KLASIFIKASI EMOSI OPINI TWITTER MENGGUNAKAN MODEL *BENCHMARK* INDONLU

TUGAS AKHIR

Diajukan oleh:
FARLA NARA VIANA
NIM. 190705037
Mahasiswa Fakultas Sains dan Teknologi
Program Studi Teknologi Informasi

PRODI TEKNOLOGI INFORMASI
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI AR-RANIRY
BANDA ACEH
2023 M/1445 H

LEMBAR PERSETUJUAN

KLASIFIKASI EMOSI OPINI TWITTER MENGGUNAKAN MODEL *BENCHMARK* INDONLU

TUGAS AKHIR

Diajukan Kepada Fakultas Sains dan Teknologi
Universitas Islam Negeri (UIN) Ar-Raniry Banda Aceh
Sebagai Salah Satu Beban Studi Memperoleh Gelar Sarjana
pada Prodi Teknologi Informasi

Oleh:
FARLA NARA VIANA
NIM. 190705037
Mahasiswa Fakultas Sains dan Teknologi
Program Studi Teknologi Informasi

Disetujui untuk Dimunaqasyahkan Oleh:

Pembimbing I,	Pembimbing II,
Hendri Ahmadian, M.I.M	**Aulia Syarif Aziz, S.Kom., M.Sc**
NIP. 198301042014031002	**NIP. 199305212022031001**

Mengetahui,
Ketua Program Studi Teknologi Informasi

Ima Dwitawati, MBA
NIP. 198210132014032002

LEMBAR PENGESAHAN

KLASIFIKASI EMOSI OPINI TWITTER MENGGUNAKAN MODEL *BENCHMARK* INDONLU

TUGAS AKHIR

Telah Diuji Oleh Panitia Ujian Munaqasah Tugas Akhir
Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh dan Dinyatakan Lulus
Serta Diterima Sebagai Salah Satu Beban Studi Program Sarjana (S-1)
Pada Prodi Teknologi Informasi

Pada Hari/Tanggal: Jumat, 22 Desember 2023
di Darussalam, Banda Aceh
Panitia Ujian Munaqasah Tugas Akhir

Ketua,	Sekretaris,
Hendri Ahmadian, S.Si., M.I.M	**Aulia Syarif Aziz, S.Kom., M.Sc**
NIP. 198301042014031002	**NIP. 199305212022031001**
Penguji I,	Penguji II,
Malahayati, MT	**Baihaqi, M.T**
NIP. 198301272015032003	**NIP. 198802212022031001**

Mengetahui,
Dekan Fakultas Sains dan Teknologi
UIN Ar-Raniry Banda Aceh

Dr. Ir. M. Dirhamsyah, M.T.,IPU
NIP.196210021988111001

LEMBAR PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama	: Farla Nara Viana
NIM	: 190705037
Program Studi	: Teknologi Informasi
Fakultas	: Sains dan Teknologi
Judul Tugas Akhir	: Klasifikasi Emosi Opini Twitter Menggunakan Model *Benchmark* IndoNLU

Dengan ini menyatakan bahwa dalam penulisan tugas akhir ini, saya:

1. Tidak menggunakan ide orang lain tanpa mampu mengembangkan dan mempertanggungjawabkan;
2. Tidak melakukan plagiasi terhadap naskah orang lain;
3. Tidak menggunakan karya orang lain tanpa menyebutkan sumber asli atau tanpa izin pemilik karya;
4. Tidak memanipulasi dan memalsukan data;
5. Mengerjakan sendiri karya ini dan mampu mempertanggungjawab atas karya ini;

Bila kemudian hari ini ada tuntutan dari pihak lain atas karya saya, dan telah melalui pembuktian yang dapat mempertanggungjawabkan dan ternyata memang ditemukan bukti bahwa saya telah melanggar pernyataan ini, maka saya siap dikenakan sanksi berdasarkan aturan yang berlaku di Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh.

Demikian pernyataan ini saya buat dengan sesungguhnya dan tanpa paksaan dari pihak manapun.

Banda Aceh,
Yang Menyatakan,

(Farla Nara Viana)

ABSTRAK

Nama : Farla Nara Viana
NIM : 190705037
Program Studi : Teknologi Informasi

Judul : Klasifikasi Emosi Opini Twitter Menggunakan Model *Benchmark* IndoNLU
Tanggal Sidang : 22 Desember 2023
Jumlah Halaman : 79 Halaman
Pembimbing I : Hendri Ahmadian, M.I.M
Pembimbing II : Aulia Syarif Aziz, S.Kom., M.Sc

Emosi sebagai pengalaman sadar yang melibatkan unsur perasaan, penyesuaian batiniah, dan ekspresi melalui perilaku verbal dan nonverbal. Penggunaan media sosial yang mulanya sebagai tempat bertukar informasi kini bertambah menjadi tempat menyalurkan emosi seseorang secara verbal. Kecenderungan emosi pada suatu topik yang dibahas dapat diketahui dengan dilakukan klasifikasi emosi pada teks. Metode yang digunakan pada penelitian ini adalah dengan menerapkan variasi model IndoBERT yang dikembangkan oleh *benchmark* IndoNLU.

Penelitian ini menggunakan 3000 dataset yang terdiri dari 6 label emosi yaitu *anger, fear, joy, love, sad,* dan *neutral*. Hasil klasifikasi emosi pada opini Twitter menunjukkan tingkat akurasi yang signifikan. Hasil akhir menunjukkan bahwa model IndoBERT variasi *base* p1 mencapai 95.67%, *base* p2 senilai 95.33%, *large* p1 senilai 96.00%, dan *large* p2 senilai 95.33%. Manfaat penelitian ini mencakup pemahaman lebih lanjut tentang performa model *benchmark* IndoNLU dalam konteks klasifikasi emosi pada teks berbahasa Indonesia, serta memberikan acuan bagi penelitian selanjutnya dalam bidang ini.

Kata Kunci: Klasifikasi, IndoNLU, IndoBERT, Akurasi

ABSTRACT

Name	: Farla Nara Viana
Student Number	: 190705037
Department	: Information Technology
Title	:Twitter Opinion Emotion Classification Using IndoNLU Benchmark Model
Date	: 22 December 2023
Number of Pages	: 79 Pages
Supervisor I	: Hendri Ahmadian, M.I.M
Supervisor II	: Bustami, M.Sc

Emotions are conscious experiences that involve elements of feeling, inner adjustments, and expression through verbal and nonverbal behavior. The use of social media, which was originally a place to exchange information, has now become a place to channel one's emotions verbally. The emotional tendency of a topic being discussed can be determined by classifying emotions in the text. The method used in this research is to apply a variation of the IndoBERT model developed by the IndoNLU benchmark.

This research uses 3000 datasets consisting of 6 emotion labels, namely anger, fear, joy, love, sad, and neutral. The results of emotion classification in Twitter opinions show a significant level of accuracy. The final results show that the IndoBERT model base p1 variation reaches 95.67%, base p2 is worth 95.33%, large p1 is worth 96.00%, and large p2 is worth 95.33%. The benefits of this research include further understanding the performance of the IndoNLU benchmark model in the context of emotion classification in Indonesian language texts, as well as providing a reference for further research in this field.

Keywords: Classification, IndoNLU, IndoBERT, Accuracy.

KATA PENGANTAR

Dengan menyebut nama Allah SWT yang Maha Pengasih lagi Maha Penyayang. Puji syukur penulis panjatkan panjatkan atas kehadirat Allah SWT yang telah melimpahkan rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan tugas akhir dengan judul "**Klasifikasi Emosi Opini Twitter Menggunakan Model *Benchmark* IndoNLU**". Shalawat serta salam kepada Rasulullah SAW yang mengantarkan manusia dari zaman kegelapan ke zaman yang terang benderang.

Dalam proses penyusunan tugas akhir ini penulis menyadari bahwa tidak dapat menyelesaikan tugas akhir jika tidak adanya dukungan, bimbingan, motivasi dan bantuan dari berbagai pihak yang telah membantu penulis dalam menyelesaikan tugas akhir ini. Dengan kerendahan hati penulis ingin berterimakasih kepada :

1. Kedua orangtua yang senantiasa memberikan dukungan dan doa kepada saya dalam menyusun tugas akhir ini.
2. Ketua dan Sekretaris Prodi Teknologi Informasi Ibu Ima Dwitawati, M.B.A dan Bapak Khairan AR, M.Kom yang telah memberikan bimbingan dan arahan dalam penyusunan tugas akhir ini.
3. Bapak Hendri Ahmadian, M.I.M selaku Pembimbing I dan Bapak Aulia Syarif Aziz, S.Kom., M.Sc. selaku Pembimbing II yang senantiasa memberikan arahan dan bimbingan dalam penyusunan tugas akhir ini.
4. Ibu Cut Ida Rahmadiana, S.Si selaku *staff* Prodi Teknologi Informasi, yang senantiasa membantu penulis dalam pemberkasan administrasi.
5. Bapak Dr. Ir. M. Dirhamsyah, M.T.,IPU selaku Dekan Fakultas Sains dan Teknologi UIN Ar-Raniry yang telah mendukung dan memberi motivasi untuk kami.
6. Bapak dan Ibu dosen Program Studi Teknologi Informasi yang telah memberikan ilmu kepada penulis dalam bidang Teknologi Informasi.
7. Sahabat dan teman-teman yang selalu memberikan dukungan moral dalam menyelesaikan penyusunan tugas akhir ini.

8. Semua pihak yang telah membantu dalam menyelesaikan proposal tugas akhir yang tidak dapat disebutkan satu persatu.

Penulis menyadari bahwa masih banyak kekurangan dari tugas akhir ini. Oleh karena itu, penulis memohon maaf atas segala kekurangan yang terjadi selama proses penyusunan tugas akhir. Akhir kata, semoga tugas akhir ini dapat memberikan kontribusi yang bermanfaat bagi penulis dan pembaca khususnya.

Banda Aceh, 18 Desember 2023

Penulis,

Farla Nara Viana

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

I.1 Latar belakang

Emosi adalah pengalaman sadar, kompleks dan meliputi unsur perasaan, yang mengikuti keadaan-keadaan psikologis dan mental yang muncul serta penyesuaian batiniah dan mengekspresikan dirinya dalam tingkah laku yang nampak (Fallis, 2013). Emosi biasa diekspresikan secara verbal maupun nonverbal. Emosi verbal berupa emosi yang ditunjukkan secara lisan atau tulisan, sedangkan emosi nonverbal adalah emosi yang ditunjukkan melalui perilaku atau gerak tubuh.

Emosi yang dihasilkan oleh seseorang berupa emosi positif dan emosi negatif. Emosi positif ditandai dengan munculnya perasaan positif sehingga menghasilkan energi positif yang berdampak pada diri sendiri dan orang lain, contohnya seperti bahagia, cinta dan sebagainya. Sebaliknya dengan emosi negatif, perasaan negatif yang dihasilkan dapat berdampak negatif kepada diri sendiri dan orang lain, seperti kebencian, kesedihan dan lainnya.

Seiring berjalannya waktu, seseorang tidak hanya mengekspresikan emosinya secara nyata atau langsung, namun juga memanfaatkan peranan sosial media yang mulanya sebagai tempat bertukar informasi kini bertambah fungsi sebagai media untuk seseorang menyalurkan emosinya. Tidak sedikit orang yang memanfaatkan social media sebagai tempat untuk berbagi keseharian atau mencurahkan isi hati dan perasaan mereka. Faktor yang mempengaruhi hal tersebut adalah pikiran, dimana saat seseorang ingin menampilkan situasi yang sedang dilakukan atau sedang terjadi, maka hal tersebut akan terus dilakukan melalui foto atau video agar mendapatkan respon dari orang lain (Estiyani, 2018).

Dari laporan We Are Social, pengguna sosial media di Indonesia mencapai 167 juta orang pada januari 2023. Hal ini menunjukkan tingkat antusias masyarakat dalam menggunakan sosial media cukup tinggi. Beberapa jenis sosial media yang cukup populer yaitu Whatsapp. Instagram, Facebook, Twitter dan lain-lain. Dari berbagai jenis sosial media tersebut, Twitter dianggap paling tepat untuk dijadikan bahan penelitian.

Pemilihan Twitter sebagai bahan penelitian dikarenakan jumlah pengguna Twitter di Indonesia mencapai 24 juta orang yang menduduki peringkat kelima sebagai platform dengan user terbanyak di dunia. Dibandingkan platform lain, Twitter memiliki fitur *trending* yang membuat para penggunanya mengetahui topik hangat yang tengah diperbincangkan di dunia maya. Fitur ini dapat dimanfaatkan untuk mengumpulkan opini-opini dengan topik yang sama untuk mengetahui kecenderungan emosi pada topik tersebut. Adapun untuk mengetahui kecenderungan emosi tersebut, perlu dilakukan klasifikasi emosi melalui teks opini dari Twitter.

Melakukan klasifikasi emosi melalui tulisan akan lebih sulit daripada secara lisan. Hal ini dikarenakan tidak ada ekspresi atau nada suara yang dihasilkan. Klasifikasi emosi pada teks berguna untuk mengetahui keadaan emosi dari suatu topik. Klasifikasi merupakan proses pengelompokan beberapa data sesuai kriteria. Penelitian tentang klasifikasi emosi pada teks telah diuji dengan beberapa metode, seperti LSTM (*Long Short Term Memory*), BiLSTM (*Bi-Directional Long Short Term Memory*), Word2Vec (*Word to Vector*), SVM (*Support Vector Machine*), dan CNN (*Support Vector Machine*). Dari penelitian-penelitian tersebut dihasilkan akurasi dari 70% hingga 90% (Widianto, 2022).

Pada penelitian (Widianto, 2022) klasifikasi emosi pada teks menggunakan metode *deep learning*, penelitian tersebut menggunakan metode *Bidirectional Encoder Representation from Transformer* (BERT). Pengujian dilakukan dengan menggunakan dua model BERT yaitu model BERT *Uncased* dan model *benchmark* IndoNLU. Secara singkat BERT *Uncased* menganalisis teks berbahasa inggris sedangkan model *benchmark* IndoNLU menganalisis teks berbahasa Indonesia. Dataset yang digunakan berasal dari Twitter, sebanyak 2515 tweets dikumpulkan dengan taknik *scraping* dan dilakukan klasifikasi emosi menjadi 9 jenis emosi yaitu, marah, bahagia, sedih, tertarik, netral, percaya, jijik, kaget, dan takut. Hasil akurasi yang didapatkan sebesar 90% pada model BERT *Uncased* dan 81% pada model *benchmark* IndoNLU.

Model BERT merupakan metode *state-of-the-art* dalam pembangunan language model dengan pendekatan *deep learning*, sedangkan model *benchmark*

IndoNLU merupakan salah satu model monolingual BERT untuk bahasa Indonesia (Hadiyan, dkk. 2021).

Penelitian ini akan menggunakan variasi dari model IndoBERT yang dikembangkan oleh *benchmark* IndoNLU sebagai metode uji pada klasifikasi emosi opini Twitter. Variasi yang akan digunakan yaitu IndoBERT *base* p1, IndoBERT *base* p2, IndoBERT *Large* p1, dan IndoBERT *large* p2. Hasil akhir yang dituju adalah melihat tingkat akurasi model IndoBERT yang dikembangkan *benchmark* IndoNLU terhadap klasifikasi emosi dengan menerapkan 6 label emosi.

I.2 Rumusan Masalah

Berdasarkan uraian dari latar belakang diatas, diambil rumusan masalah sebagai berikut:

1. Bagaimana proses klasifikasi emosi opini Twitter menggunakan variasi model IndoBERT *benchmark* IndoNLU?
2. Bagaimana tingkat akurasi klasifikasi emosi opini Twitter menggunakan variasi model IndoBERT yang dikembangkan oleh *benchmark* IndoNLU?
3. Bagaimana penggunaan *confussion matrix* pada evaluasi variasi model IndoBERT yang dikembangkan oleh *benchmark* IndoNLU?

I.3 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah:

1. Mengetahui proses klasifikasi emosi opini Twitter menggunakan variasi model IndoBERT *benchmark* IndoNLU.
2. Mengetahui tingkat akurasi klasifikasi emosi opini Twitter menggunakan variasi model IndoBERT yang dikembangkan oleh *benchmark* IndoNLU.
3. Mengetahui penggunaan *confussion matrix* pada evaluasi variasi model IndoBERT yang dikembangkan oleh *benchmark* IndoNLU.

I.4 Batasan masalah

Batasan-batasan masalah pada penelitian ini sebagai berikut:

1. Menggunakan bahasa pemograman *Python*.
2. Klasifikasi emosi perkalimat.
3. Pengklasifikasian berdasarkan 6 emosi, yaitu: *anger, fear, joy, love, sad,* dan *neutral.*

4. *Dataset* berupa opini Twitter yang diambil dari github.com.
5. Data yang digunakan berbahasa Indonesia.

I.5 Manfaat penelitian

Manfaat dari penelitian ini adalah:

1. Mengetahui performa model IndoBERT yang dikembangkan oleh *benchmark* IndoNLU dalam melakukan klasifikasi emosi pada teks berbahasa Indonesia.
2. Hasil penelitian ini dapat menjadi bahan acuan dan berpandingan untuk penelitian-penelitian selanjutnya.

BAB II
LANDASAN TEORI

II.1 Penelitian Terdahulu

Penelitian terdahulu digunakan sebagai acuan penulis dalam meneliti dan menghindari kesamaan hasil dari penelitian. Selain itu, penelitian terdahulu merupakan sumber inspirasi bagi penelitian lain untuk menghasilkan suatu penemuan yang baru, atau diperbarui dengan berbagai metode dan cara yang ilmiah.

Pertama, penelitian yang dilakukan oleh Alan Tusa Bagus Widianto pada tahun 2022 dengan judul "Klasifikasi Emosi pada Teks Menggunakan Metode *Deep learning*". Metode yang diterapkan pada penelitian ini adalah metode klasifikasi menggunakan BERT (*Bidirectional Encoder Representation From Transformers*). Data yang diuji sebanyak 2515 data yang diklasifikasikan kedalam sembilan bagian yaitu marah, takut, sedih, netral, bahagia, tertarik, percaya, kaget, dan jijik. Dalam penelitian ini, *hyperparameters* yang digunakan pada kedua model adalah 10 epoch dan validasi 0.1. Khusus untuk model *benchmark* IndoNLU menggunakan *hyperparameters* diantaranya *learning rate* 5e-05, *epsilon* 1e-08, *decay* 0.01, dan *clipnorm* 1.0. Klasifikasi emosi menggunakan BERT menghasilkan nilai akurasi sebesar 90% pada BERT *Uncased* dan menghasilkan akurasi 81% pada model *benchmark* IndoNLU (Widianto, 2022).

Kedua, penelitian yang dilakukan oleh Alia Sri Rezki pada 2021 dengan judul "Klasifikasi Emosi Pada Twitter Dengan Menggunakan Metode *K-Nearest Neighbor* (KNN)". Penelitian ini mengklasifikasikan data menjadi 8 kategori emosi yaitu: antisipasi, marah, jijik, sedih, terkejut, takut, percaya, dan gembira. Setiap kategori emosi menggunakan 300 *tweets* sebagai *dataset* dengan jumlah keseluruhan *dataset* sebanyak 2400 *tweets*. Metode yang digunakan pada penelitian ini adalah *K-Nearest Neighbor*. Hasil akurasi yang diperoleh dari proses pengujian menggunakan *confussion matrix* yaitu nilai akurasi tertinggi sebesar 91,04%, *recall* 91,04%, *precision* 91,44% dan f1-*score* 91,02% pada model *dataset* 80%:20% dengan nilai k = 17 dan *threshold* bernilai 5 (Rezki, 2021).

Ketiga, penelitian dengan judul "Analisis Sentimen Twitter Pada Isu *Mental Health* Dengan Algoritma Klasifikasi *Naive Bayes*" oleh Karina Aulia dan Lia

Amelia pada 2020. Data yang digunakan sebanyak 77.458 *tweets*. Penelitian ini melakukan klasifikasi terhadap 6 kategori emosi yaitu: mengejutkan, kesedihan, kegembiraan, takut, jijik, dan marah. Hasil klasifikasi yang diperoleh lebih dari 50.000 *tweets* tidak diketahui jenis emosinya. Sedangkan pada analisis sentimen, sentimen positif berada di jumlah tertinggi dengan perolehan data lebih dari 40.000 *tweets*, selanjutnya sentiment negatif dengan perolehan data lebih dari 25.000 *tweets* dan terakhir netral lebih dari 10.000 *tweets* (Aulia & Amelia, 2020).

Kempat, penelitian oleh Azward Nurfauzan dan Warih Maharani dengan judul "Klasifikasi Emosi Pada Pengguna Twitter Menggunakan Metode Klasifikasi *Decision Tree*" pada 2021. Data yang digunakan sebanyak 4401 *tweets* dan diklasifikasikan kedalam 5 jenis emosi yaitu: marah, cinta, sedih, senang dan takut. Penelitian ini menggunakan metode *decision tree* untuk melakukan klasifikasi emosi pada tweet dan menggunakan metode TF-IDF untuk melakukan pembobotan kata. Hasil akurasi yang diperoleh pada penelitian sebesar 55% dengan rata-rata nilai *precision* 54%, *recall* 53%, dan f1-*score* 53% (Nurfauzan & Maharani, 2021). Ringkasan mengenai penelitian terdahulu yang dugunakan pada penelitian ini dapat dilihat pada tabel 2.1.

Tabel 2.1 Perbandingan Penelitian Terdahulu

No	Nama Peneliti	Judul Penelitian	Hasil Penelitian
1	Alan Tusa Bagus Widianto (2022)	Klasifikasi Emosi pada Teks Menggunakan Metode *Deep learning*	Klasifikasi emosi menggunakan BERT menghasilkan nilai akurasi sebesar 90% pada BERT Uncased dan menghasilkan akurasi 81% pada model *benchmark* IndoNLU
2	Alia Sri Rezki (2021)	Klasifikasi Emosi Pada Twitter dengan Menggunakan Metode *K-Nearest Neighbor* (KNN).	Hasil akurasi dari proses pengujian menggunakan *confusion matrix*

			memperoleh akurasi tertinggi sebesar 91,04%, *recall* 91,04%, *precision* 91,44% dan *f1-score* 91,02% pada model *dataset* 80%:20% dengan nilai k = 17 dan *threshold* = 5.
3	Karina Aulia dan Lia Amelia (2020)	Analisis Sentimen Twitter Pada Isu *Mental Health* Dengan Algoritma Klasifikasi *Naive Bayes.*	Hasil klasifikasi emosi didominasi oleh emosi yang tidak diketahui sebanyak lebih dari 50.000 *tweets*. Sedangkan pada analisis sentimen menunjukkan *tweets* dengan sentimen positif mendominasi dengan perolehan data lebih dari 40.000 *tweets*, disusul dengan sentimen negatif lebih dari 25.000 *tweet*s dan netral lebih dari 10.000 *tweets*.
4	Azward Nurfauzan dan Warih Maharani (2021)	Klasifikasi Emosi Pada Pengguna Twitter Menggunakan Metode Klasifikasi *Decision Tree*	Pada penelitian ini mendapatkan hasil dengan tingkat akurasi terbaik sebesar 55% dengan rata-rata nilai

			precision 54%, *recall* 53%, dan *f1-score* 53%.

II.2 Twitter

Twitter adalah *platform* media sosial yang memungkinkan pengguna untuk berbagi pesan singkat dengan panjang maksimal 280 karakter, yang dikenal sebagai "*tweets*". Twitter didirikan oleh Jack Dorsey, Noah Glass, Biz Stone dan Evan Williams di San Francisco pada 19 April 2007.

Twitter menjadi salah satu media sosial dengan jumlah pengguna sebanyak 556 juta dari seluruh dunia pada Januari 2023 (berdasarkan laporan We Are Social dan Hootsuite). Jumlah tersebut meningkat 27,4% dibandingkan pada periode yang sama tahun sebelumnya. Hal tersebut disebabkan penggunaannya yang mudah untuk saling bertukar informasi sehingga setiap individu dari berbagai penjuru dunia dapat saling terhubung. Misinya adalah untuk memberi kesempatan bagi setiap orang untuk dapat saling menciptakan berbagai ide dan gagasan baru, serta informasi secara langsung tanpa hambatan (Azeharie & Kusuma, 2014).

II.3 Analisis emosi

Sistem survei analisis emosi dilakukan pertama kali pada tahun 1966, ini bisa dikatakan fase awal dari analisis emosi pada teks. Robert Plutchik membagi emosi menjadi delapan kategori utama. Separuhnya emosi positif dan separuhnya lagi emosi negatif. Setiap emosi memiliki sub-kelompok. Sebagai contoh pada emosi bahagia dapat menggambarkan keadaan kegembiraan atau kesenangan tergantung pada koteks yang diberikan. Robert Plutchik juga membuat roda emosi sebagai ilustrasi hubungan emosi satu dengan lainnya (Widianto, 2022).

Salah satu tujuan dilakukan analisis emosi adalah untuk mengidentifikasi kecenderungan emosi pada suatu topik. Presentase yang dihasilkan dapat dijadikan acuan dalam pengambilan langkah selanjutnya seperti pemilihan solusi atau perubahan rencana awal.

II.4 Identifikasi Emosi Pada Teks

Emosi merupakan salah satu aspek dengan pengaruh besar terhadap sikap manusia dengan lingkungan. Emosi sendiri terbagi menjadi dua yaitu emosi positif dan negatif. Untuk mengidentifikasinya perlu mengetahui beberapa aspek pemicunya. Adanya media sosial diera digital ini, membuat seseorang sering berkomunikasi di dunia maya. Identifikasi emosi dilakukan dengan mengetahui pemicu yang muncul dan mengetahui respon seseorang tersebut. Emosi sendiri dibagi menjadi dua yaitu verbal dan non-verbal, emosi verbal dapat diartikan emosi langsung atau menggunakan lisan maupun tulisan sebagai medianya. Sedangkan non-verbal biasanya menggunakan bahasa tubuh untuk mengekspresikannya.

Emosi verbal khususnya pada tulisan atau teks perlu dilakukan identifikasi lebih lanjut untuk mengetahui emosi yang benar-benar terjadi agar tidak terjadi kesalahan identifikasi karena pada tulisan bisa bermaksud multimakna di dalamnya. Teks sendiri adalah media utama dalam komunikasi pada sosial media, contoh sosial media yang hampir semua proses komunikasinya menggunakan teks adalah Twitter. Twitter sendiri selain menjadi media komunikasi, pengguna juga dapat membagikan informasi, opini, curahan hati, dan berdiskusi. Identifikasi emosi pada teks dilakukan dengan menggunakan pendekatan klasifikasi emosi (Widianto, 2022).

II.5 Klasifikasi Teks

Klasifikasi teks adalah proses pemberian *tag* atau kategori ke teks menurut isinya. Proses pengklasifikasian dapat dilakukan secara manual dan otomatis. Secara manual, manusia akan memahami teks kemudian memberi *tag*. Pengklasifikasian secara manual tentu lebih sulit dan membutuhkan waktu yang lama sehingga tidak efesien. Sedangkan klasifikasi secara otomatis dapat dilakukan dengan memanfaatkan *machine learning* atau teknik lainnya sehingga waktu yang digunakan lebih singkat. Pengklasifikasian secara otomatis dikelompokkan menjadi tiga sistem yaitu *rule-based system, machine learning based system* dan *hybrid system*.

II.5.1 *Rule Based System*

Rule Based System adalah sebuah cara untuk mengaplikasikan pengetahuan seorang pakar ke dalam sebuah sistem otomatis. *Rule based system* biasa menggunakan aturan dalam bentuk sekumpulan *if-then*. Sekelompok aturan ini kemudian digunakan untuk menganalisa data dalam sistem pakar yang diharapkan dapat bekerja seperti seorang pakar atau setidaknya mendekati.

II.5.2 *Machine learning based system*

Machine learning based system adalah sistem perangkat lunak yang mencakup satu atau lebih komponen yang mempelajari cara melakukan tugas dari kumpulan data yang diberikan. Secara umum, *machine learning* menawarkan teknik statistik untuk mempelajari fungsi (pola) kompleks dari kumpulan data pelatihan yang disediakan, memungkinkan untuk menerapkan fungsi yang dipelajari pada titik data baru yang sebelumnya tidak terlihat. Kemampuan untuk menggeneralisasi ini sering digunakan untuk membuat prediksi tentang sifat yang tidak diketahui dari data yang diamati (Riccio et al., 2020).

II.5.3 *Hybrid system*

Hybrid system adalah pendekatan yang menggabungkan pengklasifikasi dasar yang telah dilatih menggunakan *machine learning* dengan *rule-based system*. Tujuan dari penggabungan ini adalah untuk meningkatkan hasil yang dicapai. Dalam *hybrid system*, pengklasifikasi dasar menggunakan metode *machine learning* untuk memprediksi dan mengklasifikasikan data, sementara *rule-based system* digunakan sebagai mekanisme tambahan untuk memperbaiki atau mengatasi situasi yang mungkin tidak tercakup dengan baik oleh pengklasifikasi dasar. Dengan adanya *rule-based system*, *hybrid system* dapat dengan mudah disesuaikan dengan menambahkan aturan khusus untuk penanganan kasus yang spesifik dan belum diakomodasi dengan baik oleh pengklasifikasi dasar.

II.6 *Deep learning*

Deep learning adalah jenis metode *machine learning* berdasarkan representasi data pembelajaran. Teknik *deep learning* adalah bagian dari *neural network* yang terkenal ketika struktur *multilayer*, banyak dipakai karena dapat menangani banyak masalah sekaligus dan memberikan solusi yang unik (Amigo, 2021). *Deep learning* dibagi menjadi 3 metode, diantaranya:

a. Supervised

Supervised adalah sebuah sistem yang memiliki variabel x sebagai *input* dan variabel y sebagai *output*, dengan persamaan $Y = f$, sistem ini memetakan variabel *input* ke variabel *output* menggunakan fungsi pemetaan (x). Tujuannya adalah untuk memperkirakan secara akurat fungsi pemetaan sehingga dapat memprediksi *output* dari variabel *input* baru.

b. Semi supervised

Algoritma *semi supervised* berada diantara algoritma *supervised* dan *unsupervised*. Cara kerjanya adalah menemukan dan menganalisis struktur dalam variabel *input*. Setelah itu, akan dibuat prediksi terbaik dari data yang tidak berlabel dan memberikan data tersebut ke algoritma *supervised* sebagai data latih untuk membuat prediksi baru.

c. Unsupervised

Algoritma yang tidak memiliki variabel *output* yang sesuai, hanya variabel *input*. Memodelkan struktur atau distribusi yang dapat memahami data dengan lebih baik adalah tujuan dari metode *unsupervised*. Untuk menemukan dan menunjukkan struktur data yang menarik, algoritma akan dibiarkan sendiri (Widianto, 2022).

II.7 *Natural Language Processing* (NLP)

Natural language processing (NLP) adalah kumpulan teknik komputasi yang didasarkan pada teori linguistik untuk menganalisis dan merepresentasikan teks yang dibuat oleh manusia. NLP berfokus pada pemahaman bahasa alami dan bertujuan mencapai pemrosesan bahasa yang mirip dengan cara manusia melalui berbagai tingkat analisis linguistik. Tujuan utama NLP adalah untuk digunakan dalam berbagai tugas atau aplikasi yang melibatkan bahasa manusia.

Dalam konteks tersebut, tujuan dari sistem NLP adalah untuk mewakili arti dan maksud yang sebenarnya dari permintaan pengguna. Sistem NLP bertujuan untuk memahami permintaan yang diekspresikan secara alami dalam bahasa sehari-hari, seolah-olah pengguna sedang berbicara dengan seorang pustakawan referensi. Dengan menggunakan teknik-teknik NLP, sistem dapat menganalisis dan menginterpretasi permintaan pengguna dengan akurasi dan memprosesnya untuk memberikan respons yang sesuai (D. Lizzy, 2001).

II.7.1 Sintak (*Syntax*)

Sintak dapat dipahami sebagai susunan kata dalam sebuah kalimat sehingga kalimat tersebut dapat dimengerti dan benar secara tata bahasa. Analisis syntax digunakan oleh NLP untuk menerapkan aturan tata bahasa pada berbagai kata yang ditemukan. Ada beberapa teknik syntax yang bisa digunakan:

- *Lemmatization*, yaitu mengubah kata berbeda namun bermakna sama menjadi satu bentuk untuk mempermudah komputer dalam menganalisis. Contoh, kata "Terbaik" atau "Lebih baik" dapat dikelompokkan dalam kata "Baik".
- Segmentasi morfologis, merupakan proses membagi kata menjadi morfem-morfem yang membentuk kata tersebut. Ini membantu memahami struktur kata dan identifikasi akar kata.
- Segmentasi kata, merupakan proses membagi teks berkelanjutan menjadi unit-unit kata yang berbeda. Ini penting dalam analisis dan pemrosesan bahasa alami.
- Penandaan *part-of-speech*, merupakan proses penandaan kata pada teks dalam suatu kelas kata tertentu berdasarkan definisi dan maknanya.
- *Parsing*, merupakan proses penentuan struktur sebuah kalimat berdasarkan aturan tata bahasa (grammar) dan kamus kata (lexicon) yang spesifik.
- Pemutusan kalimat, merupakan proses menempatkan batas kalimat pada teks yang lebih panjang.
- *Stemming*, merupakan proses mencari kata dasar dari sebuah kata seperti menghilangkan imbuhan pada awal dan akhir kata.

II.7.2 Semantik (*Semantics*)

Semantik adalah ilmu yang mempelajari makna bahasa. Dalam konteks pemrosesan bahasa alami, semantik menjadi aspek yang paling sulit dianalisis dan masih belum sepenuhnya dipelajari dengan baik. Analisis semantik melibatkan penggunaan algoritme komputer untuk memahami makna dan interpretasi kata-kata, serta memahami struktur kalimat secara keseluruhan. Melalui penerapan teknik komputasional, tujuan analisis semantik adalah untuk mengekstraksi dan memahami makna dalam teks yang kompleks, serta mengenali cara kalimat-kalimat tersebut dibangun secara sintaksis dan semantis. Walaupun telah ada kemajuan yang signifikan dalam bidang ini, tantangan masih ada dalam upaya memperoleh

pemahaman yang lebih mendalam tentang bahasa manusia dan menerapkannya dalam sistem pemrosesan bahasa alami. Berikut beberapa teknik semantik yang biasa digunakan dalam NLP:

- *Named Entity Recognition* (NER) adalah teknik dalam pemrosesan bahasa alami untuk mengidentifikasi dan mengkategorikan entitas bernama dalam teks, seperti nama orang dan nama tempat. NER membantu dalam pemahaman teks dan memungkinkan pengolahan lebih lanjut terhadap entitas tersebut. Contohnya, "John tinggal di New York" akan diidentifikasi sebagai entitas bernama orang dan tempat oleh NER.
- Disambiguasi arti kata adalah proses untuk memberikan makna yang tepat pada suatu kata berdasarkan konteksnya. Hal ini penting untuk memahami makna kata yang ambigu dalam kalimat.
- *Natural language generation*, melibatkan penggunaan database untuk mendapatkan maksud semantik dari teks dan mengubahnya menjadi bahasa manusia.

II.8 *Bidirectional Encoder Representation From Transformers* (BERT)

Bidirectional encoder representation from transformers (BERT) (Luo & Wang, 2019) merupakan teknik yang berbasiskan jaringan saraf untuk *pre-training natural language*. BERT dirancang untuk memahami bahasa pada kalimat yang ambigu menggunakan teks disekitarnya untuk membangun konteks yang lebih jelas.

BERT mengandalkan transformer (mekanisme untuk mempelajari konteks kalimat dengan mempelajari hubungan kontekstual kata-kata dalam teks). Transformer dapat belajar dan mengubah pemahaman yang diperoleh dari mekanisme *self-attention*. Mekanisme *self-attention* adalah cara transformator memodifikasi kata terkait dan diubah oleh mekanisme tersebut. Transformer terdiri dari dua mekanisme *encoder* dan *decoder*.

II.8.1 *Encoder*

Encoder digunakan untuk membaca data input teks. *Encoder* terdiri dari tumpukan N = 6 lapisan identik. Setiap lapisan memiliki dua sublapisan, lapisan self-attention dan jaringan saraf feedforward. Dengan lapisan *self-attention*, *encoder* dapat membantu node yang tidak hanya fokus pada kata yang

divisualisasikan, tetapi juga mendapatkan konteks dari kata tersebut. Setiap posisi di *encoder* dapat memproses semua posisi lapisan sebelumnya di *encoder*.

II.8.2 *Decoder*

Decoder berfungsi untuk menghasilkan urutan keluaran yang diprediksi. *Decoder* juga mencakup tumpukan N = 6 lapisan yang dapat diidentifikasi. Setiap lapisan terdiri dari dua sublapisan yang sama dengan lapisan *encoder*. Dengan *attention layer* tambahan di antara keduanya untuk membantu node saat ini mengakses konten utama yang diinginkan dengan melakukan perhatian *multi-head* pada *output encoder*. Seperti pada *encoder*, lapisan *self-attention* di *decoder* memungkinkan setiap posisi di *decoder* untuk menangani semua posisi sebelumnya dan saat ini.

II.9 IndoNLU (*Indonesian Natural Language Understanding*)

Indonesian natural language understanding (IndoNLU) adalah sebuah pustaka (library) *python* yang dikembangkan untuk pemrosesan bahasa alami (*Natural Language Processing*/NLP) dalam bahasa Indonesia. IndoNLU diusulkan atas permasalahan kelangkaan data bahasa Indonesia. Meskipun sejumlah besar data Indonesia tersedia melalui internet, kemajuan penelitian NLP dalam bahasa Indonesia bergerak lambat. Hal ini terjadi karena kumpulan data yang tersedia tersebar dengan kurangnya dokumentasi dan keterlibatan komunitas yang minim. Selain itu, banyak penelitian yang ada di NLP Indonesia tidak memberikan kode dan test split, sehingga tidak memungkinkan untuk mereproduksi hasil (Wilie et al., 2020).

IndoNLU memiliki 12 tugas untuk pemahaman bahasa alami Indonesia yang dibagi menjadi empat kategori : *single-sentence classification* (klasifikasi kalimat tunggal), *single-sentence sequencetagging* (penandaan urutan kalimat tungga), *sentence-pair classification* (klasifikasi pasangan kalimat), dan *sentence-pair sequence labeling* (pelabelan urutan pasangan kalimat).

II.10 *Confusion Matrix*

Confussion matrix adalah tabel yang merangkum kinerja model. Penggunaan *confussion matrix* dapat mengidentifikasi kelemahan dalam oprasi

algoritma. Ini digunakan untuk mengevaluasi evektivitas model klasifikasi. Dengan menggunakan *confussion matrix* peneliti dapat mengetahui *error* pada operasi algoritma yang dijalankan. Terdapat 4 metrik di dalam *confusion matrix*, yaitu:

a. TP adalah *true positive* yaitu sebuah data dengan kondisi aktual yang mampu diprediksi dengan benar.
b. TN adalah *true negatif* adalah data negatif yang diprediksi dengan benar.
c. FP adalah *false positive* yaitu sebuah data positif yang diprediksi tidak sesuai.
d. FN adalah *false negatif* yaitu sebuah data negatif yang diprediksi tidak sesuai.

Bentuk dasar *confussion matrix* dapat dilihat pada tabel 2.2.

Tabel 2.2 *Confussion Matrix*

		Predicted Class	
		Positive	*Negatif*
Actual Class	*Positive*	TP	FN
	Negatif	FP	TN

Nilai akurasi model dapat dihitung dengan memanfaatkan hasil dari confussion matrix. Rumus untuk menghitung akurasi model menggunakan confussion matrix dapat dilihat pada rumus 2.1.

$$Accuracy = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \qquad 2.1$$

Pada klasifikasi *multiclass, matrix* yang ditentukan untuk klasifikasi tidak berlaku secara penuh. *Confussion matrix multiclass* memiliki dimensi $N \times N$ dimana N adalah jumlah label kelas yang berbeda $C0$, $C1$, … , CN oleh karena itu karakterisasi TP, TN, FP, dan FN tidak berlaku dalam kasus ini. Dengan demikian dimungkinkan untuk memberikan pengukuran confussion matrix penuh tergantung pada parameter. Tabel *confussion matrix multiclass* dapat dilihat pada tabel 2.3 dibawah ini.

	Predicted Class			
	C_1	C_2	...	C_N

Actual Class	C_1	$C_{1.1}$	*FP*	...	$C_{1.N}$
	C_2	*FN*	*TP*	...	*FN*
	...	...	...	...	...
	C_N	$C_{N.1}$	*FP*	...	$C_{N.N}$

Rumus yang digunakan untuk menghitung nilai akurasi model berdasarkan *confussion matrix multiclass* dapat dilihat pada rumus 3.1.

$$Accuracy = \frac{\sum_{i=1}^{N} TP(Ci)}{\sum_{i=1}^{N}\sum_{j=1}^{N} Ci,j} \quad 3.1$$

II.11 *Python*

Python adalah bahasa pemrograman interpretatif multiguna yang didesain dengan fokus pada keterbacaan kode. Bahasa ini menggabungkan kapabilitas dan kemampuan yang luas dengan sintaksis yang sangat jelas. Python juga dilengkapi dengan pustaka standar yang besar serta komprehensif, yang memungkinkan pengembang untuk dengan mudah mengakses berbagai modul dan paket yang siap digunakan. Dengan filosofi perancangan yang mengedepankan keterbacaan, Python telah menjadi bahasa pemrograman yang populer dan banyak digunakan di berbagai bidang pengembangan perangkat lunak.

Python adalah bahasa pemrograman yang mendukung beberapa paradigma, termasuk pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional. *Python* juga dikenal sebagai bahasa pemrograman dinamis yang dilengkapi dengan manajemen memori otomatis. Meskipun umumnya digunakan sebagai bahasa script, *Python* digunakan secara luas dalam berbagai konteks pengembangan perangkat lunak. *Python* dapat digunakan di berbagai platform sistem operasi dan memiliki fleksibilitas yang tinggi dalam memenuhi kebutuhan pengembangan perangkat lunak.

Saat ini kode *python* dapat dijalankan di berbagai platform sistem operasi, beberapa diantaranya adalah Linux/Unix, Windows, Mac OS X, Java Virtual Machine, Amiga, Palm, Symbian (untuk produk-produk Nokia) (Syahrudin & Kurniawan, 2018).

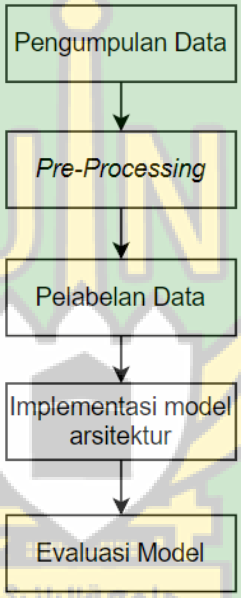
II.12 *Google Colaboratory*

Google colaboratory atau *google colab* adalah sebuah *Integrated Development Environment* (IDE) untuk pemrograman *python* dimana pemrosesan akan dilakukan oleh server *google* yang memiliki perangkat keras dengan performa yang tinggi. Dari sisi perangkat lunak, *google colab* telah menyediakan hampir sebagian besar pustaka (*library*) yang dibutuhkan. Bahkan seluruh versi disediakan, misalnya TensorFlow versi 1.x maupun 2.x tersedia, begitu juga versi *ython* yang mulai dari versi 2.x hingga 3.x. Dari sisi perangkat keras, *google colab* menyediakan layanan berupa media penyimpan yang terintegrasi dengan g*oogle drive*, prosesor yang berupa *central processing unit* (CPU), *graphics processing unit* (GPU), dan *tensor processing unit* (TPU), serta *random-access memory* (RAM). Dengan jaminan kemampuan servernya yang stabil hampir keseluruhan pemrosesan tidak menemukan kendala dengan *google colab* selama koneksi jaringan internet lancar (Gelar Guntara, 2023).

BAB III
METODOLOGI PENELITIAN

III.1 Tahapan penelitian

Model yang akan digunakan dalam klasifikasi emosi ini adalah *model benchmark* IndoNLU. Hasil akhir yang ingin dicapai yaitu mengetahui tingkat akurasi dan performa model *benchmark* IndoNLU dalam melakukan klasifikasi emosi pada teks. Berikut tahapan penelitian yang akan dilakukan dapat dilihat pada gambar 3.1.



Gambar 3.1 Tahapan Penelitian

III.2 Pengumpulan data

Dataset yang akan diuji pada penelitian ini berupa tweet bahasa Indonesia yang diambil dari github[1]. Github menyediakan dataset yang dibentuk dari tweet bahasa Indonesia dan sudah diklasifikasikan kedalam enam jenis emosi yaitu *anger, fear, joy, love, sad,* and *neutral*.

[1] https://github.com/Ricco48/Emotion-Dataset-from-Indonesian-Public-Opinion

Total data yang digunakan pada penelitian ini berjumlah 7.080 tweet yang terdiri dari label *anger* 1.130 data, *fear* 911 data, *joy* 1.275 data, *love* 760 data, *sad* 1.003 data, dan *neutral* 2.001 data.

III.3 *Pre-processing*

Pre-processing (pra-pemrosesan) adalah tahap dalam analisis data di mana data mentah dipersiapkan dan dibersihkan sebelum diterapkan pada model atau algoritma tertentu. Tujuannya adalah untuk meningkatkan kualitas data, menghilangkan noise atau gangguan, mengurangi dimensi data, dan mempersiapkan data agar sesuai dengan kebutuhan analisis yang akan dilakukan.

Tahapan *pre-processing* yang dapat dilakukan dalam pengolahan teks menggunakan IndoNLU meliputi:

- Tokenisasi: Memecah teks menjadi unit-unit yang lebih kecil, seperti kata atau frasa.
- Normalisasi: Melakukan normalisasi teks untuk mengubah teks menjadi format yang konsisten seperti mengubah huruf menjadi huruf kecil, mengubah bahasa slang, menghilangkan tanda baca atau karakter khusus, atau mengganti bentuk singkatan atau akronim dengan bentuk lengkapnya.
- Penghapusan stopword: Stopword adalah kata-kata umum yang sering muncul dalam teks, namun memiliki kontribusi yang relatif kecil dalam pemahaman konten atau makna teks.
- *Stemming*: *Stemming* adalah teknik untuk mengubah kata-kata dalam teks menjadi bentuk dasar atau kata dasar.
- Pembersihan teks tambahan: Langkah-langkah tambahan dapat mencakup penghapusan karakter atau simbol yang tidak diinginkan, penghilangan URL, atau penghilangan entitas khusus seperti tag HTML.

Data yang diperoleh dari github sudah melalui sebagian tahap pre-processing seperti menghilangkan duplikasi teks, mengubah teks menjadi huruf kecil, dan pembersihan teks tambahan (penghapusan karakter atau simbol, hastag, URL dan emotikon), sehingga hanya perlu dilakukan sebagian tahap pre-processing lainnya seperti tokenisasi, penghapusan stopword, mengubah bahasa slang, dan stemming.

III.4 Pelabelan data

Pelabelan data dilakukan pada teks yang sudah melalui tahap pre-processing. Pada penelitian ini dataset dilabelkan kedalam enam emosi yaitu *anger, fear, joy, love, sad, neutral*.

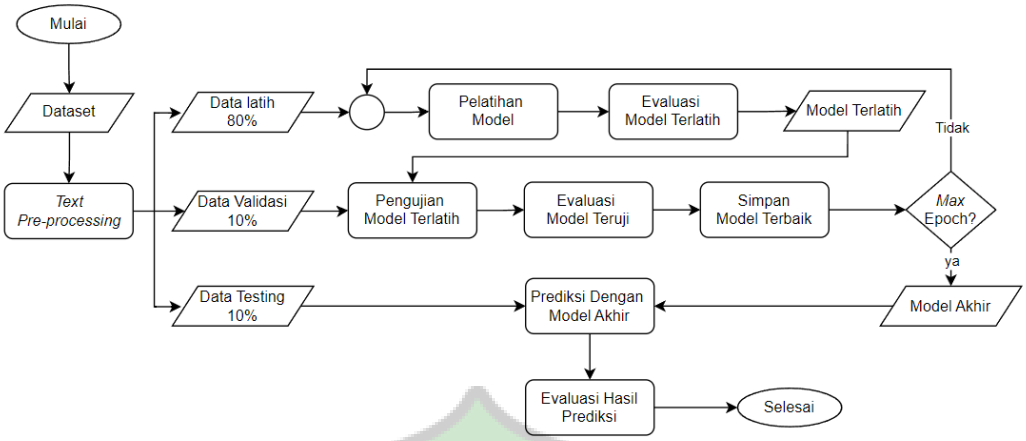
Dataset yang diperoleh dari *github* sudah memiliki label pada setiap teksnya. Contoh kalimat yang sudah ditentukan emosinya pada dataset dapat dilihat pada tabel 3.1.

Tabel 3.1 Contoh Pelabelan Data

No	Contoh data	Label
1	Capee lagi puasa ada aja yg dibikin marah nya tu	*Anger*
2	Bisa gak sih langsung tanggal 11 aja gausah ada tanggal 10 gua takut banget mana 2 hari lagi	*Fear*
3	Jangan lupa sarapann semangat buat hari inii ayang semoga harimu menyenangkan !!	*Joy*
4	Suka banget sama mereka berdua	*Love*
5	Sedih emg kalo ditinggal temen ngebucin, sedih ga ada waktu main sm temen, sedih krna pengen ngebucin jg	*Sadness*
6	Yah gimana sih yang yaudah susul aku dulu kita sarapan bubur di tempat biasa	*Neutral*

III.5 Implementasi Metode dan Arsitektur

Tahap implementasi metode dan arsitektur merupakan tahapan untuk mentransformasikan dan merealisasikan rumusan konsep penelitian pada sistem yang telah dibangun. Implementasi metode dan arsitektur terdiri dari alur sistem yang dikembangkan pada penelitian. Berikut alur sistem dalam proses klasifikasi emosi opini Twitter menggunakan model *benchmark* IndoNLU dapat dilihat pada gambar 3.2.



Gambar 3.2 Alur Implementasi Sistem

Rincian alur implementasi sistem adalah dataset yang digunakan dalam penelitian ini merupakan opini para pengguna Twitter yang diambil dari *github*. Selanjutnya, dilakukan tahap pre-processing sebagai tahap membersihkan teks pada data lalu diberi label pada setiap teks.

Kemudian dilakukan *splitting* atau pemisahan data dengan perbandingan 80% untuk data latih, 10% untuk data validasi, dan 10% untuk data testing. Data latih merupakan dataset yang digunakan untuk melatih sistem model *benchmark* IndoNLU dengan tujuan agar model dapat mengenali pola dari data. Data validasi adalah data yang digunakan untuk menguji kinerja model atau sistem yang telah dilatih dengan tujuan untuk memantau sejauh mana kinerja model dalam mengenali pola dari data. Kemudian Data testing, adalah data yang digunakan untuk menguji performa akhir model.

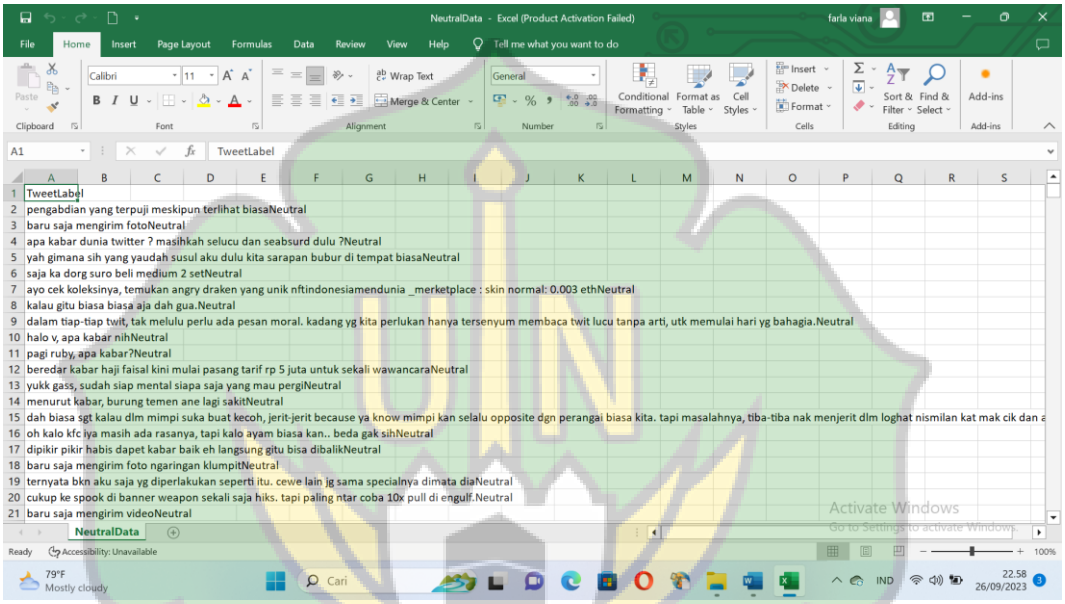
III.6 Evaluasi

Evaluasi dilakukan untuk mengetahui kelayakan hasil dari model yang sudah diuji dengan melihat hasil akurasi pada data tes. Selain itu juga akan dilakukan pengujian dengan memasukan kalimat opini untuk mengetahui kelayakan model dalam menentukan label emosi.

BAB IV
HASIL DAN PEMBAHASAN

IV.1 Dataset

Dataset yang digunakan pada penelitian ini bersumber dari github. Data yang diambil sebanyak 7080 tweet yang disimpan dalam format csv. Dataset yang disediakan oleh github sudah melewati beberapa tahap *pre-processing* dan sudah dilabeli. Berikut tampilan dataset dengan label *neutral* seperti pada gambar 4.1.



Gambar 4.1 Dataset Dengan Label *Neutral*

IV.2 Pelabelan data

Penelitian ini menggunakan enam jenis label emosi yaitu *anger*, *fear*, *joy*, *love*, *sad*, *neutral*. Data yang diperoleh dari github sudah memiliki label pada masing masing teks. Total data yang digunakan pada penelitian ini berjumlah 7.080 tweet yang terdiri dari label *anger* 1.130 data, *fear* 911 data, *joy* 1.275 data, *love* 760 data, *sad* 1.003 data, dan *neutral* 2.001 data.

IV.3 *Pre-Processing*

Tahap selanjutnya yaitu *pre-processing* data. *Pre-processing* merupakan tahapan untuk membersihkan dataset yang akan digunakan sebagai bahan uji, tujuannya agar dataset dapat dibaca dengan baik oleh model sehingga hasil akhir yang diperoleh akan lebih akurat.

Dataset yang diambil dari github sudah melewati sebagian tahap pre-processing seperti menghilangkan duplikasi teks, mengubah teks menjadi huruf kecil, dan pembersihan teks tambahan (penghapusan karakter atau simbol, hastag, URL dan emotikon). Pada model benchmark indoNLU, dibutuhkan beberapa tahap *pre-processing* lainnya seperti tokenisasi, penghapusan *stopword*, mengubah bahasa slang, dan *stemming*.

a. Tokenisasi

Tokenisasi adalah proses mengubah teks menjadi unit-unit yang lebih kecil atau memisahkan sebuah kalimat menjadi kata perkata yang disebut token. Token biasanya merupakan kata-kata yang memiliki makna atau arti. Proses tokenisasi bertujuan untuk memecah teks menjadi bagian-bagian yang lebih kecil agar mudah dikelola atau diproses. Kode pemograman yang digunakan dalam proses tokenisasi dapat dilihat pada gambar 4.2.

```
from keras.preprocessing.text import Tokenizer
from nltk.tokenize import word_tokenize

#NLTK word tokenize
def word_tokenize_wrapper(text):
 return word_tokenize(text)
data['teks'] = data['teks'].apply(word_tokenize_wrapper)
data.head()
```

Gambar 4.2 Kode Pemograman Tokenisasi

b. Normalisasi Kata

Normalisasi kata adalah proses mengubah kata-kata dalam teks menjadi bentuk yang lebih standar, konsisten, atau terstruktur. Tujuannya untuk menyederhanakan teks sehingga lebih mudah untuk dianalisis, diproses, atau dibandingkan dengan teks lainnya. Normalisasi teks yang dilakukan pada penelitian meliputi beberapa langkah yaitu menghapus tanda baca, dan mengubah bahasa slank. Kode pemograman untuk proses normalisasi teks dapat dilihat pada gambar 4.3.

```
# Membaca data normalisasi dari file CSV
normalizad_word = pd.read_csv("/content/datafix3k.csv")

normalizad_word_dict = {}

# Membuat kamus normalisasi
for index, row in normalizad_word.iterrows():
    normalizad_word_dict[row[0]] = row[1]

# Fungsi untuk normalisasi term
def normalized_term(document):
    return [normalizad_word_dict[term] if term in normalizad_word_dict else term for term in document]

# Mengaplikasikan normalisasi ke kolom 'teks'
data['teks'] = data['teks'].apply(normalized_term)

data['teks'].head(20)
```

Gambar 4.3 Kode Pemograman Normalisasi Teks

c. Penghapusan *Stopword*

Penghapusan *stopword* adalah proses mengidentifikasi dan menghapus kata-kata berhenti (*stop words*) dari dataset. *Stopword* merupakan kata-kata yang sering muncul namun tidak memiliki nilai informasi yang penting. Penghapusan *stopword* bertujuan untuk membersihkan teks dan meningkatkan relevansi dan efisiensi dalam analisis teks. Contoh kata-kata berhenti dalam bahasa Indonesia termasuk "yang", "dan", "di", "dari", "ke", "ada", "ini", "itu", dan sebagainya. Berikut kode pemograman untuk penghapusan *stopword* dapat dilihat pada gambar 4.4.

```
nltk.download('stopwords')
from nltk.corpus import stopwords

# Membuat daftar stop words untuk bahasa Indonesia
stop_words_id = set(stopwords.words('indonesian'))

# Fungsi untuk menghapus stop words dari daftar token kata
def stopword_removal(tokens):
    filtered_tokens = [token for token in tokens if token.lower() not in stop_words_id]
    return filtered_tokens

# Menghapus stop words dalam bahasa Indonesia dari kolom 'teks'
data['teks'] = data['teks'].apply(stopword_removal)
data.head(10)
```

Gambar 4.4 Kode Pemograman *Stopword*

d. *Stemming*

Stemming adalah proses mengubah kata-kata ke dalam bentuk dasar dengan menghilangkan akhiran atau awalan kata. Tujuannya adalah untuk mengurangi

variasi kata ke bentuk yang lebih sederhana. Kode pemograman untuk proses *stemming* dapat dilihat pada gambar 4.5.

```
!pip install sastrawi
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

# Fungsi untuk stemming
def stemming(tokens):
    factory = StemmerFactory()
    stemmer = factory.create_stemmer()
    stemmed_tokens = [stemmer.stem(token) for token in tokens]
    stemmed_text = ' '.join(stemmed_tokens)
    return stemmed_text

data['teks'] = data['teks'].apply(stemming)

data.to_csv('databersih3k.csv', index=False)
data_clean = pd.read_csv('databersih3k.csv', encoding='latin1')
data_clean.head()
```

Gambar 4.5 Kode Pemograman *Stemming*

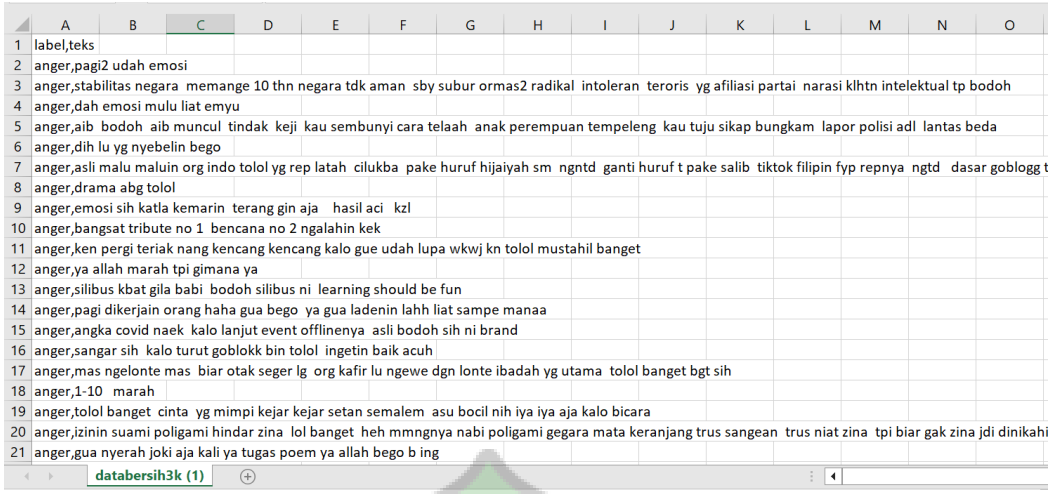
Setelah *stemming* dilakukan, dataset yang sudah bersih akan disimpan kembali ke dalam file csv pada fungsi data.to_csv(‘databersih3k.csv’, index=false).

Perbandingan teks sebelum dan sesudah di lakukan preprocessing dapat dilihat pada gambar 4.6 dan 4.7.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	label,teks,,														
2	anger,pagi2 udah di buat emosi :),,														
3	anger,"kok stabilitas negara, memange 10 thn negara tdk aman, bahkan sby menyuburkan ormas2 radikal, intoleran, teroris, yg berafiliasi ke partai tertentu..na														
4	anger,dah lah emosi mulu liat emyu,,														
5	anger,"aib? bodoh benar! sebelum kata aib itu muncul, terlebih dahulu sudah ada tindakan. yakni kekejian! jangan kau sembunyikan caramu menelaah masala														
6	anger,dih lu yg nyebelin bego,,														
7	anger,"asli malu maluin org indo tolol yg rep latah ""cilukba"" pake huruf hijaiyah sm ""ngntd"" sama ganti huruf t pake salib, ada tiktok filipin lewat fyp aku da														
8	anger,drama abg tolol,,														
9	anger,masih emosi sih sama katla kemarin. mana keterangannya gini aja. ((hasil mengaci)) kzl.,,														
10	anger,"bangsat tribute no.1, bencana no.2 mau ngalahin ini keknya",,														
11	anger,pengen pergi jauh terus teriak sambil nangis sekencang kencangnya nanti balik kalo gue udah lupa segalanya wkwj kn tolol mustahil banget,,														
12	anger,ya allah mau marah tpi gimana ya,,														
13	anger,silibus kita memang kbat gila babi. bodoh yang buat silibus ni. learning should be fun.,,														
14	anger,"pagi-pagi dikerjain orang haha dikira gua bego, ya gua ladenin lahh liat sampe manaa",,														
15	anger,"angka covid lagi naek. kalo masih dilanjut event offlinenya, asli bodoh sih ni brand.",,														
16	anger,"bukan lebih ke sangar sih , kalo menurutku goblokk bin tolol . di ingetin untuk kebaikan malah acuh",,														
17	anger,"mas ngelonte dulu mas, biar otaknya jadi seger lg, bagi org kafir seperti lu ngewe dgn lonte ibadah yg utama. punya tolol kok kebangetan bgt sih",,														
18	anger,1-10!! marah aku,,														
19	anger,tolol banget. mana ada bercinta. yg ada mimpi di kejar kejar setan semalem. asu bocil nih yang iya iya aja kalo bicara,,														
20	anger,"izinin suami poligami buat menghindari zina? lol banget, heh mmngnya nabi saya poligami gegara mata keranjang trus sangean? trus ada niatan zina, tp														
21	anger,gua nyerah joki aja kali ya tugas poem ya allah kenapa aku bego b ing,,														

datafix3k

Gambar 4.6 Teks Sebelum Tahap *Preprocessing*



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	label,teks														
2	anger,pagi2 udah emosi														
3	anger,stabilitas negara memange 10 thn negara tdk aman sby subur ormas2 radikal intoleran teroris yg afiliasi partai narasi klhtn intelektual tp bodoh														
4	anger,dah emosi mulu liat emyu														
5	anger,aib bodoh aib muncul tindak keji kau sembunyi cara telaah anak perempuan tempeleng kau tuju sikap bungkam lapor polisi adl lantas beda														
6	anger,dih lu yg nyebelin bego														
7	anger,asli malu maluin org indo tolol yg rep latah cilukba pake huruf hijaiyah sm ngntd ganti huruf t pake salib tiktok filipin fyp repnya ngtd dasar goblogg t														
8	anger,drama abg tolol														
9	anger,emosi sih katla kemarin terang gin aja hasil aci kzl														
10	anger,bangsat tribute no 1 bencana no 2 ngalahin kek														
11	anger,ken pergi teriak nang kencang kencang kalo gue udah lupa wkwj kn tolol mustahil banget														
12	anger,ya allah marah tpi gimana ya														
13	anger,silibus kbat gila babi bodoh silibus ni learning should be fun														
14	anger,pagi dikerjain orang haha gua bego ya gua ladenin lahh liat sampe manaa														
15	anger,angka covid naek kalo lanjut event offlinenya asli bodoh sih ni brand														
16	anger,sangar sih kalo turut goblokk bin tolol ingetin baik acuh														
17	anger,mas ngelonte mas biar otak seger lg org kafir lu ngewe dgn lonte ibadah yg utama tolol banget bgt sih														
18	anger,1-10 marah														
19	anger,tolol banget cinta yg mimpi kejar kejar setan semalem asu bocil nih iya iya aja kalo bicara														
20	anger,izinin suami poligami hindar zina lol banget heh mmngnya nabi poligami gegara mata keranjang trus sangean trus niat zina tpi biar gak zina jdi dinikahi														
21	anger,gua nyerah joki aja kali ya tugas poem ya allah bego b ing														

databersih3k (1)

Gambar 4.7 Teks Setelah Tahap *Preprocessing*

IV.4 Split Data

Dataset yang sudah dilakukan *preprocessing* selanjutnya akan dibagi menjadi tiga bagian, yaitu data training sebanyak 80% dari total data, data valid sebanyak 10% dan data testing sebanyak 10% dari total data. Split data dilakukan menggunakan kode phyton yang terlihat pada gambar 4.8.

```
import pandas as pd
from sklearn.model_selection import train_test_split

# Data Anda
path_ = "/content/databersih3k.csv"
data = pd.read_csv(path_, delimiter=';')

# Membagi data menjadi data latih, data validasi, dan data pengujian
train_ratio = 0.8
validation_ratio = 0.1
test_ratio = 0.1

# Membagi data menjadi data latih (80%) dan sisanya (20%)
train_data, temp_data = train_test_split(data, test_size=1 - train_ratio, random_state=42)

# Membagi data sisanya menjadi data validasi (10%) dan data pengujian (10%)
validation_data, test_data = train_test_split(temp_data, test_size=test_ratio / (test_ratio + validation_ratio), random_state=42)

# Cetak panjang masing-masing bagian data
print(f"Data Latih: {len(train_data)}")
print(f"Data Validasi: {len(validation_data)}")
print(f"Data Tes: {len(test_data)}")
```

Gambar 4.8 Kode Pemograman Split Data

Data train adalah data yang digunakan untuk melatih model agar mendapatkan nilai akurasi yang sesuai. Data valid digunakan untuk menilai performa model selama pelatihan dan membantu dalam penyesuaian hyperparameter. Sedangkan data test digunakan untuk mengukur kinerja model pada data yang tidak pernah dilihat selama pelatihan atau validasi. Pembagian data train, data valid, dan data test dengan total data yang peneliti gunakan sebesar 3000 data yang berisikan opini pengguna Twitter.

IV.5 Perancangan Model

Penelitian ini menerapkan model IndoBERT yang dikembangkan oleh model *benchmark* indoNLU. Beberapa variasi dari IndoBERT diuji untuk mengetahui perbandingan tingkat akurasi dari masing-masing variasi dengan menggunakan dataset yang sama. Berikut variasi yang akan dibandingkan yaitu IndoBERT Base p1, IndoBERT Base p2, IndoBERT Large p1, dan IndoBERT Large p2.

a. IndoBERT Base P1

Pada perancangan model pertama menggunakan IndoBERT base p1. Penerapan model dapat dilihat pada gambar 4.9.

```
# Load Tokenizer and Config
tokenizer = BertTokenizer.from_pretrained('indobenchmark/indobert-base-p1')
config = BertConfig.from_pretrained('indobenchmark/indobert-base-p1')
config.num_labels = EmotionDetectionDataset.NUM_LABELS

# Instantiate model
model = BertForSequenceClassification.from_pretrained('indobenchmark/indobert-base-p1', config=config)
```

Gambar 4.9 Kode Penerapan Indobert Base P1

Kode ini berfungsi untuk memuat tokenizer, konfigurasi, dan model BERT untuk tugas klasifikasi urutan (sequence classification) menggunakan *pre-trained weights* dari model Indobert base p1.

Sebelum memulai pelatihan, model diuji terlebih dahulu untuk membandingkan nilai probabilitas atau nilai keyakinan model sebelum dan sesudah pelatihan dilakukan. Kode pengujian model dapat dilihat pada gambar 4.10.

```
text = 'Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] * 100:.3f}%)')
```

```
Text: Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita | Label : neutral (32.396%)
```

Gambar 4.10 Pengujian IndoBERT Base P1 Sebelum Dilatih

Hasil pengujian pada teks ini menunjukkan bahwa model memprediksi teks “Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita” dalam kategori "*neutral* (netral)" dengan tingkat keyakinan sekitar 32.396%.

b. IndoBERT Base P2

Perancangan model kedua menggunakan IndoBERT base p2. Penerapan model dapat dilihat pada gambar 4.11.

```
# Load Tokenizer and Config
tokenizer = BertTokenizer.from_pretrained("indobenchmark/indobert-base-p2")
config = BertConfig.from_pretrained('indobenchmark/indobert-base-p2')
config.num_labels = EmotionDetectionDataset.NUM_LABELS

# Instantiate model
model = BertForSequenceClassification.from_pretrained('indobenchmark/indobert-base-p2', config=config)
```

Gambar 4.11 Kode Penerapan Indobert Base P2

Pengujian teks menggunakan model IndoBERT base p2 yang belum dilatih dapat dilihat pada gambar 4.12.

```
text = 'Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] * 100:.3f}%)')
```

```
Text: Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita | Label : neutral (28.808%)
```

Gambar 4.12 Pengujian IndoBERT Base P2 Sebelum Dilatih

Hasil pengujian menunjukkan bahwa model memprediksi teks “Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita” dalam kategori "*neutral* (netral)" dengan tingkat keyakinan sekitar 28.808%.

c. Indobert Large P1

Perancangan model ketiga menggunakan IndoBERT large p1. Penerapan model dapat dilihat pada gambar 4.13.

```
# Load Tokenizer and Config
tokenizer = BertTokenizer.from_pretrained("indobenchmark/indobert-large-p1")
config = BertConfig.from_pretrained('indobenchmark/indobert-large-p1')
config.num_labels = EmotionDetectionDataset.NUM_LABELS

# Instantiate model
model = BertForSequenceClassification.from_pretrained('indobenchmark/indobert-large-p1', config=config)
```

Gambar 4.13 Kode Penerapan IndoBERT Large P1

Pengujian teks menggunakan model IndoBERT large p1 yang belum dilatih dapat dilihat pada gambar 4.14.

```
text = 'Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] * 100:.3f}%)')
```

```
Text: Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita | Label : sadness (39.224%)
```

Gambar 4.14 Pengujian2 IndoBERT Large P1 Sebelum Dilatih

Hasil pengujian pada teks ini menunjukkan bahwa model memprediksi teks "Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita" kedalam kategori "*sadness* (kesedihan)" dengan tingkat keyakinan sekitar 39.224%.

d. IndoBERT Large P2

Perancangan model keempat menggunakan IndoBERT large p2. Penerapan model dapat dilihat pada gambar 4.15.

```
# Load Tokenizer and Config
tokenizer = BertTokenizer.from_pretrained("indobenchmark/indobert-large-p2")
config = BertConfig.from_pretrained('indobenchmark/indobert-large-p2')
config.num_labels = EmotionDetectionDataset.NUM_LABELS

# Instantiate model
model = BertForSequenceClassification.from_pretrained('indobenchmark/indobert-large-p2', config=config)
```

Gambar 4.15 Kode Penerapan IndoBERT Large p2

Pengujian teks menggunakan model IndoBERT large p2 yang belum dilatih dapat dilihat pada gambar 4.16.

```
text = 'Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] * 100:.3f}%)')
```

```
Text: Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita | Label : sadness (39.768%)
```

Gambar 4.16 Pengujian IndoBERT Large P2 Sebelum Dilatih

Nilai prediksi yang diperoleh pada model IndoBERT large p2 yang belum dilatih sebesar 39.768% pada teks "Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita" *sadness* (kesedihan).

IV.6 Melatih Model

Setelah model dibangun, tahapan selanjutnya adalah menjalankan pelatihan untuk masing-masing model. Model akan dilatih untuk mendapatkan nilai akurasi yang baik dan layak untuk menguji emosi dengan tepat. Pada penelitian ini terdapat empat variasi model yang akan dilatih yaitu, IndoBERT base p1, IndoBERT base p2, IndoBERT large p1, dan IndoBERT Large p2.

Keempat model menggunakan kode pemograman dan dataset yang sama dalam pelatihan. Dataset yang sudah melalui tahapan *pre-processing* dimasukkan kedalam model dengan tujuan mempersiapkan data dan data *loader* untuk melatih, memvalidasi, dan menguji model pada tugas klasifikasi emosi.

Sebelum pelatihan dijalankan, penulis mengatur jumlah *learning rate* dan perangkat yang digunakan saat proses pelatihan. Kode pemograman yang digunakan dapat dilihat pada gambar 4.17.

```
optimizer = optim.Adam(model.parameters(), lr=5e-6)
model = model.cuda()
```

Gambar 4.17 Jumlah *Learning Rate* Dan Perangkat Yang Digunakan

Dua baris kode diatas menciptakan model pelatihan dengan tingkat pembelajaran (*learning rate*) sebesar 5e-6, dan model tersebut akan proses pada perangkat CUDA (GPU). *Learning rate* merupakan parameter yang menentukan seberapa besar langkah-langkah pembaruan yang diterapkan pada bobot model selama proses pelatihan. Nilai *learning rate* sangat mempengaruhi kecepatan dan stabilitas konvergensi model. Selanjutnya kode pelatihan model dapat dilihat pada gambar 4.18.

```
# Train
n_epochs = 10
for epoch in range(n_epochs):
    model.train()
    torch.set_grad_enabled(True)

    total_train_loss = 0
    list_hyp, list_label = [], []

    train_pbar = tqdm(train_loader, leave=True, total=len(train_loader))
    for i, batch_data in enumerate(train_pbar):
```

Gambar 4.18 Kode Pelatihan Model

Epoch adalah satuan waktu dalam pelatihan model di mana data latih dan data valid diproses satu kali oleh model. Dalam penilitian ini, masing-masing model

dilatih sebanyak 10 epoch. Artinya, model akan melihat dan mempelajari seluruh dataset pelatihan sebanyak 10 kali.

Epoch mencakup dua tahap utama yaitu pelatihan (*training*) dan validasi. Tahap pelatihan dilakukan untuk mengoptimalkan parameter model, sedangkan tahap validasi membantu mengevaluasi kinerja model pada data yang tidak digunakan selama pelatihan. Dalam satu epoch, setiap sampel data dari dataset pelatihan melewati model dua kali, yaitu:

1. *Forward Pass* (Langkah Maju): Setiap sampel data dimasukkan ke dalam model untuk menghasilkan prediksi.
2. *Backward Pass* (Langkah Mundur): Model menggunakan prediksi tersebut untuk menghitung *gradien loss* terhadap parameter-parameter model.

Fungsi dari variabel list_hyp dan list_label digunakan untuk menyimpan prediksi model dan label sebenarnya pada setiap *batch*.

Dalam proses pelatihan, model akan menghitung *training loss*, evaluasi pada data validasi, dan menghitung *validation loss*. Masing-masing hasil dari *training loss* dan *validation loss* akan dicetak dalam bentuk matrik (seperti akurasi dan presisi) setelah satu epoch selesai.

IV.7 Hasil Pelatihan

Dalam melakukan klasifikasi emosi pada teks, peneliti membagi dataset menjadi tiga bagian, yaitu data *train*, data *valid*, dan data *test*. Dalam pembagiannya peneliti menggunakan proposi 80% data train, 10% data valid dan 10% data test.

Data valid = 80% x 7080
= 5664
Data valid = 10% x 7080
= 708
Data test = 10% x 7080
= 708

Di bawah ini adalah hasil dari pelatihan dari setiap model yaitu IndoBERT base p1, IndoBERT base p2, IndoBERT large p1, dan IndoBERT large p2 sebagai berikut s:

a. IndoBERT Base P1

Hasil dari pelatihan model IndoBERT base p1 menggunakan epoch 10 dapat dilihat pada tabel 4.1 dan tampilan dalam bentuk grafik pada gambar 4.19.

Tabel 4.1 Hasil Pelatihan Model Indobert Base P1

Epoch	Train_Loss	Train_Accuracy	Val_Loss	Val_Accuracy
1	0.8258	0.72	0.7884	0.74
2	0.6715	0.77	0.7459	0.76
3	0.5690	0.81	0.7652	0.73
4	0.4784	0.84	0.7658	0.77
5	0.3875	0.87	0.8129	0.75
6	0.2909	0.91	0.8495	0.76
7	0.2289	0.93	0.8957	0.75
8	0.1713	0.95	0.9279	0.75
9	0.1201	0.97	1.0367	0.75
10	0.1017	0.97	1.0934	0.74

Gambar 4.19 Grafik IndoBERT Base P1 (*Loss* dan *Accuracy*)

Tabel dan grafik diatas menunjukkan niali *train loss* menurun secara konsisten setiap epoch, sedangkan *train accuracy* meningkat secara konsisten dan mencapai nilai tertinggi pada epoch ke-10. Ini menunjukkan bahwa model semakin baik dalam mempelajari data pelatihan.

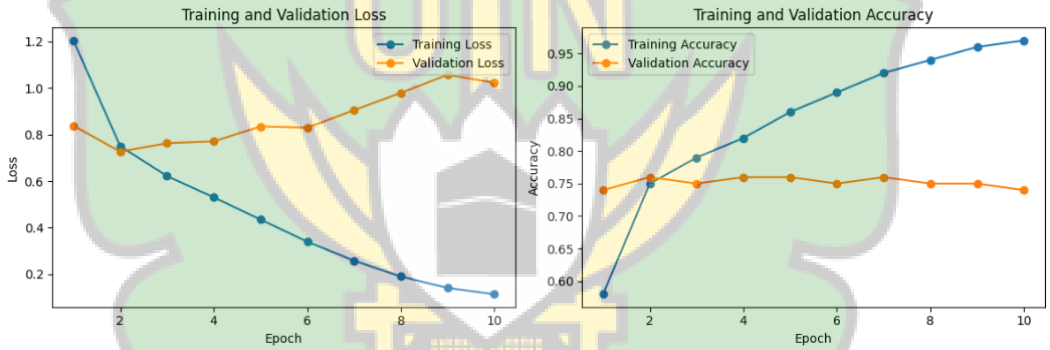
Nilai *validation loss* cenderung naik setelah epoch ke-4, menunjukkan kemungkinan adanya *overfitting*. Ini adalah tanda bahwa model terfokus terhadap data pelatihan dan kehilangan kemampuan untuk melakukan prediksi pada data validasi. Nilai *validation accuracy* naik dan turun setiap epoch. Hal ini menunjukkan model tidak sepenuhnya menggeneralisasi dengan baik dari data pelatihan ke data validasi.

b. IndoBERT Base P2

Hasil dari pelatihan model IndoBERT base p2 menggunakan epoch 10 dapat dilihat pada tabel 4.2 dan 4.20.

Tabel 4.2 Hasil Pelatihan IndoBERT Base P2

Epoch	Train_Loss	Train_Accuracy	Val_Loss	Val_Accuracy
1	1.2044	0.58	0.8380	0.74
2	0.7502	0.75	0.7266	0.76
3	0.6217	0.79	0.7628	0.75
4	0.5304	0.82	0.7708	0.76
5	0.4349	0.86	0.8347	0.76
6	0.3400	0.89	0.8295	0.75
7	0.2583	0.92	0.9049	0.76
8	0.1902	0.94	0.9790	0.75
9	0.1412	0.96	1.0566	0.75
10	0.1140	0.97	1.0226	0.74



Gambar 4.20 IndoBERT Base P2 (*Loss* dan *Accuracy*)

Pada model IndoBERT base p2, *train loss* menurun dan *train accuracy* meningkat secara konsisten di setiap epoch, menunjukkan bahwa model terus belajar dari data pelatihan. Hal ini juga berarti bahwa model semakin baik dalam memprediksi data pelatihan.

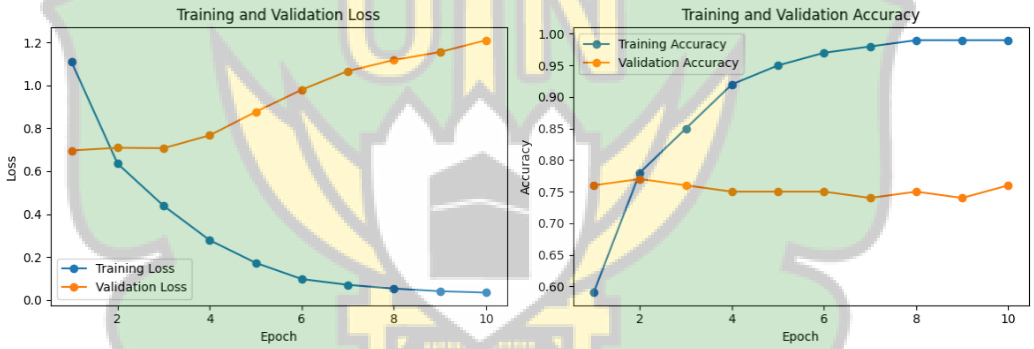
Validation loss cenderung bervariasi dari epoch ke epoch, namun mulai meningkat setelah beberapa epoch. Peningkatan *validation loss* menunjukkan bahwa model kemungkinan mengalami *overfitting*.

c. IndoBERT Large P1

Hasil dari pelatihan model IndoBERT large p1 menggunakan epoch 10 dapat dilihat pada tabel 4.3 dan gambar 4.21.

Tabel 4.3 Hasil Pelatihan Indobert Large P1

Epoch	Train_Loss	Train_Accuracy	Val_Loss	Val_Accuracy
1	1.1089	0.59	0.6962	0.76
2	0.6352	0.78	0.7088	0.77
3	0.4385	0.85	0.7070	0.76
4	0.2785	0.92	0.7668	0.75
5	0.1729	0.95	0.8762	0.75
6	0.0974	0.97	0.9793	0.75
7	0.0706	0.98	1.0662	0.74
8	0.0531	0.99	1.1174	0.75
9	0.0409	0.99	1.1548	0.74
10	0.0344	0.99	1.2099	0.76



Gambar 4.21 IndoBERT Large P1 (*Loss* Dan *Accuracy*)

Pada pelatihan model IndoBERT large p1, *train loss* menurun dan *train accuracy* meningkat secara konsisten di setiap epoch. Hal ini menunjukkan bahwa model mempelajari data pelatihan dengan baik.

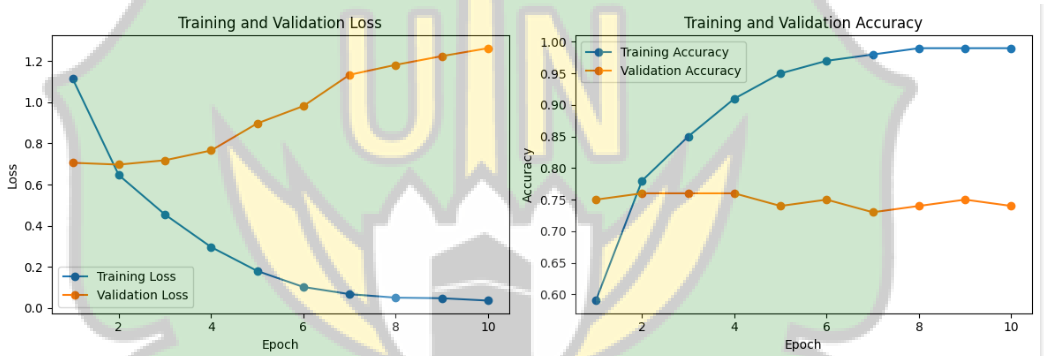
Grafik *validation loss* cenderung bervariasi dari epoch ke epoch dan mengalami peningkatan setelah beberapa epoch. Sama seperti model sebelumnya, pelatihan pada model ini menunjukkan adanya *overfitting*.

d. IndoBERT Large P2

Hasil dari pelatihan model IndoBERT large p2 menggunakan epoch 10 dapat dilihat pada tabel 4.4 dan gambar 4.22.

Tabel 4.4 Hasil Pelatihan Model Indobert Large P2

Epoch	Train_Loss	Train_Accuracy	Val_Loss	Val_Accuracy
1	1.1149	0.59	0.7058	0.75
2	0.6460	0.78	0.6969	0.76
3	0.4542	0.85	0.7181	0.76
4	0.2959	0.91	0.7655	0.76
5	0.1795	0.95	0.8974	0.74
6	0.1017	0.97	0.9818	0.75
7	0.0668	0.98	1.1337	0.73
8	0.0497	0.99	1.1821	0.74
9	0.0470	0.99	1.2246	0.75
10	0.0354	0.99	1.2632	0.74



Gambar 4.22 Indobert Large P2 (*Loss* Dan *Accuracy*)

Tabel diatas menunjukkan nilai *train loss* menurun dan *train accuracy* meningkat secara konsisten setiap epoch, menunjukkan bahwa model terus belajar dari data pelatihan. Sedangkan *validation loss* cenderung bervariasi dari epoch ke epoch, kemudian meningkat setelah beberapa epoch. Hal ini menunjukkan model tidak dapat melakukan prediksi pada data valid dengan baik.

IV.8 Evaluasi Model

Setelah proses pelatihan selesai, model akan melakukan evaluasi menggunakan data test. Kode evaluasi model dapat dilihat pada gambar 4.23.

```
# Evaluate on test
model.eval()
torch.set_grad_enabled(False)

total_loss, total_correct, total_labels = 0, 0, 0
list_hyp, list_label = [], []

pbar = tqdm(test_loader, leave=True, total=len(test_loader))
for i, batch_data in enumerate(pbar):
    _, batch_hyp, batch_label = forward_sequence_classification(model, batch_data[:-1], i2w=i2w, device='cuda')
    list_hyp += batch_hyp
    list_label += batch_label
```

Gambar 4.23 Kode Pemograman Evaluasi Model

Proses evaluasi model adalah langkah-langkah yang dilakukan untuk mengukur kinerja suatu model mesin pembelajaran terhadap data yang tidak terlihat selama pelatihan. Pada proses evaluasi, model akan menyimpan label prediksi didalam kode list_hyp dan label sebenarnya pada kode list_label. Dua variable ini selanjutnya dibandingkan untuk menghitung akurasi dari model. Kode untuk menghitung akurasi dapat dilihat pada gambar 4.24.

```
# Calculate accuracy
correct_predictions = sum(np.array(list_hyp) == np.array(list_label))
total_samples = len(list_label)
accuracy = correct_predictions / total_samples
```

Gambar 4.24 Kode Menghitung Akurasi

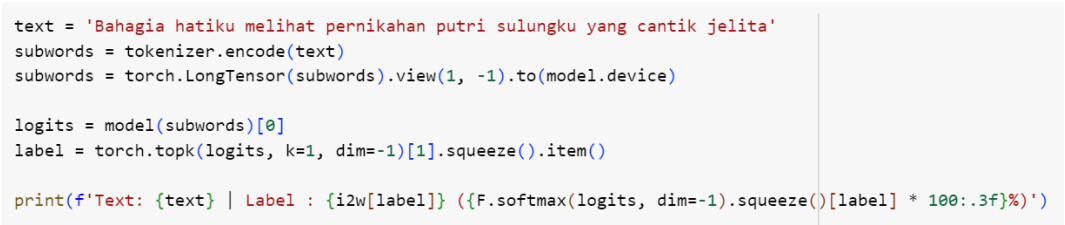
Hasil akurasi dari masing-masing model dapat dilihat pada tabel 4.5.

Tabel 4.5 Perbandingan Akurasi Setiap Model

Model	Akurasi
Pre-trained IndoBERT base p1	71.47%
Pre-trained IndoBERT base p2	71.19%
Pre-trained IndoBERT large p1	73.59%
Pre-trained IndoBERT large p2	72.60%

Tabel 4.5 menunjukkan bahwa akurasi model IndoBERT large p1 memiliki akurasi tertinggi sebesar 73.59%. Selanjutnya dilakukan dua tahap pengujian yaitu tahap *predict single text* dan *confussion matrix*.

a. ***Predict Single Text***, yaitu mengetahui label emosi yang ada pada text yang dimasukan. Pengujian ini dilakukan dengan menginput teks opini untuk mengetahui emosi yang terkandung dalam teks tersebut. Hasil pengujian dari masing masing model dapat dilihat pada gambar-gambar dibawah ini.

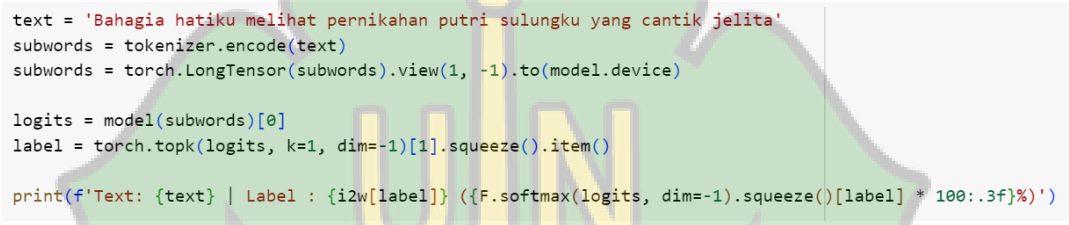


```
text = 'Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] * 100:.3f}%)')
```

```
Text: Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita | Label : Joy (98.367%)
```

Gambar 4.25 Pengujian Teks Dengan Model IndoBERT Base P1

Gambar 4.25 menunjukkan hasil pengujian menggunakan IndoBERT base p1 yang telah dilatih, model memprediksi teks “Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita” dalam kategori "*joy* (kegembiraan)" dengan tingkat keyakinan sebesar 98.367%.

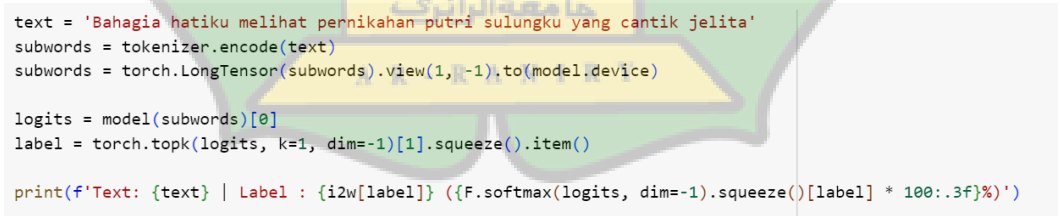


```
text = 'Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] * 100:.3f}%)')
```

```
Text: Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita | Label : Joy (98.754%)
```

Gambar 4.26 Pengujian Teks Pada Model IndoBERT Base P2

Pada gambar 4.26 dapat dilihat bahwa model IndoBERT base p2 yang telah dilatih memperoleh nilai prediksi sebesar 98.754% pada teks “Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita” dengan kategori “*joy* (kegembiraan)”.



```
text = 'Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] * 100:.3f}%)')
```

```
Text: Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita | Label : Joy (95.337%)
```

Gambar 4.27 Pengujian Teks Menggunakan Model IndoBERT *Large* P1

Gambar 4.27 memperlihatkan hasil pengujian pada teks “Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita” mengunakan model IndoBERT Large p1 yang telah dilatih, nilai prediksinya sebesar 95.337% dengan kataegori label “*joy* (kegembiraan)”.

```
text = 'Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] * 100:.3f}%)')
```

```
Text: Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita | Label : Joy (97.607%)
```

Gambar 4.28 Pengujian Teks Menggunakan Model IndoBERT *Large* P2

Gambar 4.28 menunjukkan bahwa model IndoBERT large p2 memperoleh nilai prediksi sebesar 97.607% pada pengujian teks "Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita" dengan kategori label "*joy* (kegembiraan)".

Perbandingan nilai prediksi atau probabilitas terhadap teks yang sama menggunakan model yang belum dan sudah dilatih dapat dilihat pada tabel 4.6.

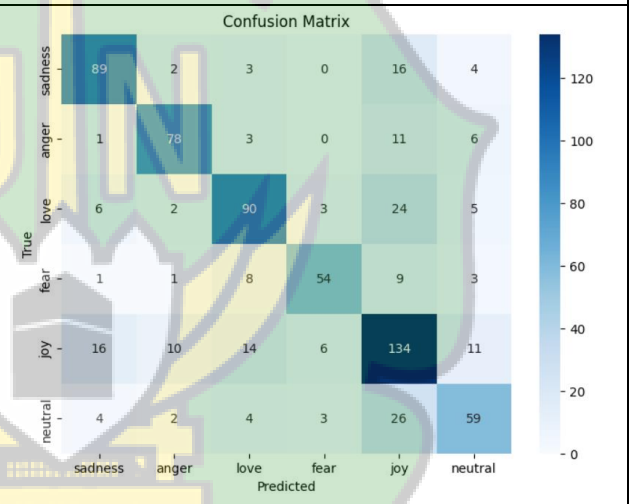
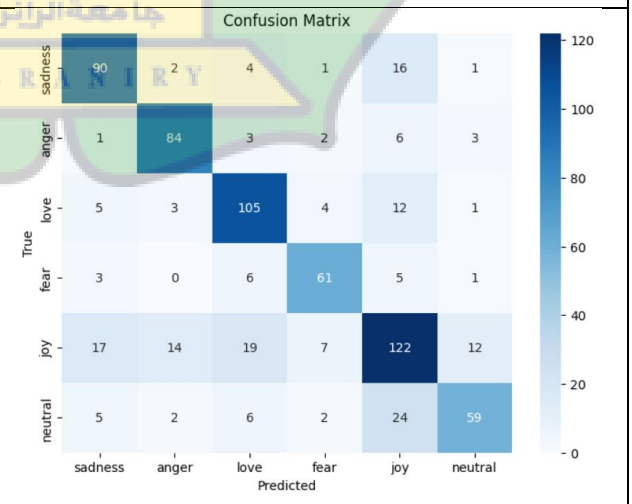
Tabel 4.6 Perbandingan Nilai Probabilitas Sebelum Dan Sesudah Model Dilatih

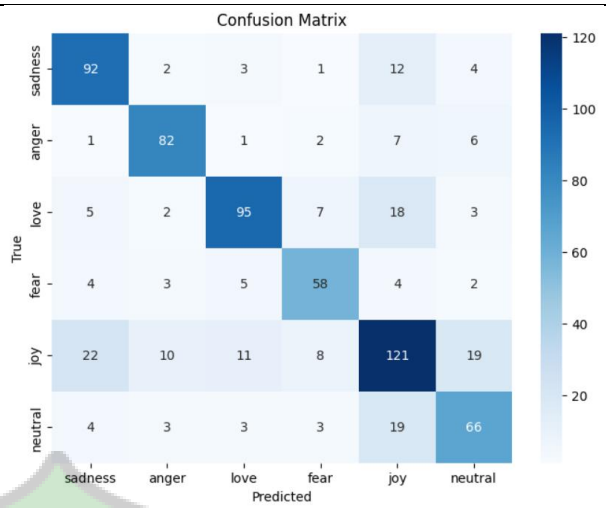
Model	Sebelum	Sesudah
Pre-trained IndoBERT base p1	32.396%	98.367%
Pre-trained IndoBERT base p2	28.808%	98.754%
Pre-trained IndoBERT large p1	39.224%	95.337%
Pre-trained IndoBERT large p2	39.768%	97.607%

b. *Confussion matrix*

Hasil pengujian *confussion matrix* pada masing-masing model dapat dilihat pada tabel 4.7.

Tabel 4.7 Perbandingan *Confussion Matrix*

No	Confusion matrix	Model
1	Pre-trained IndoBERT Base P1	
2	Pre-trained IndoBERT Base P2	
3	Pre-trained IndoBERT Large P1	

4	Pre-trained IndoBERT Large P2	

Confussion matrix menjelaskan data prediksi per tiap label *sadness, anger, love, fear, joy*, dan *neutral* didapatkan hasil yang sesuai dengan data uji sebesar 708 data. Penjelasan mengenai tabel *confussion matrix* pada masing masing model adalah sama, sehingga penulis hanya menjelaskan salah satu dari keempat *confussion matrix* diatas yaitu *confussion matrix* indoBERT base p1.

Model mendapatkan beberapa data *error* pada masing masing label. Label *sadness* memiliki bobot 91 data namun ada 1 data terprediksi ke dalam label *anger*, 4 data terprediksi kedalam label *love*, 1 data terprediksi kedalam label *fear*, 12 data terprediksi kedalam label *joy*, dan 5 data terprediksi kedalam label *neutral*. Label *anger* memiliki bobot 75 data tetapi ada 3 data yang diprediksi ke label *sadness*, 4 data terprediksi kedalam label *love*, 1 data terprediksi kedalam label *fear*, 8 data terprediksi kedalam label *joy*, dan 8 data terprediksi kedalam label *neutral*. Label *love* memiliki bobot 94 data, terdapat 4 data yang terprediksi sebagai *sadness*, 2 data terprediksi sebagai *anger*, 11 data terprediksi sebagai *fear*, 15 data terprediksi sebagai *joy*, dan 4 data terprediksi sebagai *neutral*. Label *fear* yang memiliki bobot 65 data, terdapat 1 data yang terprediksi sebagai *anger*, 4 data terprediksi sebagai *love*, 5 data terprediksi sebagai *joy*, dan 1 data terprediski sebagai *neutral*. Label *joy* memliki bobot 121 data dengan 17 data yang terprediksi sebagai *sadness*, 12 data terprediksi sebagai *anger*, 20 data terprediksi sebagai *love*, 9 data terprediksi sebagai *fear*, dan 12 data terprediksi sebagai *neutral*. Dan yang terakhir label *neutral* dengan bobot 60 data, namun ada 4 data yang terprediksi *sadness*, 3 data

terprediksi sebagai *anger*, 4 data terprediksi sebagai *love*, 5 data terprediksi sebagai *fear*, dan 22 data terprediksi sebagai *joy*.

BAB V
KESIMPULAN DAN SARAN

V.1 Kesimpulan

Pada penelitian ini dilakukan klasifikasi emosi pada opini twiter dengan jumlah data sebanyak 7080 data yang terdiri dari label *anger* 1.130 data, *fear* 911 data, *joy* 1.275 data, *love* 760 data, *sad* 1.003 data, dan *neutral* 2.001 data. Penelitian ini menggunakan model indoBERT yang dikembangkan oleh IndoNLU, model yang diuji merupakan variasi dari model indoBERT yaitu indoBERT base p1, indoBERT base p2, indoBERT large p1, dan indoBERT large p2. Adapun hasil yang diperoleh dari penelitian ini adalah:

a. Proses klasifikasi emosi opini Twitter dengan model *benchmark* IndoNLU menggunakan jumlah parameter *learning rate* sebesar 5e-6 dan epoch 10 pada pelatihan.
b. Berdasarkan hasil pengujian dari implementasi model *benchmark* IndoNLU dalam melakukan klasifikasi emosi, maka diperoleh kesimpulan sebagai berikut: Klasifikasi emosi menggunakan model IndoBERT base p1 menghasilkan akurasi sebesar 71.47%, IndoBERT base p2 memperoleh hasil akurasi 71.19%, IndoBERT large p1 memperoleh akurasi sebesar 73.59%, dan IndoBERT large p2 memperoleh nilai akurasi sebesar 72.60%.
c. Integrasi *confussion matrix* dalam evaluasi model *benchmark* IndoNLU memiliki dampak positif yang signifikan. *Confussion matrix* tidak hanya memberikan gambaran komprehensif tentang performa model, tetapi juga mengidentifikasi kelemahan dan kelebihan dalam model.

V.1 Saran

Terdapat beberapa saran dari penelitian ini untuk pengembangan penelitian selanjutnya:

a. Memperhatikan keseimbangan jumlah data pada setiap label agar menghasilkan akurasi yang lebih baik pada model.
b. Melakukan pelatihan dengan menggunakan nilai parameter lain untuk memaksimalkan hasil pelatihan model.

DAFTAR PUSTAKA

Aulia, K., & Amelia, L. (2020). Analisis Sentimen Twitter Pada Isu Mental Health Dengan Algoritma Klasifikasi Naive Bayes. *Siliwangi Journal (Seri Sains and Teknologi)*, *6*(2), 60–65.

Azeharie, S., & Kusuma, O. (2014). *Analisis Penggunaan Twitter Sebagai Media Komunikasi Selebritis Di Jakarta*. 83–98.

D. Lizzy, E. (2001). Natural language processing. *Artificial Intelligence*, *19*(2), 131–136. https://doi.org/10.1016/0004-3702(82)90032-7

Estiyani, R. (2018). Ekspresi Diri Melalui Media SOsial dan Makna dari Psikologis. *Ejournal.UMS.Bycore*, 274–282.

Fallis, A. . (2013). Kajian Pustaka Emosi. *Journal of Chemical Information and Modeling*, *53*(9), 1689–1699.

Gelar Guntara, R. (2023). Pemanfaatan Google Colab Untuk Aplikasi Pendeteksian Masker Wajah Menggunakan Algoritma Deep Learning YOLOv7. *Jurnal Teknologi Dan Sistem Informasi Bisnis*, *5*(1), 55–60. https://doi.org/10.47233/jteksis.v5i1.750

Hadiyan, K. P., Arif Bijaksana, M., & Romadhony, A. (2021). Deteksi Penggunaan Kalimat Abusive Pada Teks Bahasa Indonesia Menggunakan Metode IndoBERT. *E-Proceeding of Engineering*, *Vol.8*, *No.*(2), 3028–3038.

Iskandar Zulkarnain Maulana Putra, T., Farhan Bukhori, A., Ilmu Pengetahuan Alam, dan, & Gadjah Mada, U. (2022). Model Klasifikasi Berbasis Multiclass Classification dengan Kombinasi Indobert Embedding dan Long Short-Term Memory untuk Tweet Berbahasa Indonesia (Classification Model Based on Multiclass Classification with a Combination of Indobert Embedding and Long Short-Term Memory for Indonesian-language Tweets). *Jurnal Ilmu Siber Dan Teknologi Digital (JISTED)*, *1*(1), 1–28. https://doi.org/10.35912/jisted.v1i1.1509

Luo, L., & Wang, Y. (2019). *EmotionX-HSU: Adopting Pre-trained BERT for Emotion Classification*. http://arxiv.org/abs/1907.09669

Nurfauzan, A., & Maharani, W. (2021). Klasifikasi Emosi Pada Pengguna Twitter Menggunakan Metode Klasifikasi Decision Tree. *EProceedings …*. https://repository.telkomuniversity.ac.id/home/catalog/id/168836/slug/klasifi

kasi-emosi-pada-pengguna-twitter-menggunakan-metode-klasifikasi-decision-tree.html

Rahman, F., & Fithriasari, K. (2018). *KLASIFIKASI EMOSI UNTUK TEKS BERBAHASA INDONESIA PADA PENGGUNA TWITTER MENGENAI PRESIDEN JOKO WIDODO*. 1–8.

Rezki, A. S. (2021). *Klasifikasi Emosi Pada Twitter Dengan Metode K-Nearest Neighbor (Knn) Tugas Akhir*.

Riccio, V., Jahangirova, G., Stocco, A., Humbatova, N., Weiss, M., & Tonella, P. (2020). Testing machine learning based systems: a systematic mapping. *Empirical Software Engineering*, *25*(6), 5193–5254. https://doi.org/10.1007/s10664-020-09881-0

Syahrudin, A. N., & Kurniawan, T. (2018). Input dan Output pada Bahasa Pemrograman Python. *Jurnal Dasar Pemrograman Python STMIK, June 2018*, 1–7. https://www.researchgate.net/publication/338385483

Widianto, A. T. B. (2022). *Klasifikasi Emosi pada teks menggunakan Deep Learning*. *6*(1).

Wilie, B., Vincentio, K., Winata, G. I., Cahyawijaya, S., Li, X., Lim, Z. Y., Soleman, S., Mahendra, R., Fung, P., Bahar, S., & Purwarianti, A. (2020). *IndoNLU: Benchmark and Resources for Evaluating Indonesian Natural Language Understanding*. http://arxiv.org/abs/2009.05387

LAMPIRAN

Adapun lampiran lengkap hasil dari penelitian ini sebagai berikut :

Code berikut merupakan program *python* untuk mengimport *library* dan modul yang dibutuhkan oleh ke-empat model yaitu IndoBERT Base p1, IndoBERT Base p2, IndoBERT Large p1, dan IndoBERT Large p2.

```
!pip install transformers
!unzip /content/utils.zip

import os, sys
sys.path.append('../')
os.chdir('../')

import random
import numpy as np
import pandas as pd
import torch
from torch import optim
import torch.nn.functional as F
from tqdm import tqdm

from transformers import BertForSequenceClassification, BertConfig, BertTokenizer
from nltk.tokenize import TweetTokenizer

from utils.forward_fn import forward_sequence_classification
from utils.metrics import document_sentiment_metrics_fn
from utils.data_utils import EmotionDetectionDataset, EmotionDetectionDataLoader

###
# common functions
###
def set_seed(seed):
    random.seed(seed)
    np.random.seed(seed)
    torch.manual_seed(seed)
    torch.cuda.manual_seed(seed)

def count_param(module, trainable=False):
    if trainable:
        return sum(p.numel() for p in module.parameters() if p.requires_grad)
    else:
        return sum(p.numel() for p in module.parameters())

def get_lr(optimizer):
    for param_group in optimizer.param_groups:
```

```
        return param_group['lr']

def metrics_to_string(metric_dict):
    string_list = []
    for key, value in metric_dict.items():
        string_list.append('{}:{:.2f}'.format(key, value))
    return ' '.join(string_list)

# Set random seed
set_seed(26092020)
```

Baris code dibawah ini (*load tokenizer and config*) disesuaikan dengan variasi model yang ingin dilatih.

```
# Load Tokenizer and Config
tokenizer = BertTokenizer.from_pretrained('indobenchmark/indobert-base-p1')
config = BertConfig.from_pretrained('indobenchmark/indobert-base-p1')
config.num_labels = EmotionDetectionDataset.NUM_LABELS

model = BertForSequenceClassification.from_pretrained('indobenchmark/indobert-base-p1',
config=config)

count_param(model)

# Prepare dataset
train_dataset_path = '/content/DataTrain.csv'
valid_dataset_path = '/content/DataValid.csv'
test_dataset_path = '/content/DataTest.csv'

train_dataset = EmotionDetectionDataset(train_dataset_path, tokenizer, lowercase=True)
valid_dataset = EmotionDetectionDataset(valid_dataset_path, tokenizer, lowercase=True)
test_dataset = EmotionDetectionDataset(test_dataset_path, tokenizer, lowercase=True)

train_loader = EmotionDetectionDataLoader(dataset=train_dataset, max_seq_len=512,
batch_size=32, num_workers=2, shuffle=True)
valid_loader = EmotionDetectionDataLoader(dataset=valid_dataset, max_seq_len=512,
batch_size=32, num_workers=2, shuffle=False)
test_loader = EmotionDetectionDataLoader(dataset=test_dataset, max_seq_len=512,
batch_size=32, num_workers=2, shuffle=False)

w2i, i2w = EmotionDetectionDataset.LABEL2INDEX,
EmotionDetectionDataset.INDEX2LABEL
print(w2i)
print(i2w)

# Test model on sample sentences
# sample one
```

```
text = 'Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] *
100:.3f}%)')

# sample two
text = 'Budi pergi ke pondok indah mall membeli cakwe'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] *
100:.3f}%)')

# sample three
text = 'Dasar anak sialan!! Kurang ajar!!'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] *
100:.3f}%)')

# fine tuning & evaluation
optimizer = optim.Adam(model.parameters(), lr=5e-6)
model = model.cuda()

# Train
n_epochs = 10
for epoch in range(n_epochs):
    model.train()
    torch.set_grad_enabled(True)

    total_train_loss = 0
    list_hyp, list_label = [], []

    train_pbar = tqdm(train_loader, leave=True, total=len(train_loader))
    for i, batch_data in enumerate(train_pbar):
```

```
        # Forward model
        loss, batch_hyp, batch_label = forward_sequence_classification(model, batch_data[:-1],
i2w=i2w, device='cuda')

        # Update model
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

        tr_loss = loss.item()
        total_train_loss = total_train_loss + tr_loss

        # Calculate metrics
        list_hyp += batch_hyp
        list_label += batch_label

        train_pbar.set_description("(Epoch {}) TRAIN LOSS:{:.4f}
LR:{:.8f}".format((epoch+1),
            total_train_loss/(i+1), get_lr(optimizer)))

    # Calculate train metric
    metrics = document_sentiment_metrics_fn(list_hyp, list_label)
    print("(Epoch {}) TRAIN LOSS:{:.4f} {} LR:{:.8f}".format((epoch+1),
        total_train_loss/(i+1), metrics_to_string(metrics), get_lr(optimizer)))

    # Evaluate on validation
    model.eval()
    torch.set_grad_enabled(False)

    total_loss, total_correct, total_labels = 0, 0, 0
    list_hyp, list_label = [], []

    pbar = tqdm(valid_loader, leave=True, total=len(valid_loader))
    for i, batch_data in enumerate(pbar):
        batch_seq = batch_data[-1]
        loss, batch_hyp, batch_label = forward_sequence_classification(model, batch_data[:-1],
i2w=i2w, device='cuda')

        # Calculate total loss
        valid_loss = loss.item()
        total_loss = total_loss + valid_loss

        # Calculate evaluation metrics
        list_hyp += batch_hyp
        list_label += batch_label
        metrics = document_sentiment_metrics_fn(list_hyp, list_label)
```

```
        pbar.set_description("VALID LOSS:{:.4f} {}".format(total_loss/(i+1),
metrics_to_string(metrics)))

    metrics = document_sentiment_metrics_fn(list_hyp, list_label)
    print("(Epoch {}) VALID LOSS:{:.4f} {}".format((epoch+1),
        total_loss/(i+1), metrics_to_string(metrics)))

# Evaluate on test
model.eval()
torch.set_grad_enabled(False)

total_loss, total_correct, total_labels = 0, 0, 0
list_hyp, list_label = [], []

pbar = tqdm(test_loader, leave=True, total=len(test_loader))
for i, batch_data in enumerate(pbar):
    _, batch_hyp, batch_label = forward_sequence_classification(model, batch_data[:-1],
i2w=i2w, device='cuda')
    list_hyp += batch_hyp
    list_label += batch_label

# Calculate accuracy
correct_predictions = sum(np.array(list_hyp) == np.array(list_label))
total_samples = len(list_label)
accuracy = correct_predictions / total_samples

# Save prediction
df = pd.DataFrame({'label': list_hyp}).reset_index()
df.to_csv('pred.txt', index=False)

# Print accuracy
print(f"Accuracy on test set: {accuracy * 100:.2f}%")
print(df)

from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Load true labels from the test set
y_true = np.array(list_label)

# Load predicted labels from the saved predictions
df_pred = pd.read_csv('pred.txt')
y_pred = df_pred['label'].values

# Calculate confusion matrix
cm = confusion_matrix(y_true, y_pred)
```

```
# Display confusion matrix as a heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['sadness', 'anger', 'love',
'fear', 'joy', 'neutral'], yticklabels=['sadness', 'anger', 'love', 'fear', 'joy', 'neutral'])
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.show()

# Test fine-tuned model on sample sentences
# sample one
text = 'Bahagia hatiku melihat pernikahan putri sulungku yang cantik jelita'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] *
100:.3f}%)')

# sample two
text = 'Budi pergi ke pondok indah mall membeli cakwe'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] *
100:.3f}%)')

# sample three
text = 'Dasar anak sialan!! Kurang ajar!!'
subwords = tokenizer.encode(text)
subwords = torch.LongTensor(subwords).view(1, -1).to(model.device)

logits = model(subwords)[0]
label = torch.topk(logits, k=1, dim=-1)[1].squeeze().item()

print(f'Text: {text} | Label : {i2w[label]} ({F.softmax(logits, dim=-1).squeeze()[label] *
100:.3f}%)')
```