ke dalam sebuah data, distribusi, tugas, dsb pada domain baru melalui mekanisme *fine tuning* dengan memanfaatkan ilmu yang telah dipelajari pada domain yang bersesuaian (Liu, Shi, Ji, & Jia, 2019). *Transfer learning*, khususnya dalam *Natural Language Processing* (NLP), banyak digunakan karena beberapa metode *machine learning* ataupun *deep learning* seperti *Convolution Neural Network* (CNN), *Long Short-Term Memory* (LSTM), dsb memiliki kelemahan dalam menangkap generalisasi pada tugas analisis sentimen (Mozafari, Farahbakhsh, & Crespi, 2019). Penggunaan *transfer learning* pada NLP menghasilkan skor F1 rata-rata yang paling baik dibandingkan menggunakan metode *machine learning* konvensional (Prottasha, et al., 2022). Beberapa cara untuk melakukan *transfer learning*, khususnya dalam NLP, adalah dengan menambahkan layer sebelum, sesudah, atau diantara model BERT dan disesuaikan dengan jumlah kelas yang ada. *Layer* tambahan tersebut dapat berupa kombinasi antara *dense*, *convolution*, *long short-term memory*, dsb atau bahkan dipasangkan dengan *classifier* seperti *Support Vector Machine* (SVM), *Random Forest* (RF), dsb (Wadud, Shin, Mridha, & Nur, 2022).

2.2.3.1 Dense Layer

Dense layer merupakan *layer* atau lapisan pemrosesan di dalam *neural network* yang sering digunakan dalam model *deep neural network*. *Dense layer*, selayaknya lapisan pemrosesan lainnya pada *neural network* akan terhubung dengan lapisan pemrosesan lainnya (Josephine, Nirmala., & Alluri, 2021). Adapun operasi matematis dari *dense layer* adalah sebagai berikut.

$$\text{output} = \text{activation}(\text{dot}(\text{input}, \text{kernel}) + \text{bias}) \quad (2.1)$$

Pada Persamaan (2.1), *output* dapat digunakan kembali sebagai *input* dari *layer dense* atau *layer* pemrosesan lain yang selanjutnya. Adapun aktivasi merupakan fungsi aktivasi yang terdiri dari beberapa macam, beberapa yang sering digunakan adalah linear, ReLU, *sigmoid*, dan fungsi aktivasi lainnya.

2.2.3.2 Fungsi Aktivasi

Fungsi aktivasi merupakan suatu proses mengubah suatu *input* menjadi *output* menggunakan fungsi pada setiap neuron pada suatu layer *neural network*. Terdapat beberapa macam fungsi aktivasi yang disesuaikan dengan tujuannya masing-masing. Berikut adalah penjelasan dari beberapa fungsi aktivasi yang sering digunakan.

A. Fungsi Aktivasi Linear

Fungsi aktivasi linear menghasilkan kombinasi linear antara *input* dan parameter. Fungsi aktivasi ini adalah jenis fungsi aktivasi yang paling sederhana dan menjadi dasar bagi pengembangan fungsi aktivasi lainnya. Fungsi ini mengubah input berupa vektor \mathbf{x} menjadi output tunggal melalui hubungan linear yang direpresentasikan oleh sebuah persamaan.

$$z = \mathbf{w} \cdot \mathbf{x} + b \quad (2.2)$$

Pada Persamaan (2.2), \mathbf{x} merupakan *input vector*, \mathbf{w} adalah vektor parameter yang akan dilakukan optimasi, b merupakan bias, dan z adalah hasil kombinasi linear.

B. Fungsi Aktivasi ReLU

Fungsi aktivasi *Rectified Linear Unit* merupakan pengembangan dari fungsi aktivasi linear, dimana ReLU memiliki *range* $[0; \infty)$. Sehingga ReLU didefinisikan dengan persamaan berikut.

$$y_{ReLU} = \text{ReLU}(z) = \max(z, 0) \quad (2.3)$$

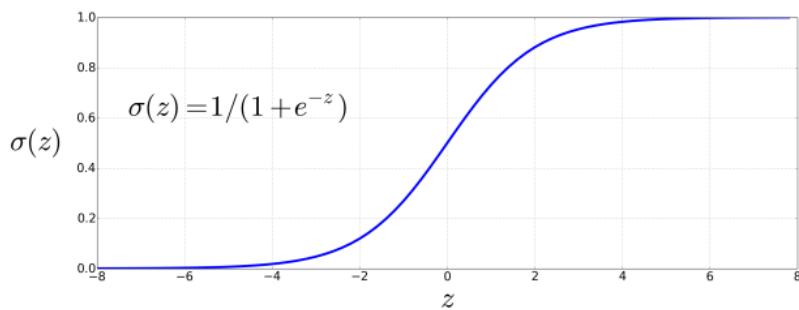
Cara intuitif untuk mendefinisikan Persamaan (2.3), ketika z bernilai negatif, maka y_{ReLU} bernilai nol, namun apabila $z \geq 0$, maka nilai y_{ReLU} adalah z .

C. Fungsi Aktivasi Sigmoid

Fungsi aktivasi *sigmoid* biasanya digunakan sebagai *layer* terakhir dalam tugas klasifikasi dua kelas. Fungsi ini akan menghasilkan *output* probabilitas dari dua kelas.

$$y_\sigma = \sigma(z) = \frac{1}{1 + e^{-z}} \quad (2.4)$$

Menggunakan properti dari bilangan euler, Persamaan (2.4) akan menghasilkan nilai probabilitas dengan range $(0; 1)$.



Gambar 2.1 Grafik Fungsi Aktivasi *Sigmoid*
(Sumber: Jurafsky & Martin, 2024)

Penggunaan bilangan euler akan membuat *outlier* terhimpit pada nilai 0 ataupun 1 seperti pada Gambar 2.1. Sifat lain dari fungsi *sigmoid* adalah *differentiable* yang memungkinkannya dapat digunakan untuk mengkategorikan dua macam kelas.

D. Fungsi Aktivasi *Softmax*

Fungsi *softmax* memiliki tugas yang sama seperti fungsi *sigmoid*, yakni menghasilkan output probabilitas kelas dan biasa digunakan pada *layer* terakhir. Namun, alih-alih mengategorikan terhadap dua kelas, *softmax* mengategorikan berdasarkan beberapa kelas. Membuatnya cocok untuk tugas klasifikasi multi kelas.

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{n_K} e^{z_j}}, \text{ untuk } 1 \leq i \leq K \quad (2.5)$$

Pada Persamaan (2.5), fungsi *softmax* akan mengambil input z_i yang merupakan elemen ke- i dari setiap neuron berjumlah n_K kelas yang perhitungannya diulang k kali. Sehingga, *output* dari fungsi aktivasi ini adalah sebuah matriks probabilitas \mathbf{z} yang memiliki ukuran $K \times 1$ dengan K adalah jumlah kelas.

2.2.3.3 Convolution Layer

Convolution atau konvolusi merupakan jenis *layer* pada *neural network* yang biasa digunakan dalam pemrosesan citra komputer. Kendati demikian, konvolusi juga dapat digunakan dalam dataset atau tugas lainnya termasuk pemrosesan bahasa alami. Dalam NLP, konvolusi akan bekerja sangat bagus dalam memprediksi kata pada jangka panjang, namun tidak mengakomodasi fitur dominan (Anggraeni, et al., 2024). Pada pemrosesan bahasa alami, konvolusi bekerja dengan *input* vektor kata dalam bentuk matriks \mathbf{x} dengan cara melakukan penggabungan dari vektor *token embedding* setiap kata.

$$\mathbf{X} = \mathbf{x}_1 \oplus \mathbf{x}_2 \oplus \dots \oplus \mathbf{x}_n \quad (2.6)$$

Pada Persamaan (2.6), \mathbf{X} adalah matriks *embedding* berukuran dimensi *embedding* dikalikan jumlah kata ($d_{\text{embedding}} \times n_{\text{words}}$). Sedangkan, untuk setiap elemen $\mathbf{x}_i \in R^{d_{\text{embedding}}}$.

Setelah didapatkan matriks *embedding*, operasi konvolusi dijalankan dengan menggunakan window sebesar h untuk setiap elemen dalam matriks *embedding* tersebut.

$$c_{i,j} = f(\mathbf{W}_{c(j)}^T \cdot \mathbf{x}_{i:i+h-1} + b_{c(j)}) \quad (2.7)$$

Persamaan (2.7) akan menghasilkan *output* konvolusi dimana $c_{i,j}$ merupakan hasil konvolusi untuk setiap sequens dan untuk setiap filter j , $\mathbf{x}_{i:i+h-1}$ adalah vektor *embedding* untuk setiap window $i:i+h-1$, $\mathbf{W}_{c(j)}$ adalah matriks pembobot kovolusi yang bernilai tetap ketika berjalan di window, $b_{c(j)}$ merupakan bias konvolusi, dan f merupakan fungsi aktivasi. *Output* dari konvolusi kemudian digabungkan menjadi sebuah matriks konvolusi $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_{n-h+1}]$.

Hasil konvolusi akan memiliki ukuran $(n_{\text{words}} - h + 1) \times \text{filter}$. Biasanya, fungsi aktivasi yang digunakan dalam CNN adalah fungsi aktivasi non-linear. Alasan penggunaan fungsi aktivasi non-linear adalah karena asumsi kuatnya terhadap pemrosesan bahasa alaminya yang membuatnya dapat merusak performa pemrosesan bahasa alami. Fungsi linear, mengasumsikan bahwa terdapat komposisi lokal dan tidak mengakomodasi dependensi jangka panjang. Selain itu, fungsi linear juga memakai parameter yang berbeda untuk setiap indeks waktu yang menghalangi pembagian parameter dalam kata yang sama (Yang, 2018).

2.2.3.4 Max Pooling

Max pooling merupakan *layer* untuk memaksimalkan *feature maps* hasil dari operasi konvolusi. Penggunaan *max pooling* memastikan hanya fitur yang memiliki nilai maksimal (signifikan) yang dipertahankan (Collobert, et al., 2011). Secara matematis, *max pooling* diekspresikan pada Persamaan 2.14.

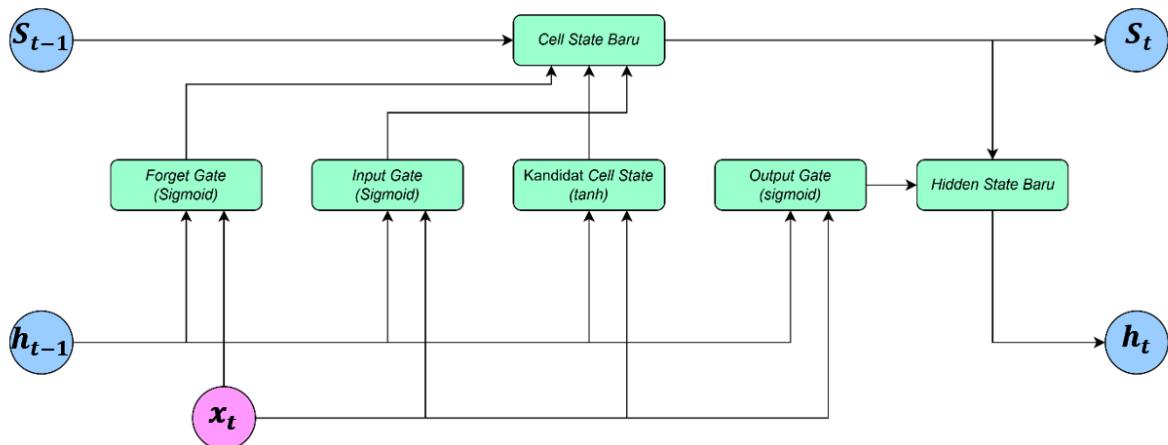
$$p_{i,j} = \max_{r=0}^{k_{pool}-1} c_{[(i:k_{pool}+r),j]} \quad (2.8)$$

$$\mathbf{p}_j = [\max(c_{0,j}, c_{1,j}, \dots, c_{k_{pool},j}), \dots, \max(c_{n_{words}-k_{pool},j}, \dots, c_{n_{words},j})]$$

Pada Persamaan (2.8), operasi maksimum akan berjalan untuk sekuens yang dihasilkan oleh operasi konvolusi, dimana c merupakan hasil *feature maps* (konvolusi) dengan i merupakan indeks untuk sekuens kata, j merupakan indeks untuk dimensi *embedding*/fitur hasil konvolusi, dan k_{pool} merupakan *pooling size*.

2.2.3.5 Long Short Term Memory (LSTM)

LSTM digunakan untuk menangkap informasi dominan, namun memiliki kelemahan dalam menangkap prediksi pada jangka panjang (Anggraeni, et al., 2024). Cara kerja satu unit LSTM terdiri atas tiga bagian utama seperti Gambar 2.2. Adapun deskripsi dari setiap bagian pada Gambar 2.2 dijelaskan pada Persamaan (2.9) – (2.14).



Gambar 2.2 Arsitektur LSTM
(Sumber: Dokumen Pribadi)

Bagian pertama adalah *forget gate*, dimana gerbang ini menentukan persentase *cell state* yang akan dilupakan.

$$f_t = \sigma(\mathbf{W}_f \cdot [x_t, h_{t-1}] + b_f) \quad (2.9)$$

Pada Persamaan (2.9), keluaran dari fungsi adalah probabilitas, jika $f_t \rightarrow 1$, maka informasi tersebut lebih cenderung dipertahankan. x_t merupakan vektor input ke- t , h_{t-1} adalah *hidden state* dari t sebelumnya, dan \mathbf{W}_f adalah matriks pembobot untuk *forget gate*. Bagian kedua adalah menetukan informasi signifikan berdasarkan *cell states* saat ini berdasarkan *input gates*.

$$i_t = \sigma(\mathbf{W}_i \cdot [x_t, h_{t-1}] + b_i) \quad (2.10)$$

Sama seperti Persamaan (2.9), Persamaan (2.10) akan mengeluarkan probabilitas, perbedaannya adalah f_t digunakan untuk menentukan apakah informasi dapat dilupakan sedangkan, i_t digunakan apakah input saat ini dapat dimasukkan ke dalam *cell state* (persen potensi *cell state* akan dipertahankan).

$$\tilde{S}_t = \tanh(\mathbf{W}_s \cdot [x_t, h_{t-1}] + b_s) \quad (2.11)$$

Pada Persamaan (2.11), \tilde{S}_t digunakan untuk membuat kandidat *cell state* baru yang dibuat berdasarkan *input gates*. Keluaran dari \tilde{S}_t adalah nilai antara -1 dan 1 sebagai akibat dari penggunaan fungsi tangen hiperbola.

$$C_t = f_t \odot S_{t-1} + i_t \odot \tilde{S}_t \quad (2.12)$$

Pada Persamaan (2.12), C_t adalah hasil *long term memory* (*cell state*) yang baru dan merupakan hasil akhir dari *input gate*. Pada persamaan ini digunakan produk Hadamard (*element wise multiplication*). Bagian terakhir dari LSTM adalah *output gate* yang digunakan untuk memperbarui *short term memory*.

$$O_t = \sigma(\mathbf{W}_o \cdot [x_t, h_{t-1}] + b_o) \quad (2.13)$$

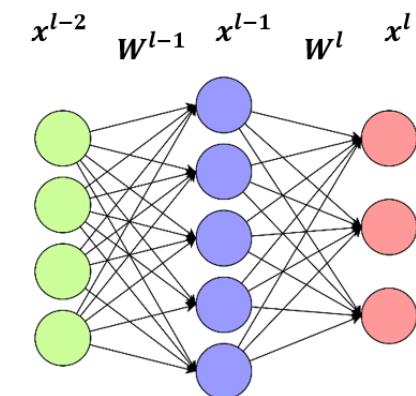
Pada Persamaan (2.13), O_t digunakan sebagai penentu berapa persentase atau bagian dari *cell state* yang menjadi *hidden state*.

$$h_t = O_t \odot \tanh(C_t) \quad (2.14)$$

Barulah *hidden state* final dihitung Persamaan (2.14) yang dimasukkan ke dalam step selanjutnya.

2.2.4 Backpropagation

Backpropagation merupakan mekanisme dimana sebuah parameter pada setiap neuron pada *layer* mempelajari nilai pembobot optimum menggunakan gradien. Mekanisme ini memerlukan nilai gradien untuk setiap pembobot menggunakan aturan rantai (Anggraeni, et al., 2024). Dalam *backpropagation* tugasnya adalah untuk mencari nilai optimum sebuah parameter menggunakan turunan fungsi *loss* terhadap parameter.



Gambar 2.3 Ilustrasi Neural Network
(Sumber: Dokumen Pribadi)

Untuk menyimulasikan sebuah mekanisme *backpropagation*, digunakan contoh *simple neural network* pada Gambar 2.3 dengan empat neuron input (*layer* ke- $(l - 2)$), kemudian lima neuron linear (*layer* ke- $(l - 1)$), dan dilanjutkan dengan tiga neuron *softmax* (*layer* ke- l).

$$\widehat{y}_k = \text{Softmax}(z_k),$$

$$L = - \sum_{k=1}^K y_k \ln \widehat{y}_k \quad (2.15)$$

Jika diambil satu buah neuron pada *layer* terakhir, maka akan menghasilkan sebuah \widehat{y}_k yang merupakan probabilitas prediksi dan y_k adalah kelas yang sebenarnya. Untuk menghitung *loss* digunakan *softmax* dari *layer* terakhir dan dimasukkan ke dalam fungsi *loss* pada Persamaan (2.15) yang disebut sebagai *cross entropy*. Dalam menghitung turunan *loss* terhadap parameter, digunakan turunan parsial dan aturan rantai (Anggraeni, et al., 2024). Hal ini dikarenakan, fungsi *loss* merupakan fungsi dari *softmax* yang merupakan fungsi dari kombinasi linear, $L = L(\text{Softmax}(z(x, w)))$. Sehingga sesuai dengan aturan rantai, turunan fungsi *loss* L terhadap parameter *neuron* ke- k dan *layer* ke- l (\widehat{w}_k^l) dapat dituliskan pada Persamaan (2.16).

$$\frac{\partial L}{\partial \widehat{w}_k^l} = \frac{\partial L}{\partial z_k^l} \cdot \frac{\partial z_k^l}{\partial w_k^l} \quad (2.16)$$

Pada Persamaan (2.16), diketahui bahwa $\frac{\partial L}{\partial z_k^l} = \widehat{y}_k - y_k$, kemudian z^l merupakan kombinasi linear dari keluaran *layer* ke- $(l-1)$. Menggunakan Persamaan 2.2, maka turunan parsial z^l terhadap parameter adalah x^{l-1} (keluaran dari *layer* sebelumnya).

$$\frac{\partial L}{\partial \widehat{w}_k^l} = (\widehat{y}_k - y_k) \cdot x_k^{l-1} \quad (2.17)$$

Sehingga, untuk meminimalkan fungsi *loss* berdasarkan aturan rantai didapatkanlah Persamaan (2.17). Dengan cara yang sama, algoritma ini dapat diterapkan untuk *layer-layer* sebelumnya. Menggunakan aturan rantai turunan parsial, gradien terhadap bobot *layer* yang sebelumnya dapat dipecah menjadi beberapa bagian seperti Persamaan (2.18).

$$\frac{\partial L}{\partial \widehat{w}_k^{l-1}} = \frac{\partial L}{\partial z_k^l} \cdot \frac{\partial z_k^l}{\partial z_k^{l-1}} \cdot \frac{\partial z_k^{l-1}}{\partial \widehat{w}_k^{l-1}} \quad (2.18)$$

Diketahui bahwa $\frac{\partial L}{\partial z_k^l} = \widehat{y}_k - y_k$, z_k^l mendapatkan *output* yang berasal dari kombinasi linear yang sebelumnya yakni z_k^{l-1} , sehingga $\frac{\partial z_k^l}{\partial z_k^{l-1}} = w_k^l$. Kemudian, turunan $\frac{\partial z_k^{l-1}}{\partial \widehat{w}_k^{l-1}}$ adalah x_k^{l-2} , karena merupakan kombinasi linear dengan input di-($l-2$).

Untuk melakukan pembaruan parameter, digunakan algoritma optimasi. Algoritma optimasi yang sering digunakan adalah Adam (*adaptive moment*). Algoritma ini menggunakan *exponential moving average* untuk melakukan pembaruan parameter (Kingma & Ba, 2015). Adam memerlukan parameter η sebagai *learning rate* dan $0 \leq \beta_1, \beta_2 < 1$ sebagai parameter *moving average* momen pertama dan kedua. Parameter-parameter ini dimasukkan ke dalam sebuah persamaan *exponential moving average* seperti pada Persamaan (2.19) dan (2.20), dimana pada inisialisasi, $m_{w^l}^t$ dan $v_{w^l}^t$ diisi oleh nilai 0.

$$m_{w^l}^{t+1} = \beta_1 \cdot m_{w^l}^t + (1 - \beta_1) \cdot \frac{\partial L}{\partial \widehat{w}_k^l} \quad (2.19)$$

$$\boldsymbol{v}_{w^l}^{t+1} = \beta_2 \cdot \boldsymbol{v}_{w^l}^t + (1 - \beta_2) \cdot \left[\frac{\partial L}{\partial \widehat{\boldsymbol{w}_k}^l} \odot \frac{\partial L}{\partial \widehat{\boldsymbol{w}_k}^l} \right] \quad (2.20)$$

Pada proses *training* awal, dikarenakan penentuan bobot (secara acak untuk *initial weight*) W^l bisa saja bernilai $\mathbf{0}$, maka efeknya adalah *moving average* yang cenderung menuju nilai $\mathbf{0}$ untuk nilai β_1 dan β_2 yang besar. Oleh karena itu, diperlukan suatu koreksi yang ditunjukkan pada Persamaan (2.21) dan (2.22).

$$\widehat{\mathbf{m}_{w^l}^{t+1}} = \frac{\mathbf{m}_{w^l}^{t+1}}{(1 - \beta_1)} \quad (2.21)$$

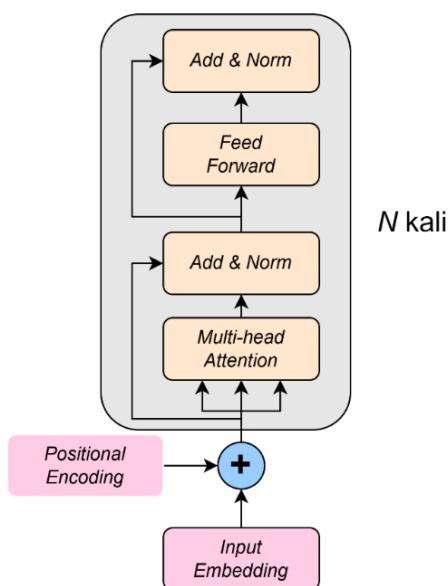
$$\widehat{\boldsymbol{v}_{w^l}^{t+1}} = \frac{\boldsymbol{v}_{w^l}^{t+1}}{(1 - \beta_2)} \quad (2.22)$$

$$\hat{w}_k^{l(t+1)} = \hat{w}_k^{l(t)} - \eta \cdot \frac{\hat{m}_{w^l}^{t+1}}{\sqrt{\hat{v}_{w^l}^{t+1}} + \epsilon} \quad (2.23)$$

Persamaan (2.23) digunakan untuk memperbarui parameter dengan nilai yang telah diuji pada *machine learning* adalah $\beta_1 = 0,9$; $\beta_2 = 0,99$; dan $\epsilon = 10^{-8}$ (Kingma & Ba, 2015).

2.2.5 Bidirectional Encoder Representation from Transformer (BERT)

BERT adalah model *large language* berbasis *transformer* yang dikembangkan oleh Google pada tahun 2018. BERT merupakan pengembangan dari mekanisme *transformer*. Perbedaan utama antara BERT dengan mekanisme *transformer* adalah, BERT hanya melakukan *encode* pada model bahasa sehingga hanya memerlukan *encoder* dari *transformer* (Pratama & Romadhony, 2020). Mekanisme *attention* (dari *transformer*) pada *encoder* BERT membuat BERT mampu memahami konteks kata dengan memperhatikan probabilitas kata. Selain itu, BERT merupakan sebuah *pre-trained model*. Sehingga untuk menyesuaikan dengan dataset yang berbeda tidak perlu dilakukan *training* ulang, cukup dengan melakukan *transfer learning* maupun *fine tuning*. Yang membuatnya fleksibel dalam skalabilitas, baik skala kecil maupun skala besar.



Gambar 2.4 Arsitektur *Encoder* dalam *Transformer*
 (Sumber: Dokumen Pribadi)

Gambar 2.4 menjelaskan arsitektur *encoder* yang ada pada *transformer*. Mulanya, model menerima sebuah input berupa *embedding* yang merupakan representasi kata dalam bentuk vektor berdimensi tinggi. Selanjutnya, matriks ini akan ditambahkan dengan *positional embedding* yang menunjukkan posisi dari sebuah kata. Lalu, sesuai dengan Gambar 2.4, hasil operasi matriks akan dibagi menjadi dua bagian, satu untuk masuk ke dalam *multihead attention* dan lainnya masuk ke bagian setelah *multihead* untuk kemudian ditambahkan dan dinormalisasi. Bagian ini kemudian diulang sebanyak N kali sesuai dengan versi BERT yang digunakan (Devlin, Chang, Lee, & Toutanova, 2019)

2.2.5.1 Embedding

Proses BERT dimulai pada ketika mendapatkan sebuah kalimat. Kalimat ini akan diubah menjadi sebuah *token embedding* dengan mekanisme WordPiece. Prinsip tokenisasi ini adalah dengan membagi sebuah kata hingga level sub-kata seperti kata dasar atau imbuhan baik itu di awal atau di akhir (Jurafsky & Martin, 2024). Misalkan, kalimat “saya membeli ikan di pasar,” jika dilakukan tokenasi, akan menjadi [“saya”, “mem##”, “beli”, “buah”, “di”, “pasar”]. Kemudian sistem *WordPiece* akan melakukan *lookup* terhadap kode/id dari kamus milik BERT. Maka, kalimat tersebut akan diubah menjadi: [90, 231, 413, 559, 7540, 952] yang disebut sebagai token. Token ini adalah representasi dari vektor *embedding* (id untuk vektor *embedding*). Misalkan, *token embedding* dari token tersebut direpresentasikan pada matriks *embedding* pada Persamaan (2.24).

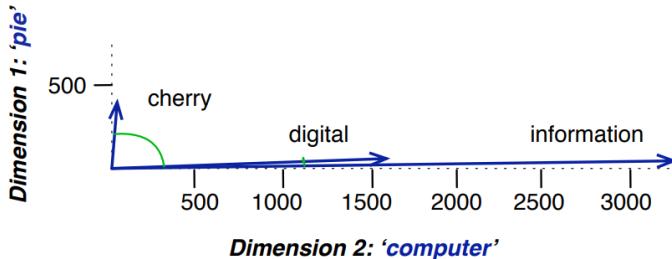
$$X_{\text{token}} = \begin{bmatrix} 0,56 & 0,92 & 0,70 & \cdots & 1,97 \\ 0,85 & 0,99 & 0,74 & \cdots & 0,45 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0,207 & 0,84 & 1,65 & \cdots & 0,529 \end{bmatrix}_{6 \times 768} \quad (2.24)$$

Ukuran dari *embedding* tersebut adalah $n_{\text{words}} \times d_{\text{model}}$, dimana ukuran *embedding* atau disebut d_{model} dari BERT-base adalah 768. Ukuran *embedding* 768 (dengan jumlah *multihead* sebanyak 12) dipilih karena mempu menghasilkan metrik yang lebih baik dibandingkan kombinasi lainnya (Devlin, Chang, Lee, & Toutanova, 2019). Pada ilustrasi di atas, jumlah token/kata yang digunakan ada enam, sehingga ukuran dari *embedding* adalah jumlah token/kata dikalikan dengan d_{model} , yakni 6×768 . Nilai-nilai pada matriks *embedding* didapatkan dengan cara melakukan *training* pada data teks. Awalnya, nilai-nilai pada matriks *embedding* diisi menggunakan nilai random pada inisialisasi, kemudian dilakukan pembaruan nilai menggunakan *backpropagation* untuk melakukan optimasi berdasarkan gradien (Jurafsky & Martin, 2024).

Penggunaan matriks *embedding* berdimensi tinggi memungkinkan untuk diketahui seberapa dekat suatu kata dengan suatu kata lainnya. *Cosine similarity* dapat digunakan untuk mengitung kesamaan antar kata yang direpresentasikan oleh vektor *embedding*. Misalkan matriks *embedding* pada Persamaan (2.24), diambil dua baris yang berbeda yang merepresentasikan kata, sehingga didapatkan dua vektor berbeda yang masing-masing berukuran 768×1 . Untuk mengetahui kedekatan nilai dua kata berbeda yang direpresentasikan oleh vektor-vektor tersebut dapat digunakan *cosine similarity*.

$$\text{cosine}(\mathbf{e}_i, \mathbf{e}_j) = \frac{\mathbf{e}_i \cdot \mathbf{e}_j}{|\mathbf{e}_i| |\mathbf{e}_j|} = \frac{\sum_{l=1}^{d_{\text{model}}} e_{i,l} \cdot e_{j,l}}{\sqrt{\sum_{l=1}^{d_{\text{model}}} e_{i,l}^2} \cdot \sqrt{\sum_{l=1}^{d_{\text{model}}} e_{j,l}^2}} \quad (2.25)$$

Pada Persamaan (2.25), e_i merupakan vektor *embedding* kata ke- i dan e_j merupakan vektor *embedding* kata ke- j yang masing-masing berukuran 768×1 , sedangkan indeks l merupakan indeks posisi di dalam vektor *embedding*. Informasi yang bisa didapatkan pada hasil Persamaan (2.25), jika nilai mendekati 1, maka kata yang diproyeksikan memiliki makna menuju satu konteks yang sama. Jika nilai mendekati -1, maka kata tersebut memiliki makna konteks yang berlawanan.



Gambar 2.5 Ilustrasi Kedekatan Nilai *Cosine Similarity*

(Sumber: Jurafsky & Martin, 2024)

Pada Gambar 2.5, kata “digital” dan “informasi” memiliki nilai *cosine* mendekati 1, sehingga kata tersebut memiliki konteks yang sama, dalam hal ini mereka memiliki konteks yang berhubungan dengan komputer. Sedangkan, kata “cherry” hampir tegak lurus dengan dua kata lainnya, sehingga kata “cherry” memiliki konteks yang berbeda dengan dua kata lainnya.

Selain *token embedding*, BERT membutuhkan input lainnya yakni *positional embedding*. *Positional embedding* pada BERT memiliki dimensi yang sama dengan *token embedding*.

$$\begin{bmatrix} pos(0,0) & pos(0,1) & pos(0,2) & \dots & pos(0,768) \\ pos(1,0) & pos(1,1) & pos(1,2) & \dots & pos(1,768) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ pos(6,0) & pos(6,1) & pos(6,2) & \dots & pos(6,768) \end{bmatrix}_{6 \times 768} \quad (2.26)$$

Baris pertama dari *positional embedding* akan memuat informasi *position* untuk token pertama, hingga token ke-6. Perhitungan position dihitung menggunakan fungsi sinus dan cosinus.

$$Pos(pos, 2i) = \sin\left(\frac{\sin}{10000^{\frac{2i}{d_{model}}}}\right) \quad (2.27)$$

$$Pos(pos, 2i + 1) = \cos\left(\frac{\cos}{10000^{\frac{2i}{d_{model}}}}\right) \quad (2.28)$$

Pada Persamaan (2.27) dan (2.28), *pos* dalam argumen merupakan posisi dari token di sequens (kalimat), *i* merupakan indeks elemen di dalam vektor *embedding*, dan d_{model} merupakan dimensi *embedding* (Saeed, 2023). Penggunaan fungsi sinusoid akan menghasilkan panjang gelombang yang membentuk deret geometri dari 2π hingga $2\pi \cdot 10000$ (Vaswani, et al., 2017).

Kemudian, *segment embedding* digunakan untuk membedakan kalimat berdasarkan tanda bacanya. Terakhir, pada *position embedding*, BERT akan mengambil informasi dari posisi token di dalam kalimat tersebut.

$$Input = Token + Positional + Segment \quad (2.29)$$

Pada Persamaan (2.29), *input embedding* menghasilkan dimensi yang sama dengan *token embedding*. *Input embedding* kemudian akan masuk ke dalam pemrosesan utama dari BERT.

2.2.5.2 Mekanisme Attention

Mekanisme *transformer* pada BERT menggunakan skema *multi head attention*. Artinya, setiap layer *attention* pada BERT memiliki beberapa pemrosesan *self attention*. Untuk memudahkan penjelasan, diilustrasikan proses pada *single head attention*. Sebagai input, misalkan \mathbf{X} adalah matriks *embedding*, matriks ini akan dilakukan *dot product* dengan matriks pembobot yang menghasilkan matriks \mathbf{Q} , \mathbf{K} , dan \mathbf{V} yang merupakan perkalian antara matriks *embedding* dengan pembobot \mathbf{W}^Q , \mathbf{W}^K , dan \mathbf{W}^V yang masing-masing matriks pembobot berukuran $d_{model} \times d_{model}$. Sehingga, ukuran dari masing-masing matriks \mathbf{Q} , \mathbf{K} , dan \mathbf{V} tersebut adalah $n_{words} \times d_{model}$. Matriks ini kemudian akan dimasukkan ke dalam fungsi softmax dengan persamaan berikut.

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{model}}}\right)\mathbf{V} \quad (2.30)$$

Pada Persamaan (2.30), *input* pada *softmax* adalah normalisasi dari perkalian matriks \mathbf{Q} dan \mathbf{K}^T , dan d_{model} adalah jumlah dimensi dari embedding yakni 768. Kembali pada Persamaan (2.5), keluaran dari layer *softmax* adalah sebuah probabilitas. Dalam kasus ini, probabilitas yang dikeluarkan setelah dikalikan dengan matriks \mathbf{V} adalah matriks *embedding* yang telah diboboti dengan seberapa besar kontribusi suatu token terhadap token yang lain. Hal inilah yang menyebabkan model *transformer* dapat menangkap konteks karena *attention* menghasilkan pembobutan kata berdasarkan probabilitas yang menghasilkan *embedding* baru.

Mekanisme *transformer* pada BERT bukanlah *single head attention* melainkan *multihead attention*. Dalam *multihead attention*, alih-alih menggunakan keseluruhan dimensi *embedding* secara langsung, matriks *embedding* akan dipecah-pecah dalam prosesnya. Pertama, didapatkan matriks *embedding* \mathbf{X} dan akan dilakukan perkalian dengan matriks pembobot \mathbf{W}^Q , \mathbf{W}^K , dan \mathbf{W}^V , dimana matriks \mathbf{W} adalah matriks pembobot berukuran $d_{model} \times d_{model}$. Perkalian ini akan menghasilkan sebuah matriks baru \mathbf{Q} , \mathbf{K} , dan \mathbf{V} . Matriks ini kemudian dipotong sehingga ukurannya masing-masing adalah $d_{model} \times d_{head}$, dimana $d_{head} = d_{model}/head$ dan *head* adalah jumlah *multihead attention* (dalam BERT-base ukuran *head* adalah 12) (Jurafsky & Martin, 2024). Sehingga, akan terdapat 12 buah matriks \mathbf{Q}_i , \mathbf{K}_i , dan \mathbf{V}_i .

$$\mathbf{head}_i = \text{Attention}(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i) \quad (2.31)$$

Persamaan (2.31), merupakan persamaan *attention* dari Persamaan (2.30), dimana ini akan menghasilkan *attention* untuk setiap bagian *head* dengan pembobot yang berbeda-beda (Vaswani, et al., 2017). Setiap bagian *head* ini kemudian disatukan menggunakan proses konkatenasi.

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\mathbf{head}_1, \mathbf{head}_2, \dots, \mathbf{head}_{12})\mathbf{W}^O \quad (2.32)$$

Pada Persamaan (2.32), pembobot \mathbf{W}^O adalah pembobot integrasi dari proses pemotongan sebelumnya, sehingga keluarannya adalah matriks berukuran $n_{words} \times 768$.

2.2.6 IndoBERTweet

IndoBERTweet merupakan model *large language* yang didasarkan pada mekanisme BERT yang dikhususkan untuk *corpus* Bahasa Indonesia yang dilakukan *training* pada bahasa media sosial Twitter (atau sekarang disebut X). Mekanisme *pre-training* yang dilakukan oleh IndoBERTweet adalah *training* berdasarkan *masked language* menggunakan *framework* Huggingface, sama seperti proses yang digunakan dalam proses *training* IndoBERT. Dimana untuk model IndoBERT-*base-uncased* memiliki konfigurasi 12 *hidden layer* dengan total 768 dimensi di dalamnya, 12 *multihead attention*, dan total *feed forward neural network* sebanyak 3.072 dimensi. Pada proses pengembangan, model BERT dilakukan *training* data X dengan total 26 juta cuitan dan 409 juta token kata dengan ukuran tumpak 128, *learning rate* 10^{-4} dengan *Adam optimizer*, dan penjadwal linier, kemudian didapatkanlah model IndoBERTweet (Koto, Lau, & Baldwin, 2021).

Sebagai perbandingan dalam mengevaluasi kinerja model untuk tugas analisis sentimen pada dataset Twitter berbahasa Indonesia, dilakukan perbandingan antara beberapa model berbasis Transformer, yaitu IndoBERTweet, mBERT (Multilingual BERT), MalayBERT (BERT untuk bahasa Melayu), serta IndoBERT.

Tabel 2.2 Perbandingan Model BERT untuk Tugas Analisis Sentimen

Model	Skor F1 (%)
mBERT	76,60
MalayBERT	82,00
IndoBERT	84,13
IndoBERTweet	86,20

Pada Tabel 2.2 Perbandingan Model BERT untuk Tugas Analisis Sentimen, IndoBERTweet memiliki skor F1 tertinggi untuk tugas analisis sentimen atau klasifikasi teks (Koto, Lau, & Baldwin, 2021). Hal ini menunjukkan bahwa untuk korpus bahasa Indonesia khususnya pada korpus bahasa Twitter, IndoBERTweet memiliki performa yang lebih mumpuni dalam melakukan berbagai tugas yang biasa dilakukan oleh model BERT, bahkan ketika dibandingkan dengan model pendahulunya yakni IndoBERT.

2.2.7 Penanganan Ketidakseimbangan Kelas dan *Overfitting*

2.2.7.1 Class Weight

Dalam permasalahan ketidakseimbangan kelas, dalam *neural network* dapat diatasi salah satunya adalah dengan menggunakan *class weight* atau pembobot kelas. Dengan menggunakan *class weight*, nilai *loss* yang digunakan dalam *back propagation* akan berfokus lebih tinggi pada optimisasi parameter model pada kelas dengan *class weight* yang tinggi. Sehingga, persamaan *loss* pada (2.15) akan diubah menjadi Persamaan (2.33), dimana ω_k merupakan pembobot pada kelas k .

$$L_\omega = - \sum_{k=1}^K \omega_k \cdot y_k \ln \hat{y}_k \quad (2.33)$$

$$\omega_k = \frac{n}{K \cdot n_k} \quad (2.34)$$

Sedangkan, untuk menghitung pembobot untuk kelas k , digunakan Persamaan (2.34), dimana K merupakan jumlah kelas, n_k merupakan jumlah data untuk kelas k , dan n adalah jumlah data.

2.2.7.2 Augmentasi Data *Back Translation*

Data Augmentation merupakan sebuah teknik untuk memperbanyak dataset dan dibuktikan mampu memberikan performa lebih baik pada *machine learning* atau *deep learning* (Pellicer, Ferreira, & Costa, 2023). Pada NLP, *data augmentation* bisa dilakukan dengan menggunakan *Back Translation* (BT). *Back translation* merupakan salah satu teknik *Data Space Data Augmentation*, dimana pembuatan dataset baru didasarkan pada teks. Hasil modifikasi ini akan menghasilkan dataset baru yang bervariasi pada level karakter hingga level dokumen (parafrasa).



Gambar 2.6 Ilustrasi Proses *Back Translation*
(Sumber: Dokumen Pribadi)

Back translate bekerja dengan menerjemahkan kalimat dari bahasa awal ke dalam bahasa lain B, kemudian menerjemahkannya kembali ke bahasa awal seperti pada Gambar 2.6 (Xie, Dai, Hovy, Luong, & Le, 2020). Hal ini bertujuan untuk membuat teks baru hasil dari pengungkapan kembali teks awal tanpa mengubah pengertian atau artinya. Jika dilakukan dengan benar, dapat memperkaya frasa yang digunakan model dan meningkatkan performa model untuk *question answering* dan klasifikasi.

2.2.7.3 Regularisasi L2

Regularisasi merupakan teknik untuk mengurangi galat pada data *test*, sehingga meningkatkan kemampuan model dalam generalisasi (bekerja baik pada data yang belum pernah dilihat). Salah satu jenis regularisasi adalah regularisasi L2 yang bekerja dengan memodifikasi *loss* untuk menambahkan hukuman pada bobot-bobot yang telalu besar (Tewari, 2021).

$$L_\lambda = L + \lambda \sum_{i=0}^{n_{neuron}} w_i^{l^2} \quad (2.35)$$

λ pada Persamaan (2.35) merupakan parameter regularisasi yang nilainya > 0 , w_i^l merupakan pembobot untuk neuron layer ke- l dan unit ke- i , dan L_λ adalah *loss* hasil regularisasi.

Persamaan (2.35) akan menyebabkan *loss* yang besar ketika terdapat parameter yang terlampaui besar.

2.2.8 Ukuran Kebaikan Klasifikasi

Salah satu ukuran kebaikan model dalam melakukan proses klasifikasi adalah *confusion matrix*. *Confusion matrix* merupakan hasil tabulasi dari jumlah kejadian antara kombinasi benar yang dihasilkan oleh prediksi (*predicted classification*) dengan nilai aslinya (*true/actual classification*) (Grandini, Bagli, & Visani, 2020). Dalam kasus klasifikasi multikelas, *confusion matrix* memiliki ukuran $K \times K$, dimana K merupakan jumlah kelas. Sehingga, berbeda dengan *confusion matrix* untuk kelas biner, TP, TN, FP , dan FN tidak dapat ditentukan langsung elemennya dalam *confusion matrix* tersebut, namun dapat dilakukan perhitungan dengan cara berfokus pada kelas spesifik (Markoulidakis, et al., 2021).

Tabel 2.3 *Confusion Matrix* Multi Kelas

Kelas	Klasifikasi			
	C_1	C_2	...	C_K
Nilai Aktual	$C_{1,1}$ [$C_1 \rightarrow C_1$]	$C_{1,2}$ [$C_1 \rightarrow C_2$]	...	C_{1,n_K} [$C_1 \rightarrow C_{n_K}$]
	$C_{2,1}$ [$C_2 \rightarrow C_1$]	$C_{2,2}$ [$C_2 \rightarrow C_2$]	...	C_{2,n_K} [$C_2 \rightarrow C_{n_K}$]
	:	:	:	:
	$C_{K,1}$ [$C_K \rightarrow C_1$]	$C_{K,2}$ [$C_K \rightarrow C_2$]	...	$C_{K,K}$ [$C_K \rightarrow C_K$]

Tabel 2.3 merupakan contoh bentuk umum dari *confusion matrix* multi kelas, dimana kelas yang digunakan adalah C_1, C_2, \dots, C_K dengan K adalah jumlah kelas. $C_{1,1}$ pada Tabel 2.3 menunjukkan jumlah tabulasi kelas yang diprediksi benar sebagai C_1 ($C_1 \rightarrow C_1$), sedangkan $C_{2,1}$ menunjukkan jumlah tabulasi kelas yang secara aktual adalah C_2 , namun diprediksi sebagai C_1 ($C_2 \rightarrow C_1$), begitu seterusnya untuk setiap kelas. Untuk menghitung ukuran kebaikan model pada klasifikasi multi kelas, *confusion matrix* ini dapat difokuskan untuk setiap kelas dalam keseluruhan kelas dan dapat digunakan untuk menghitung kebaikan model keseluruhan.

Misalkan, dari Tabel 2.3 akan difokuskan untuk kelas C_2 , maka akan menghasilkan *confusion matrix* berikut.

Tabel 2.4 *Confusion Matrix* yang Difokuskan Untuk Kelas C_2

Kelas	Klasifikasi			
	C_1	C_2	...	C_K
Nilai Aktual	$C_{1,1}$ (TN_{C_2})	$C_{1,2}$ (FP_{C_2})	...	$C_{1,K}$ (TN_{C_2})
	$C_{2,1}$ (FN_{C_2})	$C_{2,2}$ (TP_{C_2})	...	$C_{2,K}$ (FN_{C_2})
	:	:	:	:
	$C_{K,1}$ (TN_{C_2})	$C_{K,2}$ (FP_{C_2})	...	$C_{K,K}$ (TN_{C_2})

Dari Tabel 2.4, didapatkan TP_{C_2} (*True Positive* C_2) yang merupakan jumlah tabulasi kelas C_2 yang benar diklasifikasikan C_2 atau dalam artian lain, TP_{C_k} adalah elemen *confusion matrix* $C_{k,k}$. FP_{C_2} (*False Positive* C_2) merupakan jumlah tabulasi kelas yang sebetulnya bukan dari C_2 , namun diklasifikasikan C_2 , atau secara matematis FP_{C_k} adalah jumlahan semua elemen *confusion matrix* pada kolom ke- k yang bukan baris ke- k . FN_{C_2} (*False Negative* C_2) merupakan jumlah tabulasi yang berasal dari kelas C_2 , namun tidak diklasifikasikan C_2 , atau dalam penjabaran matematis, FN_{C_k} merupakan jumlahan semua elemen *confusion matrix* pada baris ke- k yang bukan kolom ke- k . Komponen lainnya adalah TN_{C_2} (*True Negative* C_2), yakni jumlah tabulasi yang bukan berasal dari C_2 dan benar tidak diklasifikasikan C_2 , atau secara matematis adalah TN_{C_k} merupakan penjumlahan semua elemen *confusion matrix* yang bukan di baris dan kolom ke- k .

Dari informasi pada *confusion matrix* dapat diperoleh beberapa metrik penting dalam klasifikasi, seperti presisi, *recall*, akurasi, dan skor F1. Akurasi digunakan untuk mengukur seberapa bagus model dalam memprediksi keseluruhan data, setiap kelas dianggap memiliki bobot dan pengaruh yang sama (Grandini, Bagli, & Visani, 2020). Akurasi memiliki formula sebagai berikut.

$$\text{Akurasi} = \frac{\sum_{m=1}^K C_{m,m}}{\sum_{m=1}^K \sum_{n=1}^K C_{m,n}} \quad (2.36)$$

Akurasi pada Persamaan (2.36) menunjukkan seberapa sering atau tingkat model mengklasifikasikan benar kelas dari data yang diberikan, tanpa memperhatikan distribusi antar kelas. Artinya, Jika model memprediksi 80 dari 100 data dengan benar (terlepas dari jenis kelasnya), maka akurasi adalah 80%

Berbeda dengan akurasi, *recall* mengukur kebaikan model dalam menemukan data yang berasal dari kelas acuan. Sehingga, semakin tinggi *recall*, maka semakin sedikit data yang terlupakan untuk kelas acuan tersebut.

$$Recall_k = \frac{C_{k,k}}{C_{k,k} + \sum_{m=1, m \neq k}^K C_{k,m}} \quad (2.37)$$

Merujuk pada Tabel 2.4, maka *recall* untuk kelas C_2 adalah kelas yang jumlah yang diprediksi C_2 dengan pembagi seluruh *instance* yang berasal dari kelas C_2 seperti pada Persamaan (2.37). Berbeda dengan *recall*, presisi mengukur kebenaran prediksi kelas acuan terhadap semua yang diprediksi kelas acuan (Grandini, Bagli, & Visani, 2020). Presisi memiliki formula berikut.

$$Presisi_k = \frac{C_{k,k}}{C_{k,k} + \sum_{m=1, m \neq k}^K C_{m,k}} \quad (2.38)$$

Menurut Persamaan (2.38), dari studi kasus yang sama, presisi untuk kelas C_2 adalah jumlah yang berasal dari kelas C_2 dan benar diklasifikasikan C_2 dibagi dengan jumlah yang diprediksi C_2 . Salah satu metrik yang sering digunakan dalam prediksi multi kelas adalah skor F1 yang merupakan kombinasi antara *recall* dan presisi.

$$F1_k = \frac{2}{\frac{1}{Presisi_k} + \frac{1}{Recall_k}} \quad (2.39)$$

Skor F1 pada Persamaan (2.39) menggunakan konsep *harmonic mean* dari *recall* dan presisi dalam formulanya. Dari skor metrik setiap kelas, untuk mengetahui kebaikan model secara

keseluruhan, maka dapat digunakan konsep *weighted F1 Score* dimana itu adalah skor F1 yang mengakomodir ketidakseimbangan kelas.

$$\text{Weighted F1 Score} = \sum_{k=1}^K \left(\frac{n_k}{N} \text{F1}_k \right) \quad (2.40)$$

Pada Persamaan (2.40), n_k adalah jumlah data pada kelas ke k dan N adalah jumlah keseluruhan data

2.2.9 Kendall's Tau

Kendall's Tau bekerja dengan mengukur tingkat hubungan antara dua variabel dengan skala minimal ordinal berdasarkan perbedaan jumlah pasangan data yang wajar (*concordant*) dan tidak wajar (*discordant*) dalam urutan peringkatnya (Daniel, 1990). Hipotesis untuk pengujian ini digunakan untuk menguji adanya korelasi peringkat, dimana $H_0: \tau = 0$ dimana tidak ada *concordant* atau *discordant* antara dua variabel, sebaliknya $H_1: \tau \neq 0$ artinya terdapat *concordant* atau *discordant* antara dua variabel.

$$\hat{\tau} = \frac{n_c - n_d}{\frac{n(n-1)}{2}} \quad (2.41)$$

Pada Persamaan (2.41), n_c merupakan jumlah kejadian urutan wajar (*concordant*), n_d merupakan jumlah kejadian urutan tidak wajar (*discordant*), dan n adalah jumlah observasi. Apabila semua pasangan data berada dalam urutan yang wajar, maka $n_c - n_d$ pada Persamaan (2.41) adalah $n(n-1)/2$. Hal ini menunjukkan adanya korelasi positif sempurna antara peringkat dua variabel tersebut. Sedangkan, apabila semua pasang data berada pada urutan yang tidak wajar, maka $n_c - n_d = -n(n-1)/2$, menunjukkan korelasi negatif sempurna.

Secara matematis, perhitungan jumlah $K_\tau = n_c - n_d$ dimulai dengan mengurutkan pasangan data. Kemudian, untuk $1 \leq i < j \leq n$, dihitung terlebih dahulu $\xi(x_i, x_j, y_i, y_j)$ (Hollander & Wolfe, 1973). Dimana fungsi tersebut dijelaskan pada Persamaan (2.42) dengan x_i, x_j dan y_i, y_j merupakan pasangan data berurut.

$$\xi(x_i, x_j, y_i, y_j) = \begin{cases} 1, & \text{jika } (x_i - x_j)(y_i - y_j) > 0 \\ -1, & \text{jika } (x_i - x_j)(y_i - y_j) < 0 \end{cases} \quad (2.42)$$

$$K_\tau = n_c - n_d = \sum_{i=1}^{n-1} \sum_{j=i+1}^n \xi(x_i, x_j, y_i, y_j) \quad (2.43)$$

Kemudian, untuk mencari K_τ , digunakan penjumlahan fungsi $\xi(x_i, x_j, y_i, y_j)$ untuk setiap permutasi pasangan data berurutan menggunakan Persamaan (2.43).

(Halaman ini sengaja dikosongkan)

BAB III

METODOLOGI

3.1 Sumber Data

Data yang digunakan dalam penelitian ini adalah data opini masyarakat terkait Pemilihan Gubernur Jawa Timur (Pilkada Jatim) 2024 yang diperoleh dari media sosial X. Pengumpulan data opini mengenai Pilkada Jatim dilakukan dengan metode pengambilan data melalui *scraping/crawling* cuitan X pada masa kampanye yakni pada 25 September 2024 – 23 November 2024. Data opini yang dianalisis merupakan tweet yang ditulis dalam bahasa Indonesia. Selain itu, digunakan pula data hasil rekapitulasi KPU serta hasil survei elektabilitas dari berbagai lembaga survei pada rentang waktu 1 Oktober – 19 November 2024.

3.2 Variabel Penelitian

Dalam penelitian ini terdapat dua tahapan analisis yakni klasifikasi sentimen dan analisis korelasi *rank* menggunakan Kendall Tau. Untuk klasifikasi sentimen, digunakan nama pasangan calon, cuitan, dan label seperti diilustrasikan pada Tabel 3.1.

Tabel 3.1 Variabel Penelitian Analisis Sentimen

Variabel	Deskripsi
Nomor Pasangan Calon	Nomor pasangan calon, hanya digunakan untuk melakukan tabulasi
Cuitan	Teks yang dituliskan oleh pengguna di X
Label (y)	Label sentimen, terdiri atas (0) negatif, (1) netral, dan (2) positif

Sedangkan, variabel penelitian untuk analisis korelasi *rank* Kendall Tau diperoleh dari hasil tabulasi persentase jumlah sentimen berdasarkan nomor pasangan calon seperti diilustrasikan pada Tabel 3.2.

Tabel 3.2 Variabel Penelitian Analisis Korelasi Rank Kendall Tau

Variabel	Deskripsi
Nomor Pasangan Calon	Nomor pasangan calon, hanya digunakan untuk melakukan tabulasi
Persentase Sentimen Positif	Hasil tabulasi sentimen positif setiap pasangan calon dalam bentuk persentase
Persentase Sentimen Negatif	Hasil tabulasi sentimen negatif setiap pasangan calon dalam bentuk persentase
Hasil KPU	Hasil rekapitulasi KPU dalam bentuk persentase

3.3 Struktur Data

Dalam penelitian ini terdapat dua tahapan analisis yakni klasifikasi sentimen dan analisis korelasi *rank*. Sehingga terdapat dua macam struktur data yang digunakan pada penelitian ini.

Tabel 3.3 Struktur Data Analisis Sentimen

Pasangan Calon	Cuitan	Label
1	$Cuitan_{1,1}$	$y_{1,1}$
	\vdots	\vdots
2	$Cuitan_{2,n_1}$	y_{1,n_1}
	\vdots	\vdots
3	$Cuitan_{2,n_2}$	y_{2,n_2}
	\vdots	\vdots
	$Cuitan_{3,1}$	$y_{3,1}$
	\vdots	\vdots
	$Cuitan_{3,n_3}$	y_{3,n_3}

Pada Tabel 3.3, cuitan digolongkan berdasarkan pasangan calonnya. Hal ini digunakan untuk tabulasi pada hasil sentimen klasifikasi. Namun pada proses training, cuitan akan digunakan sekaligus dan tidak dibedakan berdasarkan pasangan calon. Cuitan-cuitan tersebut kemudian dilakukan tokensisasi dan diubah menjadi *token embedding* sesuai Persamaan (2.24) dengan tambahan *padding*. *Padding* merupakan cara agar panjang *input* sekuensial memiliki panjang yang sama dalam satu tumpak (Prottasha, et al., 2022). Sehingga, struktur data *input embedding* dapat direpresentasikan pada Tabel 3.4.

Tabel 3.4 Struktur Data *Input Embedding*

Cuitan ke-	Vektor kata ke-1	Vektor kata ke-2	Vektor kata ke-3	...	Vektor kata ke- $n_{max length}$	Label
1	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$...	$x_{1,n_{max length}}$	y_1
2	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$...	$x_{2,n_{max length}}$	y_2
\vdots	\vdots	\vdots	\vdots	\ddots	\vdots	\vdots
n_{cuitan}	$x_{n_{cuitan},1}$	$x_{n_{cuitan},2}$	$x_{n_{cuitan},3}$...	$x_{n_{cuitan},n_{max length}}$	$y_{n_{cuitan}}$

Setelah didapatkan hasil klasifikasi, selanjutnya jumlah setiap sentimen ditabulasikan berdasarkan pasangan calon dan persentase untuk setiap sentimennya dan dibandingkan dengan data hasil rekapitulasi KPU.

Tabel 3.5 Struktur Data Analisis Korelasi Rank Kendall

Pasangan Calon	Persentase Positif	Persentase Negatif	Hasil KPU
1	$p_{1,positif}$	$p_{1,negatif}$	$p_{1,KPU}$
2	$p_{2,positif}$	$p_{2,negatif}$	$p_{2,KPU}$
3	$p_{3,positif}$	$p_{3,negatif}$	$p_{3,KPU}$

Pada Tabel 3.5, skala yang digunakan pada analisis korelasi adalah berupa proporsi (persentase) menyesuaikan dengan hasil yang didapat pada data KPU.

3.4 Tahapan Penelitian

Tahap penelitian untuk analisis sentimen menggunakan BERT model seperti IndoBERTweet berbeda dengan penggunaan model *machine learning* konvensional. Berikut merupakan tahapan yang digunakan dalam pemrosesan bahasa.

a. *Crawling* Data X dengan

Dengan menggunakan API dari X, data dari X dapat diambil dengan cara *crawling*. Informasi yang dapat diambil antara lain cuitan, *username*, dan informasi lainnya yang tersedia untuk publik. *Crawling* dilakukan dengan mencari *keyword* berdasarkan nama pasangan calon. *Crawling* dilakukan di Python dengan menggunakan modul *tweet-harvest*. Tidak terdapat batasan mengenai jumlah data X yang dapat diambil, namun hanya batasan waktu yakni 500-1000 cuitan per 15 menit.

b. *Filtering* dan Pemrosesan Data Teks

Filtering dilakukan untuk menghilangkan data-data cuitan yang tidak sesuai dengan kriteria seperti cuitan iklan, cuitan kosong, cuitan non Indonesia, dan duplikat. Sedangkan, pemrosesan data teks digunakan untuk menghilangkan karakter yang tidak diperlukan seperti tanda baca, karakter bukan kata, emoji, dan melakukan normalisasi dengan mengganti kata yang tidak baku dan singkatan menjadi kata baku.

c. Analisis Deskriptif dan Visualisasi

Pada tahap ini, dilakukan analisis deskriptif mengenai karakteristik data, seperti jumlah data per pasangan calon, jumlah data tidak valid, dsb. Sedangkan, visualisasi berisi keseluruhan sentimen maupun per pasangan calon menggunakan *word cloud* dan *bar chart*. Visualisasi kata dilakukan dengan penghapusan *stopwords*.

d. *Case Folding*

Case Folding merupakan tahapan untuk mengubah seluruh kata menjadi kata kecil semua atau lowercase.

e. Pelabelan Data

Data yang sudah dilakukan pemrosesan kemudian akan dilakukan pelabelan menggunakan Vader. Namun, karena Vader merupakan alat untuk melakukan analisis sentimen berdasarkan *lexicon* pada Bahasa Inggris, maka teks akan diterjemahkan ke dalam Bahasa Inggris terlebih dahulu untuk memastikan kekayaan korpus. Pelabelan ini dilakukan untuk semua data baik *training* maupun *test*. Data yang sudah dilabel oleh Vader kemudian dievaluasi secara manual untuk memastikan akurasi pelabelan.

f. *Tokenizing* dan *Padding*

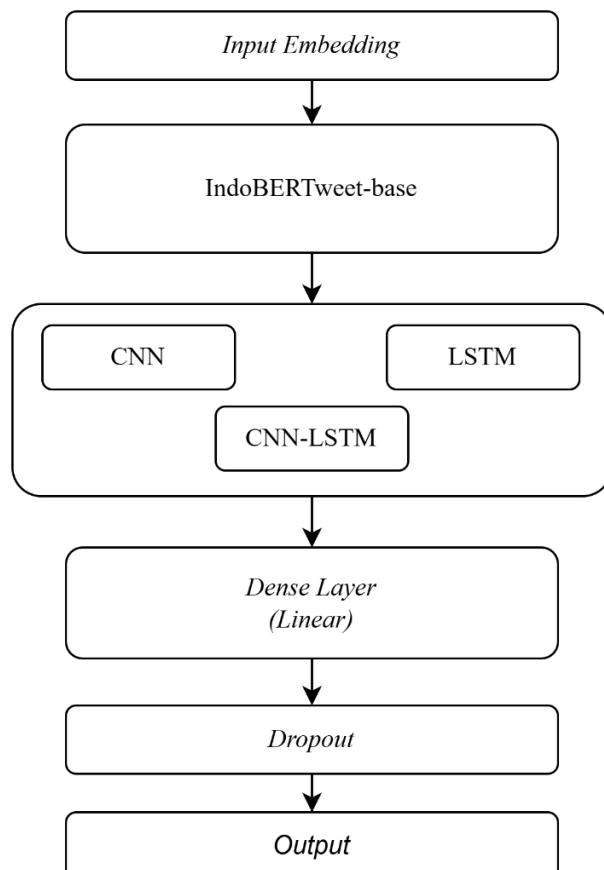
Tokenizing adalah proses untuk mengubah data teks menjadi potongan teks (kata) yang kemudian pada model IndoBERTweet dimasukkan ke dalam tokenizernya (WordPiece) untuk diubah menjadi input embedding dan dilakukan *padding* seperti pada Tabel 3.4.

g. *Split Dataset*

Split dataset digunakan untuk membagi data secara rata menjadi data *train* dan data *test* menggunakan pembagian data 80% untuk *train* dan 20% untuk *test*. Pembagian data ini tidak melihat golongan pasangan calon.

h. Melatih model *transfer learning* IndoBERTweet

Input embedding akan dimasukkan ke dalam model IndoBERTweet. Proses akan dilanjutkan ke *custom layer* seperti CNN, LSTM, *dense*, dan/atau *dropout* seperti pada Gambar 3.1.



Gambar 3.1 Arsitektur Utama Model
(Sumber: Dokumen Pribadi)

Adapun beberapa kombinasi pada *layer* CNN, LSTM, maupun *hybrid* CNN-LSTM dijelaskan pada Tabel 3.6.

Tabel 3.6 Skenario Pemodelan

Skenario	Detail
IndoBERTweet-CNN layers	<ul style="list-style-type: none">• IndoBERTweet• CNN (filters=512, kernel_size=3)• MaxPooling1D (pool size=4)• CNN (filters=256, kernel_size=5)• MaxPooling1D (pool size=4)• Dense (64 units)

Tabel 3.6 Skenario Pemodelan (Lanjutan)

Skenario	Detail
	<ul style="list-style-type: none">• Dropout (0.3)• Dense (32 units)• Dropout (0.3)• <i>Output</i>
IndoBERTweet-Bi LSTM layers	<ul style="list-style-type: none">• IndoBERTweet• <i>input</i> Bi-LSTM (32 units)• Bi-LSTM (16 units)• Dense (64 units)• Dropout (0.3)• Dense layer (32 units)• Dropout (0.3)• <i>Output</i>
IndoBERTweet-Hybrid CNN-Bi LSTM layers	<ul style="list-style-type: none">• IndoBERTweet• CNN (filters=512, kernel_size=3)• MaxPooling1D (pool size=4)• <i>input</i> Bi-LSTM (32 units)• Bi-LSTM (16 units)• Dense (64 units)• Dropout (0.3)• Dense Layer (32 units)• Dropout(0.3)• <i>output</i>

Kesamaan pada Tabel 3.6 adalah menggunakan model LLM yang sama dan dengan output yang sama, yakni *dense layer* dengan fungsi aktivasi *softmax* sebanyak tiga neuron. Skenario ini hanya berfungsi sebagai panduan umum. Peneliti dapat mengubah beberapa detail mengenai skenario, misalnya mengurangi/menambah jumlah *layer* atau mengubah parameter setiap *layer* untuk mendapatkan hasil optimal.

i. Evaluasi Model

Evaluasi model dilihat menggunakan *confusion matrix* dan juga melihat apakah terdapat *overfitting/underfitting* antara data *train* dan *test*.

j. Penanganan *Overfitting* dan Ketidakseimbangan kelas

Jika terlihat indikasi adanya *overfitting* atau ketidakseimbangan kelas yang memengaruhi model, dilakukan penanganan menggunakan augmentasi data *Back Translate* (BT) dan *class weight* untuk mengatasi ketidakseimbangan kelas, serta penggunaan regularisasi L2 pada *dense layer* untuk mengatasi *overfitting*.

k. Analisis Deskriptif dan Visualisasi Sentimen

Pada tahap ini, dilakukan analisis deksriptif hasil tabulasi sentimen baik pada data *test* dan *train* visualisasi sentimen baik keseluruhan maupun per pasangan calon menggunakan *word cloud* dan *bar chart* seperti pada Poin c.

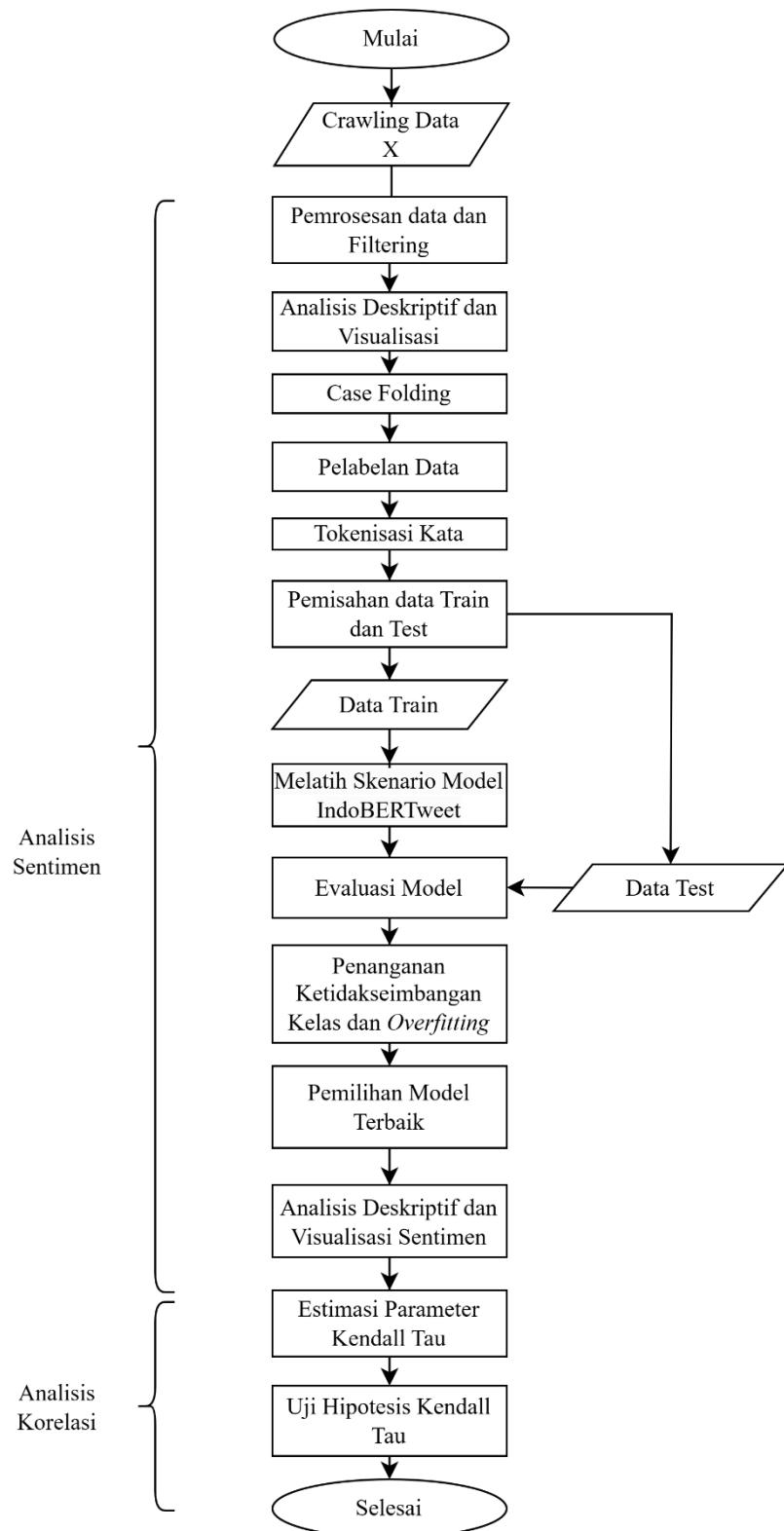
1. Analisis Hubungan Sentimen dan Hasil Survei

Hasil prediksi sentimen untuk rentang periode akan dibandingkan dengan hasil survei secara deskriptif untuk melihat apakah terdapat hubungan antar keduanya yang berubah pada aspek waktu.

m. Analisis Korelasi *Rank Sentimen* Terhadap *Real Count*

Hasil prediksi sentimen, baik pada data test maupun data train ditabulasikan berdasarkan jumlah sentimen negatif dan positifnya per pasangan calon, kemudian dikomparasikan dengan hasil real count KPU tentang pemilihan Gubernur Jawa Timur 2024 secara deskriptif dan Kendall's Tau untuk mengetahui adanya korelasi antara sentimen media sosial khususnya X dengan hasil rekapitulasi KPU.

3.5 Diagram Alir Penelitian



Gambar 3.2 Diagram Alir Penelitian
(Sumber: Dokumen Pribadi)

(Halaman ini sengaja dikosongkan)

BAB IV

ANALISIS DAN PEMBAHASAN

4.1 Karakteristik Data Umum

Data pada penelitian ini didapatkan dari aplikasi media sosial X (Twitter) menggunakan teknik *crawling* pada rentang waktu kampanye Pilkada Jatim dengan menggunakan *keyword* nama pasangan calon. Didapatkan sebanyak 4332 data, dengan rincian yang dijabarkan pada Tabel 4.1.

Tabel 4.1 Data Hasil *Crawling* per Paslon

Pasangan Calon	Jumlah Data Hasil <i>Crawling</i>
1	281
2	3460
3	591
Jumlah	4332

Dari Tabel 4.1, terdapat ketimpangan jumlah cuitan yang membahas antara paslon 1, 2, dan 3. Dari data ini, kemudian dilakukan *preprocessing* berupa identifikasi cuitan iklan, identifikasi data duplikat, dan identifikasi cuitan kosong. Cuitan iklan dapat diidentifikasi dengan melihat tanggal unggah cuitan, dimana cuitan iklan tidak memiliki metadata seperti tanggal unggah cuitan.

Tabel 4.2 Hasil *Preprocessing* Cuitan Iklan dan Duplikat

Pasangan Calon	Jumlah Cuitan Iklan	Jumlah Missing Value	Jumlah Duplikat
1	0	0	63
2	0	0	734
3	0	1	19

Hasil pada Tabel 4.2, menunjukkan bahwa untuk setiap cuitan pasangan calon tidak terdapat cuitan iklan, namun terdapat satu *missing value* pada data cuitan dan terdapat duplikat yang kemudian dihapus.

Tabel 4.3 Jumlah Data Setelah *Preprocessing*

Pasangan Calon	Jumlah Data Setelah <i>Preprocessing</i>
1	218
2	2726
3	571
Jumlah	3515

Sehingga, hasil *preprocessing* menghasilkan 3515 data yang memiliki jumlah cuitan detail per pasangan calon pada Tabel 4.3.

4.2 Karakteristik Cuitan Umum

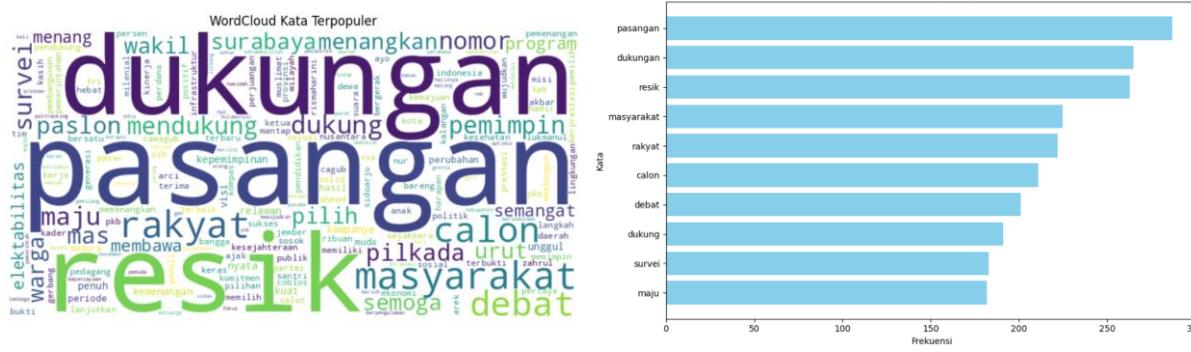
Untuk mengetahui bagaimana kondisi cuitan X pada masa Pilkada Jatim, terlebih dahulu dilakukan pemrosesan teks. Sesuai dengan langkah penelitian, pemrosesan teks dimulai dengan *case folding*, dilanjutkan dengan menghapus *username*, *hashtag*, teks terenkripsi, emoji, kata berulang, dan tanda baca. Setelahnya, dilakukan normalisasi, hal ini untuk mengganti kata gaul/tak baku menjadi kata baku, kemudian dilanjutkan dengan menghapus kata dalam *stopwords list*. Adapun untuk *stopword list* digunakan *stopword* bahasa Indonesia dari modul Python Sastrawi dan NLTK, *stopword* bahasa Jawa, dan *stopword* manual berisi nama pasangan calon dan kata yang menurut peneliti kurang relevan.

Tabel 4.4 Ilustrasi Tahapan Pemrosesan Teks Untuk *Word Cloud*

Tahapan	Hasil Pemrosesan
Cuitan Asli	Jawa Timur Butuh Pemimpin Visioner! Gus Imin @cakimiNOW meminta kader PKB tak berpangku tangan! Pilih Luluk-Lukman untuk Jawa Timur yang adil dan makmur. #LulukLukman https://t.co/Teh1stply2
Case Folding	jawa timur butuh pemimpin visioner! gus imin @cakiminow meminta kader pkb tak berpangku tangan! pilih luluk-lukman untuk jawa timur yang adil dan makmur. #luluklukman https://t.co/teh1stply2
Penghapusan <i>username</i> , <i>hashtag</i> , teks terenkripsi, emoji, kata berulang, dan tanda baca	jawa timur butuh pemimpin visioner gus imin meminta kader pkb tak berpangku tangan pilih luluk lukman untuk jawa timur yang adil dan makmur
Normalisasi	jawa timur butuh pemimpin visioner gus imin meminta kader pkb tak berpangku tangan pilih luluk lukman untuk jawa timur yang adil dan makmur
Penghapusan <i>Stopwords</i>	pemimpin visioner imin kader pkb berpangku tangan pilih adil makmur

Proses pembersihan teks untuk pembuatan *word cloud* dilakukan seperti ilustrasi pada Tabel 4.4, dimana hasil akhirnya merepresentasikan cuitan tanpa nama Paslon, *username*, *hashtag*, non-teks.

Setelah didapatkan teks bersih, dilanjutkan dengan identifikasi kata dengan distribusi terbanyak secara umum dan pembuatan *word cloud*.

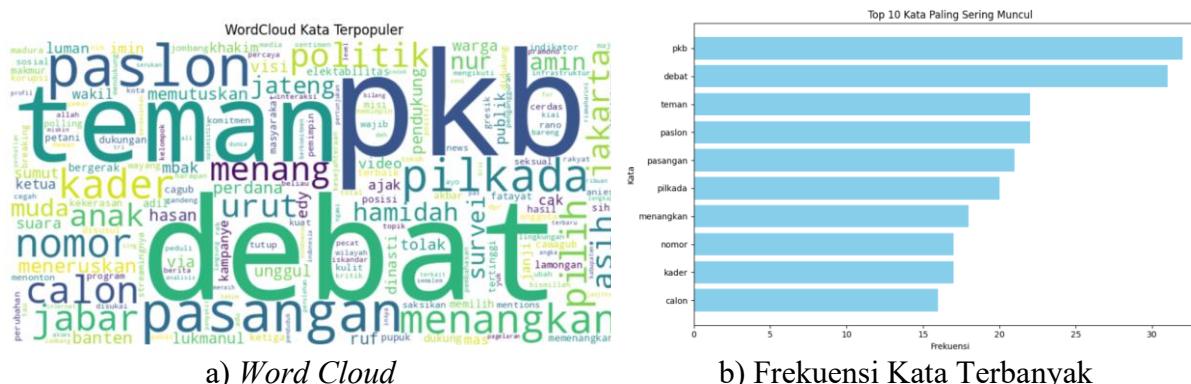


Gambar 4.1 a) Word Cloud Umum, b) Frekuensi Kata Terbanyak Umum

Dari Gambar 4.1a, diketahui bahwa mayoritas cuitan membicarakan tentang dukungan terhadap pasangan calon. Selain itu, terlihat juga kata resik yang kemungkinan merujuk pada kampanye milik pasangan calon ketiga. Jika dilihat lebih detail pada Gambar 4.1b mayoritas cuitan memiliki kata yang berkonotasi netral seperti debat, survei, calon, dsb hingga positif yang memuat kata dukungan, maju, mendukung, dsb, sangat minim terlihat adanya kata berkonotasi negatif.

4.3 Karakteristik Cuitan Per Pasangan Calon

Untuk mengetahui lebih lanjut bagaimana secara kilas kondisi dari cuitan-cuitan per pasangan calon, digunakan langkah pemrosesan yang sama seperti pada Tabel 4.4 kemudian dilanjutkan dengan pembuatan *word cloud* dan identifikasi distribusi kata terbanyak.

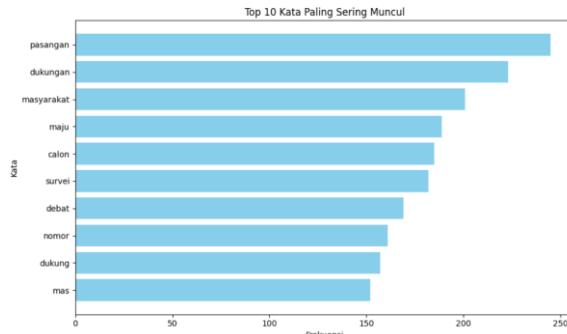


Gambar 4.2 a) Word Cloud Cuitan Paslon 1, b) Frekuensi Kata Terbanyak Cuitan Paslon 1

Dari Gambar 4.2a) diketahui bahwa mayoritas cuitan memuat kata-kata netral hingga positif seperti teman, pkb, debat, pasangan, paslon, menang, dsb. Hal ini menunjukkan minimnya cuitan yang menyerang Paslon tersebut. Pada Gambar 4.2b), frekuensi kata terbanyak diraih oleh kata pkb yang kemungkinan merujuk pada PKB (Partai Kebangkitan Bangsa) yang merupakan partai pengusung pasangan calon dengan nomor urut pertama ini. Selain itu, terdapat kata yang memiliki nuansa politik lainnya seperti kader dan politik.



a) Word Cloud



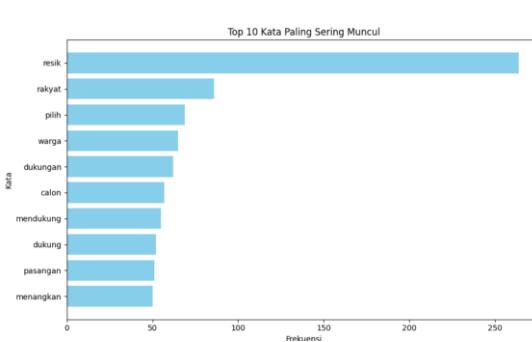
b) Frekuensi Kata Terbanyak

Gambar 4.3 a) *Word Cloud* Cuitan Paslon 2, b) Frekuensi Kata Terbanyak Cuitan Paslon 2

Untuk pasangan calon 2, dari Gambar 4.3a) diketahui bahwa mayoritas cuitan memuat kata-kata netral hingga positif seperti pasangan, dukungan, masyarakat, survei, maju, dsb. Hal ini menunjukkan minimnya cuitan yang menyerang Paslon tersebut. Jika dilihat lebih lanjut pada Gambar 4.3b), tidak terdapat kata yang bernuansa negatif. Selain itu, mayoritas cuitan-cuitan ini juga membawa kata survei dan pemimpin yang kemungkinan menunjukkan keunggulan Paslon 2 sebagai petahana dan Paslon dengan survei elektabilitas yang tinggi.



a) Word Cloud



b) Frekuensi Kata Terbanyak

Gambar 4.4 a) *Word Cloud* Cuitan Paslon 3, b) Frekuensi Kata Terbanyak Cuitan Paslon 3

Untuk pasangan calon 3, dari Gambar 4.4a) diketahui bahwa mayoritas cuitan memuat kata-kata netral hingga positif seperti resik, dukungan, rakyat, pilih, warga, dsb. Hal ini menunjukkan minimnya cuitan yang menyerang Paslon tersebut. Jika dilihat lebih lanjut pada Gambar 4.4b), tidak terdapat kata yang bernuansa negatif. Selain itu, kata dengan frekuensi paling tinggi ada pada kata resik yang kemungkinan merukup pada kampanye Resik milik Paslon 3 dan surabaya yang merujuk pada pengalaman Calon Gubernur nomor urut tiga ini saat menjabat sebagai kepala daerah Kota Surabaya.

4.4 Pemrosesan Teks Untuk Pemodelan

Sebelum dilakukan pemodelan untuk mengetahui sentimen cuitan X dalam studi kasus Pilkada Jawa Timur menggunakan model transfer learning IndoBERTweet, terlebih dahulu dilakukan beberapa pemrosesan. Pemrosesan pertama, menggabungkan data Paslon pertama, Paslon kedua, dan Paslon ketiga pada Tabel 4.3. Setelah dilakukan penggabungan, diidentifikasi cuitan yang duplikat terhadap id cuitan (45 jumlah duplikat) dan kemudian menghapusnya. Selanjutnya, dilakukan proses pembersihan teks yang hampir sama dengan

yang telah dilakukan sebelumnya, yakni dimulai dengan *case folding*, dilanjutkan dengan menghapus *username*, *hashtag*, teks terenkripsi, emoji, kata berulang, dan tanda baca. Setelahnya, dilakukan normalisasi. Perbedaannya, dalam pemrosesan untuk pemodelan ini tidak menggunakan *stopwords* merujuk pada penelitian-penelitian terdahulu.

Tabel 4.5 Ilustrasi Tahapan Pemrosesan Teks Untuk Pemodelan

Tahapan	Hasil Pemrosesan
Cuitan Asli	Jawa Timur Butuh Pemimpin Visioner! Gus Imin @cakimiNOW meminta kader PKB tak berpangku tangan! Pilih Luluk-Lukman untuk Jawa Timur yang adil dan makmur. #LulukLukman https://t.co/Teh1stply2
<i>Case Folding</i>	jawa timur butuh pemimpin visioner! gus imin @cakiminow meminta kader pkb tak berpangku tangan! pilih luluk-lukman untuk jawa timur yang adil dan makmur. #luluklukman https://t.co/teh1stply2
Penghapusan <i>username</i> , tagar, enkripsi, emoji, kata berulang, dan tanda baca	jawa timur butuh pemimpin visioner gus imin meminta kader pkb tak berpangku tangan pilih luluk lukman untuk jawa timur yang adil dan makmur
Normalisasi	jawa timur butuh pemimpin visioner gus imin meminta kader pkb tak berpangku tangan pilih luluk lukman untuk jawa timur yang adil dan makmur

Setelah didapatkan hasil cuitan bersih Tabel 4.5, didapatkan data bersih sebanyak 3468 teks yang siap untuk pemrosesan selanjutnya. Data bersih tersebut merupakan kalimat yang akan dimasukkan ke dalam *tokenizer* milik IndoBERTweet.

Sebelum dilakukan *tokenizing*, terlebih dahulu dilakukan pelabelan menggunakan Vader Sentiment. Vader bekerja dengan mengandalkan kamus leksikal dan aturan untuk mendeteksi polaritas serta intensitas sentimen dari kata-kata dalam teks. Namun, untuk saat ini keandalan Vader dalam menyimpulkan sentimen pada teks berbahasa Indonesia masih rendah, karena kamus leksikon untuk Vader dalam bahasa Indonesia masih terbatas. Sehingga, salah satu solusi dalam menggunakan Vader untuk menyimpulkan sentimen pada teks berbahasa Indonesia adalah dengan menerjemahkan teks bersih ke dalam bahasa Inggris (bahasa *native* Vader) dengan harapan akan memberikan konteks dan leksikon yang lebih kaya, sehingga kualitas sentimennya dapat diandalkan. Dalam menerjemahkan, peneliti menggunakan Google Translate yang kemudian dimasukkan ke dalam pemrosesan Vader sehingga dikeluarkan sebuah *output* sentimen.

Tabel 4.6 Ilustrasi Tahapan Pemrosesan Teks Untuk Pelabelan Vader

Tahapan	Hasil Pemrosesan
Teks Bersih (Hasil Pemrosesan)	jawa timur butuh pemimpin visioner gus imin meminta kader pkb tak berpangku tangan pilih luluk lukman untuk jawa timur yang adil dan makmur
Penerjemahan ke dalam bahasa Inggris	<i>East Java needs a visionary leader Gus Imin asks PKB cadres not to stand idly by choosing Luluk Lukman for East Java and prosperous</i>

Tabel 4.6 Ilustrasi Tahapan Pemrosesan Teks Untuk Pelabelan Vader (Lanjutan)

Tahapan	Hasil Pemrosesan
Case Folding	<i>east java needs a visionary leader gus imin asks pkb cadres not to stand idly by choosing luluk lukman for east java and prosperous</i>
Compound Score Vader (Polaritas)	0,75
Label Sentimen	Positif

Seperti diilustrasikan pada Tabel 4.6, hasil pemrosesan teks pada Tabel 4.5 dimasukkan ke dalam mesin penerjemah Google Translate dan diterjemahkan ke dalam bahasa Inggris, kemudian dilakukan *case folding* dan dilakukan pelabelan dengan Vader. Hasil dari Vader akan mengeluarkan *compound score* yang merepresentasikan sentiment, dimana semakin rendah sentiment akan semakin negatif dengan nol menjadi nilai netral. Peneliti menggunakan rentang $(-0,5; 0,5)$ untuk kategori sentimen netral, $\geq 0,5$ untuk kategori positif, dan $\leq -0,5$ untuk kategori negatif. Hasil pelabelan ini kemudian diambil sampel untuk dilakukan perbaikan label oleh peneliti dan mengetahui berapa persen ketidaksesuaian label yang diperoleh dari polaritas Vader dengan hasil pelabelan oleh peneliti. Untuk mendapatkan sampel yang sesuai, digunakan tabel ukuran sampel n yang didasarkan pada distribusi multinomial (Thompson, 1987). Berdasarkan tabel pada Lampiran 2 untuk tiga kategori, didapatkan ukuran sampel minimal 510, dengan $\alpha = 5\%$ dan *bound of error* 5%. Dari sampel tersebut, setelah diidentifikasi terdapat 23% label dari ukuran sampel minimal yang diperbaiki oleh peneliti.

Setelah didapatkan label yang sesuai, langkah selanjutnya adalah melakukan *tokenizing*. Teks akan diubah menjadi kode/id yang merepresentasikan *embedding* oleh mekanisme tokenasi *WordPiece*, kemudian kode tersebut akan dilakukan *lookup* untuk menghasilkan *embedding*nya.

Tabel 4.7 Ilustrasi Tahapan Tokenasi BERT

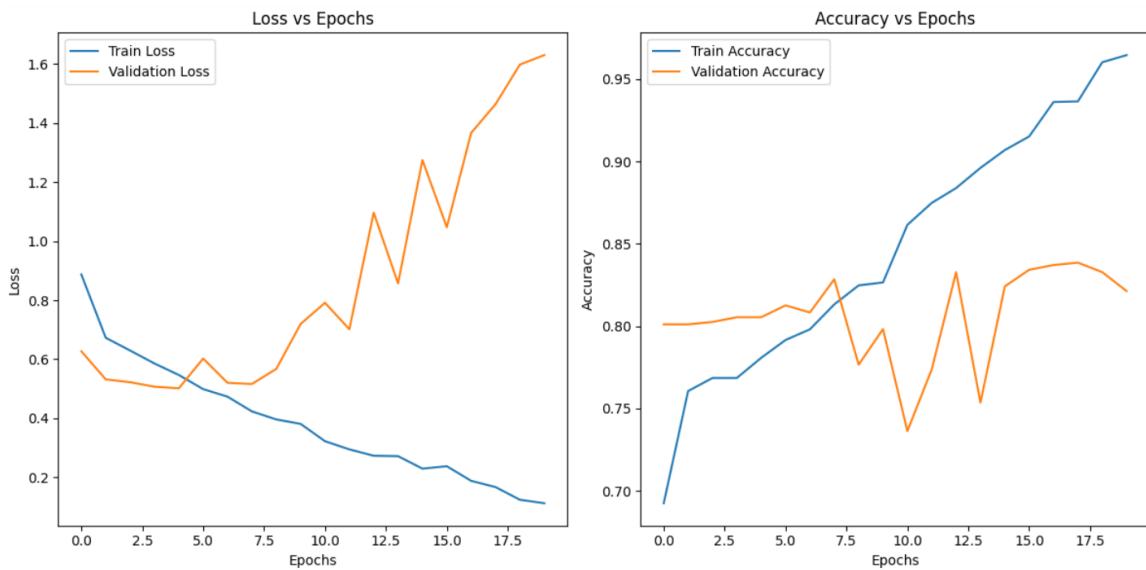
Tahapan	Hasil Pemrosesan
Teks Bersih (Hasil Pemrosesan)	jawa timur butuh pemimpin visioner gus imin meminta kader pkb tak berpangku tangan pilih luluk lukman untuk jawa timur yang adil dan makmur
Hasil Tokenasi	$[3, 11922, 12309, 936, 20359, 944, \dots, 0, 0, 0, 0]$
Word Embedding	$\begin{bmatrix} 0,39 & -0,14 & -1,2 & \cdots & 1,77 \\ 0,23 & -0,46 & -0,66 & \cdots & 1,54 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0,34 & -0,57 & -0,27 & \cdots & 2,17 \end{bmatrix}_{100 \times 768}$

Hasil dari Tabel 4.7 kemudian dimasukkan ke dalam model *transfer learning* IndoBERTweet.

4.5 Pemodelan

4.5.1 Skenario Pemodelan IndoBERTweet-CNN

Sesuai dengan skenario pemodelan pada Tabel 3.6, dilakukan pemodelan transfer learning IndoBERTweet-CNN. Hal ini dilakukan dengan cara membuka parameter IndoBERTweet dan hanya memperbarui parameter dari *custom layer*, yakni CNN. Dalam pemodelan ini, digunakan *optimizer* Adam dan dilakukan *training* sebanyak 20 *epochs*.



a) b)
Gambar 4.5 Grafik Skenario CNN per Epochs: a) Loss, b) Akurasi

Berdasarkan Gambar 4.5a), grafik *loss* untuk data *training* semakin menurun yang menunjukkan bahwa model bisa mempelajari dan mampu memahami teks pada data *train* setiap *epochnya* secara konsisten. Namun, hal yang kontras terlihat pada *loss* data *test* (pada Gambar 4.5a) disimbolkan dengan *validation loss*) terlihat bertambah dan tidak sejalan dengan tren *loss* data *train* setelah 7 *epochs*. Kenaikan ini menunjukkan bahwa model mulai kehilangan kemampuannya untuk melakukan generalisasi terhadap data di luar data *train* (*overfitting*). Selain itu, pada Gambar 4.5b), tren akurasi memperkuat indikasi tersebut. Akurasi data *train* mengalami peningkatan signifikan dan konsisten naik hingga mencapai nilai di atas 0,95 di akhir *training*. Sebaliknya, akurasi validasi hanya mengalami sedikit peningkatan di awal dan kemudian tidak sejalan dengan tren akurasi data *train* setelah 7 *epochs*. Gap ini menjelaskan bahwa model belum belajar terlalu baik pada data *train*, tetapi gagal mempertahankan performa pada data *test*. Secara keseluruhan, pola yang ditunjukkan oleh kedua grafik ini merupakan permasalahan dari *overfitting*.

Tabel 4.8 Akurasi Data *Test* Skenario Pemodelan CNN

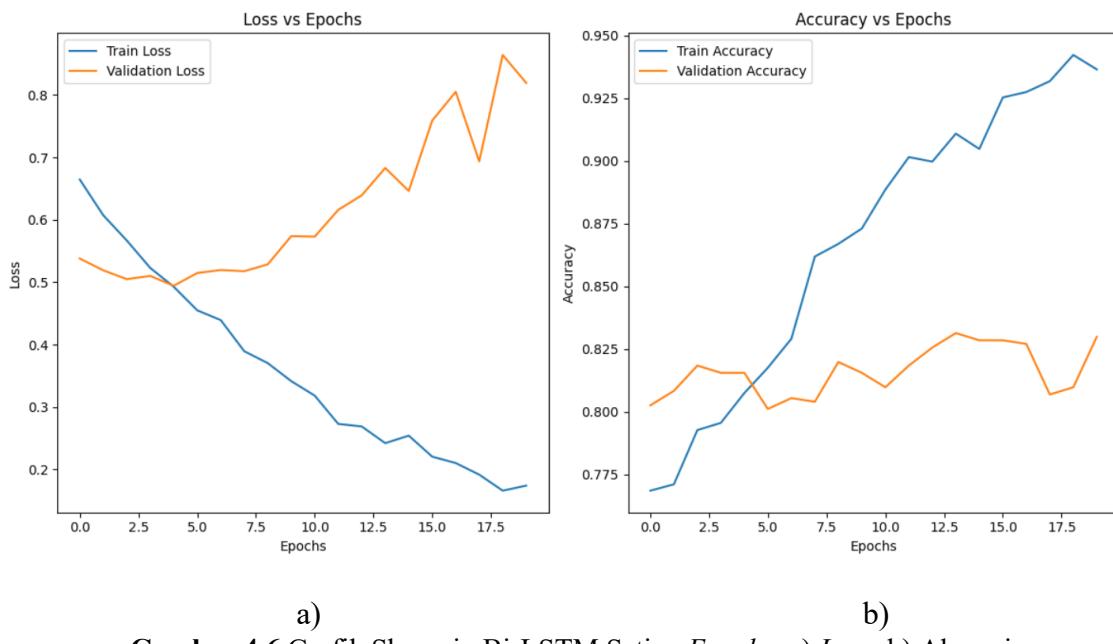
Kelas	Presisi (%)	Recall (%)	Skor F1 (%)
Negatif	45	17	24
Netral	54	46	50
Positif	87	93	90
Akurasi (%)		82	
Weighted F1 Score (%)		81	

Berdasarkan Tabel 4.8, akurasi keseluruhan model pada data *test* mencapai 82% dengan nilai *weighted F1 Score* sebesar 81%. Secara umum, angka ini menunjukkan bahwa model mampu memprediksi sebagian besar cuitan dengan benar. Namun, jika ditelusuri berdasarkan masing-masing kelas, terlihat adanya ketimpangan performa klasifikasi. Model menunjukkan performa sangat baik pada kelas positif dengan *precision* sebesar 87%, *recall* mencapai 93%, dan skor F1 yang tinggi sebesar 90%. Hal ini mengindikasikan bahwa model sangat efektif dalam

mendeteksi dan dengan benar mengidentifikasi sentimen positif. Sebaliknya, performa kelas negatif dan netral jauh lebih rendah. Kelas negatif hanya memperoleh skor F1 sebesar 24% dan kelas netral sebesar 50%. Rendahnya skor F1 pada kedua kelas ini menunjukkan bahwa model kesulitan dalam mengenali dan mengklasifikasikan sentimen nonpositif. Performa timpang ini dapat menjadi indikasi bahwa model bias terhadap kelas dominan. Hal ini juga selaras dengan indikasi *overfitting*.

4.5.2 Skenario Pemodelan IndoBERTweet Bi-LSTM

Dalam skenario pemodelan ini, digunakan arsitektur yang sama dengan skenario pemodelan pada Tabel 3.6 dan dengan konfigurasi yang sama seperti pemodelan sebelumnya, yakni menggunakan *optimizer* Adam dan dilatih sebanyak 20 *epochs*.



Gambar 4.6 Grafik Skenario Bi-LSTM Setiap *Epochs*: a) *Loss*, b) Akurasi

Berdasarkan Gambar 4.6a), grafik *loss* pada data *train* menurun secara konsisten hingga di bawah 20%, yang menunjukkan bahwa model mampu menyesuaikan dengan pola dalam data *train*. Namun, tren yang berbeda terjadi pada data *test*, di mana nilai *loss* data *test* cenderung meningkat setelah 5 *epochs*. Fenomena ini mengindikasikan bahwa model mulai kehilangan kemampuan untuk memprediksi data yang tidak pernah dilihat sebelumnya secara akurat, mengindikasikan adanya *overfitting*. Gambar 4.6b) mendukung kesimpulan tersebut. Grafik menunjukkan adanya perbedaan yang melebar antara akurasi *train* dan *test* setelah 5 *epochs*. Sementara akurasi data *train* terus mengalami kenaikan signifikan dan mencapai nilai hingga 92%, akurasi pada data *test* stagnan di rentang 80% hingga 83%. Pola ini mengindikasikan adanya *overfitting*.

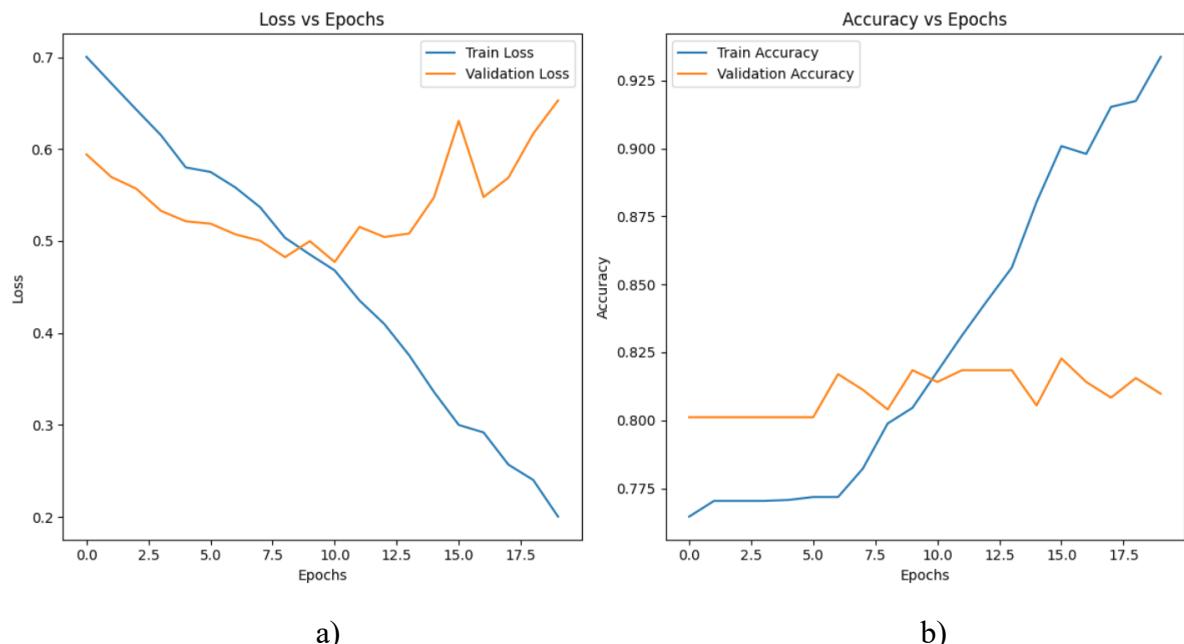
Tabel 4.9 Akurasi Data *Test* Skenario Pemodelan Bi-LSTM

Kelas	Presisi (%)	Recall (%)	Skor F1 (%)
Negatif	38	10	16
Netral	57	54	55
Positif	88	93	90
Akurasi (%)			83
Weighted F1 Score (%)			82

Berdasarkan Tabel 4.9, hasil akurasi data *test* secara keseluruhan adalah 83% dan untuk *weighted F1 score* sebesar 82%. Artinya, model secara keseluruhan sudah memprediksi mayoritas cuitan ke dalam sentimen yang bersesuaian. Namun, sama seperti skenario sebelumnya, jika dipecah dan dilihat akurasi per kelasnya, kelas positif memiliki skor F1 yang paling besar menunjukkan bahwa model dapat mengklasifikasikan cuitan positif secara akurat. Sebaliknya, kelas negatif dan netral memiliki skor F1 yang masih buruk. Pada kelas negatif, nilai *recall* 10% dan presisi 38% menunjukkan bahwa mayoritas data negatif tidak dikenali oleh model dan tidak dapat diprediksikan dengan tepat. Kelas netral, tampak memiliki nilai *recall* dan presisi yang mirip dengan skor F1 yang masih rendah. Melihat dari besarnya skor F1 kelas netral dan rendahnya skor F1 pada kelas negatif dan netral, model tampak bias terhadap kelas positif, yang kemungkinan besar merupakan kelas mayoritas atau paling mudah dikenali selama proses *training*.

4.5.3 Skenario Pemodelan IndoBERTweet CNN Bi-LSTM

Dalam skenario pemodelan ini, digunakan arsitektur yang sama dengan skenario pemodelan pada Tabel 3.6 dan dengan konfigurasi yang sama seperti pemodelan sebelumnya, yakni menggunakan *optimizer* Adam dan dilatih sebanyak 20 *epochs*.



Gambar 4.7 Grafik Skenario CNN Bi-LSTM Setiap *Epochs*: a) *Loss*, b) Akurasi

Berdasarkan Gambar 4.7a), grafik *loss* untuk data *train* menurun secara stabil dan mencapai nilai di bawah 30% yang menandakan bahwa model mampu mempelajari struktur dalam data *train* secara efektif. Namun, pola berbeda terlihat pada *loss* data *test* yang semula menurun dan relatif sejajar dengan *train loss* hingga sekitar epoch ke-10, tetapi kemudian stagnan dan mulai berfluktuasi di kisaran 60–70%. Hal ini merupakan indikasi dari *overfitting*. Hal ini diperkuat pada Gambar 4.7b), terlihat bahwa akurasi antara akurasi data *train* dan *test* masih cukup berdekatan. Namun, setelah melewati epoch ke-10, akurasi pada data *train* terus meningkat secara signifikan hingga melewati 90%, sementara akurasi *test* stagnan di rentang 80%–81%. Ditambah dengan gap yang semakin lebar antara keduanya menjadi indikasi kuat terjadinya *overfitting*, yakni ketika model terlalu menghafal data *train* tanpa benar memahami pola yang dapat digeneralisasi ke data baru.

Tabel 4.10 Akurasi Data Test Skenario Pemodelan CNN Bi-LSTM

Kelas	Presisi (%)	Recall (%)	Skor F1 (%)
Negatif	50	37	42
Netral	49	53	51
Positif	89	89	89
Akurasi (%)			81
Weighted F1 Score			81

Pada Tabel 4.10, model gabungan CNN-BiLSTM menghasilkan akurasi sebesar 81% dan *weighted F1 score* sebesar 81%, menjadikannya kombinasi paling buruk jika dibandingkan dua pemodelan sebelumnya. Jika ditinjau lebih dalam setiap kelas, terlihat bahwa dominasi performa tetap berada pada kelas positif dengan skor F1 sedang–tinggi sebesar 89%. Hal ini menandakan bahwa model sangat mampu mengenali dan mengklasifikasikan sentimen positif dengan akurasi yang tinggi dan konsisten. Namun, tantangan utama masih terdapat pada dua kelas lainnya. Kelas negatif hanya memperoleh skor F1 sebesar 42%, dengan *recall* yang rendah yaitu 37%, sedangkan kelas netral mencatatkan skor F1 sebesar 51%. Nilai-nilai ini menunjukkan bahwa model masih mengalami kesulitan dalam menangkap pola-pola linguistik yang membedakan sentimen negatif dan netral. Rendahnya performa pada kelas negatif dan netral sangat mungkin berkaitan dengan ketidakseimbangan distribusi data. Untuk mengatasi hal tersebut, diperlukan strategi penanganan data seperti *class weight* atau augmentasi data untuk memperkuat representasi kelas minoritas.

4.6 Mengatasi Ketidakseimbangan Data dan *Overfitting*

Dari berbagai pencobaan sebelumnya, diagnosis awal dalam rendahnya skor F1 tiap kelas dan adanya *overfitting* adalah karena ketidakseimbangan data, dimana rasio dari data *train* dan *test* antara negatif, netral, dan positif adalah sekitar 16:4:1. Sehingga, diperlukan cara untuk mengatasi hal ini, salah satunya adalah menggunakan teknik augmentasi pada kelas minoritas. Peneliti memilih untuk menggunakan teknik augmentasi berbasis data (*data based augmentation*) menggunakan teknik *backtranslation*. Selain itu, dilakukan pula pemodelan menggunakan *class weight* untuk mengatasi ketidakseimbangan kelas dan regularisasi L2 di *layer dense* untuk mengurangi *overfitting*.

4.6.1 Teknik Augmentasi *Backtranslation*

Teknik agmentasi ini, menggunakan NMT (*neural machine translation*) seperti Google Translate untuk menerjemahkan teks bahasa Indonesia ke dalam satu/beberapa bahasa lain, kemudian diterjemahkan kembali ke dalam bahasa Indonesia. Pada penelitian ini, peneliti menggunakan teknik augmentasi menggunakan 5 bahasa dengan alur bahasa Indonesia, bahasa Jawa, bahasa Inggris, bahasa Cina, bahasa Melayu, dan ke bahasa Indonesia kembali. Hal ini didasarkan pada penelitian klasifikasi teks berbahasa Indonesia pada studi kasus deteksi judul dan komentar hate speech pada berita online yang menunjukkan peningkatan yang cukup ketika menggunakan *back translation* lima bahasa yang serupa (Rahma & Suadaa, 2023).

Tabel 4.11 Ilustrasi Tahapan Back Translate Lima Bahasa

Tahapan	Hasil Pemrosesan
Teks Bersih (Hasil Pemrosesan)	jawa timur butuh pemimpin visioner gus imin meminta kader pkb tak berpangku tangan pilih luluk lukman untuk jawa timur yang adil dan makmur
Penerjemahan ke dalam bahasa Jawa	<i>Jawa Wétan butuh guser pimpinan wétan sing cocog karo PKB PKB kadres supaya ora milih kanthi milih Luluk Lukman kanggo Jawa Timur lan makmur</i>
Penerjemahan ke dalam bahasa Inggris	<i>East Java need East's Eastern leader guser matching PKB PKB Cadres to pick up by choosing Lukman and prosperity</i>
Penerjemahan ke dalam bahasa Mandarin	东爪哇需要东方的东部领导人古瑟 (Guser) 匹配 PKB PKB 干部, 以选择卢克曼 (Lukman) 和繁荣
Penerjemahan ke dalam bahasa Melayu	Jawa Timur Memerlukan Pemimpin Timur Guser untuk Memadankan PKB PKB PKB untuk memilih Lukman dan Prosper
Penerjemahan kembali ke dalam bahasa Indonesia	Jawa Timur Membutuhkan Pemimpin Guser Timur untuk mencocokkan CPB PKB PKB untuk memilih Lukman dan Prosper

Tabel 4.11 adalah ilustrasi dari proses *back translate*, dimana teks berbahasa Indonesia akan diubah kembali menjadi teks berbahasa Indonesia dengan penggantian diksi dan struktur kalimat. Hasil ini kemudian digunakan untuk menambah jumlah kelas negatif dan netral pada data *train*.

Selain menggunakan *back translate* lima bahasa, dicobakan juga proses *back translate* empat bahasa, yakni teks awal bahasa Indonesia, kemudian bahasa Inggris, bahasa Mandarin, bahasa Melayu, dan kembali lagi ke bahasa Indonesia.

Tabel 4.12 Ilustrasi Tahapan Back Translate Empat Bahasa

Tahapan	Hasil Pemrosesan
Teks Bersih (Hasil Pemrosesan)	jawa timur butuh pemimpin visioner gus imin meminta kader pkb tak berpangku tangan pilih luluk lukman untuk jawa timur yang adil dan makmur
Penerjemahan ke dalam bahasa Inggris	<i>East Java needs a visionary leader Gus Imin asks PKB cadres not to stand idly by choosing Luluk Lukman for East Java and prosperous</i>
Penerjemahan ke dalam bahasa Mandarin	东爪哇需要一个有远见的领导人古斯·艾明 (Gus Imin)
Penerjemahan ke dalam bahasa Melayu	Jawa Timur Memerlukan Pemimpin Berwawasan Gus Imin
Penerjemahan kembali ke dalam bahasa Indonesia	Jawa Timur Membutuhkan Pemimpin Visioner Gus Imin

Tabel 4.12 merupakan ilustrasi dari proses *back translate* empat bahasa yang menghasilkan representasi teks yang sangat berbeda dari teks awal dan digunakan untuk memperkaya korpus kelas minoritas. Sehingga, setelah augmentasi, distribusi kalimat dari setiap kelas ditampilkan pada Tabel 4.13.

Tabel 4.13 Distribusi Kelas Data *Train* Setelah Augmentasi

Kelas	Distribusi Sebelum	Distribusi Sesudah
<i>Back Translation Empat Bahasa</i>		
Negatif	127	252
Netral	462	916
Positif	1986	1986
<i>Back Translation Lima Bahasa</i>		
Negatif	127	252
Netral	462	916
Positif	1986	1986

Penampilan distribusi kelas dilakukan untuk BT empat bahasa dan lima bahasa dikarenakan hasil yang berbeda, dimungkinkan memiliki duplikat yang berbeda walaupun berdasarkan Tabel 4.13 jumlah distribusi kelas untuk dua skenario *back translation* yang berbeda adalah mirip dengan perbedaan di kelas positif untuk lima bahasa. Hasil BT ini meningkatkan rasio kelas hingga 8:4:1, dari yang awalnya 16:4:1 pada data *train*.

Setelah didapatkan hasil teks dengan *back translation*, dilakukan pemodelan dengan skenario pemodelan dan *layer-layer* yang sama seperti pemodelan awal, digunakan pula pengaturan *hyperparameter* yang sama yakni dilakukan sebanyak 20 *epoch* dengan *learning rate* default. Hasil pemodelan *back translate* untuk semua skenario model dan jenis *back translate* ditampilkan pada Tabel 4.14

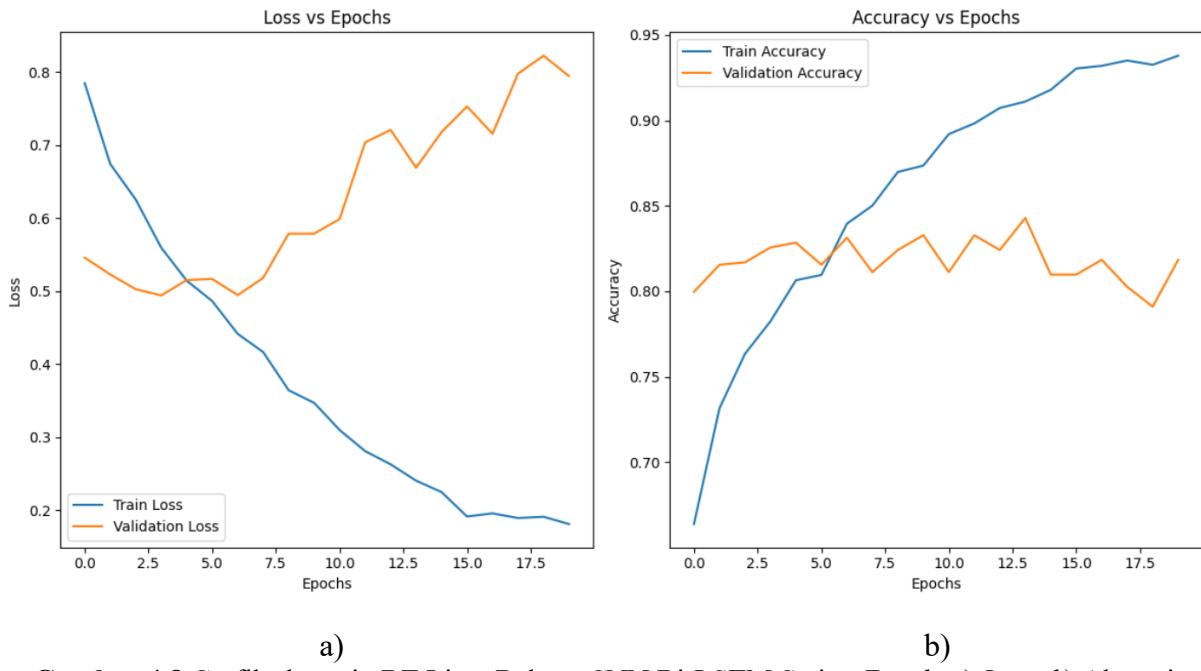
Tabel 4.14 Metrik Kebaikan Model Klasifikasi *Back Translation* Pada Data *Test*

Kelas	Akurasi Model (%)	Weighted F1 Score (%)	Selisih Akurasi Train-Test (%)
<i>Back Translation Empat Bahasa</i>			
Skenario CNN	79	79	16
Skenario Bi-LSTM	78	79	15
Skenario CNN Bi-LSTM	80	79	16
<i>Back Translation Lima Bahasa</i>			
Skenario CNN	77	78	18
Skenario Bi-LSTM	82	81	12
Skenario CNN Bi-LSTM	77	78	17

Hasil dari pemodelan menggunakan *back translation* lima bahasa dan empat bahasa jika dilihat menggunakan akurasi model dan *weighted F1 Score* pada Tabel 4.14 menunjukkan bahwa model memiliki variasi akurasi, *weighted F1 Score*, dan kriteria overfitting yang mirip dengan model awal, bahkan beberapa tidak lebih baik dari pada sebelum ditangani dengan augmentasi data, kecuali untuk model CNN Bi-LSTM yang mendapatkan kenaikan dari keseluruhan metrik

kebaikan model. Sekilas pandang, *back translate* dengan lima bahasa memiliki performa yang lebih baik dibandingkan dengan *back translate* dengan empat bahasa, walaupun perbedaannya sangat sedikit. Dari keseluruhan penambahan *back translate* tersebut, model Bi-LSTM pada BT lima bahasa mendapatkan performa terbaik dengan akurasi sebesar 82% dan weighted F1 score sebesar 81%, meskipun terdapat selisih akurasi *train-test* sebesar 15%, yang mengindikasikan kecenderungan *overfitting*.

Pada model terbaik berdasarkan Tabel 4.14, dilakukan analisis lebih lanjut untuk melihat kebaikan model.



Gambar 4.8 Grafik skenario BT Lima Bahasa CNN Bi-LSTM Setiap *Epoch*: a) *Loss*, b) Akurasi

Pada Gambar 4.8a), *train loss* mengalami penurunan tajam dan konsisten hingga mendekati nol, sementara *test loss* justru meningkat drastis setelah sekitar *epoch* ke-6 dan menunjukkan fluktuasi yang besar, mencapai nilai 0,8–1. Hal ini menunjukkan bahwa meskipun model mampu belajar dengan sangat baik pada data pelatihan, kemampuannya dalam mengenali pola pada data validasi semakin menurun seiring waktu. Pola yang sama tampak pada Gambar 4.8b), di mana akurasi *train* terus meningkat secara signifikan hingga melewati 90%, sementara akurasi *test* stagnan di kisaran 80–82% tanpa peningkatan berarti. Perbedaan ini menandakan bahwa model mengalami *overfitting* dan gagal melakukan generalisasi terhadap data baru.

Tabel 4.15 Metrik Setiap Kelas Pemodelan BT Lima Bahasa Bi-LSTM

Kelas	Presisi (%)	Recall (%)	Skor F1 (%)
Negatif	40	20	27
Netral	51	53	52
Positif	89	91	90
Akurasi (%)		82	
Weighted F1 Score		81	

Dari Tabel 4.15, model terbaik pada skenario *back translation* lima bahasa dengan arsitektur Bi-LSTM mempertahankan performa sangat baik pada kelas positif, dengan presisi 89%, *recall* 91%, dan skor F1 mencapai 90%. Namun, kendala klasifikasi pada kelas negatif dan netral masih cukup signifikan, masing-masing hanya memperoleh skor F1 sebesar 27% dan 52%. Rendahnya *recall* pada kelas negatif (20%) mengindikasikan bahwa model masih mengalami kesulitan dalam mendeteksi sentimen negatif yang cenderung jarang atau kurang eksplisit. Meskipun strategi augmentasi berbasis *back translation* mampu meningkatkan akurasi dan performa keseluruhan model, tantangan dalam menangani ketidakseimbangan kelas serta peningkatan deteksi pada kelas minoritas tetap menjadi isu yang perlu ditangani lebih lanjut.

4.6.2 Penggunaan *Class Weight* dan Regularisasi L2

Penggunaan *back translation* merupakan cara untuk mengatasi ketidakseimbangan data melalui augmentasi (penambahan) data dan efektif dalam menambah kebaikan model secara keseluruhan. Selain melakukan manipulasi pada data, dilakukan juga skenario pemodelan menggunakan algoritma yang *robust* terhadap *data imbalance* dan *overfitting*. *Class weights* digunakan untuk memberi bobot lebih besar pada kelas minoritas selama proses pelatihan, sehingga model tidak hanya fokus pada kelas mayoritas. Bobot pada *class weight* dihitung berdasarkan proporsi kebalikan (*inverse proportion*) dari distribusi kelas dalam data *train*. Sehingga, saat proses *training*, kelas minoritas dapat memberikan kontribusi *loss* yang lebih besar dalam model, sehingga model dapat lebih memperhatikan kelas minoritas.

Regularisasi L2 diterapkan untuk mengurangi risiko *overfitting*, yaitu kecenderungan model untuk terlalu menyesuaikan diri terhadap data *train*. Regularisasi L2 menambahkan penalti terhadap bobot-bobot model yang terlalu besar, sehingga parameter model cenderung lebih kecil dan generalisasi ke data baru menjadi lebih baik. Pada bagian ini, pemodelan akan dilakukan dengan skenario penambahan *class weight* dengan kriteria *balanced* dan regularisasi L2 dengan parameter regularisasi 0,01 untuk set data hasil pemberian label untuk model awal, dan hasil BT lima bahasa.

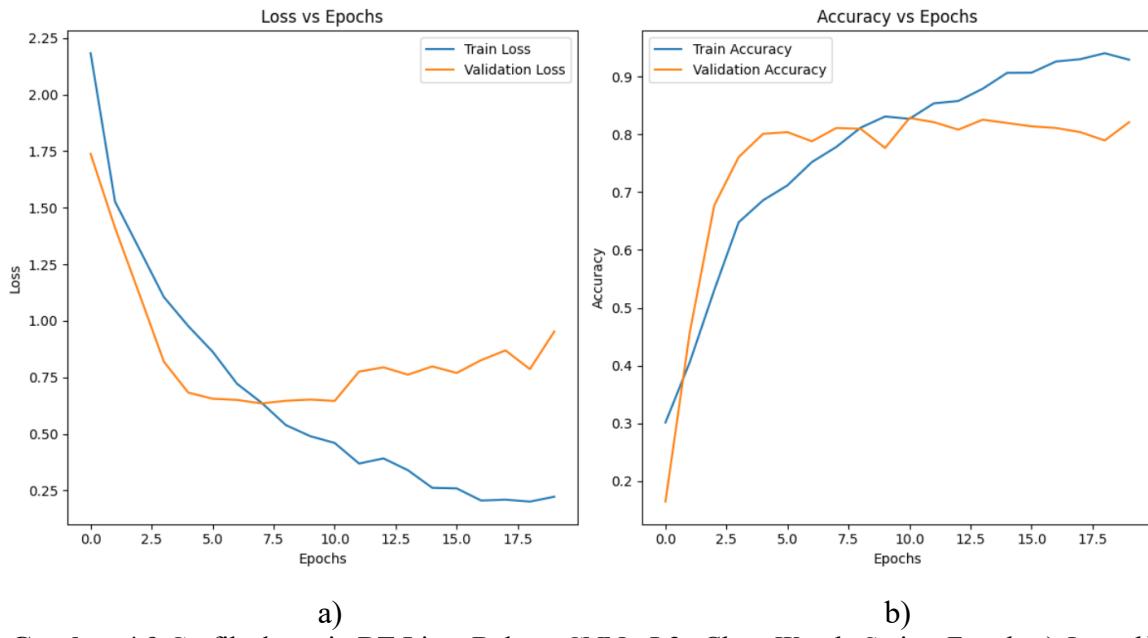
Tabel 4.16 Kebaikan Model Dengan Penambahan Regularisasi L2 dan *Class Weight*

Model	Akurasi (%)	Weighted F1 Score (%)	Selisih Akurasi Train-Test (%)
Data Awal			
CNN+CW	78	78	1
CNN+L2	79	79	18
CNN+L2+CW	73	76	8
LSTM+CW	79	80	2
LSTM+L2	83	81	10
LSTM+CW+L2	78	79	2
CNN LSTM+CW	81	81	3
CNN LSTM+L2	80	79	8
CNN LSTM+CW+L2	65	68	8
Back Translation Lima Bahasa			
CNN+CW	80	71	30
CNN+L2	80	80	16
CNN+L2+CW	82	81	10
LSTM+CW	75	76	14
LSTM+L2	78	79	14
LSTM+CW+L2	75	77	13
CNN LSTM+CW	82	79	6
CNN LSTM+L2	82	80	12
CNN LSTM+CW+L2	73	75	4
Back Translation Empat Bahasa			
CNN+CW	80	71	30
CNN+L2	80	80	16
CNN+L2+CW	82	81	10
LSTM+CW	75	76	14
LSTM+L2	78	79	14
LSTM+CW+L2	75	77	13
CNN LSTM+CW	82	79	6
CNN LSTM+L2	82	80	12
CNN LSTM+CW+L2	73	75	5

*)LSTM yang digunakan adalah Bi-LSTM dan CW adalah Class Weight

Tabel 4.16 menampilkan performa model seluruhnya pada skenario pemodelan awal, BT empat bahasa, dan lima bahasa. Secara keseluruhan, hasil pemodelan dengan menambah regularisasi L2 dan *class weight* memberikan hasil yang berbeda-beda. Dari pemodelan tersebut, setelah diurutkan berdasarkan akurasi dan *weighted F1 score* terbaik serta diurutkan berdasarkan kriteria *overfitting* terkecil, maka didapatkan model terbaik dari *back translation* lima bahasa dengan menggunakan model CNN dengan tambahan regularisasi L2 dan *class*

weight. Keunggulan model ini, dibandingkan model yang lain juga memiliki kriteria *overfitting* yang paling rendah dibanding peringkat-peringkat berikutnya.



Gambar 4.9 Grafik skenario BT Lima Bahasa CNN+ L2+*Class Weight* Setiap *Epoch*: a) *Loss*, b) Akurasi

Gambar 4.9a), menunjukkan bahwa loss *train* dan *test* menurun tajam pada *epochs* awal, lalu stabil mendekati nilai 0,6 tanpa gap yang terlalu jauh. Namun, setelah 7 *epoch*, loss pada *test* terlihat stagnan di angka 0,75–1. Pada Gambar 4.9b), akurasi *train* dan *test* juga menunjukkan tren kenaikan yang serupa dan relatif sejajar hingga akhir, kemudian stagnan pada angka 80%, sehingga tidak terdeteksi adanya *overfitting* yang parah.

Tabel 4.17 Akurasi Data *Test* Skenario BT Lima Bahasa CNN Dengan Regularisasi L2 dan *Class Weight*

Kelas	Presisi (%)	Recall (%)	Skor F1 (%)
Negatif	30	10	15
Netral	55	57	56
Positif	88	91	90
Akurasi (%)			82
Weighted F1 Score			81

Berdasarkan Tabel 4.18, model CNN dengan tambahan regularisasi L2 dan *class weight* menghasilkan akurasi keseluruhan sebesar 82% dan *weighted F1 score* sebesar 81%, yang merupakan performa terbaik sejauh ini. Model menunjukkan presisi dan *recall* yang sangat baik pada kelas positif, dengan skor F1 mencapai 90%, menandakan kemampuannya dalam mengenali sentimen yang kuat secara konsisten. Sementara itu, kelas netral mengalami peningkatan dibanding skenario sebelumnya, dengan skor F1 56%, yang mengindikasikan bahwa model mulai mampu membedakan ekspresi netral secara lebih akurat. Namun, performa pada kelas negatif masih sangat terbatas (skor F1 15%).

4.7 Penetapan Model Terbaik

Setelah didapatkan model terbaik dari eksperimen menggunakan skenario CNN, Bi-LSTM, dan CNN Bi-LSTM, dengan tambahan augmentasi, *class weight*, dan regularisasi L2, diambil beberapa model terbaik dari percobaan-percobaan tersebut, dan dibandingkan dengan model awal untuk mendapatkan model terbaik. Dari pemodelan awal yang belum menerapkan metode untuk mengatasi ketidakseimbangan data dan *overfitting*, diperoleh model terbaik LSTM dengan akurasi 83% dan *weighted F1 score* sebesar 72%, namun dengan kriteria *overfitting* yang masih moderat. Kemudian dari *back translate* didapatkan model Bi-LSTM menggunakan lima bahasa yang mendapatkan akurasi 82% dan 81%, dengan kriteria *overfitting* yang lebih tinggi dibanding pemodelan awal terbaik. Model terbaik selanjutnya ada pada *back translate* lima bahasa CNN dengan regularisasi L2 dan *class weight* mendapatkan akurasi 82% dengan *weighted F1 score* 81% dan dengan *overfitting* yang moderat. Karena penelitian ini memfokuskan pada akurasi model dalam memprediksi sentimen untuk langkah selanjutnya, dipilihlah model Bi-LSTM menggunakan data awal.

4.8 Karakteristik Sentimen Setiap Pasangan Calon

Setelah didapatkan label hasil prediksi model terbaik, dilanjutkan dengan melihat secara umum bagaimana karakteristik kata yang ada di dalam setiap sentimen dan pasangan calon.

4.8.1 Karakteristik Sentimen Pasangan Calon Pertama

Untuk pasangan calon pertama, yakni Luluk-Lukman, mendapatkan total sentimen negatif, netral, dan positif berturut-turut adalah 6,48%, 25,92%, dan 68,05%. Kemudian, jika dilihat bagaimana karakteristik kata untuk setiap sentimen terhadap Paslon ini, digunakan *word cloud*.

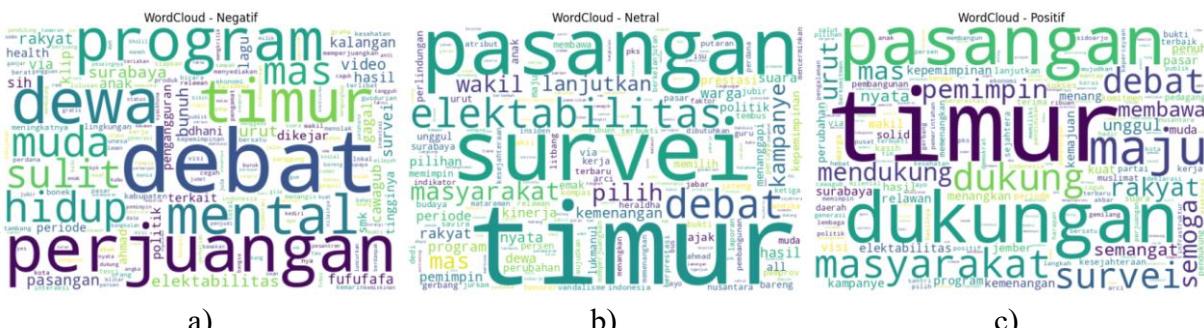


Gambar 4.10 Word cloud Paslon 1: a) Sentimen Negatif, b) Sentimen Netral, c) Sentimen Positif

Dari Gambar 4.10a), diketahui bahwa sentimen negatif terhadap pasangan calon pertama terdapat kata-kata berkonotasi negatif seperti pecat dan debat. Namun, pada sentimen ini lebih didominasi oleh kata-kata yang berkonotasi netral seperti pkb, debat, cak, dsb. Gambar 4.10b) menunjukkan sentimen netral didominasi oleh kata-kata berita seperti saksikan, perdana, dan debat. Gambar 4.10c) menunjukkan bahwa sentimen positif terhadap Paslon 1 didominasi oleh kata teman, pasangan, dan PKB (partai pengusung Paslon).

4.8.2 Karakteristik Sentimen Pasangan Calon Kedua

Pasangan calon kedua, sekaligus petahana, Khofifah-Emil mendapatkan total sentimen negatif, netral, dan positif berturut-turut adalah 3,59%, 16,59%, dan 79,84%.



a)

b)

c>

Gambar 4.11 Word cloud Paslon 2: a) Sentimen Negatif, b) Sentimen Netral, c) Sentimen Positif

Dari Gambar 4.11a), sentimen negatif terhadap pasangan calon didominasi oleh kata-kata seperti program, perjuangan, dan sulit. Untuk sentimen netral pada Gambar 4.11b) didominasi oleh kata elektabilitas, pasangan, survei, dan masyarakat, mencerminkan narasi yang bersifat informatif atau deskriptif seperti berita. Kemudian, untuk sentimen positif pada Gambar 4.11c) didominasi antara lain pasangan, dukungan, maju, masyarakat, dan semangat, menunjukkan harapan terhadap pasangan calon.

4.8.3 Karakteristik Sentimen Pasangan Calon Ketiga

Pasangan calon ketiga, Risma-Hans mendapatkan total sentimen negatif, netral, dan positif berturut-turut adalah 4,38%, 21,92%, dan 73,85%.



a)

b)

c)

Gambar 4.12 Word cloud Paslon 3: a) Sentimen Negatif, b) Sentimen Netral, c) Sentimen Positif

Dari Gambar 4.12a), sentimen negatif didominasi oleh kata-kata seperti pemimpin, pilih, warga, dan resik (slogan/program Paslon). Untuk sentimen netral pada Gambar 4.12b), kata-kata yang menonjol antara lain resik, bareng, rakyat, dan pilih. Sedangkan, pada Gambar 4.12c), kata-kata positif seperti resik, dukungan, rakyat, memilih, dan dukung menjadi dominan, menunjukkan dukungan terhadap pasangan calon ini. Kata yang muncul paling banyak di setiap sentimen untuk Paslon ini adalah resik, yang kemungkinan merujuk pada program atau slogan Paslon.

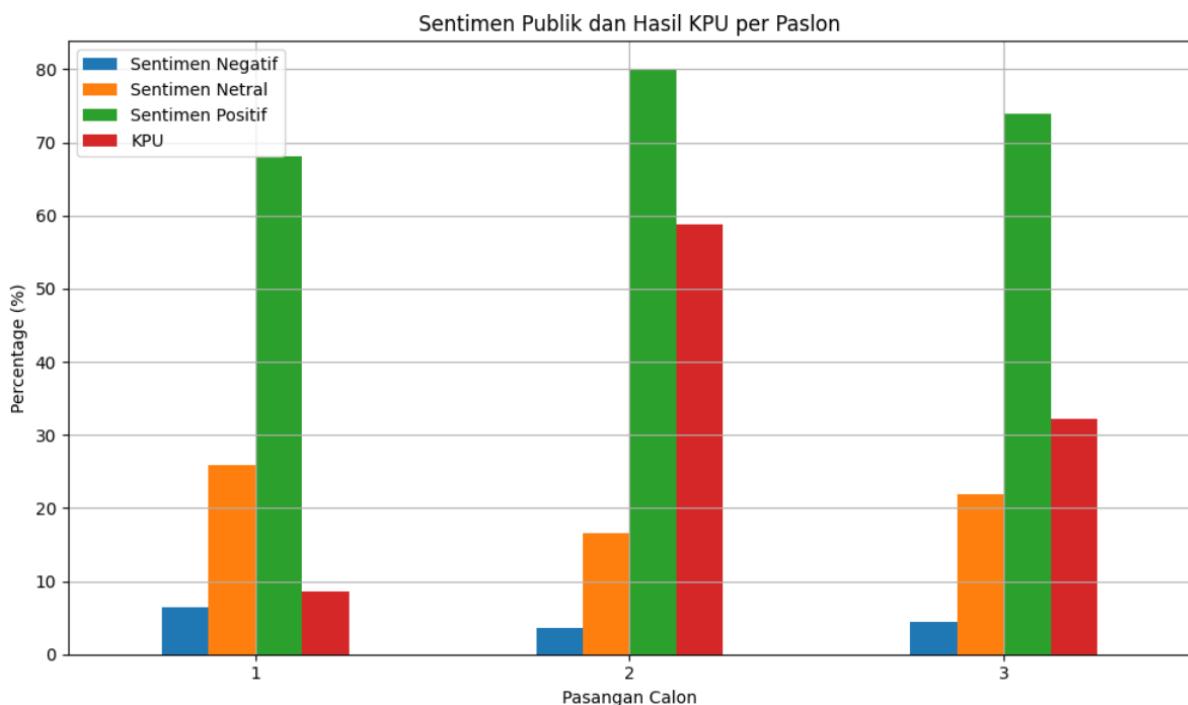
4.9 Analisis Hubungan Sentimen Paslon dengan Hasil Pilkada

Dari hasil prediksi, hasil kemudian ditabulasikan untuk mendapatkan jumlah sentimen negatif, netral, ataupun positif untuk setiap pasangan calon.

Tabel 4.18 Hasil Tabulasi Prediksi per Pasangan Calon dan Jenis Sentimen

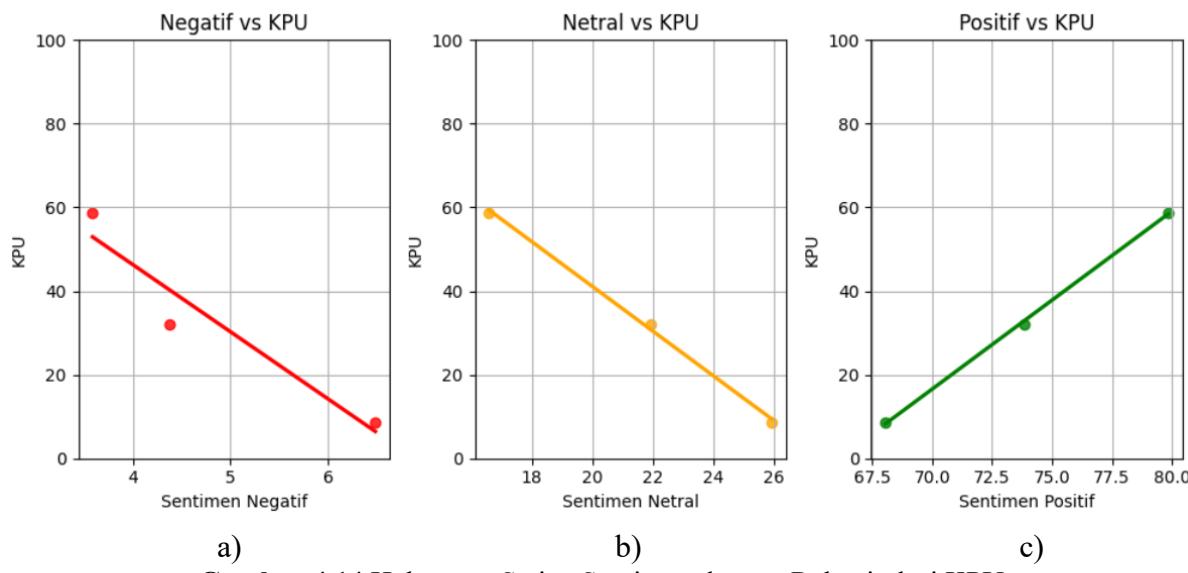
Pasangan Calon	Sentimen Negatif (%)	Sentimen Netral (%)	Sentimen Positif (%)	Rekapitulasi KPU (%)
1	6,48	25,92	68,05	8,67
2	3,59	16,59	79,84	58,81
3	4,38	21,92	73,85	32,2

Sekilas pada Tabel 4.18, terlihat bahwa dengan banyaknya sentimen positif dan negatif akan bersesuaian pada peningkatan rekapitulasi KPU.



Gambar 4.13 Visualisasi Sentimen dan Rekapitulasi KPU

Pada Gambar 4.13, pasangan Calon 2 memperoleh sentimen positif tertinggi (79,84%) dan secara konsisten mencatat perolehan suara tertinggi dalam rekapitulasi KPU (58,81%). Sebaliknya, Pasangan Calon 1 yang memiliki sentimen positif terendah (68,05%) juga meraih suara terendah (8,67%). Paslon 3 berada di posisi menengah untuk kedua variabel tersebut. Pola ini mengindikasikan bahwa terdapat kecenderungan antara sentimen positif publik dengan hasil suara resmi, di mana semakin positif persepsi publik terhadap kandidat, semakin besar kemungkinan kandidat tersebut memperoleh suara lebih tinggi.



Gambar 4.14 Hubungan Setiap Sentimen dengan Rekapitulasi KPU

Pada Gambar 4.14a), terlihat adanya korelasi negatif yang cukup jelas antara sentimen negatif dan hasil KPU. Pasangan calon dengan sentimen negatif tertinggi memperoleh suara terendah, berbanding terbalik dengan Paslon dengan perolehan suara tertinggi. Hal ini mengindikasikan bahwa semakin tinggi persepsi negatif publik terhadap paslon, semakin kecil peluangnya untuk memperoleh suara besar. Begitu pula untuk Gambar 4.14b), Pasangan calon dengan sentimen netral tertinggi memperoleh suara terendah. Sedangkan untuk Gambar 4.14c), tampak jelas hubungan linear positif antara sentimen positif dan hasil KPU. Paslon dengan sentimen positif tertinggi secara konsisten memperoleh suara tertinggi, sementara paslon dengan persepsi positif yang lebih rendah cenderung meraih suara lebih sedikit. Ini memperkuat dugaan bahwa tingginya sentimen positif publik dapat menjadi indikator kuat terhadap elektabilitas pasangan calon.

Untuk menentukan dugaan tersebut, digunakan koefisien korelasi Kendall's Tau. Untuk menghitung koefisien korelasi Kendall's Tau antara tabulasi sentimen dengan rekapitulasi KPU, digunakan *software R* dengan *library VGAM*.

Tabel 4.19 Hubungan Antara Sentimen dengan Rekapitulasi KPU

Hubungan	Koefisien Kendall Tau
Sentimen Negatif dan Rekapitulasi KPU	-1
Sentimen Netral dan Rekapitulasi KPU	-1
Sentimen Positif dan Rekapitulasi KPU	1

Tabel 4.19 menunjukkan bahwa koefisien Kendall's Tau untuk sentimen negatif dan netral terhadap rekapitulasi KPU adalah -1, mengindikasikan korelasi negatif, sedangkan sentimen positif memiliki korelasi positif sebesar 1. Hasil ini memperkuat temuan pada Gambar 4.14 bahwa semakin tinggi sentimen negatif atau netral, semakin rendah perolehan suara, dan sebaliknya, sentimen positif yang tinggi terkait erat dengan suara yang besar. Selain hubungan dari sentimen warganet, jumlah cuitan yang memuat pasangan calon juga memiliki hubungan dengan hasil KPU. Berdasarkan data, Paslon 2 memiliki peringkat jumlah cuitan terbanyak, disusul oleh Paslon 3 dan Paslon 1. Ketika hal ini dikorelasikan dengan korelasi Kendall Tau,

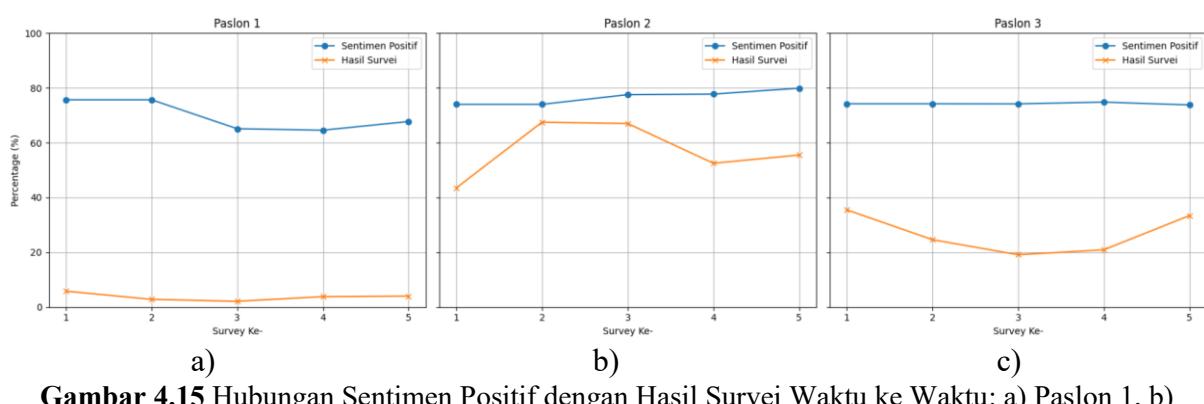
didapatkan nilai 1 yang artinya terdapat hubungan linear positif antara jumlah cuitan Paslon dengan hasil KPU. Dengan demikian, sentimen publik dan jumlah cuitan dapat mencerminkan elektabilitas pasangan calon dalam studi kasus ini.

Selain dibandingkan berdasarkan rekapitulasi KPU, hasil sentimen positif juga dibandingkan secara deskriptif dengan hasil survei selama masa kampanye (5 survei). Hasil survei ini, dimuat pada Tabel 4.20.

Tabel 4.20 Perbandingan Survei Selama Masa Kampanye dengan Sentimen Positif

Survei Ke-	Pasangan Calon	Sentimen Positif (%)	Hasil Survei (%)	Hasil
1	1	75,68	5,8	Tidak Konsisten
	2	74,01	43,5	
	3	74,23	35,5	
2	1	75,68	2,8	Tidak Konsisten
	2	74,01	67,5	
	3	74,23	24,6	
3	1	65,09	2,1	Konsisten
	2	77,55	67	
	3	74,2	19,1	
4	1	64,57	3,8	Konsisten
	2	77,75	52,5	
	3	74,82	20,9	
5	1	67,74	4	Konsisten
	2	79,9	55,5	
	3	73,8	33,4	

Dari Tabel 4.20, didapatkan bahwa hubungan antara sentimen positif dengan survei tidak selalu konsisten jika dibandingkan dengan hubungan sentimen positif dengan hasil KPU. Dapat dilihat pada survei 1–3, bahwa paslon dengan peringkat sentimen positif terbanyak berubah-ubah, padahal, hasil survei secara konsisten menunjukkan bahwa Paslon 2 selalu unggul dalam hasil survei. Selengkapnya, hasil Tabel 4.20 divisualisasikan menggunakan *line chart* sederhana pada Gambar 4.15.



Gambar 4.15 Hubungan Sentimen Positif dengan Hasil Survei Waktu ke Waktu: a) Paslon 1, b) Paslon 2, c) Paslon 3

Dari Gambar 4.15, Paslon 1 terlihat memimpin sentimen positif pada periode survei 1–2, lalu menurun sembari Paslon 2 mengalami peningkatan sentimen positif. Sedangkan, Paslon 3 mengalami penurunan yang stabil dan baru mengikuti hasil yang konsisten ketika survei ke-5 dimana jumlah sentimen positif Paslon 3 sedikit dibawah Paslon 2 dan Paslon 1 memiliki sentimen positif sedikit lebih rendah dibandingkan keduanya. Secara keseluruhan, grafik ini menunjukkan bahwa meskipun ada indikasi korelasi, hubungan antara sentimen positif di media sosial dan elektabilitas dari waktu ke waktu tidak selalu linier dan dapat dipengaruhi oleh banyak faktor lainnya. Kemudian, jika data pada Tabel 4.20, jika dikorelasikan menggunakan korelasi Pearson, didapatkan nilai 0,63. Angka ini menunjukkan bahwa persentase sentimen positif masih memiliki hubungan yang positif dengan hasil survei dari waktu ke waktu, namun angka ini menunjukkan hubungan yang sedang–kuat . Sehingga, menunjukkan pula bahwa hubungan ini bisa jadi konsisten dalam waktu ke waktu.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan analisis dan pembahasan yang telah dilakukan pada data cuitan X pada masa Pilkada 2024 pada studi kasus Pemilihan Gubernur Jawa Timur 2024, didapatkan beberapa simpulan.

1. Kecenderungan warganet X, dalam menanggapi pemilihan Gubernur Jawa Timur 2024 mayoritas memiliki karakteristik kata positif untuk keseluruhan cuitan. Jika dilihat per pasangan calon, terdapat perbedaan karakteristik sentimen. Paslon 1 memperoleh sentimen positif sebesar 68,05%, yang menunjukkan dukungan cukup kuat, tetapi paling rendah dibandingkan Paslon lainnya. Paslon 2 memiliki dominasi sentimen positif tertinggi sebesar 79,84% (tertinggi). Sementara itu, Paslon 3 mendapatkan sentimen positif sebesar 73,85%, menunjukkan tingkat dukungan yang relatif kuat namun masih berada di bawah Paslon 2.
2. Hasil sentimen tersebut didapatkan berdasarkan hasil pemodelan analisis sentimen menggunakan *transfer learning* IndoBERTweet dengan berbagai macam skenario. Hasil pemodelan terbaik didapatkan pada penggunaan model *transfer learning* IndoBERTweet-Bi-LSTM, dengan akurasi pada data *test* sebesar 83% dan *weighted F1 score* 81%. Meskipun model ini merupakan model terbaik, namun model ini masih dipengaruhi oleh ketidakseimbangan data yang ekstrem dan masih terdapat *overfitting*, walau tidak terlalu parah. Ketidakseimbangan ini menyebabkan model masih kesusahan dalam mengklasifikasikan dengan benar untuk kelas minoritas, yakni sentimen netral dan negatif.
3. Prediksi pemodelan terbaik ditabulasikan dan dibandingkan dengan hasil rekapitulasi resmi dari KPU. Hasilnya, dalam studi kasus ini, sentimen dari X memiliki hubungan terhadap hasil rekapitulasi KPU, dimana sentimen positif memiliki hubungan linear positif. Sebaliknya, sentimen negatif dan netral memiliki hubungan linear negatif dengan hasil rekapitulasi KPU. Sehingga, sentimen positif di media sosial dapat menjadi indikator awal yang cukup akurat untuk memprediksi hasil pemilu, karena menunjukkan hubungan linear positif. Sebaliknya, tingginya sentimen negatif atau netral cenderung mengindikasikan rendahnya dukungan nyata, karena berhubungan secara negatif dengan hasil pemilu. Namun, hubungan ini hanya ada ketika sentimen dihubungkan dengan hasil rekapitulasi KPU saja. Ketika sentimen ini dibandingkan dengan hasil survei dari waktu ke waktu, hubungan tersebut tidak selalu konsisten dan memiliki pengaruh positif dengan kekuatan sedang–kuat .

5.2 Saran

Meskipun dengan hasil pemodelan yang cukup secara keseluruhan, fakta bahwa masih terdapat ketidakseimbangan kelas yang ekstrem menunjukkan bahwa model masih belum bisa melakukan klasifikasi untuk kelas netral dan negatif dengan baik. Sehingga, perlu dilakukan teknik penanganan ketidakseimbangan data lainnya selain *class weight* dan augmentasi *back translation*. Selain itu, efek hubungan antara sentimen dengan rekapitulasi KPU perlu dijelajahi lebih dalam apakah efek ini berlaku pada studi kasus ini saja atau berlaku pada hampir semua kasus.

(Halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

- Al-Ayyoub, M., Khamaiseh, A. A., Jararweh, Y., & Al-Kabi, M. N. (2019). A comprehensive survey of arabic sentiment analysis. *Information Processing and Management*, 320-342.
- Alvi, Q., Ali, S. F., Ahmed, S. B., & Khan, N. A. (2023). On the frontiers of Twitter data and sentiment analysis in election prediction: a review. *PeerJ Computer Science*. doi:10.7717/peerj-cs.1517
- Anggraeni, W., Kusuma, M. F., Riksakomara, E., Wibowo, R. P., Pujiadi, & Sumpeno, S. (2024). Combination of BERT and Hybrid CNN-LSTM Models for Indonesia Dengue Tweets Classification. *Intellegent Engineering & Systems*, 17, 813-826. doi:10.22266/ijies2024.0229.68
- Arsi, P., & Waluyo, R. (2021). ANALISIS SENTIMEN WACANA PEMINDAHAN IBU KOTA INDONESIA MENGGUNAKAN ALGORITMA SUPPORT VECTOR MACHINE (SVM). *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIK)*, 147-156.
- Belle, G. V., Fisher, L., Heagerty, P. J., & Lumley, T. (2004). *Biostatistics: A Methodology for the Health Sciences (2nd Edition)*. Hoboken: John Wiley & Sons, Inc.
- Brahimi, B., Touahria, M., & Tari, A. (2021). Improving sentiment analysis in Arabic: A combined approach. *Journal of King Saud University - Computer and Information Sciences*, 1421-1250.
- Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcoulo, K., & Kuksa, P. (2011). Natural Language Processing (Almost) from Scratch. *The Journal of Machine Learning Research*, 12, 2493-2537. doi:<https://doi.org/10.48550/arXiv.1103.0398>
- Conway, B. A., Kenski, K., & Wang, D. (2014). The Rise of Twitter in the Political Campaign: Searching for Intermedia Agenda-Setting Effects in the Presidential Primary. *Computer-Mediated Communication*, 20, 363-380. doi:10.1111/jcc4.12124
- Daniel, W. W. (1990). *Applied Nonparametric Statistics 2nd Edition*. Atlanta: Cengage Learning Classic Series.
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (hal. 4171-4186). Minneapolis: Association for Computational Linguistics.
- Ernawati, S., & Wati, R. (2018). Penerapan Algoritma K-Nearest Neighbors Pada Analisis Sentimen Review Agen Travel. *JURNAL KHATULISTIWA INFORMATIKA*, VOL. VI, NO. 1 JUNI 2018, 64-69.
- Feldman, R., & Sanger, J. (2007). *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. New York: Cambridge University Press.
- Fithriasari, K., Jannah, S. Z., & Reyhana, Z. (2020). Deep Learning for Social Media Sentiment Analysis. *Matematika; MJIAM*, 36, 99-111.

Fitriyyah, S. N., Safriadi, N., & Pratama, E. E. (2019). Analisis Sentimen Calon Presiden Indonesia 2019 dari Media Sosial Twitter Menggunakan Metode Naive Bayes. *Jurnal Edukasi dan Penelitian Informatika*, 279-285.

Gamal, B. (2021). *Performance Metrics for Classification Models in Machine Learning: Part II*. Diambil kembali dari Medium: <https://bassantgz30.medium.com/performance-metrics-for-classification-models-in-machine-learning-part-ii-9303a1c7cadd>

Grandini, M., Bagli, E., & Visani, G. (2020). METRICS FOR MULTI-CLASS CLASSIFICATION: AN OVERVIEW. *A White Paper*. arXiv Preprints. Diambil kembali dari <https://arxiv.org/pdf/2008.05756>

Hollander, M., & Wolfe, D. A. (1973). *Nonparametric Statistical Methods*. New York: John Wiley & Sons.

Josephine, H., Nirmala., & Alluri, V. L. (2021). Impact of Hidden Dense Layers in Convolutional Neural Network to enhance Performance of Classification Model. *IOP Conference Series: Materials Science and Engineering*, 1-8. doi:10.1088/1757-899X/1131/1/012007

Jurafsky, D., & Martin, J. H. (2024). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. Stanford Education.

Kendall, M. G. (1938). A New Measure of Rank Correlation. *Biometrika*, 30, 81-93. doi:<https://doi.org/10.2307/2332226>

Kingma, D. P., & Ba, J. L. (2015). ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION. *3rd International Conference on Learning Representations*.

Kominfo Jatim. (2024, Agustus 30). *Dinas Kominfo Provinsi Jawa Timur*. Diambil kembali dari Resmi Ditutup. KPU : Ada Tiga Bacalon Ikut Daftar di Pilgub Jatim: <https://kominfo.jatimprov.go.id/berita/kpu-ada-tiga-bacalon-ikut-daftar-di-pilgub-jatim>

Koto, F., Lau, J. H., & Baldwin, T. (2021). INDOBERTTWEET: A Pretrained Language Model for Indonesian Twitter with Effective Domain-Specific Vocabulary Initialization. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021)* (hal. 10660-10668). Punta Cana: Association for Computational Linguistics. doi:<https://doi.org/10.18653/v1/2021.emnlp-main.833>

KPU RI. (2022). Peraturan Komisi Pemilihan Umum Nomor 8 Tahun 2022. Indonesia: JDIH KPU RI.

Lestari, A. R., Perdana, R. S., & Fauzi, M. A. (2017). Analisis Sentimen Tentang Opini Pilkada Dki 2017 Pada Dokumen Twitter Berbahasa Indonesia Menggunakan NaïveBayes dan Pembobotan Emozi. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, 1718-1724.

Liu, R., Shi, Y., Ji, C., & Jia, M. (2019). A Survey of Sentiment Analysis Based on Transfer Learning. *IEEE Access*, 85401-85412.

- Markoulidakis, I., Rallis, I., Georgoulas, I., Kopsiaftis, G., Doulamis, A., & Doulamis, N. (2021). Multiclass Confusion Matrix Reduction Method and Its Application on Net Promoter Score Classification Problem. *technologies*, 9. doi:<https://doi.org/10.3390/technologies9040081>
- Maulana, A. D., & Lhaksamana, K. M. (2023). Sentiment Analysis on Tweets of Kanjuruhan Tragedy Using Deep Learning IndoBERTweet. *Jurnal Media Informatika Budidarma*, 948-955.
- Mozafari, M., Farahbakhsh, R., & Crespi, N. (2019). A BERT-Based Transfer Learning Approach for Hate Speech Detection in Online Social Media. *Journal of Studies in Computational Intelligence* (hal. 928-940). Springer. Diambil kembali dari https://link.springer.com/chapter/10.1007/978-3-030-36687-2_77
- Nurfaizah, R. J., & Ahsan, M. (2024). MONITORING KUALITAS PELAYANAN RSUD DR. SOETOMO BERDASARKAN ULASAN PASIEN DI GOOGLE MAPS MENGGUNAKAN DIAGRAM KENDALI P BERBASIS BIDIRECTIONAL ENCODER REPRESENTATIONS FROM TRANSFORMERS (BERT). *Tugas Akhir*. Institut Teknologi Sepuluh Nopember. Diambil kembali dari <https://repository.its.ac.id/115058/>
- Pellicer, L. F., Ferreira, T. M., & Costa, a. H. (2023). Data augmentation techniques in natural language processing. *Applied Soft Computing*, 132.
- Pratama, F. A., & Romadhony, A. (2020). Identifikasi Komentar Toksik Dengan BERT. *Engineering* (hal. 7941-7949). Bandung: Telkom University.
- Prottasha, N. J., Sami, A. A., Kowsher, M., Akbar, S., Bairagi, A. K., Masud, M., & Baz, M. (2022). Transfer Learning for Sentiment Analysis Using BERT Based Supervised Fine-Tuning. *Sensors*, 22, 4157-4177.
- Rahma, I. A., & Suadaa, L. H. (2023). PENERAPAN TEXT AUGMENTATION UNTUK MENGATASI DATA YANG TIDAK SEIMBANG PADA KLASIFIKASI TEKS BERBAHASA INDONESIA STUDI KASUS: DETEKSI JUDUL CLICKBAIT DAN KOMENTAR HATE SPEECH PADA BERITA ONLINE. *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, 10, 1329-1340.
- Reginantis, I. A., Priyambodo, N. A., & Jamal, A. (2024). Analisis Penyebab Diselenggarakannya Pemungutan Suara Ulang (PSU) Di Provinsi Jawa Timur Tahun 2024. *Eksekusi: Jurnal Ilmu Hukum dan Administrasi Negara*, 268-276.
- Rinanda, H. (2024). *Pertarungan Sengit 3 Srikandi Berebut Singgasana Grahadi*. Jawa Timur: Detik Jawa Timur. Diambil kembali dari <https://www.detik.com/jatim/pilkada/d-7514085/pertarungan-sengit-3-srikandi-berebut-singgasana-grahadi>
- Rozi, I. F., Pramono, S. H., & Dahlan, E. A. (2012). Implementasi Opinion Mining (Analisis Sentimen) untuk Ekstraksi Data Opini Publik pada Perguruan Tinggi. *JEECCIS*, 6, 37-43. doi:[10.21776/jeeccis.v6i1.164](https://doi.org/10.21776/jeeccis.v6i1.164)

- Saeed, M. (2023). *A Gentle Introduction to Positional Encoding in Transformer Models, Part 1*. Diambil kembali dari <https://machinelearningmastery.com/a-gentle-introduction-to-positional-encoding-in-transformer-models-part-1/>
- Setiawan, C. J., Lhaksamana, K. M., & Bunyamin. (2023). SENTIMENT ANALYSIS OF INDONESIAN TIKTOK REVIEW USING LSTM AND INDOBERTWEET ALGORITHM. *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)*, 774-780.
- Tewari, U. (2021, November). *Regularization—Understanding L1 and L2 regularization for Deep Learning*. Dipetik Juni 11, 2025, dari Medium: <https://medium.com/analytics-vidhya/regularization-understanding-l1-and-l2-regularization-for-deep-learning-a7b9e4a409bf>
- Thompson, S. K. (1987). Sample Size for Estimating Multinomial Proportions. *The American Statistician*, 41, 42-46. doi:<https://doi.org/10.2307/2684318>
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., & Polosukhin, I. (2017). Attention is All you Need. *Advances in Neural Information Processing Systems* (hal. 5998-6008). NeurIPS Proceedings.
- Wadud, A. H., Shin, J., Mridha, F., & Nur, K. (2022). Deep-BERT: Transfer Learning for Classifying Multilingual Offensive Texts on Social Media. *Computer Systems Science and Engineering*, 1775-1791.
- Wibawa, A. P., Purnama, M. G., Akbar, M. F., & Dwiyanto, F. A. (2018). Metode-metode Klasifikasi. In *Prosiding Seminar Ilmu Komputer dan Teknologi Informasi (Vol. 3, No. 1)*, 134-138.
- Xie, Q., Dai, Z., Hovy, E., Luong, T., & Le, Q. V. (2020). Unsupervised Data Augmentation for Consistency Training. *Advances in Neural Information Processing Systems*. 33. NeurIPS. doi:<https://doi.org/10.48550/arXiv.1904.12848>
- Yang, Y. (2018). Convolutional Neural Networks with Recurrent Neural Filters. (hal. 912-917). Brussels: Association for Computational Linguistics. doi:10.18653/v1/D18-1109
- Zuhdi, A. M., Utami, E., & Raharjo, S. (2019). ANALISIS SENTIMENT TWITTER TERHADAP CAPRES INDONESIA 2019 DENGAN METODE K-NN. *Jurnal INFORMA Politeknik Indonusa Surakarta*, 1-7.

LAMPIRAN

Lampiran 1. Surat Pernyataan Data Sekunder

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini, mahasiswa Departemen Statistika FSAD ITS:

Nama : Abdillah Ilham
NRP : 5003211069

menyatakan bahwa data yang digunakan dalam **Tugas Akhir /Thesis*** ini merupakan data sekunder yang diambil dari **penelitian/buku/Tugas Akhir/Thesis/publikasi lainnya*** yaitu :

- Sumber : 1. Data Cuitan X (Twitter) dari <https://x.com>
2. Data rekapitulasi Pilgub Jatim 2024 dari <https://kominfo.jatimprov.go.id/berita/rekapitulasi-pilkada-jatim-2024-selesai-berikut-hasilnya-dari-38-kab-kota>
3. Data Survei Pilgub Jatim dengan rincian sumber:
- <https://www.metrotvnews.com/read/NG9C3o36>,
- <https://www.detik.com/jatim/pilkada/d-7610877/survei-terbaru-pilgub-jatim-elekabilitas-khofifah-emil-makin-kokoh>,
- <https://www.metrotvnews.com/read/kBVCax3Q>,
- <https://www.kompas.com/tren/read/2024/11/15/073000865/hasil-survei-litbang-kompas-pilkada-jatim-2024-khofifah-emil-525-persen?page=all>,
- <https://suarajatimpost.com/survei-pilgub-jatim-menjelang-hari-pemilihan-27-november-ini>.

- Keterangan : 1. Data cuitan dengan *keyword* pencarian nama Paslon Cagub Jatim 2024 pada masa kampanye (25 September 2024 – 23 November 2024).
2. Data rekapitulasi Pilgub Jatim 2024 (persentase pemilih).
3. Data Survei Pilgub Jatim pada masa survei: 1–9 Oktober 2024, 4–10 Oktober 2024, 27 Oktober – 3 November 2024, 2–7 November 2024, 13–19 November 2024.

Surat Pernyataan ini dibuat dengan sebenarnya. Apabila terdapat pemalsuan data maka saya siap menerima sanksi sesuai aturan yang berlaku.

Surabaya, 25 Juni 2025

Mengetahui,
Dosen Pembimbing Tugas Akhir

Mahasiswa

Adatul Mukarromah, S.Si, M.Si
NIP. 19800418 200312 2 001

Abdillah Ilham
NRP. 5003211069

Lampiran 2. Ukuran Sampel Berdistribusi Multinomial

α	$d^2 n$	n ($B = 0,05$)	m
0,5	0,44	177	4
0,4	0,51	203	4
0,3	0,60	241	3
0,2	0,75	299	3
0,1	1,01	403	3
0,05	1,27	510	3
0,025	1,56	624	2
0,02	1,66	664	2
0,01	1,97	788	2
0,005	2,28	915	2
0,001	3,03	1212	2
0,0005	3,33	1342	2
0,0001	4,11	1645	2

*Diambil dari (Thompson, 1987)

Lampiran 3. Kode Crawling Data X

```
!pip install pandas

# Install Node.js
!sudo apt-get update
!sudo apt-get install -y ca-certificates curl gnupg
!sudo mkdir -p /etc/apt/keyrings
!curl -fsSL https://deb.nodesource.com/gpgkey/nodesource-repo.gpg.key | sudo gpg
--dearmor -o /etc/apt/keyrings/nodesource.gpg

!NODE_MAJOR=20    && echo "deb [signed-by=/etc/apt/keyrings/nodesource.gpg]
https://deb.nodesource.com/node_$NODE_MAJOR.x nodistro main" | sudo tee
/etc/apt/sources.list.d/nodesource.list

!sudo apt-get update
!sudo apt-get install nodejs -y

!node -v
# Sample: Crawl Data Luluk-Lukman Bagian 4

filename = '1 part 4.csv'
search_keyword = 'luluk lukman since:2024-10-11 until:2024-11-23 lang:id'
limit = 2000

!npx -y tweet-harvest@2.6.1 -o "{filename}" -s "{search_keyword}" --tab "LATEST"
-l {limit} --token {twitter_auth_token}
```

Lampiran 4. Kode untuk Pembersihan Data untuk Pemodelan

```
# Setelah dimuat df dari setiap paslon dan setelah dilakukan drop duplikat
df = pd.concat([df1, df2, df3])
df.drop_duplicates(subset='id_str', inplace=True)
df = df.dropna(subset=['full_text'])

import pandas as pd
import re
import html
import matplotlib.pyplot as plt

def lowercase(review):
    return review.lower()

def remove_encrypted_text(review):
    pattern = r'@[A-Za-z0-9+/=]{20,60}'
    return re.sub(pattern, '', review)

def remove_username(review):
    pattern = r'@\w+'
    return re.sub(pattern, '', review).strip()

def remove_hashtags2(review):
    pattern = r'#\w+'
    return re.sub(pattern, '', review).strip()

def remove_url(review):
    return re.sub(r'http\S+', '', review)

def remove_square_brackets(review):
    return re.sub(r'\[.*?\]', '', review)

def remove_irrelevant(review):
    return re.sub('[^a-zA-Z\s]', ' ', review)

def remove_space(review):
    review= re.sub(r'\s+', ' ', review)
    review= review.replace('\n', ' ')
    review= review.replace('\\n', ' ')
    review= review.replace('\t', ' ')
    review= review.replace('\\t', ' ')
    review= review.replace('\\u', ' ')
    review= review.replace('\\', ' ')
    return review

def remove_emoji(review):
    emoji_pattern= re.compile("["
        u"\U0001F600-\U0001F64F" # emoticon
        u"\U0001F300-\U0001F5FF" # simbol & dingbat
        u"\U0001F680-\U0001F6FF" # transportasi & simbol map
        u"\U0001F700-\U0001F77F" # simbol kuno
        u"\U0001F780-\U0001F7FF" # simbol kuno tambahan
        u"\U0001F800-\U0001F8FF" # simbol tanda batas
        u"\U0001F900-\U0001F9FF" # emoticon tambahan
        u"\U0001FA00-\U0001FA6F" # simbol musik
        u"\U0001FA70-\U0001FAFF" # simbol musik tambahan
        u"\U00002702-\U000027B0" # simbol karakter
    ])
```

```

        u"\U000024C2-\U0001F251" # simbol katakter tambahan
        "]+", flags= re.UNICODE)
    return emoji_pattern.sub(r'', review)

def remove_html_entities(text):
    return html.unescape(text)

def normalisasi_kalimat(kalimat, kamus):
    tokens = kalimat.split()
    hasil = [kamus.get(token, token) for token in tokens]
    return ' '.join(hasil)

def repeat_char(review):
    review= re.sub('([A-Za-z])\\1{2,}', '\\1', review)
    return review

def remove_amp(text):
    return re.sub(r'\s+amp\s+', ' ', text).strip()

def df_prep(train):
    train['full_text'] = train['full_text'].fillna('').astype(str)
    train['filtering']= train['full_text'].apply(remove_encrypted_text)
    train['filtering']= train['filtering'].apply(lowercase)
    train['filtering']= train['filtering'].apply(remove_url)
    train['filtering']= train['filtering'].apply(remove_username)
    train['filtering']= train['filtering'].apply(remove_hashtags2)
    train['filtering']= train['filtering'].apply(remove_square_brackets)
    train['filtering']= train['filtering'].apply(remove_irrelevant)
    train['filtering']= train['filtering'].apply(remove_space)
    train['filtering']= train['filtering'].apply(remove_emoji)
    train['filtering']= train['filtering'].apply(repeat_char)
    train['filtering']= train['filtering'].apply(lambda x: normalisasi_kalimat(x, kamus))
    train['filtering']= train['filtering'].apply(remove_html_entities)
    train['filtering']= train['filtering'].apply(remove_amp)
    return train

# Now apply the function
df_clean = df_prep(df)
df1_clean = df_prep(df1)
df2_clean = df_prep(df2)
df3_clean = df_prep(df3)

```

Lampiran 5. Kode untuk Pembersihan Data untuk Statistika Deskriptif

```
# Setelah dimuat df dari setiap paslon dan setelah dilakukan drop duplikat
df = pd.concat([df1, df2, df3])
df.drop_duplicates(subset='id_str', inplace=True)
df = df.dropna(subset=['full_text'])

import pandas as pd
import re
import html
import matplotlib.pyplot as plt

def lowercase(review):
    return review.lower()

def remove_encrypted_text(review):
    pattern = r'@[A-Za-z0-9+/=]{20,60}'
    return re.sub(pattern, '', review)

def remove_username(review):
    pattern = r'@\w+'
    return re.sub(pattern, '', review).strip()

def remove_hashtags2(review):
    pattern = r'#\w+'
    return re.sub(pattern, '', review).strip()

def remove_url(review):
    return re.sub(r'http\S+', '', review)

def remove_square_brackets(review):
    return re.sub(r'\[.*?\]', '', review)

def remove_irrelevant(review):
    return re.sub('[^a-zA-Z\s]', ' ', review)

def remove_space(review):
    review= re.sub(r'\s+', ' ', review)
    review= review.replace('\n', ' ')
    review= review.replace('\\n', ' ')
    review= review.replace('\t', ' ')
    review= review.replace('\\t', ' ')
    review= review.replace('\\u', ' ')
    review= review.replace('\\', ' ')
    return review

def remove_emoji(review):
    emoji_pattern= re.compile("["
        u"\U0001F600-\U0001F64F" # emoticon
        u"\U0001F300-\U0001F5FF" # simbol & dingbat
        u"\U0001F680-\U0001F6FF" # transportasi & simbol map
        u"\U0001F700-\U0001F77F" # simbol kuno
        u"\U0001F780-\U0001F7FF" # simbol kuno tambahan
        u"\U0001F800-\U0001F8FF" # simbol tanda batas
        u"\U0001F900-\U0001F9FF" # emoticon tambahan
        u"\U0001FA00-\U0001FA6F" # simbol musik
        u"\U0001FA70-\U0001FAFF" # simbol musik tambahan
        u"\U00002702-\U000027B0" # simbol karakter
    ])"
```

```

        u"\U000024C2-\U0001F251" # simbol katakter tambahan
        "]+", flags= re.UNICODE)
    return emoji_pattern.sub(r'', review)

def remove_html_entities(text):
    return html.unescape(text)

def normalisasi_kalimat(kalimat, kamus):
    tokens = kalimat.split()
    hasil = [kamus.get(token, token) for token in tokens]
    return ' '.join(hasil)

def repeat_char(review):
    review= re.sub('([A-Za-z])\\1{2,}', '\\1', review)
    return review

def remove_amp(text):
    return re.sub(r'\s+amp\s+', ' ', text).strip()

def df_prep(train):
    train['full_text'] = train['full_text'].fillna('').astype(str)
    train['filtering']= train['full_text'].apply(remove_encrypted_text)
    train['filtering']= train['filtering'].apply(lowercase)
    train['filtering']= train['filtering'].apply(remove_url)
    train['filtering']= train['filtering'].apply(remove_username)
    train['filtering']= train['filtering'].apply(remove_hashtags2)
    train['filtering']= train['filtering'].apply(remove_square_brackets)
    train['filtering']= train['filtering'].apply(remove_irrelevant)
    train['filtering']= train['filtering'].apply(remove_space)
    train['filtering']= train['filtering'].apply(remove_emoji)
    train['filtering']= train['filtering'].apply(repeat_char)
    train['filtering']= train['filtering'].apply(lambda x: normalisasi_kalimat(x, kamus))
    train['filtering']= train['filtering'].apply(remove_html_entities)
    train['filtering']= train['filtering'].apply(remove_amp)
    return train

# Now apply the function
df_clean = df_prep(df)
df1_clean = df_prep(df1)
df2_clean = df_prep(df2)
df3_clean = df_prep(df3)

```

Lampiran 6. Kode untuk Statistika Deskriptif (Wordcloud)

```
import matplotlib.pyplot as plt
from collections import Counter

def get_top_words(df, stopwords_all, top_n=10, column='filtering'):
    # Gabungkan semua teks dari kolom yang telah dibersihkan
    all_text = ' '.join(df[column])

    # Tokenisasi dan filter stopwords
    words = [word for word in all_text.split() if word not in stopwords_all and len(word) > 2]

    # Hitung frekuensi kata
    word_freq = Counter(words)
    top_words = word_freq.most_common(top_n)

    # Pisahkan kata dan frekuensinya untuk plot
    labels, counts = zip(*top_words)

    # Plot bar chart
    plt.figure(figsize=(10, 6))
    plt.barh(labels, counts, color='skyblue')
    plt.xlabel('Frekuensi')
    plt.ylabel('Kata')
    plt.title(f'Top {top_n} Kata Paling Sering Muncul')
    plt.gca().invert_yaxis() # Bar horizontal dari atas ke bawah
    plt.tight_layout()
    plt.show()

    return top_words, word_freq

def generate_wordcloud(word_freq):
    wc = WordCloud(width=800, height=400,
                   background_color='white').generate_from_frequencies(word_freq)
    plt.figure(figsize=(10, 5))
    plt.imshow(wc, interpolation='bilinear')
    plt.axis('off')
    plt.title("WordCloud Kata Terpopuler")
    plt.show()

# Call Function for df all
# Analisis kata
top_words, word_freq = get_top_words(df, stop_words)

print("Top 10 Kata Terpopuler:")
for word, freq in top_words:
    print(f"{word}: {freq}")

# Tampilkan WordCloud
generate_wordcloud(word_freq)

# Call Function for df Paslon 1 (example), must have df1_cleaned first
# Analisis kata
top_words, word_freq = get_top_words(df1_cleaned, stop_words)

print("Top 10 Kata Terpopuler:")
for word, freq in top_words:
```

```
print(f"{word}: {freq}")

# Tampilkan WordCloud
generate_wordcloud(word_freq)
```

Lampiran 7. Kode untuk *Back Translation* (Contoh dengan bahasa Jawa)

```
import pandas as pd
import asyncio
from googletrans import Translator

translator = Translator()

async def translate_safe(text, src, dest):
    try:
        translated = await translator.translate(text, src=src, dest=dest)
        return translated.text
    except Exception as e:
        return f"Error: {str(e)}"

async def backtranslate_deconstructed(text):
    step1 = await translate_safe(text, 'id', 'jw')
    step2 = await translate_safe(step1, 'jw', 'en')
    step3 = await translate_safe(step2, 'en', 'zh-cn')
    final = await translate_safe(step3, 'zh-cn', 'id')
    return step1, step2, step3, final

# Call Function
async def process_dataframe(df):
    results = await asyncio.gather(*(backtranslate_deconstructed(text) for text
in df['filtering']))
    df[['bt_javanese', 'bt_english', 'bt_mandarin', 'bt_ind_final']] =
pd.DataFrame(results, index=df.index)
    return df
```

Lampiran 8. Kode untuk *Split* dan Tokenisasi BERT

```
df['num_sentiment'] = df['sentiment_label_new'].map({'negative': 0, 'neutral': 1, 'positive': 2})

import tensorflow as tf
from sklearn.model_selection import train_test_split
test_size = 0.2

tf.random.set_seed(42)
train_df, test_df = train_test_split(df, test_size=test_size, random_state=42)

test_df = test_df[['id_str', 'filtering', 'num_sentiment']]
train_df = train_df[['id_str', 'filtering', 'num_sentiment']]

from     sklearn.metrics      import      classification_report,      confusion_matrix,
ConfusionMatrixDisplay
import numpy as np
from transformers import BertTokenizer, TFBertModel

tokenizer = BertTokenizer.from_pretrained("indolem/indobertweet-base-uncased")

max_len = 100

def tokenize_texts(texts):
    return tokenizer(
        texts.tolist(), # Konversi Series ke list
        padding='max_length',
        truncation=True,
        max_length=max_len,
        return_tensors="tf",
        add_special_tokens=True
    )

# 1. Tokenizing and converting label to numerical

train_encodings = tokenize_texts(train_df['filtering'])
test_encodings = tokenize_texts(test_df['filtering'])

# Convert labels to tensors
train_labels = tf.convert_to_tensor(train_df['num_sentiment'].values)
test_labels = tf.convert_to_tensor(test_df['num_sentiment'].values)
```

Lampiran 9. Kode untuk Latih Skema CNN

```
# 1. Build Model
input_ids      = tf.keras.layers.Input(shape=(max_len,),           dtype=tf.int32,
name="input_ids")
attention_mask = tf.keras.layers.Input(shape=(max_len,),           dtype=tf.int32,
name="attention_mask")

# 2. Setting up IndoBERTweet
bert_model = TFBertModel.from_pretrained(
    "indolem/indobertweet-base-uncased",
    from_pt=True,
    output_hidden_states=True
)

for layer in bert_model.layers:
    layer.trainable = False # Set each BERT layer to non-trainable

# Get last hidden state of the IndoBERTweet Transformer
last_hidden_state = bert_model(input_ids,
                                attention_mask=attention_mask).last_hidden_state

# 3. CNN Scheme
# CNN Layers
x = tf.keras.layers.Conv1D(filters=512, kernel_size=3,
activation='relu')(last_hidden_state)
x = tf.keras.layers.MaxPooling1D(pool_size=4)(x)
x = tf.keras.layers.Conv1D(filters=256, kernel_size=5, activation='relu')(x)
x = tf.keras.layers.MaxPooling1D(pool_size=4)(x)

# Dense Layers
x = tf.keras.layers.Flatten()(x)
x = tf.keras.layers.Dense(64, activation='relu')(x)
x = tf.keras.layers.Dropout(0.3, seed = SEED)(x)
x = tf.keras.layers.Dense(32, activation='relu')(x)
x = tf.keras.layers.Dropout(0.3, seed = SEED)(x)

# Output Layer
output = tf.keras.layers.Dense(3, activation='softmax')(x)

# 4. Compile Model
model_cnn1 = tf.keras.Model(inputs=[input_ids, attention_mask], outputs=output)
model_cnn1.compile(optimizer=tf.keras.optimizers.Adam(),
                   loss=tf.keras.losses.SparseCategoricalCrossentropy(),
metrics=[tf.keras.metrics.SparseCategoricalAccuracy(name='accuracy')])

# 5. Training
history_cnn1 = model_cnn1.fit(
    [train_encodings['input_ids'], train_encodings['attention_mask']],
    train_labels,
    validation_data=(
        [test_encodings['input_ids'], test_encodings['attention_mask']],
        test_labels
    ),
    epochs=20,
    batch_size=32,
```

```
shuffle=False,
workers=1,
use_multiprocessing=False
)

# 6. Evaluation

predictions = model_cnn1.predict([test_encodings['input_ids'],
test_encodings['attention_mask']])
predicted_classes = np.argmax(predictions, axis=1)

# Classification Report
print(classification_report(test_labels, predicted_classes,
target_names=['Negatif', 'Netral', 'Positif']))
```

Lampiran 10. Kode untuk Latih Skema Bi-LSTM

```
# 1. Build Model
input_ids      = tf.keras.layers.Input(shape=(max_len,),           dtype=tf.int32,
name="input_ids")
attention_mask = tf.keras.layers.Input(shape=(max_len,),           dtype=tf.int32,
name="attention_mask")

# 2. Setting up IndoBERTweet
bert_model = TFBertModel.from_pretrained(
    "indolem/indobertweet-base-uncased",
    from_pt=True,
    output_hidden_states=True
)

for layer in bert_model.layers:
    layer.trainable = False # Set each BERT layer to non-trainable

# Get last hidden state of the IndoBERTweet Transformer
last_hidden_state = bert_model(input_ids,
                                attention_mask=attention_mask).last_hidden_state

# 3. CNN Scheme
# Bi-LSTM Layers
x = tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(32,
return_sequences=True))(last_hidden_state)
x = tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(16,
return_sequences=False))(x)

# Dense Layers
x = tf.keras.layers.Dense(64, activation='relu')(x)
x = tf.keras.layers.Dropout(0.3, seed = SEED)(x)
x = tf.keras.layers.Dense(32, activation='relu')(x)
x = tf.keras.layers.Dropout(0.3, seed = SEED)(x)

# Output Layer
output = tf.keras.layers.Dense(3, activation='softmax')(x)

# 4. Compile Model
model_lstm3 = tf.keras.Model(inputs=[input_ids, attention_mask], outputs=output)
model_lstm3.compile(optimizer=tf.keras.optimizers.Adam(),
                    loss=tf.keras.losses.SparseCategoricalCrossentropy(),
metrics=[tf.keras.metrics.SparseCategoricalAccuracy(name='accuracy')])

# 5. Training
history_lstm3 = model_lstm3.fit(
    [train_encodings['input_ids'], train_encodings['attention_mask']],
    train_labels,
    validation_data=(
        [test_encodings['input_ids'], test_encodings['attention_mask']],
        test_labels
    ),
    epochs=20,
    batch_size=32,
    shuffle=False,
    workers=1,
    use_multiprocessing=False
```

```
)  
  
# 6. Evaluation  
  
predictions = model_lstm3.predict([test_encodings['input_ids'],  
test_encodings['attention_mask']])  
predicted_classes = np.argmax(predictions, axis=1)  
  
# Classification Report  
print(classification_report(test_labels, predicted_classes,  
target_names=['Negatif', 'Netral', 'Positif']))
```

Lampiran 11. Kode untuk Latih Skema CNN Bi-LSTM

```
# 1. Build Model
input_ids      = tf.keras.layers.Input(shape=(max_len,),           dtype=tf.int32,
name="input_ids")
attention_mask = tf.keras.layers.Input(shape=(max_len,),           dtype=tf.int32,
name="attention_mask")

# 2. Setting up IndoBERTweet
bert_model = TFBertModel.from_pretrained(
    "indolem/indobertweet-base-uncased",
    from_pt=True,
    output_hidden_states=True
)

for layer in bert_model.layers:
    layer.trainable = False # Set each BERT layer to non-trainable

# Get last hidden state of the IndoBERTweet Transformer
last_hidden_state = bert_model(input_ids,
                                attention_mask=attention_mask).last_hidden_state

# 3. CNN-Bi LSTM Scheme
# Bi-LSTM Layers
x = tf.keras.layers.Conv1D(filters=512, kernel_size=3,
activation='relu')(last_hidden_state)
x = tf.keras.layers.MaxPooling1D(pool_size=4)(x)

x = tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(32,
return_sequences=True))(x)
x = tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(16,
return_sequences=False))(x)

# Dense Layers
x = tf.keras.layers.Dense(64, activation='relu')(x)
x = tf.keras.layers.Dropout(0.3, seed=SEED)(x)
x = tf.keras.layers.Dense(32, activation='relu')(x)
x = tf.keras.layers.Dropout(0.3, seed=SEED)(x)

# Output Layer
output = tf.keras.layers.Dense(3, activation='softmax')(x)

# 4. Compile Model
model_cnn_lstm1 = tf.keras.Model(inputs=[input_ids,           attention_mask],
outputs=output)
model_cnn_lstm1.compile(optimizer=tf.keras.optimizers.Adam(),
loss=tf.keras.losses.SparseCategoricalCrossentropy(),
metrics=[tf.keras.metrics.SparseCategoricalAccuracy(name='accuracy')])

# 5. Training
history_cnn_lstm1 = model_cnn_lstm1.fit(
    [train_encodings['input_ids'], train_encodings['attention_mask']],
    train_labels,
    validation_data=(
        [test_encodings['input_ids'], test_encodings['attention_mask']],
        test_labels
```

```
        ),
        epochs=20,
        batch_size=32,
        shuffle=False,
        workers=1,
        use_multiprocessing=False
    )

# 6. Evaluation

predictions      =      model_cnn_lstm1.predict([test_encodings['input_ids'],
test_encodings['attention_mask']])
predicted_classes = np.argmax(predictions, axis=1)

# Classification Report
print(classification_report(test_labels,                                     predicted_classes,
target_names=['Negatif', 'Netral', 'Positif']))
```

Lampiran 12. Kode Pemodelan CNN Dengan *Class Weight*

```
# Pembuatan bobot

from sklearn.utils import class_weight
labels_train_np = np.array(train_labels)

# Calculate class weights
class_weights = class_weight.compute_class_weight(
    class_weight='balanced',
    classes=np.unique(labels_train_np),
    y=labels_train_np
)
class_weights = {i: weight for i, weight in enumerate(class_weights)}
class_weights

# 1. Build Model
input_ids      = tf.keras.layers.Input(shape=(max_len,),          dtype=tf.int32,
name="input_ids")
attention_mask = tf.keras.layers.Input(shape=(max_len,),          dtype=tf.int32,
name="attention_mask")

# 2. Setting up IndoBERTweet
bert_model = TFBertModel.from_pretrained(
    "indolem/indobertweet-base-uncased",
    from_pt=True,
    output_hidden_states=True
)

for layer in bert_model.layers:
    layer.trainable = False # Set each BERT layer to non-trainable

# Get last hidden state of the IndoBERTweet Transformer
last_hidden_state           = bert_model(input_ids,
attention_mask=attention_mask).last_hidden_state

# 3. CNN-Bi LSTM Scheme
# Bi-LSTM Layers
x                  = tf.keras.layers.Conv1D(filters=512,           kernel_size=3,
activation='relu')(last_hidden_state)
x = tf.keras.layers.MaxPooling1D(pool_size=4)(x)

x                  = tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(32,
return_sequences=True))(x)
x                  = tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(16,
return_sequences=False))(x)

# Dense Layers
x = tf.keras.layers.Dense(64, activation='relu')(x)
x = tf.keras.layers.Dropout(0.3, seed=SEED)(x)
x = tf.keras.layers.Dense(32, activation='relu')(x)
x = tf.keras.layers.Dropout(0.3, seed=SEED)(x)

# Output Layer
output = tf.keras.layers.Dense(3, activation='softmax')(x)

# 4. Compile Model
```

```

model_cnn_lstm2      =      tf.keras.Model(inputs=[input_ids,      attention_mask],
outputs=output)
model_cnn_lstm2.compile(optimizer=tf.keras.optimizers.Adam(),
                        loss=tf.keras.losses.SparseCategoricalCrossentropy(),
metrics=[tf.keras.metrics.SparseCategoricalAccuracy(name='accuracy')])

# 5. Training
history_cnn_lstm2 = model_cnn_lstm2.fit(
    [train_encodings['input_ids'], train_encodings['attention_mask']],
    train_labels,
    validation_data=(
        [test_encodings['input_ids'], test_encodings['attention_mask']],
        test_labels
    ),
    epochs=20,
    batch_size=32,
    class_weight=class_weights,
    shuffle=False,
    workers=1,
    use_multiprocessing=False
)

# 6. Evaluation

predictions      =      model_cnn_lstm2.predict([test_encodings['input_ids'],
test_encodings['attention_mask']])
predicted_classes = np.argmax(predictions, axis=1)

# Classification Report
print(classification_report(test_labels,                                predicted_classes,
target_names=['Negatif', 'Netral', 'Positif']))

```

Lampiran 13. Kode Pemodelan CNN Dengan Regularisasi L2

```
# Setup Regulizer
regularizers = tf.keras.regularizers

# 1. Build Model
input_ids      = tf.keras.layers.Input(shape=(max_len,),           dtype=tf.int32,
name="input_ids")
attention_mask = tf.keras.layers.Input(shape=(max_len,),           dtype=tf.int32,
name="attention_mask")

# 2. Setting up IndoBERTweet
bert_model = TFBertModel.from_pretrained(
    "indolem/indobertweet-base-uncased",
    from_pt=True,
    output_hidden_states=True
)

for layer in bert_model.layers:
    layer.trainable = False # Set each BERT layer to non-trainable

# Get last hidden state of the IndoBERTweet Transformer
last_hidden_state          = bert_model(input_ids,
attention_mask=attention_mask).last_hidden_state

# 3. CNN Scheme
# CNN Layers
x      = tf.keras.layers.Conv1D(filters=512,           kernel_size=3,
activation='relu')(last_hidden_state)
x = tf.keras.layers.MaxPooling1D(pool_size=4)(x)
x = tf.keras.layers.Conv1D(filters=256, kernel_size=5, activation='relu')(x)
x = tf.keras.layers.MaxPooling1D(pool_size=4)(x)

# Dense Layers
x = tf.keras.layers.Flatten()(x)
x      = tf.keras.layers.Dense(64,           activation='relu',
kernel_regularizer=regularizers.l2(0.01))(x)
x = tf.keras.layers.Dropout(0.3, seed = SEED)(x)
x      = tf.keras.layers.Dense(32,           activation='relu',
kernel_regularizer=regularizers.l2(0.01))(x)
x = tf.keras.layers.Dropout(0.3, seed = SEED)(x)

# Output Layer
output = tf.keras.layers.Dense(3, activation='softmax')(x)

# 4. Compile Model
model_cnn3 = tf.keras.Model(inputs=[input_ids, attention_mask], outputs=output)
model_cnn3.compile(optimizer=tf.keras.optimizers.Adam(),
                   loss=tf.keras.losses.SparseCategoricalCrossentropy(),
metrics=[tf.keras.metrics.SparseCategoricalAccuracy(name='accuracy')])

# 5. Training
history_cnn3 = model_cnn3.fit(
    [train_encodings['input_ids'], train_encodings['attention_mask']],
    train_labels,
    validation_data=
```

```
[test_encodings['input_ids'], test_encodings['attention_mask']],
test_labels
),
epochs=20,
batch_size=32,
shuffle=False,
workers=1,
use_multiprocessing=False
)

# 6. Evaluation

predictions = model_cnn3.predict([test_encodings['input_ids'],
test_encodings['attention_mask']])
predicted_classes = np.argmax(predictions, axis=1)

# Classification Report
print(classification_report(test_labels, predicted_classes,
target_names=['Negatif', 'Netral', 'Positif']))
```

Lampiran 14. Kode Pemodelan CNN Dengan Regularisasi L2 dan *Class Weight*

```
# 1. Build Model
input_ids      = tf.keras.layers.Input(shape=(max_len,),           dtype=tf.int32,
name="input_ids")
attention_mask = tf.keras.layers.Input(shape=(max_len,),           dtype=tf.int32,
name="attention_mask")

# 2. Setting up IndoBERTweet
bert_model = TFBertModel.from_pretrained(
    "indolem/indobertweet-base-uncased",
    from_pt=True,
    output_hidden_states=True
)

for layer in bert_model.layers:
    layer.trainable = False # Set each BERT layer to non-trainable

# Get last hidden state of the IndoBERTweet Transformer
last_hidden_state = bert_model(input_ids,
                                attention_mask=attention_mask).last_hidden_state

# 3. CNN Scheme
# CNN Layers
x = tf.keras.layers.Conv1D(filters=512, kernel_size=3,
activation='relu')(last_hidden_state)
x = tf.keras.layers.MaxPooling1D(pool_size=4)(x)
x = tf.keras.layers.Conv1D(filters=256, kernel_size=5, activation='relu')(x)
x = tf.keras.layers.MaxPooling1D(pool_size=4)(x)

# Dense Layers
x = tf.keras.layers.Flatten()(x)
x = tf.keras.layers.Dense(64, activation='relu',
kernel_regularizer=regularizers.l2(0.01))(x)
x = tf.keras.layers.Dropout(0.3, seed = SEED)(x)
x = tf.keras.layers.Dense(32, activation='relu',
kernel_regularizer=regularizers.l2(0.01))(x)
x = tf.keras.layers.Dropout(0.3, seed = SEED)(x)

# Output Layer
output = tf.keras.layers.Dense(3, activation='softmax')(x)

# 4. Compile Model
model_cnn2 = tf.keras.Model(inputs=[input_ids, attention_mask], outputs=output)
model_cnn2.compile(optimizer=tf.keras.optimizers.Adam(),
                   loss=tf.keras.losses.SparseCategoricalCrossentropy(),
metrics=[tf.keras.metrics.SparseCategoricalAccuracy(name='accuracy')])

# 5. Training
history_cnn2 = model_cnn2.fit(
    [train_encodings['input_ids'], train_encodings['attention_mask']],
    train_labels,
    validation_data=(
        [test_encodings['input_ids'], test_encodings['attention_mask']],
        test_labels
    ),
    epochs=20,
```

```
batch_size=32,
class_weight=class_weights,
shuffle=False,
workers=1,
use_multiprocessing=False
)

# 6. Evaluation

predictions = model_cnn2.predict([test_encodings['input_ids'],
test_encodings['attention_mask']])
predicted_classes = np.argmax(predictions, axis=1)

# Classification Report
print(classification_report(test_labels, predicted_classes,
target_names=['Negatif', 'Netral', 'Positif']))
```

Lampiran 15. Kode Mengeluarkan *Classification Report* dan Grafik Akurasi dan *Loss* setiap *Epoch*

```
# === Predict for TRAIN set ===
train_predictions = model.predict([train_encodings['input_ids'],
train_encodings['attention_mask']])
train_predicted_classes = np.argmax(train_predictions, axis=1)

# === Predict for TEST set ===
test_predictions = model.predict([test_encodings['input_ids'],
test_encodings['attention_mask']])
test_predicted_classes = np.argmax(test_predictions, axis=1)

# === Classification Report ===
print("== Classification Report: TRAIN ==")
print(classification_report(train_labels, train_predicted_classes,
target_names=['Negatif', 'Netral', 'Positif']))

print("\n== Classification Report: TEST ==")
print(classification_report(test_labels, test_predicted_classes,
target_names=['Negatif', 'Netral', 'Positif']))

# === F1 Scores for TRAIN ===
from sklearn.metrics import f1_score
print("F1 Macro (Train):", f1_score(train_labels, train_predicted_classes,
average='macro'))
print("F1 Micro (Train):", f1_score(train_labels, train_predicted_classes,
average='micro'))
print("F1 Weighted (Train):", f1_score(train_labels, train_predicted_classes,
average='weighted'))

# === F1 Scores for TEST ===
print("F1 Macro (Test):", f1_score(test_labels, test_predicted_classes,
average='macro'))
print("F1 Micro (Test):", f1_score(test_labels, test_predicted_classes,
average='micro'))
print("F1 Weighted (Test):", f1_score(test_labels, test_predicted_classes,
average='weighted'))

# === Confusion Matrices ===
fig, axes = plt.subplots(1, 2, figsize=(14, 5))

cm_train = confusion_matrix(train_labels, train_predicted_classes)
cm_test = confusion_matrix(test_labels, test_predicted_classes)

disp_train = ConfusionMatrixDisplay(confusion_matrix=cm_train,
display_labels=['Negatif', 'Netral', 'Positif'])
disp_test = ConfusionMatrixDisplay(confusion_matrix=cm_test,
display_labels=['Negatif', 'Netral', 'Positif'])

axes[0].set_title('Confusion Matrix - Train')
disp_train.plot(ax=axes[0], cmap='Blues', colorbar=False)

axes[1].set_title('Confusion Matrix - Test')
disp_test.plot(ax=axes[1], cmap='Greens', colorbar=False)

plt.tight_layout()
plt.show()
```

```
# Training and Validation Loss
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
plt.plot(history_model.history['loss'], label='Train Loss')
plt.plot(history_model.history['val_loss'], label='Validation Loss')
plt.title('Loss vs Epochs')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

# Training and Validation Accuracy
plt.subplot(1, 2, 2)
plt.plot(history_model.history['accuracy'], label='Train Accuracy')
plt.plot(history_model.history['val_accuracy'], label='Validation Accuracy')
plt.title('Accuracy vs Epochs')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.tight_layout()
plt.show()
```

BIODATA PENULIS



Penulis dilahirkan di Surabaya pada bulan Juni 2003, merupakan bagian dari 5 bersaudara. Penulis telah menempuh pendidikan wajib di SD Ta'Miriyah Surabaya, SMP Negeri 3 Surabaya, dan SMA Negeri 6 Surabaya. Setelah lulus dari pendidikan wajib, Penulis mengikuti Seleksi Bersama Masuk PTN dan diterima di Departemen Statistika FSAD ITS pada tahun 2021 dan terdaftar dengan NRP 5003211069.

Selama berkuliah di ITS, Penulis aktif di beberapa organisasi dan kegiatan seperti Statistics Computer Course (SCC) sebagai staff, yang kemudian diamanahi menjadi Ketua Divisi. Selain itu, peneliti pernah berkecimpung di bidang UKM Olahraga Billiard menjadi staff dan Pekan Raya Statistika menjadi Wakil Ketua Divisi Event STATION. Selain aktif pada kegiatan kemahasiswaan, penulis aktif mengikuti berbagai kompetisi akademik dan sukses meraih: Insentif PKM Artikel Ilmiah 2024, Pendanaan PKM Pengabdian Masyarakat 2024, Juara 3 Infografis UHO, dan Juara Favorit Infografis SSF UNS. Tahun lalu, penulis dan tim menginisiasi program Keputih Cinta Statistik dalam rangka pengabdian masyarakat untuk membangkitkan penggunaan data di tingkat kelurahan. Saat ini penulis sedang mendalami bidang ilmu praktis data analitik dan sains data dan telah memperoleh berbagai macam pengalaman praktis melalui magang dan studi independen. Penulis mengikuti studi independen di program Bangkit oleh Google pada jalur *machine learning* dan meraih *distinction graduate* dan *dev.cert Tensorflow Developer*, dimana program ini memfokuskan pada pengembangan *machine learning* dan AI. Peneliti juga pernah melakukan magang di PT Telekomunikasi Indonesia, Tbk sebagai *machine learning engineer* dan di PT Pelindo Terminal Petikemas sebagai Analis Pengembangan Operasi. Penulis juga beberapa kali berkesempatan untuk menjadi asisten dosen pada mata kuliah: Basis Data, Pemrograman *Open Source*, dan *Data Warehouse* (ETL dan ELT).

Bagi pembaca yang ingin untuk berdiskusi, bercengkrama, memberikan kritik atau masukan terkait tugas akhir, silakan menghubungi peneliti melalui surel yang ada pada laman portofolio penulis pada ailham4321.github.io. Semoga tugas akhir ini bermanfaat bagi anda.