ANALISIS PENGGUNAAN KALIMAT *ABUSIVE* PADA MEDIA SOSIAL *TWITTER* MENGGUNAKAN METODE *MULTILINGUAL BERT*

TUGAS AKHIR

Diajukan Oleh :

NURHALIZA
NIM. 190705015
Mahasiswa Fakultas Sains dan Teknologi
Program Studi Teknologi Informasi

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

LEMBAR PERSETUJUAN

ANALISIS PENGGUNAAN KALIMAT *ABUSIVE* PADA MEDIA SOSIAL *TWITTER* MENGGUNAKAN METODE *MULTILINGUAL BERT*

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:
NURHALIZA
NIM. 190705015
Mahasiswa Fakultas Sains dan Teknologi
Program Studi Teknologi Informasi

Disetujui untuk Dimunaqasyahkan Oleh:

Pembimbing I,	Pembimbing II,
Hendri Ahmadian, M.I.M	**Mulkan Fadhli, M.T**
NIP. 198301042014031002	**NIP. 198811282020121006**

Mengetahui,
Ketua Program Studi Teknologi Informasi

Ima Dwitawati, MBA
NIP. 198210132014032002

LEMBAR PENGESAHAN

ANALISIS PENGGUNAAN KALIMAT *ABUSIVE* PADA MEDIA SOSIAL *TWITTER* MENGGUNAKAN METODE *MULTILINGUAL BERT*

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: Selasa, 15 Agustus 2023
28 Muharram 1445 H

di Darussalam, Banda Aceh
Panitia Ujian Munaqasah Tugas Akhir

Ketua,

Hendri Ahmadian, M.I.M
NIP. 198301042014031002

Sekretaris,

Mulkan Fadhli, M.T
NIP. 198811282020121006

Penguji I,

Bustami, M.Sc
NIP. 198210132014032002

Penguji II,

Khairan AR, M.Kom
NIP. 198607042014031001

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 : Nurhaliza

NIM : 190705015

Program Studi : Teknologi Informasi

Fakultas : Sains dan Teknologi

Judul Tugas Akhir : Analisis Penggunaan Kalimat *Abusive* Pada Media Sosial *Twitter* Menggunakan Metode *Multilingual BERT*

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

(Nurhaliza)

ABSTRAK

Nama : Nurhaliza

NIM : 190705015

Program Studi : Teknologi Informasi

Judul : Analisis Penggunaan Kalimat *Abusive* Pada Media Sosial *Twitter* Menggunakan Metode *Multilingual BERT*

Tanggal Sidang : 15 Agustus 2023

Jumlah Halaman : 70 Halaman

Pembimbing I : Hendri Ahmadian, M.I.M

Pembimbing II : Mulkan Fadhli, M.T

Perkembangan teknologi informasi ini memberikan dampak dan manfaat yang dapat diimplementasikan pada bidang *deep learning*. Saat ini banyak contoh yang biasa digunakan dalam pengembangan masalah deep learning, seperti *Transformers*. Penelitian ini menggunakan salah satu arsitektur dari *Transformers* yaitu *Multilingual BERT*, *Multilingual BERT* itu sendiri terdiri dari *BERT* yaitu *Bidirectional Encoder Representations from Transformers* yang biasa digunakan untuk masalah *deep learning*. Metode *Multilingual BERT* diimplementasikan untuk mendeteksi penggunaan kalimat abusive pada teks bahasa Indonesia.

Penelitian ini menggunakan dataset berjumlah 2016 data, yang terdiri dari 2 atribut yaitu *Tweet* dan Label. Pada penelitian ini menerapkan 3 percobaan dengan *learning rate, batch size* dan *epoch* yang berbeda. Evaluasi terhadap hasil analisis dengan model arsitektur *Multilingual BERT* menggunakan metode *Confusion Matrix*. Hasil evaluasi dengan metode *Confusion Matrix* memiliki nilai akurasi sebesar 81%. Secara keseluruhan hasil analisis kalimat *abusive* dengan model arsitektur *Multilingual BERT* dapat dikategorikan baik.

Kata Kunci: Kalimat *Abusive, Multilingual BERT, BERT, Confusion Matrix*

ABSTRACT

Name : Nurhaliza

Student ID : 190705015

Study Program : Information Technology

Title : Analysis of Abusive Sentence Usage on Twitter Social Media Using the Multilingual BERT Method

Session Date : 15 Agustus 2023

Number of Pages : 70 Pages

Supervisor I : Hendri Ahmadian, M.I.M

Supervisor II : Mulkan Fadhli, M.T

The advancement of information technology has brought impacts and benefits that can be implemented in the field of deep learning. Currently, many examples are commonly used in the development of deep learning problems, such as Transformers. This study utilizes one of the architectures from Transformers, namely Multilingual BERT, which consists of BERT, Bidirectional Encoder Representations from Transformers, commonly used in deep learning problems. The Multilingual BERT method is implemented to detect the usage of abusive sentences in Indonesian language texts.

This study employs a dataset consisting of 2016 data, comprising 2 attributes, namely Tweet and Label. Three experiments with different learning rates, batch sizes, and epochs are conducted in this research. Evaluation of the analysis results using the Multilingual BERT architecture model is performed using the Confusion Matrix method. The evaluation results using the Confusion Matrix method yield an accuracy value of 81%. Overall, the analysis of abusive sentences using the Multilingual BERT architecture model can be categorized as good.

Keywords: ***Abusive Language, Multilingual BERT, BERT, Confusion Matrix***

KATA PENGANTAR

بِسْـــــــــــــمِ اللهِ الرَّحْمَنِ الرَّحِيْمِ

Puji syukur penulis panjatkan atas kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya, penulis dapat menyelesaikan proposal tugas akhir yang berjudul "**Analisis Penggunaan Kalimat *Abusive* pada Media** Sosial Twitter Menggunakan Metode Multilingual BERT". Salawat beserta salam kita sampaikan kepada Rasulullah SAW berserta keluarga dan sahabat beliau sekalian yang telah memperjuangkan umat Islam kepada jalan kebenaran dengan dibekali ilmu yang bermanfaat untuk dunia dan akhirat.

Penulisan proposal tugas akhir ini diajukan sebagai salah satu syarat untuk menyelesaikan tugas akhir pada program studi Teknologi Informasi, Fakultas Sains dan Teknologi, UIN Ar-Raniry. Dalam penulisan proposal tugas akhir ini, penulis dengan segala kerendahan hati ingin mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Kedua orangtua saya, ayahanda (Mustafa.M) ibunda (Alm.Ratna Juwita) beserta ibunda sambung (Faridah) dan keluarga tercinta yang selalu memberikan doa dan dukungan.
2. Ketua Prodi Teknologi Informasi, Ibu Ima Dwitawati, M.B.A yang telah memberikan bimbingan dan arahan dalam penyusuanan proposal tugas akhir ini.
3. Dosen Pembimbing Proposal Tugas Akhir, Bapak Hendri Ahmadian, M.IM yang senantiasa memberikan arahan dan bimbingan dalam penyusunan proposal tugas akhir ini.
4. Dosen Pembimbing, Bapak Hendri Ahmadian, M.IM. yang senantiasa memberikan motivasi kepada penulis dalam kegiatan ini.
5. Bapak dan Ibu dosen Program Studi Teknologi Informasi yang telah membekali penulis dengan ilmu pengetahuan di bidang Teknologi Informasi.
6. *Staff* Prodi Teknologi Informasi, Ibu Cut Ida Rahmadiana, S.Si yang telah membantu penulis dalam pemberkasan administrasi.

7. Sahabat dan teman-teman yang selalu memberikan dorongan dan semangat kepada penulis dalam menyelesaikan penyusunan proposal tugas akhir ini.
8. Pihak-pihak terkait lainnya yang membantu penulis dalam penyusunan proposal ini.

Banda Aceh, 28 Desember 2022
Penulis,

Nurhaliza
190705015

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Teknologi informasi dan komunikasi (TIK) tumbuh sangat pesat dalam beberapa tahun terakhir, masyarakat umum kini menggunakan internet sebagai alat komunikasi utama, yang menjadi latar belakang peralihan dari teknologi komunikasi ke teknologi yang sepenuhnya digital. (Rafiq, 2020). Banyaknya jumlah pengguna media sosial seperti *Twitter* di Indonesia sebagai bagian dari komunikasi, *twitter* dapat berkomunikasi melalui *tweet* yang dikirimkan ke dalam aplikasi. *Tweet* ini bisa positif dan beberapa negatif (Perwira et al., 2021). *Twitter* dapat memudahkan pengguna dalam menerima berita, menyebarkan informasi dan berkomunikasi antar pengguna, serta kebebasan berpendapat, namun akibat dari kebebasan berpendapat, pengguna lebih leluasa menyebarkan kalimat *abusive* (ujaran kebencian) (Yazid et al., 2022).

Abusive yang berupa ekspresi, tulisan, tindakan yang ditujukan untuk memprovokasi kekerasan atau diskriminasi terhadap seseorang dengan karakteristik masyarakat seperti ras, suku, jenis kelamin, orientasi seksual, agama atau karakteristik lainnya. Ujaran kebencian sering dilontarkan karena masyarakat tidak menyadari perbedaan antara kritik dan pernyataan yang dapat berkontribusi pada kejahatan karena berfikir semua itu berupa kebebasan berpendapat (Herwanto et al., 2021).

Banyaknya kalimat *abusive* yang beredar di media sosial akan memberikan dampak yang besar bagi korban, diantaranya kalimat-kalimat *abusive* yang di hadapi membuatnya merasa sedih, marah, dan bahkan malu. Merasa tidak aman dan takut untuk berbicara atau berbagi pandangan di media sosial. menghancurkan kepercayaan diri dan mulai meragukan nilai pendapat dan perasaannya sendiri. Mendorong untuk menjauh dari interaksi *online*, memiliki dampak jangka panjang pada kesejahteraan mental, mungkin mengalami penurunan suasana hati, kecemasan, dan bahkan masalah kepercayaan diri dalam

kehidupan nyata. Pada penelitian ini dilakukan deteksi penggunaan kalimat *abusive* melalui media sosial *twitter*, menggunakan pendekatan *Machine learning Machine learning* merupakan bagian dari *Artificial Intelligence* (AI). Salah satu metode AI yang digunakan pada saat penelitian berupa *deep learning*. Penggunaan metode *deep learning* diharapkan dapat mempersingkat waktu serta memiliki keakuratan yang tinggi dalam hal akurasi. Sehingga kedepannya penyebaran kalimat *abusive* bisa berkurang.

Beberapa penelitian telah dilakukan mengenai deteksi kalimat *abusive* dengan menggunakan beberapa metedo seperti CNN, KNN, SVM, *BERT*, Indo*BERT* dan juga beberapa algoritma seperti *Naïve Bayes* dan LSTM. Dari penelitian-penelitian tersebut dihasilkan akurasi dari 64% - 93%. Diantaranya "Deteksi Ujaran Kebencian dengan Metode Klasifikasi *Naïve Bayes* dan Metode N-*Gram* pada *Dataset Multi*-Label *Twitter* Berbahasa Indonesia" oleh (Yazid et al., 2022) penelitian tersebut memperoleh nilai akurasi sebesar 64,957%. Penelitian berikutnya oleh (Perwira et al., 2021) dengan penelitian yang berjudul "Deteksi Ujaran Kebencian pada Twitter Bahasa Indonesia menggunakan *Machine Learning*: *Reviu Literatur*" pada penelitian ini menggunakan metode *Reviu Literatur* dari arsitektur RNN, peneliti mendapatkan nilai akurasi sebesar 91%.

Berdasarkan latar belakang diatas, penelitian ini mengarah pada analisis penggunaan kalimat *abusive* pada media sosial *Twitter* menggunakan metode *Multilingual BERT*. Dari hasil penelitian ini diharapkan dapat mengetahui Metode *Multilingul BERT* yang diimplementasikan untuk mendeteksi kalimat *abusive* pada *Twitter* serta mengetahui kinerja masing – masing model dalam mendeteksi kalimat *abusive* pada *Twitter*.

1.2 Rumusan Masalah

Adapun rumusan masalah pada penelitian ini, sebagai berikut:

1. Bagaimana proses analisis penggunaan kalimat *abusive* pada media sosial *twitter* menggunakan metode *Multilingual BERT*?
2. Bagaimana tingkat akurasi analisis penggunaan kalimat *abusive* pada media sosial *twitter* menggunakan metode *Multilingual BERT*?

1.3 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah:

1. Mengetahui proses analisis penggunaan kalimat *abusive* pada media sosial *twitter* menggunakan metode *Multilingual BERT*.
2. Mengetahui tingkat akurasi analisis penggunaan kalimat *abusive* pada media sosial *twitter* menggunakan metode *Multilingual BERT*.

1.4 Batasan Masalah

Adapun batasan masalah pada penelitian ini, yaitu:

1. Bahasa pemrograman yang dipakai pada penelitian ini berupa *Python* dengan memakai *Google Colab* sebagai media untuk menjalankan program.
2. Data yang digunakan merupakan *dataset twitter* yang diambil dari *https://github.com/okkyibrohim/id-abusive-language-detection.git.*
3. Metode yang digunakan yaitu *Multilingual BERT*.

1.5 Manfaat Penelitian

Adapun manfaat dari penelitian ini adalah:

1. Memberikan pengetahuan tentang implementasi *deep learning* menggunakan metode *Multilingual BERT* untuk membantu menyelesaikan permasalahan pada analisis penggunaan kalimat *abusive* pada sosial media *twitter.*
2. Mengetahui tingkat akurasi dari implementasi *deep lerning* menggunakan metode *Multilingual BERT* pada analisis penggunaan kalimat *abusive* pada media sosial *twitter.*
3. Runutan proses dan hasil akhir dari penelitian ini diharapkan menjadi penganalisis penggunaan kalimat *abusive* pada media sosial *twitter* menggunakan metode *Multilingual BERT* dengan akurasi lebih baik.

BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Berkaitan dengan penelitian yang diteliti oleh penulis dengan menggunakan teknologi *deep learning*, penelitian sebelumnya yang berkaitan dengan penelitian yang dilakukan oleh penulis sangat dibutuhkan sebagai referensi dalam mengembangkan penelitian yang akan dilakukan penulis.

Dalam penelitian mengenai analisis penggunaan kalimat *abusive*, banyak peneliti menggunakan beberapa metode, seperti KNN, SVM, *BERT*, Indo*BERT*, *Naïve Bayes*, LSTM dan CNN. Dari penelitian-penelitian tersebut dihasilkan akurasi dari 64% - 93%. Diantaranya "Deteksi Ujaran Kebencian dengan Metode Klasifikasi *Naïve Bayes* dan Metode N-*Gram* pada *Dataset Multi*-Label *Twitter* Berbahasa Indonesia" oleh (Yazid et al., 2022) penelitian tersebut memperoleh nilai akurasi sebesar 64,957%. Penelitian berikutnya oleh (Perwira et al., 2021) dengan penelitian yang berjudul "Deteksi Ujaran Kebencian pada *Twitter* Bahasa Indonesia menggunakan *Machine Learning*: *Reviu Literatur*" pada penelitian ini menggunakan metode *Reviu Literatur* dari arsitektur RNN, peneliti mendapatkan nilai akurasi sebesar 91%.

Penelitian yang diteliti (Hadiyan et al., 2021) dengan judul "Deteksi Penggunaan Kalimat *Abusive* Pada Teks Bahasa Indonesia Menggunakan Metode Indo*BERT*" mendapatkan hasil akurasi sebanyak 92,46% dengan membandingkan dengan beberapa metode yaitu KNN, SVM, *Naive Bayes*, *BERT Multilingual Base* dan *BERT Base*. "*Hate Speech Detection in Indonesian Twitter using Contextual Embedding Approach*" yang diteliti oleh (Herwanto et al., 2021) memperoleh hasil akurasi sebanyak 80% menggunakan metode RNN dan GPU sebagai arsitektur utamanya. Hasil akurasinya dipengaruhi oleh banyaknya data dan perbedaan label pada *dataset* yang diteliti. Penelitian lainnya juga diteliti oleh (Ibrohim & Budi, 2018) yang berjudul "*A Dataset and Preliminaries Study for Abusive Language Detection in Indonesian Social Media*" memiliki tingkat

akurasi 70% - 83% menggunakan algorima *Naïve Bayes* dan SVM juga menggunakan *Random Forest Decision Tree Classifier* untuk mendeteksi *tweet*.

Tabel 2. 1 perbandingan penelitian terdahulu

PENELITI	METODE	KASUS	*DATASET*	AKURASI
(Yazid et al., 2022)	*Naïve Bayes* dan N-*Gram*	Deteksi Ujaran Kebencian dengan Metode Klasifikasi *Naïve Bayes* dan Metode N-*Gram* pada *Dataset Multi*-Label *Twitter* Berbahasa Indonesia	*Dataset multi*-label dari penelitian sebelumnya sebanyak 11.809 data	64,957%.
(Perwira et al., 2021)	RNN (*Reviu Literatur*)	Deteksi Ujaran Kebencian pada *Twitter* Bahasa Indonesia menggunakan *Machine Learning*: *Reviu Literatur*	Mengumpulkan literatur – literatur dari *Google Scholar* dan IEEE *Xflore* sebanyak 34.531 *literatur*	91%

PENELITI	METODE	KASUS	DATASET	AKURASI
(Hadiyan et al., 2021)	Indo*BERT*	Deteksi Penggunaan Kalimat *Abusive* Pada Teks Bahasa Indonesia Menggunakan Metode Indo*BERT*	*Dataset* diambil dari *dataset* yang sudah dibangun dari penelitian terdahulu berupa komentar berita *online* di *Facebook* sebanyak 3184 komentar	92,46%
(Nabiilah et al., 2023)	*Multilingual BERT*, Indo*BERT*	*BERT base model for toxic comment analysis on Indonesian social media*	Menggunakan *dataset* dari peneliti terdahulu ya diambil dari peneliti Ahmad Izzan	88%
(Ibrohim & Budi, 2018)	*Naïve Bayes* dan SVM	*A Dataset and Preliminaries Study for Abusive Language Detection in Indonesian Social Media*	*Dataset Twitter* yang di *crowling* sendiri sebanyak 2.500 data	70%- 83%

Kesimpulan yang penulis dapat dari hasil perbandingan penelitian sebelumnya yang berkaitan dengan penelitian yang dilakukan oleh penulis mengenai analisis kalimat *abusive* dengan metode *Multilingual BERT* adalah:

- Jumlah dataset yang digunakan untuk *training* sangat mempengaruhi tingkat akurasi pengujian. Semakin banyak dataset yang digunakan, maka tingkat akurasi dari hasil pengujian analisis kalimat *abusive* akan lebih tinggi.
- Data yang digunakan pada proses deteksi kalimat *abusive* harus melewati tahapan *pre-processing* dengan baik untuk menjaga keseimbangan data pada saat pengujian. Data yang tidak melewati tahapan *pre-processing* dengan baik, akan mengurangi hasil akurasi.

2.2 Media Sosial *Twitter*

Twitter adalah layanan media sosial yang memungkinkan teman, keluarga, dan kolega berkomunikasi dan tetap berhubungan dengan bertukar pesan secara cepat dan sering. Pengguna mengirim *tweet*, yang dapat berisi foto, video, tautan, dan teks (*Twitter*, 2023). *Twitter* dapat berkomunikasi melalui *tweet* yang dikirimkan ke dalam aplikasi. *Tweet* ini bisa positif dan beberapa negatif. Komentar negatif menjadi masalah karena biasanya berisi ujaran kebencian dan bisa berujung pada tindakan hukum bagi pembuatnya (Perwira et al., 2021). *Twitter* juga merupakan salah satu *platform* media sosial paling populer untuk diseminasi ilmiah di bidang *kardiologi*. *Platform* ini dapat menjadi alat baru yang penting untuk diskusi ilmiah, diseminasi hasil, kolaborasi dan pengembangan pertanyaan penelitian baru.

Aktivitas *twitter* meningkat dengan munculnya pertemuan *kardiologi virtual* dan *hybrid*, didukung oleh para pemimpin opini utama di lapangan dan dibantu oleh peningkatan penggunaan duta *twitter*. Selain itu, mempromosikan artikel penelitian di *twitter* dapat memengaruhi tingkat kutipan secara positif di masa mendatang. Orang bisa berspekulasi tentang keberlanjutan media sosial, khususnya *twitter*, untuk penggunaan profesional, karena perkembangan teknologi seperti itu bisa mengalami fase "*euforia*". *Platform* ini sangat ideal untuk

membuat pembaca diperbarui dengan sangat cepat, karena *tweet* dibatasi hingga 260 karakter dan oleh karena itu hanya berisi pesan yang benar-benar diperlukan. Pengguna profesional terus bertambah karena media sosial menawarkan akses yang mudah dan cepat ke topik yang diminati dan kesempatan untuk berjejaring dengan kolega *offline* (Benjamin Fyenbo et al., 2022).

2.3 Definisi Kalimat *Abusive*

Kalimat *abusive* (ujaran kebencian) adalah pernyataan yang menyerang seseorang atau kelompok karena jenis kelamin, etnis, agama, ras, disabilitas, atau orientasi seksual. Ujaran kebencian juga memiliki tingkat ancaman, semakin tinggi tingkat ancaman ujaran kebencian, semakin cepat penyebarannya, mulai dari konflik antar individu hingga konflik antar kelompok. (Yazid et al., 2022).

Isu terkait kebebasan berbicara dan regulasi berbicara bukanlah hal yang unik di Indonesia. Negara-negara yang menghormati kebebasan berbicara dan memiliki masyarakat majemuk selalu memiliki masalah yang sama. Kebebasan berekspresi yang berlebihan dan tidak terkendali dapat membahayakan kelompok minoritas. Namun, kebebasan berekspresi masyarakat jangan terlalu dibatasi, karena bisa "membekukan" perubahan sosial.

Sesuai Surat Edaran SE/06/X/2015 Kepolisian Negara Republik Indonesia tentang penggunaan kalimat yang menyinggung (kebencian). Kalimat *abusive* dapat memicu kebencian kolektif, pengucilan, diskriminasi, kekerasan dan bahkan pada tingkat yang paling mengerikan seperti pembantaian etnis atau genosida terhadap kelompok yang terkena dampak ujaran kebencian. Melalui surat edaran ini, kami berharap masyarakat khususnya pengguna internet sangat berhati-hati dalam menyampaikan pendapatnya di ruang publik, khususnya di bidang jejaring sosial. Tindak tutur kekerasan (*hate speech*) termasuk dalam kejahatan terhadap kehormatan, istilah lain yang juga sering digunakan untuk kejahatan kehormatan adalah kejahatan pencemaran nama baik. Objek atau tujuan dari pasal tersebut, oleh karena itu melindungi kehormatan, lebih merupakan kejahatan terhadap kehormatan dari sudut pandang objek (Pasaribu et al., 2020).

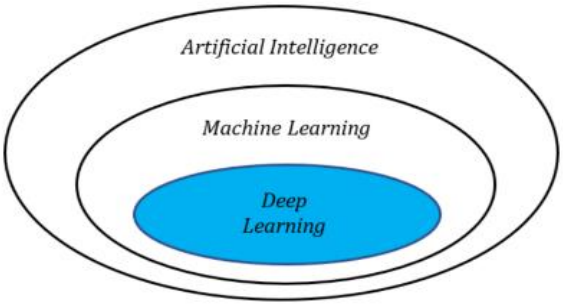
2.4 *Artificial Inteligence*

Artificial Intelligence atau AI singkatnya, adalah kecerdasan buatan yang kemudian dapat memudahkan pekerjaan manusia. AI sendiri bisa meniru perilaku manusia jika segala sesuatu yang dilakukannya dianggap pintar atau cerdas. AI juga didefinisikan sebagai kecerdasan makhluk ilmiah. Andreas Kaplan dan Michael Haenlein menerjemahkan kecerdasan buatan sebagai "kemampuan suatu sistem untuk menafsirkan data eksternal dengan benar, belajar darinya, dan menggunakan pembelajaran itu untuk mencapai tujuan dan tugas tertentu melalui pengadaptasian yang *fleksibel* (Siahaan et al., 2020).

AI sendiri bekerja dengan menggabungkan keberadaan beberapa kumpulan data, pemrosesan berulang, dan algoritma cerdas. Ini sebenarnya memungkinkan perangkat lunak secara otomatis mempelajari tentang pola atau properti data. AI juga bisa dikatakan sebuah bidang studi yang amat sangat luas. Cakupan teori, metode, teknologi dan sub bidang yang ada pada AI sangatlah banyak meliputi pembelajaraan mesin, jaringan *neural*, komputasi *kognitif*, visi komputer, kemudian pemprosesan bahasa secara ilmiah (Luh Putu Ary Sri Tjahyanti et al., 2022).

2.5 *Deep Learning*

Deep learning berupa sebuah metode eksplorasi terhadap data yang bertujuan untuk membuat representasi (abstraksi) data secara bertahap menggunakan beberapa lapisan pemrosesan data. Dalam kasus *deep learnig*, penting agar representasi data tidak dilakukan secara eksplisit oleh manusia, tetapi dibuat menggunakan algoritma pembelajaran (Yaya Heryadi, 2020).



Gambar 2. 1 *Deep Learning* sebagai cabang ilmu *Artificial Intelligence* (AI) (Yaya Heryadi, 2020)

Gambar 2. 1 menggambarkan bahwa *Deep learning* adalah sebuah bidang dalam AI yang merupakan turunan dari *machine learning*. *Machine learning* sendiri merupakan sub-bidang dari AI yang berfokus pada pengembangan algoritma dan teknik untuk mengajarkan komputer untuk belajar dari data, tanpa perlu secara *eksplisit* diprogram. *Deep learning* adalah sebuah pendekatan dalam *machine learning* yang menggunakan arsitektur jaringan saraf tiruan (*neural networks*) yang terdiri dari banyak lapisan (*deep layers*). Oleh karena itu, istilah "*deep*" digunakan untuk menggambarkan jaringan dengan banyak lapisan (Yaya Heryadi, 2020).

2.6 *Natural Language Processing*

Natural language processing atau disingkat dengan NLP adalah suatu cabang ilmu dari ilmu komputer, atau lebih spesifiknya NLP merupakan cabang ilmu dari AI (Rayyan, 2022). NLP juga dapat disebut sebagai komputasi *linguistik*, bahasa komputer, dan pemrosesan bahasa atau teknologi bahasa manusia. NLP berkaitan dengan bagaimana menganalisa bahasa manusia baik dalam bentuk tertulis maupun lisan (Mubarok, 2021).

Beberapa teknik yang sering digunakan dalam proses penyelesaian masalah menggunakan NLP, yaitu:

- *Sentence segmentation*

  Sentence segmentation merupakan teknik NLP untuk menentukan unit proses yang terdiri dari beberapa kata. Teknik ini digunakan untuk melakukan identifikasi terhadap batas kalimat.

- *Tokenization*

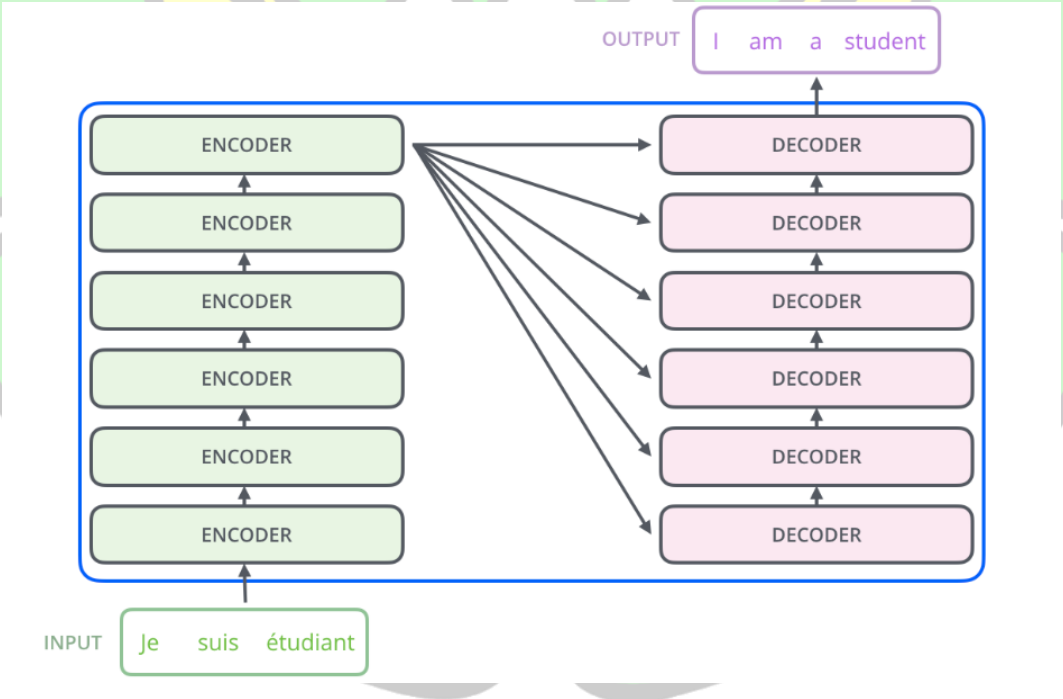
  Teknik merupakan teknik dasar pada NLP. Teknik ini bekerja dengan cara memisahkan kalimat menjadi token-token yang berisi kata.

- *Stopword removal*

  Stopword removal merupakan teknik pada NLP yang digunakan untuk menghapus kata yang bersifat *non-informatif* untuk meningkatkan tingkat akurasi hasil (Mubarok, 2021).

2.7 *Transformer*

Transformer adalah model *deep learning* yang menerapkan mekanisme dari *self-attention*, sebuah mekanisme untuk mengetahui hubungan kontekstual antar kata. *Transformer* terutama digunakan dalam pemrosesan bahasa alami dan *computer vision*. Model ini dirancang untuk memproses tipe data *sekuens* seperti bahasa alami untuk melakukan tugas seperti terjemahan bahasa dan peringkasan teks. *Transformer* terdiri dari dua buah tumpukan (*stacks*) (Rayyan, 2022).



Gambar 2. 2 Model *Transformer* (Alammar, 2018)

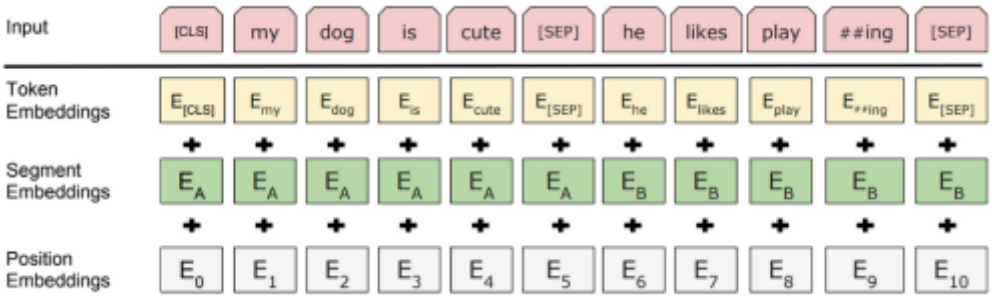
Dapat dilihat dari Gambar 2. 2 *Transformer* memiliki dua komponen utama yaitu *encoder* dan *decoder*. *Encoder* bertugas untuk membaca input dan memahami konteks dari *input*, adapun *decoder* bertugas untuk memproduksi hasil prediksi, semisal pada tugas menerjemahkan sebuah kalimat bahasa inggris ke perancis (Rayyan, 2022).

Transformer menggunakan *attention* untuk meningkatkan kecepatan yang dapat digunakan untuk melatih model. *Transformer* mengungguli model *Google Neural Machine Translation* dalam tugas tertentu. Manfaat terbesar *transformer* yaitu cocok untuk paralelisasi (Alammar, 2018).

2.8 *Bidirectional Encoder Representations from Transformers*

Bidirectional Encoder Representations from Transformers (*BERT*) berupa model sebuah *machine learning* berbasis *transformer* untuk *natural language processing* yang dibangun oleh *Google*. Model ini dapat melakukan banyak tugas dibidang *natural language processing*, seperti menerjemahkan dari suatu bahasa lain, menyimpulkan teks, dan membuat system *question answering* untuk *chatbot* (Rayyan, 2022). Selain itu, *BERT* tidak hanya mampu menganalisis teks bahasa Inggris, tetapi juga pandai menganalisis bahasa lain seperti bahasa Arab, Prancis, dan Cina, menggunakan korpus klinis tanpa label untuk menyempurnakan model bahasa *BERT* sebelum melatih pengklasifikasi teks dan mencapai hasil yang lebih baik daripada model pembelajaran mendalam lainnya yang lebih canggih (Jiang & He, 2020).

Pada dasarnya, arsitektur *BERT* merupakan tumpukan *encoder* dari *transformer* yang dirancang secara dua arah (*bidirectional*) sehingga model ini memiliki kemampuan memahami konteks dan aliran bahasa lebih baik daripada dengan model searah. Tujuan utama penggunaan model *BERT* adalah untuk menghasilkan sebuah model representasi dari suatu bahasa sehingga *BERT* hanya memerlukan *encoder* dari *transformer* saja.



Gambar 2. 3 Representation Input Model BERT (Rayyan, 2022)

Jenis *input* pada *encoder BERT* yang ditunjukkan pada Gambar 2. 3 yaitu kumpulan kata atau token yang berurutan, token-token itu telah melalui tahap *embedding* yang terdiri dari token *embeddings*, *segment embeddings* dan *positional embeddings*. Hasil dari proses *embedding* berupa *sekuens* yang bersesuaian pada *sekuens input* dengan *indeks* yang sama. Setelah itu *sekuens* akan diteruskan menuju *feed-forward neural network* dan *softmax layer*.

2.9 *Multilingual BERT*

Multilingual BERT adalah versi *BERT* dengan multi-bahasa, *Multilingual BERT* berupa salah satu *pre-trained* model *BERT* namun yang membedakan *Multilingual BERT* dengan model lainnya yaitu *Multilingual BERT* dilatih dengan 104 bahasa termasuk bahasa Indonesia (Girinoto et al., 2022). *Multilingual BERT* sangat bagus dalam *transfer* model lintas bahasa *zero-shot*, di mana anotasi khusus tugas dalam satu bahasa digunakan dalam menyempurnakan model untuk evaluasi pada bahasa lain. Kinerja *Multilingual BERT* sebagian besar tidak bergantung pada tumpang tindih bahasa, yang menunjukkan bahwa *Multilingual BERT* mempelajari representasi multi-bahasa lebih dalam daripada menghafal kosa kata sederhana (Pires et al., 2020).

2.10 Indo*BERT*

Indo*BERT* adalah model berbasis *transformer* dengan mengadaptasi *BERT*, tetapi dilatih murni sebagai *masked language* model yang dilatih menggunakan *huggingface* dengan mengikuti konfigurasi *BERT base* (*uncased*) (Amelia, 2021). Model ini mempunyai 4 versi yang berbeda, yaitu Indo*BERT*-

BASE, Indo*BERT-LARGE*, dan 2 sisanya berbasis model *ALBERT*, yaitu indo*BERT-liteBASE* dan Indo*BERT-liteLARGE*. Indo*BERT* dilatih menggunakan Indo4B *dataset* dan TPUv3-8 dalam dua fase (Rayyan, 2022).

Indo*BERT* adalah modifikasi dari *BERT Base* yang diprakarsai oleh tim IndoNLU, mahakarya untuk *sentiment* analisis dalam bahasa Indonesia. Model ini menjadi populer baru-baru ini karena dilatih dengan sekitar 4 miliar *Word* Korpus. Model ini dilatih menggunakan lebih dari 220 juta kata yang dikumpulkan dari tiga sumber utama yaitu: Bahasa Indonesia, Wikipedia, artikel berita dari Kompas, Tempo dan Liputan6 juga Indonesian *Web* Corpus. Sumber daya model pra-pelatihan ini dapat diakses dan mudah direproduksi. Indo*BERT* menggunakan mekanisme transformator yang mempelajari hubungan antar kata dalam sebuah teks/kalimat, ia dilatih murni sebagai penyamaran pelatihan model bahasa menggunakan *framework Huggingface* (Mubaraq & Maharani, 2022).

2.11 *Confusion Matrix*

Confusion Matrix merupakan teknik yang dipakai untuk menyimpulkan performa dari sebuah algoritma klasifikasi. Terdapat 4 metrik di dalam *confusion matrix*, yaitu:

- *True Positive* (TP)
  Jumlah prediksi yang berhasil diprediksi oleh model kelas positif
- *True Negative* (TN)
  Jumlah prediksi yang berhasil diprediksi oleh model kelas negatif
- *False Positive* (FP)
  Jumlah prediksi yang gagal diprediksi oleh model kelas positif sebagai negatif
- *False Negative* (FN)
  Jumlah prediksi yang gagal diprediksi oleh model kelas negatif sebagai positif (Rayyan, 2022).

Confusion matrix dapat digunakan dalam menghitung nilai performa model akurasi, presisi, rekal dan F_1 *Score*. Metrik evaluasi ini memberikan informasi yang lebih rinci tentang kinerja model.

- Akurasi mengukur proporsi prediksi yang benar dari total prediksi.

  Akurasi = $\frac{TP+TN}{TP+TN+FP+FN} * 100\%$ (10.1)

- Presisi mengukur sejauh mana prediksi positif yang dilakukan oleh model benar.

  $Presisi = \frac{TP}{FP+TP} * 100\%$ (10.2)

- Rekal mengukur sejauh mana model berhasil mengidentifikasi semua *instance* yang benar positif.

  $Rekal = \frac{TP}{FN+TP} * 100\%$ (10.3)

- F1-*Score* adalah harmonik dari presisi dan rekal, memberikan keseimbangan antara keduanya.

  $F1 = \frac{2 \times Rekal \times Presisi}{Rekal+Presisi} * 100\%$ (10.4)

2.12 *Tools*

Tools yang digunakan penulis dalam penelitian ini adalah bahasa pemrograman *Python* dan *Google Colaboratory*. *Tools* tersebut akan digunakan penulis dalam proses deteksi penggunaan kalimat *abusive* pada sosial media *twitter* menggunakan metode *Multilingual BERT*. *Tools* yang digunakan penulis dapat digunakan *online* pada *website google colab* dan instalasi semua *tools* yang dibutuhkan.

2.12.1 *Python*

Python adalah salah satu bahasa pemrograman tingkat tinggi (*high-level programming language*), berjalan dengan sistem *interpreted*, dan bisa digunakan untuk berbagai tujuan (*general purpose*) (Gumilar et al., 2021). *Python* adalah bahasa pemrograman yang ditafsirkan secara *universal* dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. *Python* dikatakan sebagai bahasa yang menggabungkan kapabilitas dan kemampuan dengan sintak kode yang sangat jelas dan dilengkapi dengan fungsionalitas pustaka standar yang luas dan *komprehensif*.

Python juga didukung oleh komunitas yang besar. *Python* sering digunakan sebagai bahasa *script*, meskipun dalam praktiknya, penggunaan bahasa yang lebih luas mencakup konteks pemanfaatan yang umumnya tidak dilakukan dengan menggunakan bahasa *script*. *Python* dapat digunakan untuk berbagai aplikasi pengembangan perangkat lunak dan dapat digunakan pada beberapa *platform* sistem operasi (Syahrudin & Kurniawan, 2018).

2.12.2 *Google Colaboratory*

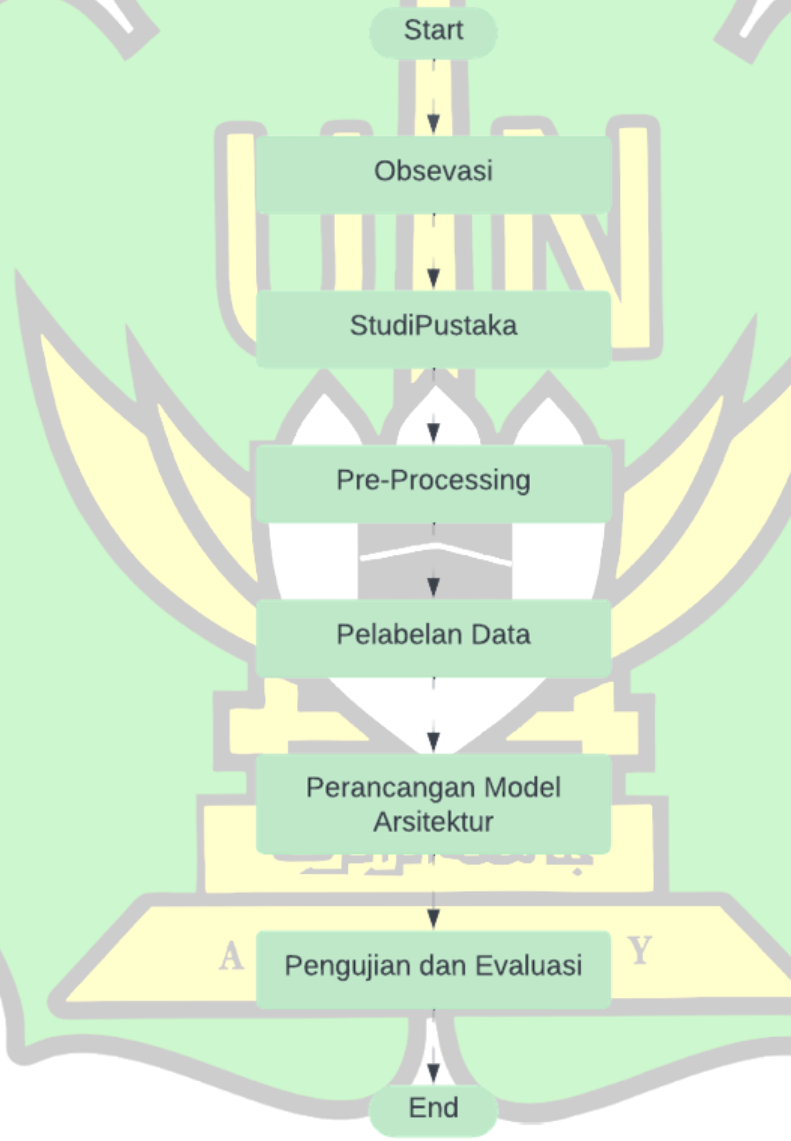
Colaboratory, atau "*Colab*" merupakan produk dari *Google Research*. *Colab* memungkinkan siapa saja menulis dan menjalankan kode *python* melalui *browser*, dan sangat cocok untuk *machine learning*, analisis data, dan pelatihan. Dalam istilah yang lebih teknis, *Colab* berupa layanan *notebook Jupyter* yang dihosting dan dapat digunakan tanpa penginstalan, serta menyediakan akses gratis ke *resource* komputasi, termasuk GPU (Soen et al., 2022).

Colab mengimplementasikan Fitur *sharing* yang berpotensi menjadi solusi untuk pembatasan akses. *Google colaburatory* dapat terkoneksi dengan *google drive* untuk mencegah hilangnya kode yang sudah dikerjakan user dikarenakan *error* atau *bug* pada *system* (Febrian Sengkey et al., 2020).

BAB III
METODOLOGI PENELITIAN

3.1 Tahapan Penelitian

Tahapan penelitian ini menjelaskan alur penelitian yang digunakan oleh penulis pada penelitian ini. Runutan alur tahapan penelitian dapat dilihat pada Gambar 3.1.



Gambar 3. 1 Diagram Alur Tahapan Penelitian

3.2 Metode Pengumpulan Data

Metode pengumpulan data yang digunakan penulis pada penelitian ini berdasarkan dari *observasi* dan studi literatur. *Observasi* dilakukan penulis untuk mengidentifikasi masalah penelitian mengenai deteksi kalimat *abusive* pada sosial media *twitter.* Studi literatur dilakukan oleh penulis untuk mendapatkan referensi tentang perumusan metode yang akan di implementasikan pada penelitian ini.

Data yang digunakan dalam penelitian ini adalah komentar pengguna di media sosial *twitter* berupa teks. peneliti mengumpulkan kembali data dari penelitian terdahulu pada penelitian yang sudah dilakukan oleh (Ibrohim & Budi, 2018) Dataset diambil melalui link *Github.com*, *dataset* yang digunakan sebanyak 2016 *tweet*. Data tersebut nantinya akan diberi label sesuai dengan tingkat *abusive* kalimat pada setiap *tweet*.

3.3 Pengambilan Data

Pada tahap ini data yang digunakan oleh penulis tidak lagi dilakukan pada tahap *pre-processing* karena kumpulan data yang digunakan pada penelitian ini sudah mencantumkan label sehingga dapat langsung digunakan pada model yang ingin dilatih. Namun, proses pra-pemrosesan diperlukan agar data mentah akan diproses untuk mengambil informasi yang dikandungnya karena masih banyak data dan simbol yang tidak perlu. Pra-pemrosesan data dilakukan dengan membuang data yang tidak perlu atau yang tidak sesuai agar data tersebut lebih banyak diolah, untuk menghasilkan data yang sesuai dengan kebutuhan penelitian.

3.4 Pelabelan Data

Setelah *dataset* telah terkumpul, tahapan selanjutnya adalah melakukan pelabelan data. Pada *dataset* yang dipakai untuk penelitian ini telah diberikan label. Setiap ulasan telah diberikan dua label yaitu label *Not Abusive Language dan Abusive Language.*

3.5 Metode Simulasi

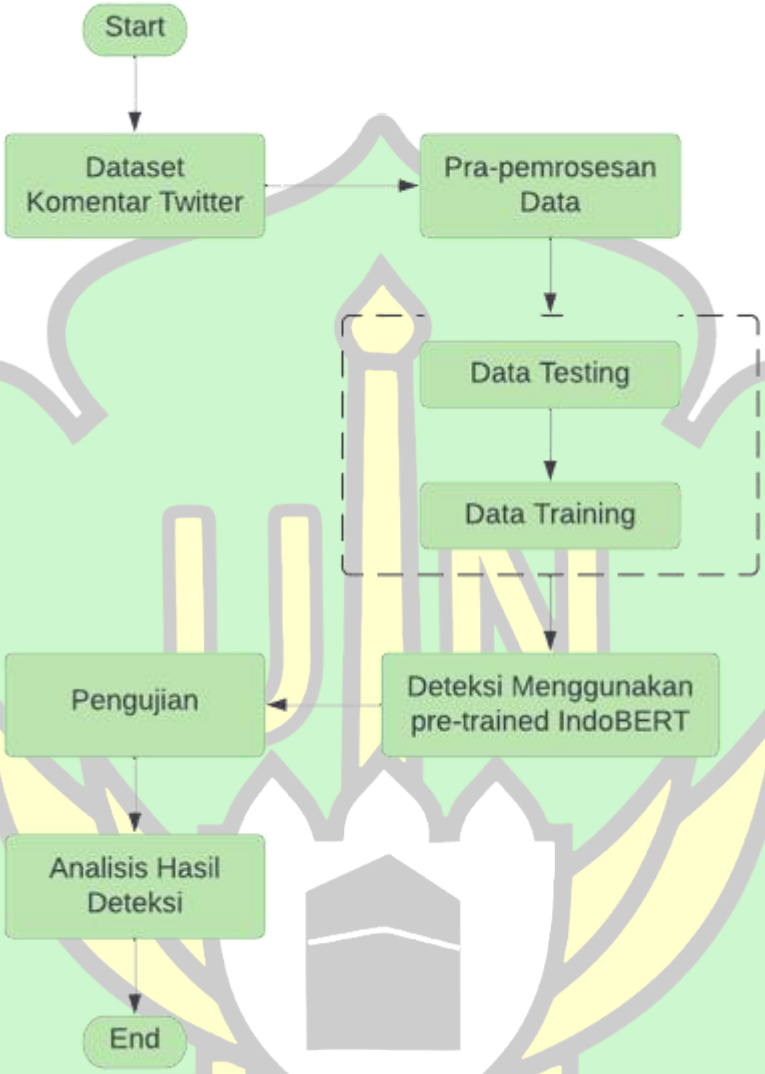
Metode simulasi merupakan skenario penelitiaan yang akan diimplementasikan pada sistem. Metode simulasi ini bertujuan untuk memberikan gambaran kinerja metode yang akan digunakan. Adapun tahapan pada metode simulasi penelitian ini teridiri dari dua tahapan utama yaitu perumusan konsep penelitian dan implementasi metode dan arsitektur.

3.5.1 Perumusan Konsep Penelitian

Konsep penelitian yang akan dilakukan penulis dirumuskan berdasarkan hasil dari identifikasi masalah dan studi literatur. Perumusan konsep penelitian merupakan tahapan penelitian yang dilakukan penulis untuk menentukan metode, arsitektur dan *tools* apa yang dapat diaplikasikan dalam studi kasus yang penulis akan teliti. Berdasarkan studi dan observasi yang dilakukan penulis, penelitian ini mengguna metode *Multilingual BERT* untuk proses deteksi penggunaan kalimat *abusive* pada media sosial *twitter.*

3.5.2 Implementasi Metode dan Arsitektur

Tahap implementasi metode dan arsitektur merupakan tahapan untuk mentransformasikan dan merealisasikan rumusan konsep penelitian pada sistem yang dibangun oleh penulis. Implementasi metode dan arsitektur terdiri dari alur sistem yang dikembangkan pada penelitian. Berikut ini penulis mengilustrasikan alur sistem dalam proses deteksi penggunaan kalimat *abusive* pada media sosial *twitter* menggunakan metode *Multilingual BERT*.



Gambar 3. 2 Diagram Alur Sistem

Rincian alur sistem pada tahapan implementasi adalah sebagai berikut:

- Pemilihan *Dataset*

Data yang digunakan dalam penelitian ini adalah komentar pengguna di media sosial *twitter* berupa teks. peneliti mengumpulkan kembali data dari penelitian terdahulu pada penelitian yang sudah dilakukan oleh (Ibrohim & Budi, 2018) *Dataset* diambil melalui *link Github.com*, *dataset* yang digunakan sebanyak 2016 *tweet*. Data tersebut nantinya akan diberi label sesuai dengan tingkat *abusive* kalimat pada setiap *tweet*. Berikutnya, *dataset* dibagi menjadi data *training* dan

data *testing* dengan proporsi masing-masing, yaitu 80% dan 20% dari keseluruhan data. Data *training* akan dibagi lagi menjadi data *training* itu sendiri dan data validasi dengan proporsi 70% dan 30% dari jumlah data *training*.

- Data *training*

Data *training* merupakan *dataset* yang digunakan untuk menghasilkan model sistem dari metode *Multilingual BERT*. Proses sistem pada tahap ini adalah untuk mempersiapkan inputan data yang akan digunakan pada tahap pelatihan.

- Data *testing*

Data *testing* adalah data yang digunakan untuk menghasilkan *value* evaluasi. Proses sistem pada tahap ini adalah untuk menginputkan data yang akan berbentuk matriks untuk menggambar hasil proses sistem.

3.6 Metode Analisis Hasil

Metode analisis hasil merupakan tahapan terakhir pada penelitian yang dikembangkan oleh penulis. Analisis hasil dilakukan dengan melakukan evaluasi terhadap sistem yang telah diterapkan penulis pada penelitian ini.

3.6.1 Evaluasi

Evaluasi model merupakan tahap yang harus selalu dilakukan untuk mengetahui apakah model tersebut bekerja dengan baik atau tidak. Evaluasi model dengan data yang sama pada saat melakukan *training* mengakibatkan *overfitting* yang mengakibatkan tidak bisa menggeneralisasikan data dengan baik dikarenakan model dapat mengingat data yang telah dipakai saat *training*.

Cara evaluasi model dengan baik yaitu menyisihkan data dari keseluruhan data untuk diuji pada model, yaitu membagi *dataset* menjadi tiga set (*training*, validasi dan *test*). Dalam *machine learning* terdapat banyak variasi metrik untuk mencari nilai, sehingga ketetapan memilih metrik yang sesuai *task* merupakan hal yang harus. Pada penelitian ini, metrik yang digunakan adalah *confusion matrix* yang digunakan untuk menghitung nilai performa model menggunakan akurasi, rekal, presisi dan F_1 *score*.

3.7 Alat Bantu Penelitian

Alat bantu pada penelitian ini adalah perangkat keras dan perangkat lunak komputer. Perangkat keras yang digunakan adalah satu unit *VivoBook* 14_ASUS Laptop X441MA_X441MA dengan spesifikasi sebagai berikut:

- *Processor* Intel(R) *Celeron*(R) N4000 CPU @ 1.10GHz 1.10 GHz.
- RAM 8.00 GB (7.83 GB *usable*).
- *Storage* 460 GB TOSHIBA MQ04ABF100 HDD

Perangkat lunak yang digunakan pada penelitian ini adalah sistem operasi *Microsoft Windows 11 Home Version 21H2* dan penggunaan *tools* bahasa pemrograman *python* dan *google colaboratory*. *Tools* tersebut akan digunakan penulis untuk proses deteksi kalimat *abusive* pada sosial media *twitter* menggunakan metode *Multilingual BERT*.

Spesifikasi *google coolab* yang dipakai untuk menjalankan penelitian ini sebagai berikut:

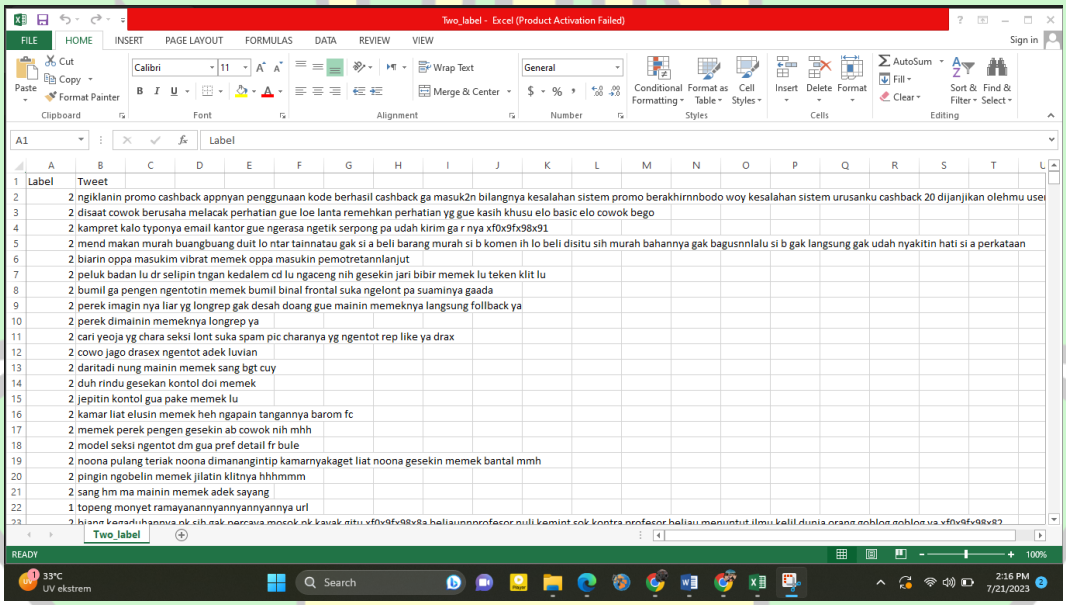
- *Python* 3 *type* T4 *Google Compute Engine backend* (GPU)
- *System* RAM 1.3 / 12.7 GB
- *Disk* 24.2 / 78.2 GB

BAB IV
HASIL DAN PEMBAHASAN

Pada bab ini, penulis akan menjelaskan hasil dari pengambilan data, proses *pre-processing*, serta menunjukkan hasil uji coba metode yang digunakan. Lalu akan ditunjukkan pula evaluasi model menggunakan *confusion matrix*.

4. 1 Pengambilan Data

Data yang diperoleh merupakan *dataset* yang diambil dari *link github*: *https://github.com/okkyibrohim/id-abusive-language-detection.git*. *Dataset* yang diambil yaitu *abusive language detection* dengan jumlah sampel data sebanyak 2016 dan 2 label. *Dataset* tersebut terdapat 2 atribut berupa *tweet* dan label, seperti yang terlihat pada Gambar 4. 1.



Gambar 4. 1 Sampel Dataset

Label 1 berupa kalimat *not abusive language*, dan label 2 berupa kalimat *abusive language*. Setiap label terbagi menjadi beberapa *sentences*, label 1 terdapat 331 *sentences* dan label 2 terdapat 1685 *sentences* yang dapat dilihat pada Tabel 4. 1.

Tabel 4. 1 Jumlah Data Setiap Label

Label	Keterangan	Jumlah
1	*Not Abusive Language*	331
2	*Abusive Language*	1685

4. 2 *Pre-processing* data

Pre-processing data merupakan tahapan untuk mempersiapkan data menjadi data yang lebih terstruktur untuk bisa diolah ke tahapan berikutnya. Langkah-langkah yang dilakukan adalah *case folding*, menghapus simbol-simbol, *remove slang*, *stopword removal*, *tokenizing* dan *stemming*. Hasil dari *pre-processing* data bisa dilihat pada Tabel 4. 2.

Tabel 4. 2 Hasil Pre-Processing *Dataset*

Label	Sebelum *Pre-Processing*	Sesudah *Pre-Processing*
1	Jangan suka pamer.. \nKarena gak ada orang yg benar2 peduli tentang apa yg kau pamer kan. \nBukan pujian yg akan kau dapatkan, melainkan mereka akan mengatakan 'KAMPUNGAN'."	suka pamer nkarena gak orang yg benar2 peduli yg kau pamer nbukan pujian yg kau dapatkan kampungan.
2	USER USER USER hahaha bener lo USER giliran majikan dia mesum 7 taon si kunyuk mingkem USER sok suci pula ngatain 'mikirin sex' muke lacur lu nyuk.."	user user user hahaha bener lo user giliran majikan mesum 7 taon si kunyuk mingkem user sok suci ngatain mikirin sex muke lacur lu nyuk

Sebelum masuk dalam pelatihan model *Multilingual BERT*, data dalam *dataset* yang masih berbentuk *sentence* akan dilakukan tokenisasi yang setiap kalimatnya akan diubah menjadi token-token kecil untuk memfasilitasi proses analisis. Tokenisasi ini diambil dari repositori model *Hugging Face* dengan Metode "*from_pretrained*" yang secara otomatis mengunduh *file tokenizer* jika belum ada dalam *cache* local dan "*uncased*" menandakan bahwa *tokenizer* tidak membedakan huruf kapital dan huruf kecil. Setelah dilakukannya tokenisasi, setiap token akan diubah menjadi ID *numerik* yang sesuai dari kamus model *BERT*. Seperti pada Gambar 4. 2.

```
from transformers import BertTokenizer

print("Loading BERT Tokenizer")
tokenizer = BertTokenizer.from_pretrained('bert-base-
multilingual-uncased', do_lower_case=True)
============================================================
print("Original: ", sentences[124])

print("Tokenized: ", tokenizer.tokenize(sentences[124]))

print("Token IDS: ",
tokenizer.convert_tokens_to_ids(tokenizer.tokenize(sentences[1
24])))
```

```
Original:  user babi kaulahhh maju lu anjinggg
Tokenized:  ['user', 'bab', '##i', 'ka', '##ula', '##h', '##h', '##h',
'maju', 'lu', 'anjing', '##gg']
Token IDS:  [24934, 39711, 10116, 10237, 13426, 10243, 10243, 10243,
24943, 12993, 68758, 20738]
```

Gambar 4. 2 Tokenisasi Data

Setelah melakukan tokenisasi, penulis akan melakukan *padding* data pada semua kalimat agar memiliki panjang yang sama sebesar "*MAX_LEN*" yang ditentukan, yaitu 64. Proses ini bertujuan untuk memastikan bahwa semua kalimat yang dimasukkan ke dalam model memiliki panjang yang sama dengan nilai *value* = 0 yang digunakan sebagai *padding* token ID. Dapat dilihat pada Gambar 4. 3

```
from tensorflow.keras.preprocessing.sequence import
pad_sequences

MAX_LEN = 64

print("Padding/truncating all sentences to %d values" %
MAX_LEN)
print('Padding token: "{:}", ID:
{:}'.format(tokenizer.pad_token, tokenizer.pad_token_id))

input_ids = pad_sequences(input_ids, maxlen=MAX_LEN,
dtype='long', value=0, truncating='post', padding='post')
```

```
print("Done")
```

```
Padding/truncating all sentences to 64 values
Padding token: "[PAD]", ID: 0
Done
```

```
==========================================================================
```

```
input_ids[124]
```

```
array([  101, 24934, 39711, 10116, 10237, 13426, 10243, 10243, 10243,
       24943, 12993, 68758, 20738,   102,     0,     0,     0,     0,
           0,     0,     0,     0,     0,     0,     0,     0,     0,
           0,     0,     0,     0,     0,     0,     0,     0,     0,
           0,     0,     0,     0,     0,     0,     0,     0,     0,
           0,     0,     0,     0,     0,     0,     0,     0,     0,
           0,     0,     0,     0,     0,     0,     0,     0,     0,
           0])
```

Gambar 4. 3 *Padding Data*

4. 3 Implementasi Model Arsitektur

Pada bagian ini, penulis akan membagi data menjadi tiga bagian utama : data *training*, data validasi dan data uji (*test*). Data *training* akan digunakan untuk melatih model, data validasi akan digunakan untuk mengevaluasi performa model selama proses pelatihan dan data uji akan digunakan untuk menguji model setelah pelatihan selesai.

Tabel 4. 3 Pembagian *Dataset*

Pembagian Data	Jumlah Data yang Dipakai
Data *Training*	1541
Data Validasi	273
Data *Test*	202

Seperti yang terlihat pada Tabel 4. 3 pembagian *dataset* untuk data *training* dan data *testing* sejumlah 80% dan 20% dari keseluruhan data. Lalu dibagi lagi untuk data validasi dengan proporsi 70% dan 30% dari jumlah data *training*.

Setelah melakukan pembagian data, penulis menginisialisasi model *BERT* untuk tugas analisis kalimat menggunakan arsitektur "*Bert For Sequence Classification"*. Model ini telah dimuat dari *pre-trained* model *"bert-base-multilingual-uncased"* yang dapat meng-*handle* beberapa bahasa dan memiliki *casing* yang tidak peka. Codingnya dapat dilihat pada Gambar 4. 4.

```
from transformers import BertForSequenceClassification, AdamW,
BertConfig

model = BertForSequenceClassification.from_pretrained(
    "bert-base-multilingual-uncased",
    num_labels = 3,
    output_attentions = False,
    output_hidden_states = False
)

model.cuda()
```

Gambar 4. 4 Inisialisasi Model *BERT*

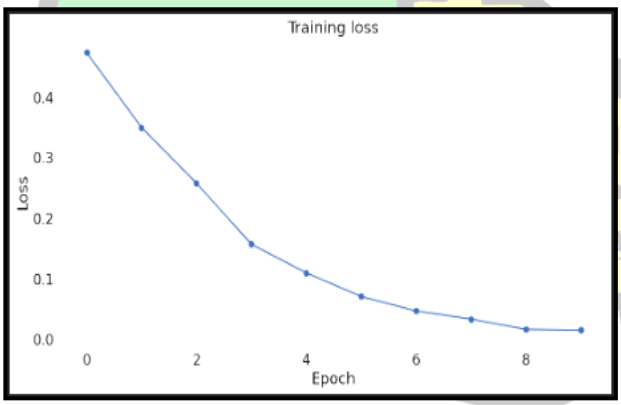
Dataset yang telah siap akan dilatih menggunakan model *multilingual BERT*. Selama pelatihan, model diberikan sejumlah besar kalimat *abusive* dan *non-abusive* untuk memahami pola dan representasi kata yang berkaitan dengan masing-masing kategori.

4. 4 Hasil Analisis terhadap Implementasi Model dan Arsitektur

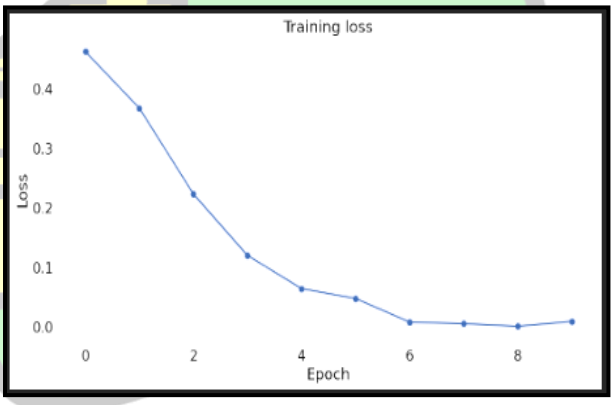
Bagian ini, penulis akan menjelaskan hasil dari beberapa percobaan yang dilakukan untuk membandingkan pengaruh parameter *learning rate*, *batch size* dan *epoch* terhadap performa model *multilingual BERT* dalam tugas klasifikasi kalimat *abusive* pada media sosial *twitter*. Percobaan ini bertujuan untuk mengetahui kombinasi parameter terbaik yang dapat menghasilkan model dengan kinerja yang optimal. Berikut berupa penjelasan dari parameter yang digunakan untuk perbandingan.

1. *Learning Rate* : penulis akan membandingkan tiga nilai *learning rate* berbeda yang akan dicoba satu nilai dari tingkatan yang berbeda. Dari tingkat rendah (1e-5, 2e-5 dan 3e-5) nilai yang dicoba 2e-5, tingkat menengah (4e-5, 5e-5 dan 1e-4) nilai yang dicoba 5e-5, tingkat tinggi (2e-4, 3e-4) nilai yang dicoba 2e-4.
2. *Bach Size* : penulis akan membandingkan dua ukuran *batch* yang berbeda : 16, 32. Penggunaan *batch size* 16 dan 32 ditujukan untuk mendapatkan keseimbangan antara kecepatan pelatihan dan penggunaan memori.
3. *Epoch* : penulis akan membandingkan dua jumlah *epoch* yang berbeda : 5 dan 10.

4. 4. 1 Percobaan 1: *Learning Rate* (2e-5, 5e-5 dan 2e-4), *Batch Size* 32 dan *Epoch* 10



(a) *Learning Rate* 2e-5



(b) *Learning Rate* 5e-5

(c) *Learning Rate* 2e-4

Gambar 4. 5 Parameter Percobaan 1

Pada gambar 4. 5 Dalam grafik *training loss*, sumbu x biasanya mewakili iterasi atau *epoch* pelatihan, sementara sumbu y mewakili nilai *loss* pada iterasi atau *epoch* tertentu. Pada gambar (a) dan (b) di awal pelatihan, *loss* umumnya tinggi karena model belum mengetahui pola-pola yang ada dalam data dengan baik. Namun, seiring berjalannya waktu dan model mengenali pola-pola tersebut, *loss* secara bertahap akan menurun. Pada gambar (c) terlihat data grafik tidak sama dengan gambar (a) dan (b), ini bisa terjadi karena parameter yang dipakai tidak cocok dengan model. Untuk data akurasi dan MCC yang didapat dari percobaan 1 tersebut dapat dilihat pada Tabel 4. 4.

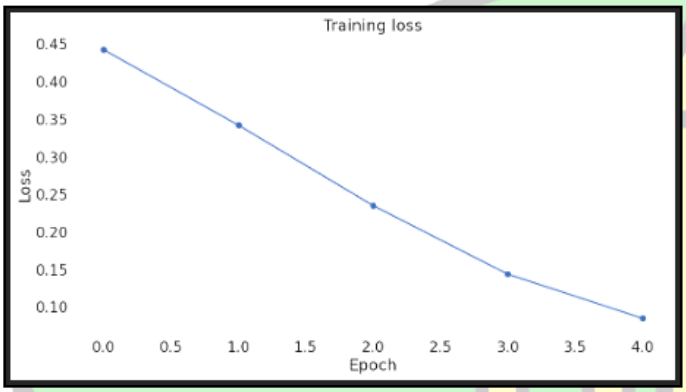
Tabel 4. 4 Parameter Percobaan 1

Learning rate	**Akurasi (ACC)**	**MCC**
2e-5	0.876	0.539
5e-5	0.866	0.465
2e-4	0.822	0.000

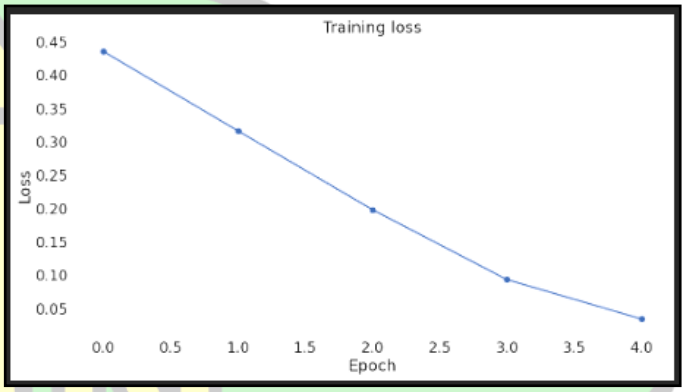
Nilai MCC pada *learning rate* 2e-4 menunjukkan angka 0.000, itu menandakan bahwa terjadi hal yang tidak stabil pada model. Ini disebabkan nilai

lerning rate yang dipakai terlalu tinggi sehingga tidak dapat bekerja dengan baik pada model dalam memahami data.

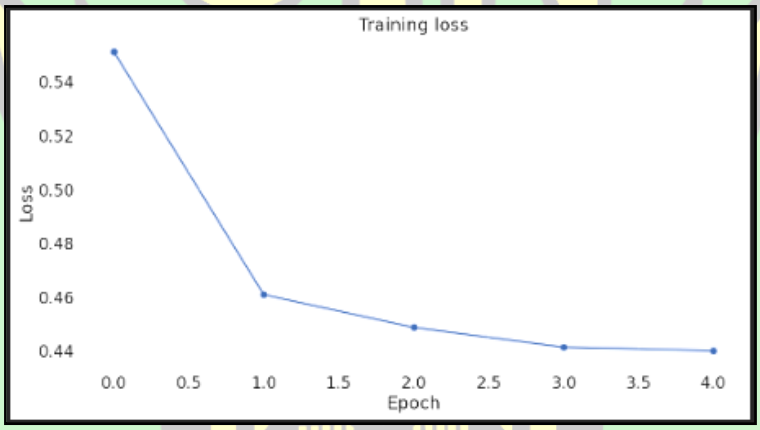
4. 4. 2 Percobaan 2: *Learning Rate* (2e-5, 5e-5 dan 2e-4), *Batch Size* 16 dan *Epoch* 5



(a) *Learning Rate* 2e-5



(b) *Learning Rate* 5e-5



(c) *Learning Rate* 2e-4

Gambar 4. 6 Parameter Percobaan 2

Pada gambar 4. 6 terlihat bahwa gambar (a) dan (b) tidak memiliki perbedaan grafik yang seknifikan untuk grafik *training loss* yang di uji dengan beberapa nilai parameter *learning rate*, *Batch size* 16 *dan epoch* 5. Untuk data akurasi dan MCC yang didapat dari percobaan 2 tersebut dapat dilihat pada Tabel 4. 4.

Tabel 4. 5 Parameter Percobaan 2

Learning rate	**Akurasi (ACC)**	**MCC**
2e-5	0.861	0.445
5e-5	0.871	0.495
2e-4	0.822	0.000

Sama halnya dengan percobaan 1, nilai MCC pada *learning rate* 2e-4 menunjukkan angka 0.000, yang menandakan bahwa terjadi *konvergen* yang tidak stabil pada model. Ini disebabkan nilai *lerning rate* yang dipakai terlalu tinggi sehingga tidak dapat bekerja dengan baik pada model dalam memahami data.

Berdasarkan hasil percobaan diatas, penulis dapat mengamati beberapa hal:

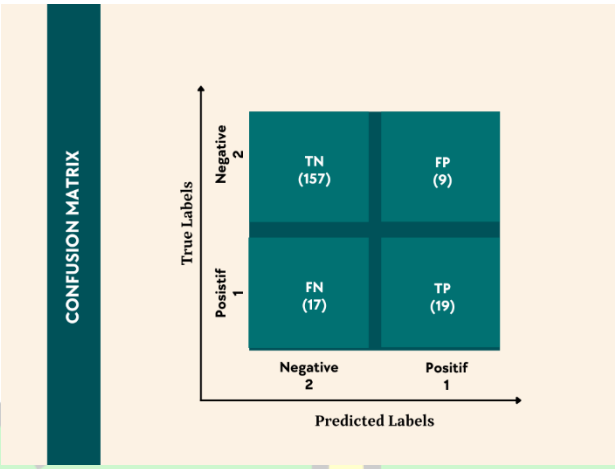
1. *Learning Rate*: Nilai *learning rate* 2e-5 memberikan hasil yang lebih baik dengan akurasi dan MCC tertinggi. *Learning rate* yang terlalu besar (2e-4) dapat mempengaruhi performa model dan menyebabkan *konvergen*.
2. *Batch size*: *Batch size* 32 memberikan hasil yang lebih baik.
3. *Epoch*: Jumlah *epoch* 10 memberikan hasil yang lebih baik, terlalu sedikit atau terlalu banyak epoch dapat menyebabkan *underfitting* atau *overfitting* pada model.

4. 5 *Confusion Matrix*

Dari hasil percobaan dan analisis yang dilakukan diatas, penulis mengambil kombinasi parameter yang memberikan hasil terbaik untuk dilakukan perhitungan keakuratan dengan menggunakan persamaan dari akurasi, *precision*, *recall* dan F1-*Score*.

- *Learning Rate*: 2e-5
- *Batch size*: 32
- *Epoch*: 10

Berikut proses pencarian perhitungan akurasi, *precision*, *recall* dan F1-*Score*.



Gambar 4. 7 Gambar *Confusion Matrix*

Dari pengujian data padaGambar 4. 7 dan Tabel 4. 6 didapatkan sebanyak 19 data TP yaitu *True Positive*, dimana data terklasifikasi oleh sistem dengan benar pada kelas *positive*. Kemudian, 9 data FP yaitu *false positive*, jumlah kelas *positive* yang salah diprediksi dengan benar oleh model. Selanjutnya, 17 data FN yaitu *False Negative*, dimana kelas *negative* namun dideteksi *positive* dan TN yaitu *true negative* sebanyak 157 data, sistem berhasil memprediksi sebagai kelas *negative.*

Berikut perhitungan akurasi, *Precision, Recall* dan F1-*Score* pada *dataset* :

Akurasi = $Accuracy = \frac{Jumlah\ Objek\ Yang\ Terdeteksi\ Benar}{Jumlah\ Keseluruhan\ Objek\ Yang\ Terdeteksi}$ **x 100%**

$= \frac{176}{202}$ **x 100 %**

= 87 %

Precision = $Precision = \frac{TP}{(TP + FP)}$ **x 100%**

$= \frac{19}{19+9}$ **x 100 %**

$= \frac{19}{28}$ **x 100 %**

= 68%

$$\textbf{Recall} = Recall = \frac{TP}{(TP + FN)} \text{ x } 100\%$$

$$= \frac{19}{19+17} \text{ x } 100\ \%$$

$$= \frac{19}{36} \text{ x } 100\ \%$$

$$= 53\ \%$$

$$\textbf{F1 Score} = F1\ Score = \frac{2\ x\ recall\ x\ precision}{rekal + precision} \text{ x } 100\%$$

$$= \frac{2x\ 68x\ 53}{68+53} \text{ x } 100\%$$

$$= \frac{6968}{119} \text{ x } 100\%$$

$$= 59\ \%$$

Tabel 4. 6 Perhitungan Akurasi, *Precision, Recall dan F1-Score*

Akurasi	**87%**
Precision	68%
Recall	53%
F1-Score	59%

Dari Tabel 4. 7 di dapatkan penjelasan sebagai berikut:

- Akurasi (*Accuracy*):
  Dalam kasus ini, akurasi mencapai 87%, yang berarti sekitar 87% dari semua sampel telah diklasifikasikan dengan benar oleh model, baik itu sebagai hasil positif maupun negatif.
- Presisi (*Precision*):
  presisi sebesar 68% menunjukkan bahwa dari semua prediksi positif yang dibuat oleh model, hanya sekitar 68% yang benar-benar *relevan* atau akurat.

- *Recall* (Sensitivitas atau *True Positive Rate*):

  Recall sebesar 53% mengindikasikan bahwa model hanya mampu mengidentifikasi sekitar 53% dari semua kasus positif yang sebenarnya ada.

- F1-*Score*:

  F1-*Score* sebesar 59% menunjukkan tingkat keseimbangan yang baik antara kemampuan model untuk menghasilkan hasil yang akurat (*precision*) dan kemampuan model untuk mengidentifikasi kasus positif yang ada (*recall*).

BAB V
KESIMPULAN DAN SARAN

5. 1 Kesimpulan

Dari hasil penelitian yang telah penulis lakukan, penulis mengambil beberapa kesimpulan pada penelitian ini, yaitu:

1. Proses dalam analisis kalimat *abusive* pada media sosial *Twitter* menggunakan metode *multilingual BERT* berjalan dengan baik dan menghasilkan akurasi 87%
2. *Dataset* yang digunakan adalah sebanyak 2016 data, terdapat dua *sentences Tweet* dan label. *Dataset* terbagi tiga yaitu data *training*, data *testing* dan data validasi. Data *training* memiliki jumlah 1541 *sentence*, data *testing* berjumlah 202 *sentence* dan data validasi sebanyak 273 *sentence*. Model *multilingual BERT* yang digunakan pada penelitian ini menghasilkan Tingkat akurasi dari dataset nilai yang cukup tinggi dengan nilai akurasi 87%.

5. 2 Saran

Berdasarkan kesimpulan dari hasil dan pembahasan pada penelitian ini, maka penulis mengemukakan beberapa saran yang dapat digunakan untuk pengembangan lanjutan dari penelitian ini, sebagai berikut:

1. Penelitian ini hanya memfokuskan pada satu metode yaitu *multilingual BERT,* untuk penelitian selanjutnya bisa digabungkan beberapa metode untuk membandingkan kinerja model dengan beberapa metode.
2. Berdasarkan hasil percobaan dan analisis, kami merekomendasikan penggunaan kombinasi parameter berikut untuk model *BERT* dalam tugas klasifikasi kalimat *abusive* pada media sosial *Twitter*: *Learning Rate*: 2e-5, *Batch Size*: 32 dan *Epoch*: 10.

Namun demikian, hasil ini dapat berbeda-beda tergantung pada *dataset* dan sifat tugas yang berbeda. Oleh karena itu, disarankan untuk melakukan eksperimen lebih lanjut dengan parameter lainnya untuk mendapatkan model yang paling sesuai dengan tugas yang spesifik.

DAFTAR PUSTAKA

Alammar, J. (2018). *The Illustrated Transformer*. Github.Com. https://jalammar.github.io/illustrated-transformer/

AMELIA, D. (2021). *Klasifikasi Emosi pada teks menggunakan Deep Learning*. *6*(1).

Benjamin Fyenbo, D., Charlotte Frederiksen, T., Linz, D., Jespersen, T., Dobrev, D., Gislason, G., Betz, K., Saljic, A., & Nielsen Holck, E. (2022). Researchers in cardiology – Why and how to get on Twitter? *IJC Heart and Vasculature*, *40*(March), 101010. https://doi.org/10.1016/j.ijcha.2022.101010

Febrian Sengkey, D., Diane Kambey, F., Paulus Lengkong, S., Reynaldo Joshua, S., & Valentino Florensius Kainde, H. (2020). Pemanfaatan Platform Pemrograman Daring dalam Pembelajaran Probabilitas dan Statistika di Masa Pandemi CoVID-19. *Jurnal Informatika*, *15*(4), 257–264.

Girinoto, G., Alwan, D. A., Gde K.T. D, G. A. N., Nabila, O. G., Arizal, A., & Priambodo, D. F. (2022). Implementasi Deteksi Judul Berita Clickbait Berbahasa Indonesia dengan pre-trained model Multilingual BERT Pada Aplikasi Berbasis Chrome Extension. *Jurnal Ilmiah SINUS*, *20*(2), 25. https://doi.org/10.30646/sinus.v20i2.624

Gumilar, M. D., Sembiring, F., & Erfina, A. (2021). Implementasi Progressive Web App pada Sistem Informasi E-learning untuk Pembelajaran Bahasa Pemrograman Python. *Jutisi : Jurnal Ilmiah Teknik Informatika Dan Sistem Informasi*, *10*(2), 309. https://doi.org/10.35889/jutisi.v10i2.658

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.

Herwanto, G. B., Ningtyas, A. M., Mujiyatna, I. G., Nugraha, K. E., & Prayana Trisna, I. N. (2021). Hate Speech Detection in Indonesian Twitter using Contextual Embedding Approach. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, *15*(2), 177. https://doi.org/10.22146/ijccs.64916

Ibrohim, M. O., & Budi, I. (2018). A Dataset and Preliminaries Study for Abusive Language Detection in Indonesian Social Media. *Procedia Computer Science*, *135*, 222–229. https://doi.org/10.1016/j.procs.2018.08.169

Jiang, D., & He, J. (2020). Tree Framework with BERT Word Embedding for the Recognition of Chinese Implicit Discourse Relations. *IEEE Access*, *8*, 162004–162011. https://doi.org/10.1109/ACCESS.2020.3019500

Luh Putu Ary Sri Tjahyanti, Putu Satya Saputra, & Made Santo Gitakarma. (2022). Peran Artificial Intelligence (Ai) Untuk MendukungPembelajaran Di Masa Pandemi Covid-19. *Komputer Dan Teknologi Sains (KOMTEKS)*, *1*(1), 15–21.

Mubaraq, M. F., & Maharani, W. (2022). *Sentiment Analysis on Twitter Social Media towards Climate Change on Indonesia Using IndoBERT Model*. *6*, 2426–2431. https://doi.org/10.30865/mib.v6i4.4570

Mubarok, M. M. (2021). *Indonesian Abstractive News Summarization Berbasis Deep Learning Dengan Metode Sequence-To-Sequence Long Short-Term Memory*.

Nabiilah, G. Z., Prasetyo, S. Y., Izdihar, Z. N., & Girsang, A. S. (2023). BERT base model for toxic comment analysis on Indonesian social media. *Procedia Computer Science*, *216*(2022), 714–721. https://doi.org/10.1016/j.procs.2022.12.188

Pasaribu, R. G. M., Mulyadi, & Wulan, G. A. (2020). Pencegahan Kejahatan Ujaran Kebencian di Indonesia. *Jurnal Ilmu Kepolisian*, *14*(3), 170–188. http://www.jurnalptik.id/index.php/JIK/article/download/278/98

Perwira, A., Dwitama, J., & Kunci, K. (2021). Deteksi Ujaran Kebencian Pada Twitter Bahasa Indonesia Menggunakan Machine Learning : Reviu Literatur. *Jurnal SNATi*, *1*(1), 31–39.

Pires, T., Schlinger, E., & Garrette, D. (2020). How multilingual is multilingual BERT? *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, 4996–5001. https://doi.org/10.18653/v1/p19-1493

Rafiq, A. (2020). Dampak Media Sosial Terhadap Perubahan Sosial Suatu Masyarakat. *Global Komunika*, *1*(1), 18–29.

Rayyan, F. (2022). *PENGEMBANGAN CHATBOT UNTUK APLIKASI ONLINE CHAT TELEGRAM DENGAN PENDEKATAN KLASIFIKASI EMOSI PADA TEKS MENGGUNAKAN METODE INDOBERT-LITE ONLINE CHAT TELEGRAM DENGAN PENDEKATAN KLASIFIKASI EMOSI PADA TEKS MENGGUNAKAN*.

Siahaan, M., Jasa, C. H., Anderson, K., & Valentino, M. (2020). Penerapan Artificial Intelligence (AI) Terhadap Seorang Penyandang Disabilitas Tunanetra. *Information System and Technology*, *01*(02), 186–193.

Soen, G. I. E., Marlina, & Renny. (2022). Implementasi Cloud Computing dengan Google Colaboratory Pada Aplikasi Pengolah Data Zoom Participants. *Journal Informatic Technology And Communication*, *6*(1), 24–30.

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

Twitter. (2023). *Apa itu Twitter?* Twitter.Com. https://help.twitter.com/id/resources/new-user-faq

Yaya Heryadi, E. I. (2020). *Deep Learning: Aplikasinya di Bidang Geospasial*

(alexander Agung Santoso Gunawan (ed.); pertama, 1). books.google.co.id. https://books.google.co.id/books?hl=id&lr=&id=UorwDwAAQBAJ&oi=fnd&pg=PA4&dq=definisi+Deep+Learning&ots=t-K0deuU9z&sig=8Jiv9gHpnR7xxvgwGfJwn7602D4&redir_esc=y#v=onepage&q=definisi Deep Learning&f=false

Yazid, R. M., Umbara, F. R., & Sabrina, P. N. (2022). *Deteksi Ujaran Kebencian dengan Metode Klasifikasi Naïve Bayes dan Metode N-Gram pada Dataset Multi-Label Twitter Berbahasa Indonesia*. *2*, 46–52.

LAMPIRAN

Berikut berupa penjelasan tentang kode yang penulis gunakan untuk analisis kalimat *abusive*.

1. Cek *resort* yang tersedia

```
import torch

if torch.cuda.is_available():
  device = torch.device('cuda')

  print('there are %d GPU(s) available.' %
torch.cuda.device_count())

  print('we will use the GPU: ',
torch.cuda.get_device_name(0))

else:
  print("No GPU available, using the CPU instead")
  device = torch.device("cpu")
```

Kode ini adalah skrip Python yang memeriksa apakah terdapat GPU (dengan dukungan CUDA) yang dapat digunakan dengan PyTorch, dan kemudian mengatur variabel **device** sesuai dengan ketersediaan tersebut.

```
!pip install transformers
```

Perintah !pip install transformers digunakan untuk mengunduh dan menginstal pustaka "transformers" dari Python Package Index (PyPI) ke dalam lingkungan Python. Setelah diinstal, dapat menggunakan pustaka ini untuk bekerja dengan berbagai model pemrosesan bahasa alami yang telah dilatih sebelumnya.

```
!pip install wget
```

Perintah !pip install wget digunakan untuk mengunduh dan menginstal pustaka "wget" dari Python Package Index (PyPI) ke dalam lingkungan Python, Pustaka wget merupakan sebuah alat baris perintah dan pustaka Python yang digunakan untuk melakukan pengunduhan berkas dari jaringan.

2. *Download* dan *load* dataset

```
import pandas as pd

# dataset berita hoax
dataset = '/content/Two_label.csv'

# Membaca dataset ke dalam DataFrame
df = pd.read_csv("Two_label.csv")

# Menampilkan dataset
print(df.head(4))
```

Kode ini membaca berkas CSV yang berisi data berita hoax, menyimpannya dalam DataFrame df, dan kemudian mencetak empat baris pertama dari DataFrame tersebut ke konsol. Fungsi ini berguna dalam memahami struktur dan konten dari dataset yang akan gunakan dalam analisis atau pemrosesan lebih lanjut.

```
import pandas as pd

df = pd.read_csv("/content/Two_label.csv")
df.shape
```

konteks kode yang diberikan, jika menjalankan df.shape, akan mendapatkan tuple yang menunjukkan jumlah baris dan jumlah kolom dalam DataFrame df. Ini berguna untuk memahami ukuran atau dimensi dari dataset yang sedang di kerjakan.

```
import matplotlib.pyplot as plt

# Menghitung jumlah label
label_counts = df['Label'].value_counts()

# Menampilkan barplot
plt.figure(figsize=(8, 6))
label_counts.plot(kind='bar')
plt.xlabel('label')
plt.ylabel('Jumlah berita')
plt.title('Jumlah Label dalam Dataset')
plt.show()
```

Kode ini menggunakan pustaka matplotlib dalam Python untuk membuat dan menampilkan barplot berdasarkan jumlah label pada DataFrame. Hasil dari kode ini adalah barplot yang menunjukkan jumlah berita untuk masing-masing jenis label pada dataset. Grafik ini membantu untuk memvisualisasikan distribusi label dalam dataset dengan jelas.

```
sentences = df.Tweet.values
labels = df.Label.values
```

Setelah kode ini dieksekusi, akan memiliki dua array terpisah: sentences yang berisi teks dari tweet dan labels yang berisi label atau kategori yang sesuai dengan masing-masing teks tweet.

3. Load BERT Tokenizer

```
from transformers import BertTokenizer

print("Loading BERT Tokenizer")
tokenizer = BertTokenizer.from_pretrained('bert-base-
multilingual-uncased', do_lower_case=True)
```

Setelah kode ini dieksekusi, akan memiliki objek tokenizer yang dapat digunakan untuk melakukan tokenisasi pada teks. Tokenisasi ini akan mengubah teks menjadi sejumlah token-token yang nantinya dapat diumpankan ke dalam model BERT untuk pemrosesan lebih lanjut.

```
print("Original: ", sentences[124])

print("Tokenized: ", tokenizer.tokenize(sentences[124]))

print("Token IDS: ",
tokenizer.convert_tokens_to_ids(tokenizer.tokenize(sentences[1
24])))
```

kita dapat melihat bagaimana teks asli diubah menjadi token, serta bagaimana token-token ini diwakili dalam bentuk ID numerik yang akan digunakan sebagai input untuk model BERT. Ini adalah langkah penting dalam mempersiapkan teks untuk pemrosesan menggunakan model-model bahasa seperti BERT.

```
input_ids = []
```

```
for sent in sentences:
  encoded_sent = tokenizer.encode(
      sent,
      add_special_tokens = True
  )
  input_ids.append(encoded_sent)

print("Original: ", sentences[124])
print("Token IDs: ", input_ids[124])
```

Kode yang di berikan adalah contoh penggunaan tokenizer BERT untuk mengonversi sejumlah kalimat (dalam variabel sentences) menjadi daftar ID token menggunakan metode encode dari tokenizer.

```
print("Max sentence length: ", max([len(sen) for sen in
input_ids]))
```

Kode yang di berikan digunakan untuk mencari panjang maksimum dari daftar ID token (input_ids). Dengan mengukur panjang maksimum dari daftar ID token, dapat memiliki gambaran tentang seberapa panjang input yang dapat diterima oleh model BERT. Panjang input ini harus sesuai dengan batas panjang yang diterima oleh model yang sedang digunakan.

```
from tensorflow.keras.preprocessing.sequence import
pad_sequences


MAX_LEN = 64

print("Padding/truncating all sentences to %d values" %
MAX_LEN)
print('Padding token: "{:}", ID:
{:}'.format(tokenizer.pad_token, tokenizer.pad_token_id))

input_ids = pad_sequences(input_ids, maxlen=MAX_LEN,
dtype='long', value=0, truncating='post', padding='post')

print("Done")
```

Kode yang di berikan menggunakan pustaka TensorFlow (tensorflow.keras.preprocessing.sequence) untuk menerapkan padding pada daftar ID token input_ids, sehingga semua kalimat memiliki panjang yang seragam sesuai dengan MAX_LEN. Ini penting karena model BERT menerima input dengan panjang yang tetap dan seragam.

```
input_ids[124]
```

Ini akan mencetak daftar ID token yang telah di-pad sesuai dengan panjang maksimum yang di tentukan sebelumnya. Daftar ini akan terdiri dari angka-angka yang mewakili token-token dari kalimat yang telah di-tokenisasi dan diubah menjadi ID token.

```
attention_mask = []

for sent in input_ids:
  att_mask = [int(token_id > 0) for token_id in sent]

  attention_mask.append(att_mask)
```

Dengan langkah-langkah ini, membuat attention_mask yang akan digunakan bersamaan dengan input_ids saat memberi tahu model BERT bagaimana mengabaikan token-token padding dan hanya memperhatikan token-token yang sebenarnya (non-padding) dalam proses komputasinya.

4. Persiapkan data

```
from sklearn.model_selection import train_test_split

train_input, test_input, train_labels, test_labels =
train_test_split(input_ids,

        labels,

        random_state=2017,

        test_size=0.1)
train_mask, test_mask, _, _ = train_test_split(attention_mask,
                                               labels,

 random_state=2017,
```

```
                                                    test_size=0.1)

train_input, validation_input, train_labels, validation_labels
= train_test_split(train_input,

                   train_labels,

                   random_state=2018,

                   test_size=0.15)
train_mask, validation_mask, _, _ =
train_test_split(train_mask,

 train_mask,

 random_state=2018,

 test_size=0.15)
```

Kode yang di berikan menggunakan pustaka scikit-learn untuk membagi data menjadi set pelatihan, validasi, dan pengujian, serta membuat masker perhatian yang sesuai untuk setiap bagian. Dengan langkah-langkah ini, memiliki set pelatihan, validasi, dan pengujian, serta masker perhatian yang sesuai untuk masing-masing bagian. Data ini dapat digunakan untuk melatih dan menguji model BERT.

```
import numpy as np
print("== Train ==")
print("Input: ", train_input.shape)
print("Label: ", train_labels.shape)
print("Mask: ", np.array(train_mask).shape)

print("\n== Validation ==")
print("Input: ", validation_input.shape)
print("Label: ", validation_labels.shape)
print("Mask: ", np.array(validation_mask).shape)

print("\n== Test ==")
print("Input: ", test_input.shape)
print("Label: ", test_labels.shape)
print("Mask: ", np.array(test_mask).shape)
```

Kode yang di berikan digunakan untuk mencetak bentuk (shape) dari setiap bagian data yang telah di bagi menjadi data pelatihan, validasi, dan pengujian, serta masker perhatian yang sesuai. Output ini akan memberikan informasi tentang ukuran (jumlah baris dan kolom) dari setiap bagian data dan masker perhatian yang telah di persiapkan.

```
train_input = torch.tensor(train_input)
train_labels = torch.tensor(train_labels)
train_mask = torch.tensor(train_mask)

validation_input = torch.tensor(validation_input)
validation_labels = torch.tensor(validation_labels)
validation_mask = torch.tensor(validation_mask)

test_input = torch.tensor(test_input)
test_labels = torch.tensor(test_labels)
test_mask = torch.tensor(test_mask)
```

Kode yang di berikan mengonversi data dari NumPy arrays menjadi tensor PyTorch. Ini adalah langkah yang diperlukan untuk mempersiapkan data dalam format yang kompatibel dengan pemrosesan di dalam framework PyTorch.

```
from torch.utils.data import TensorDataset, DataLoader,
RandomSampler, SequentialSampler

batch_size = 32

train_data = TensorDataset(train_input, train_mask,
train_labels)
train_sampler = RandomSampler(train_data)
train_dataloader = DataLoader(train_data,
sampler=train_sampler, batch_size=batch_size)

validation_data = TensorDataset(validation_input,
validation_mask, validation_labels)
validation_sampler = SequentialSampler(validation_data)
validation_dataloader = DataLoader(validation_data,
sampler=validation_sampler, batch_size=batch_size)

test_data = TensorDataset(test_input, test_mask, test_labels)
test_sampler = SequentialSampler(test_data)
```

```
test_dataloader = DataLoader(test_data, sampler=test_sampler,
batch_size=batch_size)
```

Kode yang di berikan digunakan untuk mempersiapkan data sebagai objek yang sesuai dengan struktur dataset PyTorch dan mengatur dataloader untuk pelatihan, validasi, dan pengujian. Dataloaders ini akan membantu mengelola dan memproses batch data secara efisien selama pelatihan dan pengujian.

5. Persiapkan model *pre-trained BERT*

```
from transformers import BertForSequenceClassification, AdamW,
BertConfig

model = BertForSequenceClassification.from_pretrained(
    "bert-base-multilingual-uncased",
    num_labels = 3,
    output_attentions = False,
    output_hidden_states = False
)

model.cuda()
```

Kode yang di berikan menggunakan pustaka transformers untuk mengimpor, mengkonfigurasi, dan memuat model BERT untuk tugas klasifikasi urutan (sequence classification). Setelah kode ini dijalankan, akan memiliki model BERT untuk tugas klasifikasi urutan yang siap digunakan. Dapat melatih dan menguji model ini dengan dataloaders yang telah di persiapkan sebelumnya.

```
params = list(model.named_parameters())

print("The BERT model has {:} different named
parameters.".format(len(params)))

print("==== Embedding Layer ====")
for p in params[0:5]:
  print("{:<60} {:>12}".format(p[0], str(tuple(p[1].size()))))

print("==== First Transformers ====")
for p in params[5:21]:
  print("{:<60} {:>12}".format(p[0], str(tuple(p[1].size()))))

print("==== Output Layer ====")
for p in params[-4:]:
```

```
  print("{:<60} {:>12}".format(p[0], str(tuple(p[1].size()))))
```

Kode yang di berikan digunakan untuk mencetak informasi tentang berbagai parameter dalam model BERT, termasuk ukuran (shape) dari setiap parameter. Output ini akan memberikan wawasan tentang ukuran parameter dalam berbagai bagian utama model BERT. Ini berguna untuk memahami struktur dan kompleksitas model serta untuk melakukan pemeliharaan dan penyesuaian sesuai kebutuhan

```
optimizer = AdamW(
    model.parameters(),
    lr = 2e-5,
    eps = 1e-8
)
```

Kode yang di berikan menginisialisasi optimizer AdamW untuk mengoptimasi parameter-parameter dalam model BERT selama pelatihan. Dengan menginisialisasi optimizer ini, memiliki alat yang diperlukan untuk mengoptimasi parameter-parameter model BERT selama pelatihan. Selanjutnya, dapat menggunakan optimizer ini bersama dengan fungsi kerugian (loss function) untuk melatih model.

```
from transformers import get_linear_schedule_with_warmup

epochs = 10

total_steps = len(train_dataloader) * epochs

scheduler = get_linear_schedule_with_warmup(optimizer,
                                            num_warmup_steps
= 0,

 num_training_steps = total_steps)
```

Kode yang di berikan digunakan untuk menghasilkan scheduler penjadwalan laju pembelajaran linear dengan pemanasan (warmup) untuk pelatihan model. Scheduler ini akan mengatur laju pembelajaran secara linier selama pelatihan, dengan jumlah pemanasan awal (warmup) yang tidak ada. Ini membantu dalam

stabilitas dan konvergensi pelatihan model. Dapat menggunakan scheduler ini dalam proses pelatihan model.

```
import numpy as np

def flat_accuracy(preds, labels):
  pred_flat = np.argmax(preds, axis=1).flatten()
  labels_flat = labels.flatten()
  return np.sum(pred_flat == labels_flat) / len(labels_flat)
```

Kode ini mendefinisikan fungsi flat_accuracy yang digunakan untuk menghitung akurasi dari prediksi model pada data yang diberikan. Fungsi ini akan membantu menghitung akurasi prediksi model pada data uji atau validasi selama pelatihan atau pengujian model.

```
import time
import datetime

def format_time(elapsed):
  elapsed_rounded = int(round(elapsed))
  return str(datetime.timedelta(seconds=elapsed_rounded))
```

Kode ini mendefinisikan fungsi format_time yang digunakan untuk mengonversi waktu yang telah berlalu menjadi format yang lebih mudah dibaca. Fungsi ini berguna untuk mencetak atau melaporkan waktu yang telah berlalu dalam format yang lebih mudah dimengerti, seperti dalam bentuk "3 days, 5:12:45" atau sejenisnya. Anda dapat menggunakannya untuk melacak berapa lama waktu yang telah diambil dalam proses pelatihan atau pengujian model.

6. Training *BERT*

```
import random

seed_val = 42

random.seed(seed_val)
np.random.seed(seed_val)
torch.manual_seed(seed_val)
torch.cuda.manual_seed_all(seed_val)

loss_values = []
```

```
for epoch_i in range(0, epochs):

  # ========================================
  #               Training
  # ========================================

  print("======= Epoch {:} / {:} =======".format(epoch_i+1,
epochs))
  print("Training...")

  t0 = time.time()

  total_loss = 0

  model.train()

  # For each batch of training data
  for step, batch in enumerate(train_dataloader):

    # Progress update every 40 batches
    if step % 40 == 0 and not step == 0:
      elapsed = format_time(time.time() - t0)

      print("Batch {:>5,} of {:>5,}.     Elapsed:
{:}".format(step, len(train_dataloader), elapsed))

    b_input_ids = batch[0].to(device)
    b_input_mask = batch[1].to(device)
    b_labels = batch[2].to(device)

    model.zero_grad()

    outputs = model(b_input_ids,
                    token_type_ids=None,
                    attention_mask=b_input_mask,
                    labels=b_labels)

    loss = outputs[0]

    total_loss += loss.item()

    loss.backward()

    torch.nn.utils.clip_grad_norm_(model.parameters(), 1.0)
```

```
    optimizer.step()

    scheduler.step()

  avg_train_loss = total_loss / len(train_dataloader)

  loss_values.append(avg_train_loss)

  print("   Average training loss:
{0:.2f}".format(avg_train_loss))
  print("   Training epoch took:
{:}".format(format_time(time.time() - t0)))

  # ==================================
  #             Validation
  # ==================================

  print("Running Validation...")

  t0 = time.time()

  model.eval()

  eval_loss, eval_accuracy = 0, 0
  nb_eval_steps, nb_eval_examples = 0, 0

  for batch in validation_dataloader:

    batch = tuple(t.to(device) for t in batch)

    b_input_ids, b_input_mask, b_labels = batch

    with torch.no_grad():
      outputs = model(b_input_ids,
                      token_type_ids=None,
                      attention_mask=b_input_mask)

    logits = outputs[0]
    logits = logits.detach().cpu().numpy()
    label_ids = b_labels.to('cpu').numpy()

    tmp_eval_accuracy = flat_accuracy(logits, label_ids)

    eval_accuracy += tmp_eval_accuracy
```

```
    nb_eval_steps += 1

  print("   Accuracy:
{0:.2f}".format(eval_accuracy/nb_eval_steps))
  print("   Validation took:
{:}".format(format_time(time.time() - t0)))

print("Training complete!")
```

Kode yang di berikan adalah skrip pelatihan dan pengujian model BERT untuk tugas klasifikasi. Ini adalah kerangka kerja yang umum digunakan untuk melatih model menggunakan dataset yang telah diolah sebelumnya dengan model BERT dan melakukan validasi selama pelatihan. Dapat mengadaptasi kode ini untuk tugas klasifikasi khusus.

```
import matplotlib.pyplot as plt
import seaborn as sns

sns.set(style='darkgrid')
sns.set(font_scale=1.5)
plt.rcParams["figure.figsize"] = (12,6)

plt.plot(loss_values, 'b-o')

plt.title("Training loss")
plt.xlabel("Epoch")
plt.ylabel("Loss")

plt.show()
```

Kode ini digunakan untuk menghasilkan grafik yang menunjukkan perubahan loss pelatihan selama setiap epoch. Output dari kode ini adalah grafik yang menunjukkan bagaimana loss pelatihan berubah selama setiap epoch. Ini membantu untuk melihat konvergensi pelatihan dan memastikan bahwa loss berkurang seiring berjalannya pelatihan.

7. Prediksi and evaluasi

```
print("Predicting labels for {:,} test
sentences".format(len(test_input)))
```

```
model.eval()

prediction, true_labels = [], []

for batch in test_dataloader:
  batch = tuple(t.to(device) for t in batch)

  b_input_ids, b_input_mask, b_labels = batch

  with torch.no_grad():
    outputs = model(b_input_ids,
                    token_type_ids=None,
                    attention_mask=b_input_mask)

  logits = outputs[0]

  logits = logits.detach().cpu().numpy()
  label_ids = b_labels.to('cpu').numpy()

  prediction.append(logits)
  true_labels.append(label_ids)

print(" DONE.")
```

Kode ini untuk melakukan prediksi label pada data pengujian menggunakan model BERT yang telah dilatih. Dengan kode ini, telah berhasil melakukan prediksi menggunakan model yang telah dilatih pada data pengujian. Sekarang dapat menganalisis hasil prediksi untuk mengevaluasi kinerja model.

```
from sklearn.metrics import matthews_corrcoef

flat_prediction = [item for sublist in prediction for item in
sublist]
flat_prediction = np.argmax(flat_prediction, axis=1).flatten()

flat_true_labels = [item for sublist in true_labels for item
in sublist]

mcc = matthews_corrcoef(flat_true_labels, flat_prediction)

print("MCC: %.3f" %mcc)
```

Kode ini untuk menghitung koefisien korelasi Matthews (Matthews Correlation Coefficient, MCC) sebagai metrik evaluasi kinerja model klasifikasi.

MCC adalah metrik yang berguna untuk mengukur kinerja model klasifikasi, terutama saat jumlah kelas yang tidak seimbang. Nilai MCC berkisar dari -1 hingga +1, di mana +1 menunjukkan prediksi sempurna, 0 menunjukkan prediksi acak, dan -1 menunjukkan prediksi yang salah arah. Semakin tinggi nilai MCC, semakin baik kinerja model dalam tugas klasifikasi.

```
from sklearn.metrics import accuracy_score

acc = accuracy_score(flat_true_labels, flat_prediction)

print("ACC: %.3f" %acc)
```

Kode ini digunakan untuk menghitung akurasi (Accuracy) sebagai metrik evaluasi kinerja model klasifikasi. Akurasi adalah metrik yang mengukur proporsi prediksi yang benar dari total prediksi. Ini merupakan metrik evaluasi yang umum digunakan dalam tugas klasifikasi. Semakin tinggi nilai akurasi, semakin baik kinerja model dalam memprediksi label yang benar.

```
# Convert tensor to numpy arrays
flat_true_labels = np.array(flat_true_labels)
flat_prediction = np.array(flat_prediction)

# Decode label IDs to original labels
label_map = {0: "Label_A", 1: "Label_B", 2:''}  # Ubah sesuai
dengan label yang digunakan dalam dataset Anda

true_labels = [label_map[label] for label in flat_true_labels]
predicted_labels = [label_map[label] for label in
flat_prediction]

# Print original labels and predicted labels side by side
for i in range(len(flat_true_labels)):
    print("Data ke-%d - True: %s, Predicted: %s" % (i+1,
flat_true_labels[i], flat_prediction[i]))
```

Kode ini berfokus pada mengonversi prediksi dan label sebenarnya ke dalam bentuk yang lebih mudah dibaca dan kemudian mencetak prediksi dan label sebenarnya secara berdampingan untuk beberapa contoh data. Ini adalah langkah terakhir untuk menganalisis hasil prediksi model. Dengan mencetak label asli dan

prediksi secara berdampingan, dapat dengan mudah melihat sejauh mana prediksi model cocok dengan label yang sebenarnya.

8. Confusion Matrix

```
from sklearn.metrics import confusion_matrix

# Prediksi label pada data validasi
true_labels: [2, 1, 1, 2, 2, 2, 2, 2, 1, 2, 1, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 1, 1, 2, 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 1,
2, 2, 1, 2, 2, 2, 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 1, 2,
1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1, 1, 2,
1, 1, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 1, 2, 2, 2]
predicted_labels: [2, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 1, 2, 2, 2, 2, 2, 1, 1, 1, 2, 1, 2, 2, 1, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 1, 1, 2, 2, 1, 2, 1,
1, 2, 2, 2, 2, 2, 2, 1, 1, 2, 2, 2, 2, 1, 1, 2, 2, 2, 2, 2, 2,
2, 1, 2, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 2, 1, 1, 2, 2, 2,
2, 2, 2, 1, 2, 2, 2, 2, 1, 1, 2, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2,
2]

# Hitung confusion matrix
cm = confusion_matrix(true_labels, predicted_labels)

# Tampilkan confusion matrix
print("Confusion Matrix:")
print(cm)
```

Kode ini digunakan untuk menghitung dan mencetak Confusion Matrix (matriks kebingungan) sebagai alat untuk menganalisis hasil prediksi model. Confusion Matrix adalah tabel yang menggambarkan performa model klasifikasi pada setiap kelas target. Ini membantu melihat berapa banyak prediksi yang benar dan salah untuk setiap kelas, serta bagaimana kelas-kelas tersebut saling bercampur. Confusion Matrix sangat berguna untuk mengevaluasi kinerja model dalam mengklasifikasikan data.

```
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# Tampilkan confusion matrix dalam bentuk heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Reds")

plt.title("Confusion Matrix")
plt.xlabel("Predicted Labels")
plt.ylabel("True Labels")

plt.show()
```

Kode ini digunakan untuk menghasilkan Heatmap dari Confusion Matrix. Heatmap adalah cara yang baik untuk memvisualisasikan matriks kebingungan dengan warna, yang membantu Anda memahami dengan lebih jelas di mana model cenderung membuat prediksi yang benar dan di mana model cenderung membuat kesalahan. Heatmap Confusion Matrix membantu memvisualisasikan dengan jelas di mana model berhasil dan di mana ia membuat kesalahan dalam melakukan klasifikasi. Ini dapat memberikan wawasan yang berharga tentang kinerja model terhadap setiap kelas target.

RIWAYAT HIDUP

Nurhaliza, lahir di Takengon, Aceh Tengah pada tanggal 24 April 2001. Anak kedua dari empat bersaudara, dari pasangan Ibu Ratna Juwita dan Bapak Mustafa. Penulis menyelesaikan pendidikan Sekolah Dasar di SD Negeri 4 Lut Tawar. Kemudian melanjutkan Pendidikan di jenjang Sekolah Menengah Pertama di SMP Negeri 4 Takengon dan lulus tahun 2016. Penulis menempuh pedidikan jenjang Sekolah Menengah Atas di SMA Swasta Muslimat Samalanga dan lulus pada tahun 2019. Pada tahun yang sama penulis terdaftar sebagai mahasiswi pada program studi Teknologi Informasi, Fakultas Sains dan Teknologi, Universitas Islam Negeri Ar-Raniry, Banda Aceh melalui jalur seleksi SNMPTN.