

BAB II

KAJIAN LITERATUR

2.1 Tinjauan Pustaka

Pada bagian ini berisikan landasan teori dan akan dijelaskan tinjauan pustaka yang berkaitan dengan penelitian yang dilakukan.

2.1.1 Google Play Store

Google Play Store sebelumnya dikenal sebagai Android Market adalah layanan distribusi digital yang dioperasikan dan dikembangkan oleh Google. Layanan ini berfungsi sebagai toko aplikasi resmi untuk perangkat bersertifikasi yang menjalankan sistem operasi Android dan turunannya serta ChromeOS, memungkinkan pengguna untuk menjelajahi dan mengunduh aplikasi yang dikembangkan dengan kit pengembangan perangkat lunak Android dan diterbitkan melalui Google. Google Play juga berfungsi sebagai toko media digital, dengan menawarkan berbagai media untuk dibeli (serta beberapa hal yang tersedia secara gratis) seperti buku, film, single musik, program televisi dan *video game* [9].

Aplikasi tersedia melalui Google Play baik secara gratis maupun berbayar. Aplikasi dapat diunduh langsung ke perangkat Android melalui aplikasi seluler Google Play Store atau dengan menyebarkan aplikasi ke perangkat dari situs *web* Google Play. Aplikasi yang memanfaatkan kemampuan perangkat keras suatu perangkat dapat ditargetkan pada pengguna perangkat dengan komponen perangkat keras tertentu, seperti sensor gerak (untuk *game* yang bergantung pada gerakan) atau kamera depan (untuk panggilan video *online*). Google Play Store memiliki lebih dari 82 miliar unduhan aplikasi pada tahun 2016 dan lebih dari 3,5 juta aplikasi yang diterbitkan pada tahun 2017, sementara setelah pembersihan aplikasi, jumlahnya kembali menjadi lebih dari 3 juta. Hal ini telah menjadi subjek berbagai masalah terkait keamanan, di mana perangkat lunak berbahaya telah disetujui dan diunggah ke toko dan diunduh oleh pengguna, dengan berbagai tingkat keparahan [9].

Fungsi dan fitur utama Google Play Store:

1. Tempat unduh aplikasi dan *game*: menyediakan jutaan aplikasi dan *game* Android yang dikembangkan oleh *developer* global.
2. Toko konten digital: menawarkan berbagai media lain seperti buku elektronik (*play books*) dan film.

3. Keamanan terjamin: menyediakan aplikasi yang telah diverifikasi untuk memastikan keamanan perangkat Android.
4. Pembaruan otomatis: memungkinkan pengguna untuk memperbarui aplikasi yang sudah diinstal dengan mudah.
5. Bisa gratis atau berbayar: menampilkan aplikasi gratis dan berbayar yang dapat disesuaikan dengan kebutuhan [10].

Ulasan Google Play Store adalah fitur masukan pengguna berupa *rating* bintang 1-5 dan komentar tertulis mengenai aplikasi di Google Play Store. Tujuannya untuk berbagi pengalaman, membantu pengguna lain menemukan aplikasi berkualitas, serta memberikan *feedback* bagi *developer* untuk perbaikan. Berikut adalah penjelasan detail Google Play Store terdiri dari:

1. Fungsi utama:
 - a. Bagi pengguna: membantu memutuskan apakah akan mengunduh aplikasi dengan membaca pengalaman orang lain.
 - b. Bagi *developer*: alat analisis untuk mengetahui kelebihan, kekurangan, dan *bug* aplikasi serta saran fitur.
 - c. Algoritma Google Play Store: mempengaruhi visibilitas aplikasi. Aplikasi dengan *rating* dan ulasan positif lebih mudah ditemukan.
2. Komponen ulasan: terdiri dari *rating* bintang (1 sampai 5) dan teks komentar.
3. Kebijakan Google Play Store:
 - a. Ulasan harus jujur, relevan, dan berdasarkan pengalaman pribadi.
 - b. Tidak boleh mengandung *spam*, konten seksual, ujaran kebencian, atau informasi pribadi.
 - c. Google dapat menghapus ulasan yang melanggar kebijakan.
4. Fitur terbaru: *rating* yang ditampilkan adalah kualitas terbaru aplikasi, bukan rata-rata sepanjang waktu, sehingga mencerminkan kondisi aplikasi terkini.
5. Respons *developer* dapat menanggapi langsung ulasan pengguna, memfasilitasi komunikasi untuk solusi masalah [11].

Dalam rekayasa perangkat lunak, ulasan di Google Play Store berfungsi sebagai jembatan komunikasi antara pengembang dan pengguna.

1. Identifikasi masalah: ulasan seringkali berisi laporan *bug* atau kegagalan sistem yang tidak ditemukan saat tahap pengujian internal.
2. Saran fitur: pengguna menggunakan kolom ulasan untuk menyampaikan kebutuhan fungsional baru, yang dapat menjadi dasar pengembangan aplikasi versi berikutnya [11].

Terdapat kaitan erat antara nilai kuantitatif (*rating*) dan nilai kualitatif (ulasan) yaitu:

1. Validasi data: dalam penelitian, *rating* bintang 1-5 digunakan untuk melabeli data secara otomatis (misal: bintang 1-2 sebagai sentimen negatif, bintang 4-5 sebagai positif).
2. Anomali data: terkadang ulasan teks tidak sejalan dengan *rating* (misalnya: komentar pujian tapi *rating* bintang 1). Hal ini sering diteliti dalam studi akurasi data atau perilaku pengguna.

Dalam teori pemasaran digital, ulasan di Google Play Store berfungsi sebagai *social proof* (bukti sosial) berupa:

1. Kepercayaan pengguna: calon pengguna cenderung membaca ulasan sebelum memutuskan untuk mengunduh. Ulasan positif meningkatkan kepercayaan, sementara ulasan negatif menjadi hambatan.
2. Algoritma rekomendasi: Google Play Store menggunakan volume dan sentimen ulasan sebagai salah satu faktor untuk menaikkan peringkat aplikasi di hasil pencarian [11].

Secara teoritis, ulasan adalah perwujudan dari *User Experience* (UX) yang dirasakan secara subjektif. Ini mencakup aspek:

1. *Usability*: aplikasi mudah digunakan.
2. *Reliability*: aplikasi stabil dan jarang *crash*.
3. *Utility*: aplikasi benar-benar berguna bagi pengguna.

Jadi dalam bidang ilmu komputer atau *data science*, ulasan merupakan data mentah yang sangat berharga dengan tujuan:

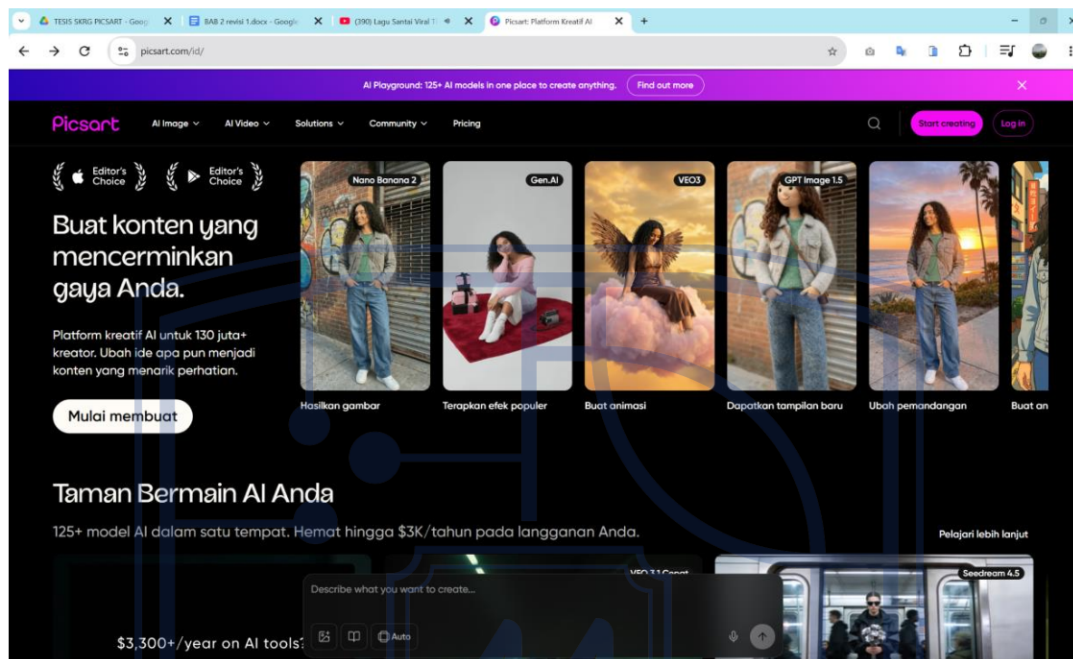
1. Analisis sentimen: mengekstrak opini untuk mengetahui tren persepsi publik terhadap sebuah layanan (misal: ulasan aplikasi m-banking atau *e-commerce*).
2. Topik *modeling*: mengelompokkan ribuan ulasan secara otomatis untuk menemukan topik yang paling sering dikeluhkan pengguna (misal: masalah "*login*", "*pembayaran*", atau "*tampilan*") [12].

2.1.2 Aplikasi Picsart

Aplikasi Picsart adalah penyunting foto dan video lengkap dengan fitur AI canggih untuk membuat editan populer seperti Foto Blur, Golden Hour, Mirror *Selfie*, filter VHS atau Y2K [13].

Aplikasi Picsart sekaligus *platform* komunitas desain visual, yang paling banyak digunakan di dunia. Tersedia untuk perangkat seluler (Android dan iOS) serta *web*, aplikasi ini dikenal karena menggabungkan alat pengeditan tingkat profesional dengan antarmuka yang ramah pengguna. Aplikasi Picsart merupakan salah satu *platform* penyuntingan visual

dan desain grafis seluler pihak ketiga (*third-party*) terbesar di dunia. Tampilan aplikasi *website* Picsart dapat dilihat pada Gambar 2.1.



Gambar 2.1 Tampilan Aplikasi Picsart [14]

Masa awal peluncurannya menempatkan *Picsart* sebagai aplikasi penyuntingan foto konvensional yang mengandalkan alat manipulasi piksel manual (seperti pemotongan, *filter* warna, dan penyesuaian kontras). Disrupsi teknologi komputasi awan dan *Machine Learning* kemudian mendorong ekosistem *Picsart* untuk mengalami evolusi fundamental. Data analitik pasar aplikasi seluler ekosistem *Android* di Indonesia menunjukkan bahwa *Picsart* saat ini menduduki peringkat teratas (*Top 3*) dengan akumulasi lebih dari ratusan juta unduhan secara global dan memimpin pangsa pasar aplikasi utilitas visual bertenaga kecerdasan buatan. Transformasi ini secara resmi ditandai dengan peluncuran ekosistem *Picsart Ignite*, sebuah portofolio yang mengintegrasikan lebih dari 20 alat berbasis kecerdasan buatan *generative artificial intelligence* (AI) langsung ke dalam antarmuka ponsel pintar pengguna. Kecerdasan buatan diskriminatif hanya membedakan pola, sedangkan generatif AI pada *Picsart* bertugas untuk mensintesis dan menciptakan piksel atau data visual baru yang sebelumnya tidak eksis, berdasarkan instruksi teks (*text-to-image prompt*). Beberapa fitur komputasi tingkat tinggi yang menjadi inti operasional *Picsart* saat ini meliputi:

1. *AI Replace (inpainting)*: fitur ini memungkinkan pengguna untuk menyorot (*masking*) objek tertentu di dalam foto, lalu menggantinya dengan objek lain yang sepenuhnya baru hanya dengan mengetikkan deskripsi teks. Model AI secara otomatis merekonstruksi area sekitarnya agar pencahayaan dan bayangan menyatu secara realistis dengan gambar asli.
2. *AI Avatar (face generator)*: sistem komputasi yang mampu menganalisis topologi wajah dari foto swafoto (*selfie*) pengguna, lalu membangun ulang wajah tersebut ke dalam puluhan gaya seni digital (seperti 3D, lukisan klasik, atau fantasi) tanpa memerlukan keterampilan desain grafis.
3. *AI Background Generator & Generative Fill*: fitur penghapusan latar belakang (*background removal*) cerdas yang tidak hanya mengeliminasi piksel di belakang objek utama, tetapi juga mampu mensintesis latar belakang pemandangan atau studio baru (berbasis *prompt*) yang disesuaikan secara proporsional dengan perspektif objek utama.
4. *AI Expand (out painting)*: algoritma yang mendeteksi pola di tepian gambar asli, lalu memperluas kanvas foto dengan menebak dan menggambar kelanjutan dari pemandangan tersebut secara mulus hingga melampaui batas bingkai asli.

Interaksi pengguna dengan fitur-fitur mutakhir *Picsart* ini dalam praktiknya sering kali menghasilkan anomali atau malfungsi yang tidak ditemukan pada aplikasi konvensional. Anomali tersebut terbagi menjadi dua kategori utama yaitu:

1. Malfungsi mekanis (sistemik)

Meliputi aplikasi yang tertutup paksa (*force close*), layar membeku (*freeze*), atau galat jaringan (*network error*) saat pengguna mencoba mengeksport gambar beresolusi tinggi hasil *render* AI.

2. Malfungsi kognitif (algoritmik)

Meliputi kegagalan model bahasa AI dalam menginterpretasikan perintah *prompt* pengguna, distorsi piksel pada hasil *generate* (seperti bentuk jari manusia yang tidak proporsional), fenomena halusinasi AI (menghasilkan objek yang tidak diminta), hingga hasil *AI Avatar* yang tidak menyerupai wajah asli pengguna.

Kompleksitas malfungsi yang terjadi di atas, memicu lonjakan keluhan teknis dari para pengguna aplikasi. Dinamika keluhan tersebut direkam secara langsung dalam bentuk ulasan aplikasi di *Play Store* maupun *platform* aduan konsumen, sehingga semua orang, baik para pengguna *Picsart*, terutama pihak pengembang aplikasi *Picsart* langsung dapat mengetahui laporan *bug* dan saran fitur yang disampaikan.

2.1.3 Siklus Pemeliharaan Perangkat Lunak

Pemeliharaan perangkat lunak (*software maintenance*) adalah proses berkelanjutan untuk memperbarui, mengoptimalkan dan menyempurnakan aplikasi perangkat lunak setelah diimplementasikan. Hal ini memastikan aplikasi tetap berfungsi, aman, dan selaras dengan kebutuhan yang terus berkembang. Ini fase krusial dan berkesinambungan dalam Siklus Hidup Pengembangan Perangkat Lunak (SDLC) yang terjadi setelah perangkat lunak dirilis ke lingkungan produksi. Fase ini mewakili sebagian besar dari total biaya siklus hidup perangkat lunak. Pemeliharaan berlanjut sepanjang masa operasional perangkat lunak. Ini mencakup perbaikan *bug*, pembaruan keamanan, optimasi kinerja, dan peningkatan fitur. Fase pemeliharaan memastikan perangkat lunak tetap aman, sesuai standar, dan bernilai tinggi dalam jangka waktu lama setelah rilis awal. Berikut beberapa aspek penting dari fase pemeliharaan dalam SDLC terdiri dari:

1. Perbaikan *bug*

Salah satu aktivitas utama selama fase pemeliharaan adalah mengidentifikasi dan memperbaiki *bug* atau cacat perangkat lunak yang mungkin terlewatkan selama fase pengujian atau muncul di lingkungan produksi. Aktivitas ini termasuk menangani masalah yang dilaporkan oleh pengguna.

2. Peningkatan

Seiring perubahan kebutuhan pengguna atau bisnis dari waktu ke waktu, fitur atau fungsionalitas baru mungkin perlu ditambahkan ke perangkat lunak. Fase pemeliharaan mencakup perencanaan, perancangan, dan implementasi peningkatan-peningkatan tersebut.

3. Adaptasi terhadap perubahan lingkungan

Baik itu perangkat keras baru, sistem operasi, atau faktor eksternal lainnya, perangkat lunak Anda harus beradaptasi agar tetap kompatibel dan berfungsi.

4. Optimalisasi kinerja

Memantau dan mengoptimalkan kinerja perangkat lunak untuk memastikan perangkat lunak berjalan efisien dan memenuhi persyaratan kinerjanya merupakan bagian penting dari pemeliharaan.

Hal ini mungkin mencakup penyetelan basis data, optimasi kode, dan peningkatan skala infrastruktur.

5. Pembaruan keamanan

Ancaman keamanan siber terus berubah. Anda perlu menjaga keamanan perangkat lunak Anda. Ini mencakup penerapan *patch* keamanan dan pembaruan pustaka. Atasi kerentanan segera setelah ditemukan.

6. Dokumentasi

Memastikan dokumentasi selalu mutakhir sangat penting untuk dukungan jangka panjang perangkat lunak. Ini termasuk manual pengguna, dokumentasi teknis, dan diagram arsitektur.

7. Dukungan pengguna

Memberikan dukungan kepada pengguna dan menangani pertanyaan atau masalah pengguna merupakan tugas berkelanjutan selama fase pemeliharaan. Hal ini mungkin mencakup pembentukan helpdesk atau tim dukungan.

8. Pencadangan dan pemulihan

Memastikan bahwa data dicadangkan secara teratur dan bahwa ada prosedur yang diterapkan untuk pemulihan bencana merupakan bagian dari menjaga ketersediaan dan keandalan perangkat lunak.

9. Pengujian regresi

Setiap kali perubahan atau pembaruan dilakukan selama fase pemeliharaan, pengujian regresi dilakukan untuk memastikan bahwa fungsionalitas yang ada tidak terpengaruh secara negatif oleh modifikasi tersebut.

10. Kontrol versi

Mengelola berbagai versi perangkat lunak sangat penting selama pemeliharaan. Sistem kontrol versi melacak perubahan dan memastikan versi yang benar digunakan.

11. Masa pensiun

Pada akhirnya, perangkat lunak mungkin akan mencapai akhir masa pakainya, dan rencana untuk penghentian atau penggantian harus dikembangkan dan dilaksanakan.

Pemeliharaan perangkat lunak dikategorikan menjadi empat jenis yang berbeda, masing-masing memiliki tujuan spesifik dalam siklus hidup perangkat lunak. Memahami jenis-jenis ini membantu organisasi mengalokasikan sumber daya secara efektif dan merencanakan strategi pemeliharaan.

1. Pemeliharaan korektif (perbaikan *bug* dan penyelesaian cacat)

Pemeliharaan korektif menangani cacat dan kesalahan perangkat lunak yang ditemukan setelah penerapan. Bentuk pemeliharaan reaktif ini berfokus pada perbaikan *bug*,

kerusakan, masalah kinerja, dan kesalahan fungsional yang dilaporkan oleh pengguna atau terdeteksi melalui pemantauan.

Contoh: memperbaiki *bug* pada proses pembayaran di aplikasi e-commerce yang mencegah pelanggan menyelesaikan pembelian.

2. Pemeliharaan adaptif (pembaruan lingkungan dan *platform*)

Pemeliharaan adaptif memodifikasi perangkat lunak agar tetap kompatibel dengan perubahan lingkungan eksternal, *platform*, dan teknologi. Seiring dengan evolusi sistem operasi, basis data, *platform cloud*, dan layanan pihak ketiga, perangkat lunak harus beradaptasi untuk mempertahankan fungsinya.

Contoh: memperbarui aplikasi perbankan seluler agar mendukung iOS 18 dan fitur keamanan iPhone yang baru.

3. Pemeliharaan perfektif (peningkatan dan optimalisasi)

Pemeliharaan meningkatkan fungsionalitas, kinerja, kegunaan, dan pemeliharaan perangkat lunak berdasarkan umpan balik pengguna dan kebutuhan bisnis yang terus berkembang. Jenis pemeliharaan proaktif ini menambah nilai dengan meningkatkan fitur yang ada atau memperkenalkan kemampuan baru.

Contoh: menambahkan fitur "simpan untuk nanti" ke keranjang belanja e-commerce berdasarkan permintaan pelanggan.

4. Pemeliharaan preventif (mitigasi risiko proaktif)

Pemeliharaan preventif mengidentifikasi dan menyelesaikan potensi masalah sebelum menyebabkan masalah atau kegagalan. Pendekatan yang berwawasan ke depan ini mengurangi biaya pemeliharaan di masa mendatang dengan mengatasi risiko sejak dini dan meningkatkan ketahanan sistem.

Contoh: memperbarui ke versi terbaru dari pustaka keamanan untuk mencegah potensi kerentanan, bahkan sebelum eksploitasi apa pun ditemukan [15].

Standar internasional IEEE 14764 menjabarkan proses siklus hidup pemeliharaan perangkat lunak. Dalam konteks pemeliharaan adaptif, siklus ini berjalan melalui tahapan sistematis berikut:

1. Analisis masalah dan modifikasi (*problem and modification analysis*)

- a. Identifikasi perubahan lingkungan: menganalisis elemen eksternal apa yang berubah (misal: *deprecation* pada pustaka lama, perubahan struktur data *cloud*).
- b. Analisis dampak (*impact analysis*): mengevaluasi komponen sistem mana saja yang akan terdampak oleh adaptasi ini. hal ini penting untuk meminimalisir risiko kerusakan pada modul lain yang sudah berjalan baik.

- c. Perencanaan: menentukan ruang lingkup adaptasi, estimasi sumber daya, dan jadwal pelaksanaan.
2. Implementasi modifikasi (*modification implementation*)
 - a. Desain adaptasi: menyesuaikan arsitektur perangkat lunak untuk mengakomodasi lingkungan baru tanpa merusak fungsionalitas inti.
 - b. Pengkodean (*coding*): proses penulisan atau refaktor kode secara aktual berdasarkan desain adaptasi.
 - c. Pengujian unit: menguji modul yang diubah untuk memastikan logika adaptasinya berjalan dengan baik di lingkungan yang baru.
3. Tinjauan pemeliharaan dan pengujian (*maintenance review/acceptance*)
 - a. Pengujian regresi (*regression testing*): memastikan bahwa adaptasi pada lingkungan baru tidak menyebabkan *bug* atau kegagalan pada fitur-fitur yang sebelumnya berfungsi normal.
 - b. Pengujian integrasi & sistem: menguji sistem secara keseluruhan dalam lingkungan sistem (OS, *database*, atau *server*) yang baru.
 - c. Penerimaan (*acceptance*): validasi akhir bahwa perangkat lunak kini sepenuhnya kompatibel dengan lingkungan operasional yang baru.
4. Migrasi (*migration*) - jika diperlukan
 Jika pemeliharaan adaptif melibatkan perpindahan *platform* secara total (misalnya migrasi dari *server on-premise* ke infrastruktur *cloud*), proses ini mencakup konversi data, parallel run (menjalankan sistem lama dan baru secara bersamaan sementara).
5. Pengiriman perangkat lunak (*software delivery*)
 Mendistribusikan perangkat lunak yang telah diadaptasi ke lingkungan *production*, disertai dengan pembaruan dokumentasi teknis perubahan lingkungan baru [15].

2.1.4 Ekstraksi Kebutuhan Otomatis (*Automated Requirement Extraction*)

Ekstraksi kebutuhan otomatis (*automated requirement extraction*) adalah proses komputasi untuk mengidentifikasi dan mengekstrak informasi kebutuhan sistem secara otomatis dari dokumen tidak terstruktur. Teknologi ini sangat krusial dalam Rekayasa Perangkat Lunak (*Requirement Engineering*) untuk mempercepat analisis dokumen teks bebas dan mencegah kesalahan interpretasi manusia. Ekstraksi kebutuhan otomatis terdiri dari:

1. Rekayasa kebutuhan (*requirement engineering* - RE)

Merupakan fase awal dalam siklus pengembangan perangkat lunak yang berfokus pada

pendefinisian tujuan, fungsi, dan batasan sistem. Proses ini melibatkan tahapan:

- a. Elisitasi: pengumpulan data dari *stakeholder*.
- b. Analisis dan negosiasi: memprioritaskan dan menyelesaikan konflik kebutuhan.
- c. Spesifikasi: Pendokumentasian kebutuhan secara formal.
- d. Validasi: Memastikan kebutuhan sesuai dengan keinginan pengguna.

2. Pemrosesan Bahasa Alami (*Natural Language Processing* - NLP)

Landasan komputasi utama dalam ekstraksi otomatis, di mana komputer membaca, menguraikan dan memahami bahasa manusia. Tahapan NLP yang digunakan meliputi:

- a. *Tokenization*: memecah teks menjadi kata atau frasa dasar (token).
- b. *Part-of-Speech tagging*: mengidentifikasi kelas kata (kata benda, kata kerja) untuk mencari subjek dan aksi fungsional.
- c. *Named entity recognition*: mendeteksi entitas penting seperti nama sistem, aktor, atau istilah teknis spesifik.

3. Analisis semantik dan sintaksis

- a. Pola linguistik (sintaksis): Menggunakan aturan tata bahasa (*grammar rules*) untuk menemukan pola kalimat kebutuhan (contoh: "Sistem harus dapat [Fitur]").
- b. Ontologi dan glosarium: pemetaan istilah ke dalam basis pengetahuan khusus domain untuk memahami konteks dan relasi antar kebutuhan.

4. Pendekatan komputasi

- a. Berbasis aturan (*rule-based*): ekstraksi berdasarkan kamus kata kerja kunci (seperti harus, dapat, wajib) dan pola kalimat yang telah ditentukan manusia.
- b. Berbasis kecerdasan buatan: menggunakan pembelajaran mesin (*machine learning*) dan *deep learning* untuk mengenali pola kompleks, di mana sistem dilatih menggunakan korpus data teks untuk meningkatkan akurasi seiring waktu. [16]

Dalam ekosistem distribusi perangkat lunak modern seperti Google Play Store, pendekatan tradisional elisitasi kebutuhan tidak lagi memiliki skalabilitas yang memadai untuk memproses volume umpan balik pengguna yang sangat masif (*crowdsourced data*). Menurut Mihany dkk. (2025), proses ekstraksi kebutuhan secara manual dari ulasan aplikasi telah menjadi tugas yang sangat kompleks dan tidak efisien akibat penggunaan bahasa informal, tingginya tingkat redundansi, serta dominasi data bising (*noisy text*) yang menyembunyikan informasi berharga bagi tim pengembang. Untuk mengatasi hambatan komputasional tersebut, ekstraksi kebutuhan otomatis memanfaatkan integrasi teknik Pemrosesan Bahasa Alami (NLP) dengan model berbasis arsitektur *transformer* guna menjembatani kesenjangan antara keluhan pengguna akhir dengan repositori *backlog*

pengembang. Dalam praktiknya, sistem ekstraksi otomatis ini menerapkan klasifikasi multi-kelas untuk mengurai niat linguistik (*linguistic* intensi) secara spesifik, yang secara langsung memisahkan ulasan ke dalam taksonomi operasional Otomatisasi berbasis pemodelan data teks ini memberikan keunggulan analitik yang sangat krusial bagi pengembang, karena memungkinkan penyaringan ribuan umpan balik ambigu menjadi wawasan yang *actionable* dan berbasis data (*data-driven*), sehingga siklus perilsan pembaruan (*update patches*) aplikasi dapat dieksekusi dengan tingkat presisi dan efisiensi waktu yang jauh lebih tinggi [17].

2.1.5 Analisis Sentimen dan Ekstraksi Opini

Analisis sentimen dikenal sebagai penambahan opini atau emosi adalah penggunaan pemrosesan bahasa alami, analisis teks, linguistik komputasional, dan biometrik untuk secara sistematis mengidentifikasi, mengekstrak, mengukur, dan mempelajari keadaan afektif dan informasi subjektif. Analisis sentimen banyak diterapkan pada materi suara pelanggan seperti ulasan dan tanggapan survei, media *online* dan media sosial, dan materi perawatan kesehatan untuk aplikasi yang berkisar dari pemasaran hingga layanan pelanggan hingga kedokteran klinis dan lain sebagainya [18].

Analisis sentimen memainkan peran penting dalam memahami berbagai opini yang berkembang di masyarakat. Dengan teknik ini, dapat mengetahui bagaimana perasaan atau sikap publik terhadap suatu topik, baik itu dalam ulasan produk, komentar di media sosial, atau artikel berita. Analisis sentimen memberikan wawasan yang sangat berguna bagi banyak pihak untuk menyesuaikan strategi mereka.

Tujuan analisis sentimen antara lain:

1. Memahami reaksi publik

Mengetahui apakah opini masyarakat terhadap suatu topik lebih condong positif, negatif, atau netral.

2. Meningkatkan efektivitas pemasaran

Menyesuaikan strategi pemasaran berdasarkan ulasan dan umpan balik (*feedback*) dari pelanggan.

3. Mengoptimalkan produk atau layanan

Mendeteksi area yang perlu diperbaiki dari ulasan negatif serta mempertahankan kekuatan dari ulasan positif.

4. Mendukung pengambilan keputusan

Menyediakan gambaran akurat tentang opini publik untuk keputusan strategis [19].

Beberapa manfaat utama dari analisis sentimen dalam penelitian meliputi:

1. Menghemat waktu dan tenaga.
Memproses data dalam jumlah besar lebih cepat daripada cara manual.
2. Memberikan evaluasi dan ide baru.
Menyediakan evaluasi yang lebih terukur untuk pengembangan studi atau produk.
3. Mengubah data tak terstruktur menjadi tersusun rapi.
4. Mengonversi data teks yang tidak teratur menjadi informasi mudah dipahami.

Terdapat dua teknik dalam klasifikasi sentimen yaitu:

1. Klasifikasi sentimen menggunakan *supervised learning*
Supervised learning bekerja dengan cara membuat *classifier* yang membutuhkan dua set dokumen untuk klasifikasi yaitu set data latih dan set data uji. Metode ini dikenal sebagai metode pembelajaran mesin atau *machine learning*. Sederhananya, pada metode ini, data akan dilatih menggunakan set data latih yang telah diberi label secara manual (misalnya positif, negatif dan netral).
2. Klasifikasi sentimen menggunakan *unsupervised learning*
Unsupervised learning bekerja dengan cara membandingkan teks dengan kata atau leksikon yang sudah diberikan nilai sentimennya. Dokumen akan diperiksa dan dibandingkan dengan kata-kata positif atau negatif. Jika sebuah dokumen atau *review* memiliki lebih banyak kata positif, maka dokumen tersebut akan diklasifikasikan sebagai positif. Jika sebaliknya maka akan diklasifikasikan sebagai negatif [20].

Beberapa jenis analisis sentimen yang dapat ditemukan meliputi:

1. *Emotion detection*
Mendeteksi emosi seperti senang, sedih, atau marah yang diekspresikan dalam teks.
2. *Aspect-based sentiment analysis*
Menentukan aspek-aspek spesifik yang disukai atau tidak disukai oleh pelanggan dalam suatu produk.
3. *Fine granularity sentiment analysis*
Menganalisis sentimen secara lebih mendetail, sering digunakan dalam analisis ulasan produk di *e-commerce* [20].

Analisis sentimen pada dasarnya diklasifikasikan menjadi tiga kategori sebagai berikut:

1. Sentimen positif

Sentimen positif berisi kumpulan kata-kata baik atau kata-kata yang bersifat positif.

Apabila kata-kata positif memiliki kuantitas yang lebih besar maka sentimen tersebut dapat dikategorikan sebagai kategori positif. Semakin banyak komentar positif terhadap suatu produk maka akan lebih banyak pelanggan yang akan membeli produk tersebut. Contoh kata kunci: bagus, mudah, cepat, mantap, oke, memuaskan, sangat membantu, oke, keren, mantap, recommended, nyaman, indah, hebat. Contoh: “Produk ini sangat bagus”.

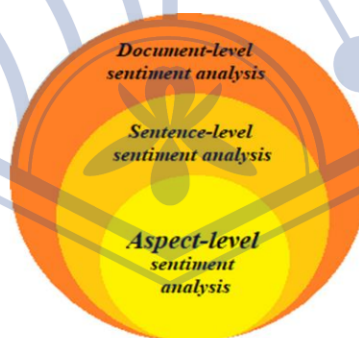
2. Sentimen negatif

Sentimen negatif berisi kumpulan kata-kata yang bersifat negatif. Semakin banyak komentar negatif terhadap suatu produk maka semakin sedikit pelanggan yang akan membeli produk tersebut. Contoh kata kunci: jelek, susah, lambat, *error*, kecewa, mahal, bau, kotor, rusak, gagal, membingungkan, lelet, buruk, susah. Contoh: “Kualitas produk ini sangat buruk”.

3. Sentimen netral

Ketika suatu sentimen tidak dianggap sebagai sentimen positif maupun negatif, maka sentimen tersebut dapat dikategorikan ke dalam sentimen netral dalam proses analisis sentimen. Contoh kata kunci: "saya sudah coba", "biasa saja", "sesuai harga", "jumat", "besok", "lokasi", "cukup", "pernah". Contoh: “Saya sudah pernah membeli produk ini” [21].

Analisis sentimen dilakukan dalam beberapa level. Terdapat 3 level dalam analisis sentimen yang dapat dilihat pada Gambar 2.2.



Gambar 2.2 Level Analisis Sentimen [22]

Penjelasan:

1. Level dokumen analisis sentimen

Level ini mengklasifikasikan keseluruhan dokumen atau *file* dalam analisis sentimen. dokumen tersebut kemudian dapat diidentifikasi sebagai opini positif, negatif atau

netral. Setiap dokumen diklasifikasikan berdasarkan sentimen keseluruhan dari opini tentang suatu entitas tunggal. Level dokumen analisis sentimen bekerja paling baik ketika dokumen ditulis oleh satu orang dan tidak cocok untuk dokumen yang mengevaluasi atau membandingkan banyak entitas.

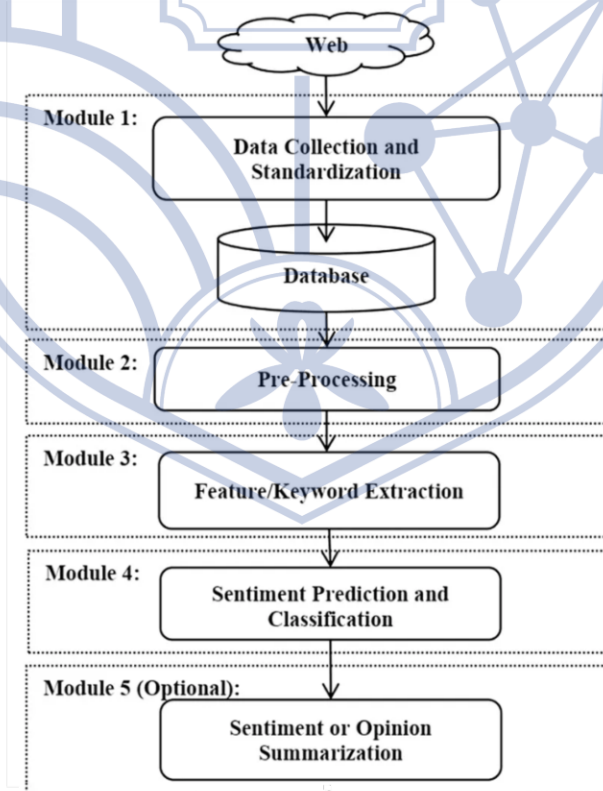
2. Level kalimat analisis sentimen

Level ini berfokus pada kalimat. setiap kalimat akan diidentifikasi sebagai opini positif, negatif atau netral. Tetapi, terlebih dahulu kalimat tersebut perlu diklasifikasikan sebagai kalimat yang mengungkapkan informasi faktual secara objektif atau mengungkapkan pandangan dan pendapat secara subjektif.

3. Level aspek analisis sentimen

Level ini mendefinisikan sentimen berdasarkan aspek tertentu dari suatu entitas. Contohnya seperti “Kamera iPhone 11 luar biasa”. Ulasan tersebut ada di “kamera” yang merupakan aspek dari entitas “iPhone 11” dan ulasannya positif. Ekstraksi aspek merupakan tugas ini untuk analisis sentimen yang dapat berupa aspek implisit maupun eksplisit [22].

Kerangka kerja analisis sentimen secara umum terdiri dari 5 modul utama ditunjuk pada gambar 2.3.



Gambar 2.3 Kerangka Kerja Analisis Sentimen [23]

Penjelasan gambar:

1. Pengumpulan dan standarisasi data (*data collecting*)

Modul ini berfokus pada pengumpulan data yang relevan untuk analisis sentimen. Data yang dikumpulkan dapat berasal dari berbagai sumber, seperti *platform online*, *web*, atau basis data publik. Selain itu, modul ini juga mencakup langkah-langkah untuk melakukan standarisasi data agar sesuai dengan kebutuhan analisis sentimen.

2. Prapemrosesan data (*data pre-processing*)

Modul ini melibatkan langkah-langkah untuk membersihkan dan mempersiapkan data sebelum dilakukan analisis sentimen. Prapemrosesan data meliputi penghapusan karakter khusus, penyeragaman huruf, penghapusan *stopword*, *stemming*, dan normalisasi teks.

3. Ekstraksi fitur atau kata kunci (*feature extraction*)

Modul ini bertujuan untuk mengidentifikasi kata-kata atau fitur yang mewakili sentimen keseluruhan dari *dataset*. Metode yang digunakan untuk ekstraksi fitur dapat beragam, seperti pendekatan berbasis kamus, pendekatan berbasis statistik, atau pendekatan berbasis pembelajaran mesin.

4. Prediksi atau klasifikasi sentimen (*sentiment classification/prediction*)

Modul ini melibatkan penggunaan teknik-teknik untuk memprediksi atau mengklasifikasikan sentimen yang terkait dengan kata kunci atau seluruh kalimat atau dokumen, sesuai dengan kebutuhan analisis. Beberapa teknik yang umum digunakan termasuk pendekatan berbasis pembelajaran mesin, seperti klasifikasi berbasis aturan, klasifikasi berbasis fitur, atau klasifikasi berbasis jaringan saraf.

5. Peringkasan sentimen atau opini (*sentiment or opinion summarization*)

Modul ini bersifat opsional (pilihan tambahan) dan merupakan tahap analitik lanjutan setelah prediksi selesai dilakukan. Setelah modul ke-4 berhasil mengklasifikasikan data ke dalam berbagai label sentimen, modul ke-5 bertugas untuk mengekstrak dan merangkum esensi dari ribuan ulasan tersebut menjadi informasi yang padat. Alih-alih hanya menampilkan data statistik mentah, sistem akan menghasilkan ringkasan representatif mengenai apa yang sebenarnya dibicarakan oleh pengguna (misalnya, menyimpulkan bahwa mayoritas sentimen negatif berpusat pada kegagalan fungsi *generative fill*). Fitur peringkasan ini sangat membantu pengembang aplikasi atau pemangku kepentingan untuk langsung menangkap wawasan utama (*key insights*) tanpa harus membaca dokumen teks satu per satu. [23].

Analisis sentimen memungkinkan pengambil keputusan untuk:

1. Menyediakan data akurat

Gambaran yang lebih akurat tentang opini masyarakat membantu dalam merumuskan strategi.

2. Mengidentifikasi area yang perlu diperbaiki

Ulasan negatif dapat membantu dalam mengidentifikasi aspek yang perlu diperbaiki.

3. Memantau reputasi

Feedback dari media sosial memungkinkan perusahaan untuk terus memantau dan menjaga citra merek.

Pengaplikasian analisis sentimen dapat dilakukan pada beberapa area seperti:

1. *E-commerce*

Banyak situs *web* yang menyediakan fitur untuk menyampaikan ulasan mereka terhadap suatu produk. Ulasan yang diberikan bermanfaat bagi pengguna lain maupun produsen dari produk tersebut.

2. *Voice of customer* (VOC)

Voice of customer berhubungan dengan teknik riset pasar yang mendefinisikan kebutuhan dan harapan dari pelanggan karena menentukan keandalan suatu produk.

3. Pemerintahan Pada bidang pemerintahan

Analisis sentimen dapat membantu untuk melihat kekuatan serta kelemahan dari ulasan publik tentang isu pemerintahan.

4. Pemasaran

Analisis sentimen membantu para produsen untuk menemukan pelanggan mana yang loyal dan tidak serta bagaimana membuat pelanggan mereka menjadi lebih loyal.

5. Politik

Sebelum masa pemilu, pandangan publik dapat dianalisis melalui komentar atau ulasan yang mereka berikan di media sosial.

6. Analisis blog

Analisis sentimen dapat dengan efektif digunakan untuk menambang pertengkaran dalam diskusi atau forum debat di mana hal ini dapat diterapkan untuk menganalisis posting blog dan melakukan subjektifitas.

7. Prediksi pasar saham

Analisis sentimen dapat digunakan untuk meramalkan setiap kejadian di pasar saham [24].

Ada beberapa tantangan dalam analisis sentimen meliputi:

1. Ambiguitas bahasa

Kata-kata yang memiliki makna ganda seringkali sulit untuk diklasifikasikan dengan tepat.

2. Data *noise*

Adanya data yang tidak relevan dapat mengganggu hasil analisis.

3. Skalabilitas

Menganalisis data dalam jumlah besar membutuhkan waktu dan sumber daya yang memadai.

Beberapa cara yang dapat dilakukan untuk meningkatkan akurasi analisis sentimen yaitu:

1. Memperluas data latih.

Menggunakan lebih banyak data untuk melatih model *machine learning*.

2. Menerapkan teknik *preprocessing*.

Membersihkan data agar lebih sesuai untuk dianalisis.

Saat ini dengan teknologi baru akan membantu proses analisis dalam hal:

1. Mempercepat proses analisis.

Memungkinkan analisis dilakukan dengan lebih cepat tanpa mengorbankan akurasi.

2. Meningkatkan pemahaman kontekstual.

Memberikan analisis yang lebih akurat terhadap konteks tertentu dalam teks [24].

Analisis sentimen memberikan wawasan mendalam tentang opini publik, membantu perusahaan memahami persepsi masyarakat dan membuat keputusan akurat. Perusahaan dapat meningkatkan strategi bisnis, memantau reputasi, dan merespons pelanggan lebih cepat. Perkembangan *platform digital* seperti *website*, aplikasi, media sosial yang sangat pesat membuat ketersediaan opini dan sentimen publik semakin meningkat. Hal ini membuat analisis sentimen menjadi alat penting untuk memahami sentimen publik terhadap suatu produk atau layanan. Dalam konteks dunia digital, analisis sentimen digunakan untuk mengetahui persepsi pengguna terhadap produk atau layanan. Analisis sentimen dapat dikategorikan ke dalam tiga bagian berdasarkan tingkat granularitas yaitu:

1. Tingkat kata, di mana analisis menentukan polaritas berdasarkan suatu kata

2. Tingkat kalimat, di mana analisis menentukan polaritas berdasarkan suatu kalimat.

Analisis ini biasa digunakan untuk analisis opini.

3. Tingkat dokumen, di mana analisis menentukan polaritas berdasarkan suatu dokumen.

Dalam analisis sentimen, berfokus terutama pada analisis orientasi sentimen terhadap

kumpulan komentar, yang menunjukkan apakah kalimat yang disampaikan menunjukkan sentimen positif, negatif, atau netral terhadap suatu produk atau layanan [25].

2.1.6 Pemrosesan Bahasa Alami (NLP)

Pemrosesan Bahasa Alami disebut *Natural Language Processing* (NLP) merupakan subdisiplin dari kecerdasan buatan (AI) yang berfokus pada interaksi antara komputer dan bahasa manusia. Dengan NLP, maka komputer dapat membaca, memahami, dan menghasilkan teks yang bisa dimengerti oleh manusia. Dengan perkembangan teknologi saat ini, di mana manusia hidup dalam era digital, NLP salah satu solusi untuk mengolah volume teks yang sangat besar yang tidak dapat dilakukan secara manual oleh manusia dalam waktu singkat. NLP mampu menangkap makna, konteks, ambiguitas, serta meniru pemrosesan bahasa manusia dalam memahami pengetahuan yang tersembunyi dalam teks [26].

Tujuan fundamental dari NLP adalah membekali mesin dengan kemampuan komputasional untuk membaca, menguraikan, memahami, dan mengekstraksi makna dari teks yang tidak terstruktur. Secara umum, NLP bekerja melalui dua tahap utama:

1. Pemrosesan data (agar teks siap dibaca mesin).
2. Pengembangan algoritma (untuk memahami makna) [26].

NLP digunakan dalam banyak teknologi *modern* yang berinteraksi dengan manusia:

1. Asisten AI generatif: produk seperti ChatGPT (OpenAI), Google Gemini, dan Claude AI populer karena kemampuannya dalam melakukan riset, penulisan, dan pengkodean secara otomatis.
2. Asisten virtual: Google Assistant, Siri (Apple), dan Alexa (Amazon) digunakan untuk perintah suara sehari-hari.
3. Penerjemahan bahasa: Google Translate yang lebih akurat karena dapat memahami konteks kalimat secara keseluruhan.
4. Mesin pencari: Google Search menggunakan NLP untuk memahami maksud pencarian bahkan jika kata kunci yang dimasukkan kurang tepat.
5. Aplikasi produktivitas: *Grammarly* atau fitur penulisan cerdas di email membantu memperbaiki tata bahasa dan gaya penulisan [27].

Secara garis besar, proses NLP terbagi menjadi dua fase utama:

1. Tahap prapemrosesan teks (*text preprocessing*)

Sebelum dianalisis, teks "mentah" harus dibersihkan dan disederhanakan agar mesin tidak bingung dengan variasi bahasa yang terdiri dari *token*, *case folding*, *stopword removal*, *stemming & lemmatisasi*, *cleansing*, *normalization*, *part-of-speech tagging*.

2. Tahap analisis dan pembelajaran

Setelah teks siap, sistem akan menggunakan algoritma kecerdasan buatan untuk memahami makna atau pola terdiri dari:

- a. *Vectorisation*: mengubah teks menjadi angka (vektor) karena komputer hanya bisa memproses data matematis.
- b. *Machine & deep learning*: mesin "belajar" dari kumpulan data raksasa untuk mengenali konteks, maksud, dan sentimen (emosi) di balik teks tersebut.
- c. *Output generation*: komputer kemudian menghasilkan respons yang relevan, baik berupa klasifikasi (seperti membedakan email spam), terjemahan, atau jawaban teks seperti pada ChatGPT [28].

Melalui kombinasi statistik dan model jaringan saraf (*neural networks*), *NLP modern* tidak hanya menghafal kata, tetapi juga memahami nuansa bahasa manusia. Di mana algoritma NLP adalah serangkaian metode yang dirancang agar komputer dapat memproses, menganalisis, dan memahami bahasa manusia. Algoritma ini telah berevolusi dari sistem berbasis aturan sederhana menjadi model kecerdasan buatan yang sangat kompleks. Berikut adalah tiga kategori utama algoritma NLP:

1. Algoritma berbasis aturan (*rule-based*)

Ini adalah pendekatan paling awal yang menggunakan aturan linguistik yang dibuat secara manual oleh ahli bahasa.

Cara kerja menggunakan logika "jika-maka" (*if-then*) dan kamus kata.

Kelebihan/kekurangan adalah sangat konsisten untuk tugas sederhana, tetapi sulit menangani variasi bahasa manusia yang fleksibel dan ambigu.

2. Algoritma statistik (*statistical NLP*)

Populer pada era 1980-an hingga 2010-an, algoritma ini menggunakan probabilitas untuk menebak makna teks berdasarkan data besar (*corpora*). Contoh Algoritma:

- a. N-grams: memprediksi kata berikutnya berdasarkan urutan kata sebelumnya.
- b. Hidden markov models: sering digunakan untuk pengenalan suara dan pelabelan bagian kalimat.

Kelebihan: Lebih fleksibel dibanding aturan manual karena "belajar" dari pola data.

3. Algoritma *deep learning & transformer* (*modern*)

Teknologi dibalik AI canggih saat ini seperti ChatGPT. *Recurrent Neural Networks* (RNN) & *Long Short-Term Memory* (LSTM) yaitu algoritma yang memproses teks secara berurutan, mirip seperti manusia membaca dari kiri ke kanan.

- a. *Transformer*: arsitektur revolusioner yang menggunakan mekanisme *self-attention*. Berbeda dengan RNN, *transformer* dapat memproses seluruh kalimat sekaligus secara paralel, sehingga jauh lebih cepat dan memahami konteks kata yang berjauhan dengan sangat baik.
- b. Contoh Model: BERT (oleh Google) dan GPT (oleh OpenAI) [29].

2.1.7 Data Preprocessing

Data preprocessing adalah proses mengubah data mentah ke dalam bentuk yang lebih mudah dipahami. Proses ini diperlukan untuk memperbaiki kesalahan pada data mentah yang seringkali tidak lengkap dan memiliki *format* yang tidak teratur. Tujuan dari data *preprocessing* yaitu untuk mentransformasikan data mentah untuk dianalisis agar dapat menghasilkan data yang berkualitas dan akurat [30]. *Preprocessing* melibatkan proses validasi dan imputasi data. Validasi bertujuan untuk menilai tingkat kelengkapan dan akurasi data yang tersaring. Sedangkan imputasi bertujuan memperbaiki kesalahan dan memasukkan nilai yang hilang, baik secara manual atau otomatis melalui program business process automation (BPA). Kualitas data memang berdampak langsung terhadap keberhasilan setiap proyek yang melibatkan analisis data. Dalam *machine learning*, data *preprocessing* berperan memastikan bahwa *big data* sudah disesuaikan dan informasi di dalamnya dapat dipahami oleh algoritma perusahaan sehingga bisa mengeluarkan hasil yang lebih akurat [31].

Standar literatur keilmuan NLP tahapan prapemrosesan teks secara berurutan meliputi:

1. Pembersihan derau (*text cleansing*)

Langkah awal untuk menghapus elemen-elemen teks yang tidak memiliki representasi makna semantik atau nilai linguistik. Proses ini mencakup penghapusan tanda baca, karakter khusus (*special characters*), angka, tag HTML, tautan *web* (URL), hingga emoticon atau emoji. Tujuannya adalah untuk mengurangi dimensi ruang fitur agar algoritma hanya berfokus pada alfabet yang membentuk bahasa.

2. Penyeragaman huruf (*case folding*)

Proses konversi atau pengubahan seluruh karakter dalam dokumen teks menjadi format huruf kecil (*lowercase*). Tahapan ini bertujuan untuk menyamakan kedudukan kata terlepas dari posisinya di dalam kalimat. Sebagai contoh, kata "Aplikasi" di awal kalimat dan "aplikasi" di tengah kalimat akan direpresentasikan sebagai satu entitas token yang sama oleh sistem.

3. Tokenisasi (*tokenization*)

Merupakan tugas memecah urutan karakter berupa dokumen, paragraf, atau kalimat menjadi unit-unit leksikal yang lebih kecil dan terisolasi, yang disebut token. Token biasanya merepresentasikan kata (*word tokenization*) atau sub-kata (*subword tokenization*). Model bahasa beroperasi dengan menganalisis susunan dan relasi antar token tersebut.

4. Normalisasi ejaan (*normalization*)

Normalisasi adalah proses standarisasi leksikal untuk mengubah kata-kata yang tidak baku, bahasa gaul (*slang*), singkatan, atau kesalahan pengetikan (*typo*) menjadi bentuk baku atau kata standar. Dalam kasus analisis teks berbahasa Indonesia, tahap ini sering kali mengandalkan pemetaan kamus (*custom dictionary mapping*) guna memastikan konsistensi makna (misalnya mengubah "bgt" dan "bngt" menjadi "banget").

5. Penghapusan kata hubung (*stopword removal*)

6. Langkah untuk membuang kata-kata umum yang memiliki frekuensi kemunculan sangat tinggi namun dianggap tidak membawa informasi krusial mengenai sentimen atau konteks utama kalimat. Contoh kata yang sering dihapus antara lain preposisi ("di", "ke", "dari") dan konjungsi ("dan", "atau", "yang").

7. Pemotongan imbuhan (*stemming*) dan lematisasi (*lemmatization*)

Stemming adalah proses heuristik yang memotong imbuhan (awalan, sisipan, akhiran) dari sebuah kata untuk mengembalikannya ke bentuk kata dasar (misalnya "berjalan" menjadi "jalan"). Sementara Lematisasi adalah proses analisis morfologis yang lebih canggih untuk mengembalikan kata ke bentuk kamus aslinya dengan memastikan bahwa bentuk dasar tersebut merupakan kata yang valid secara tata bahasa [32].

2.1.8 Arsitektur *Transformer*

Arsitektur *transformer* adalah jenis jaringan saraf dalam (*deep learning*) yang dirancang untuk memproses data sekuensial (urutan) secara paralel, menggunakan mekanisme *self-attention* untuk menangkap konteks dan hubungan antar elemen. Diperkenalkan tahun 2017 dalam makalah "*Attention Is All You Need*", arsitektur ini menggantikan RNN/LSTM, memungkinkan pelatihan lebih cepat dan akurasi tinggi pada NLP dan visi komputer [33].

Komponen utama arsitektur *transformer*:

1. *Self-attention mechanism*

Mengizinkan model mengevaluasi relevansi setiap elemen dalam urutan terhadap elemen lainnya secara simultan, memahami konteks yang lebih dalam.

2. *Encoder-decoder structure*

Terdiri dari *encoder* (memproses *input*) dan *decoder* (menghasilkan *output*). Banyak varian *modern*, seperti GPT, hanya menggunakan bagian *decoder*, sementara BERT menggunakan *encoder*.

3. *Positional encoding*

Karena *transformer* memproses data secara paralel (bukan berurutan), komponen ini menambahkan informasi mengenai posisi setiap kata/token dalam urutan.

4. *Multi-head attention*

Memungkinkan model fokus pada berbagai aspek hubungan antar kata secara bersamaan.

5. *Feed-forward networks & residual connections*

Lapisan untuk pemrosesan lebih dalam dan stabilitas pelatihan [34].

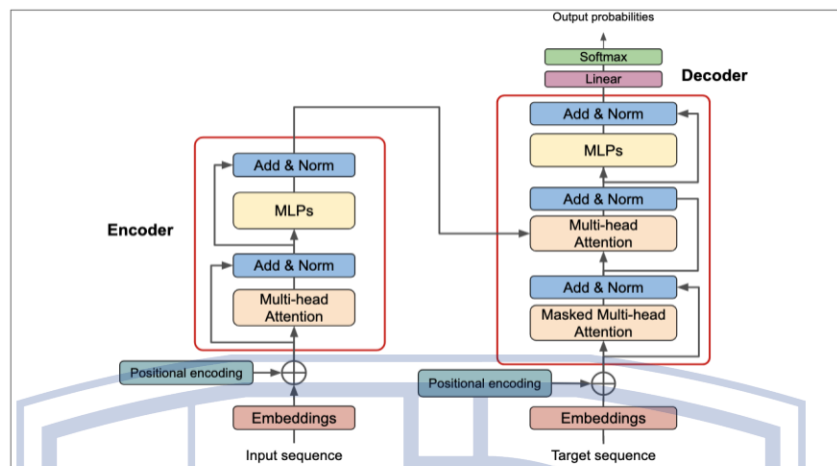
Transformer menjadi dasar bagi model bahasa besar (LLM) *modern* seperti GPT-4, Claude, Llama, dan Gemini. Kegunaannya meliputi:

1. NLP: terjemahan mesin, ringkasan teks, jawaban pertanyaan.
2. Visi komputer: *vision transformer* untuk klasifikasi gambar.
3. *Multimodal*: pemrosesan gabungan teks dan gambar [35].

Arsitektur *transformer* terdiri dari:

1. *Encoder*: bertanggung jawab untuk membaca dan memahami teks masukan.
2. *Decoder*: bertanggung jawab untuk menghasilkan teks keluaran berdasarkan informasi dari *encoder*.
3. *Self-Attention*: mekanisme yang memungkinkan model untuk fokus pada kata-kata yang relevan dalam satu kalimat.
4. *Positional encoding*: digunakan untuk memberi tahu model posisi atau urutan setiap kata dalam kalimat [35].

Diagram arsitektur *transformer* dapat dilihat pada Gambar 2.4.



Gambar 2.4 Diagram Arsitektur *Transformer* [36]

Berikut adalah rincian komponen yang ada dalam diagram tersebut:

1. *Block encoder* (pemroses input)

Encoder bertugas mengubah teks masukan menjadi representasi numerik yang kaya akan konteks.

- Input embedding & positional encoding*, kata diubah menjadi vektor, lalu ditambahkan informasi posisi karena *transformer* memproses semua kata secara paralel (tidak berurutan).
- Multi-head attention*: inti dari *transformer* yang memungkinkan model "melihat" hubungan antara setiap kata dalam kalimat tanpa mempedulikan jaraknya.
- Feedforward*: jaringan saraf standar yang memproses hasil dari mekanisme attention.

2. *Block decoder* (penghasil output)

Decoder menggunakan informasi dari *encoder* untuk menghasilkan teks keluaran satu per satu secara berurutan.

- Masked multi-head attention*: versi khusus yang mencegah model melihat kata-kata "masa depan" saat sedang berlatih memprediksi kata berikutnya.
- Encoder-decoder attention*: lapisan yang membantu *decoder* fokus pada bagian penting dari teks masukan yang diproses oleh *encoder*.
- Linear & softmax*: tahap akhir yang mengubah hasil perhitungan menjadi probabilitas kata untuk memilih kata yang paling cocok sebagai keluaran.

3. Komponen tambahan

- Add & norm (residual connections)*: garis yang melompati blok-blok utama untuk membantu aliran informasi dan menstabilkan pelatihan model yang sangat dalam.

- b. Nx: menandakan bahwa blok *encoder* dan *decoder* ini biasanya ditumpuk berkali-kali (misalnya 6 lapis pada model asli) untuk meningkatkan kemampuan pemahaman [36].

2.1.9 Model IndoBERT

Bidirectional Encoder Representations from Transformers (BERT) adalah model bahasa canggih yang dibangun menggunakan arsitektur *transformer*. BERT sebuah model yang dirancang untuk memahami arti dan hubungan kata-kata dalam suatu teks secara bidireksional dengan mempertimbangkan konteks kiri dan kanan pada semua lapisan. Tidak hanya bergantung pada satu arah, model BERT dapat mempelajari konteks kata dengan menganalisis kata-kata di sekitarnya. Fondasi dari pendekatan BERT merujuk pada teknologi *transformer*, yang terfokus pada pembelajaran relasi kontekstual antar kata dalam teks [37].

IndoBERT dibangun menggunakan kumpulan data berukuran besar yang disebut Indo4B. Indo4B adalah sebuah *dataset* atau korpus teks berskala besar dalam bahasa Indonesia yang digunakan untuk melatih kecerdasan buatan (AI). *Dataset* ini berisi sekitar 4 miliar kata dan 250 juta kalimat dari berbagai sumber. Data dalam Indo4B terdiri dari kalimat-kalimat dalam bahasa Indonesia, termasuk yang bersifat formal maupun sehari-hari. Data ini dikumpulkan dari 12 sumber berbeda, di mana dua di antaranya berfokus pada bahasa Indonesia sehari-hari, delapan berfokus pada bahasa Indonesia formal, dan sisanya menggunakan gaya campuran antara bahasa sehari-hari dan formal [37].

Berikut adalah poin-poin penting mengenai IndoBERT terdiri dari:

1. Fungsi utama: IndoBERT digunakan untuk berbagai tugas *Natural Language Processing* (NLP) atau pemrosesan bahasa alami, seperti analisis sentimen (positif/negatif), klasifikasi teks/berita, ekstraksi informasi, dan peringkasan teks.
2. Keunggulan kontekstual: berbeda dengan model NLP tradisional, IndoBERT memahami makna kata berdasarkan konteks kalimat secara menyeluruh (dua arah/*bidirectional*), baik dari kiri ke kanan maupun sebaliknya.
3. Performa unggul: penelitian menunjukkan bahwa IndoBERT secara konstan mengungguli model Multilingual BERT (M-BERT) dalam berbagai tugas NLP berbasis bahasa Indonesia karena lebih spesifik mengenali struktur bahasa lokal.
4. Teknologi: model ini menggunakan arsitektur *transformer* yang efisien untuk *transfer learning*, memungkinkan pengembang AI untuk melakukan *fine-tuning* pada tugas-tugas spesifik dengan data yang lebih sedikit [38].

Terobosan paling revolusioner dari BERT adalah penerapan paradigma *transfer learning* (pembelajaran pindahan). Konsep ini diadaptasi dari cara manusia belajar, di mana mesin tidak perlu belajar bahasa dari nol setiap kali membaca dokumen baru, melainkan menggunakan pengetahuan bahasa dasar yang sudah dimiliki sebelumnya. Pembelajaran pindahan ini dibagi menjadi dua fase utama yaitu:

1. Fase pertama adalah *pre-training*, di mana model BERT dilatih dengan membaca miliaran kata teks umum selama berbulan-bulan menggunakan superkomputer untuk menyerap tata bahasa dan logika secara mendalam.
2. Fase kedua adalah *fine-tuning*, di mana model yang sudah terlatih tersebut dilatih ulang menggunakan *dataset* yang jauh lebih kecil dan spesifik guna menyesuaikan pengetahuannya untuk mengenali pola pelaporan *bug* atau saran fitur secara spesifik [39].

Model BERT dapat memahami dan menganalisis teks dengan mempertimbangkan struktur, konteks, dan hubungan antar kata-kata serta kalimat. Hal ini dikarenakan BERT menggunakan tiga komponen penting yang menjadi *input* dari model BERT terdiri dari:

1. *Token embedding*

Token embedding adalah representasi dari kata-kata dalam teks. *Token embedding* berfungsi untuk mengubah teks masukan menjadi token-token terpisah dan memberikan identifikasi unik (ID) untuk setiap token. Setelah ID ditetapkan, setiap token akan diubah menjadi representasi khusus untuk token tersebut. Langkah berikutnya adalah memasukkan token [CLS] ke setiap *input* dan Token [SEP] digunakan sebagai penanda akhir atau pemisah antara segmen kalimat. Token [CLS] bertujuan sebagai penanda awal dari teks dan [SEP] bertindak sebagai penanda pembatas antara teks yang diberikan.

2. *Segment embedding*

Segment embedding adalah bagian komponen yang ditambahkan pada token *input* yang bertujuan untuk memungkinkan pengidentifikasian antara kalimat-kalimat dan menetapkan urutan relatif di antara mereka. Dengan *segment embedding*, model dapat membedakan antara bagian-bagian dari teks yang berasal dari kalimat yang berbeda, membantu dalam memahami konteks dan hubungan antara kalimat-kalimat tersebut.

3. *Position embedding*

Pada tahap *position embedding*, masing-masing token *input* diberikan lapisan tambahan yang memungkinkan penentuan posisinya dalam urutan. *Position embedding* penting karena tanpa pengetahuan tentang posisi relatif dari setiap token dalam urutan, model tidak akan dapat memahami hubungan antara kata-kata dalam sebuah frasa atau kalimat.

Position embedding memungkinkan model untuk membedakan antara kata-kata yang memiliki makna berbeda dalam konteks yang sama [40].

Secara teknis, IndoBERT bekerja dengan cara "belajar" dari jutaan kalimat bahasa Indonesia untuk memahami pola, tata bahasa, dan hubungan antar kata. Berikut adalah penjelasan cara kerjanya yang dibagi menjadi dua tahap utama:

1. Tahap Pra-pelatihan (*Pre-training*)

Pada tahap ini, IndoBERT dilatih menggunakan *dataset* masif (seperti Indo4B yang berisi miliaran kata dari berita, media sosial, dan Wikipedia). IndoBERT menggunakan dua metode belajar utama:

- a. *Masked Language Modeling* (MLM): Bayangkan sebuah kalimat yang satu katanya ditutup (dihapus). IndoBERT bertugas menebak kata tersebut berdasarkan kata-kata di kiri dan kanannya.
Contoh: "Ibu sedang memasak [MASK] di dapur." IndoBERT belajar bahwa kata yang paling mungkin mengisi [MASK] adalah "nasi" atau "sayur", bukan "sepatu".
- b. *Next Sentence Prediction* (NSP): IndoBERT belajar memahami hubungan antar kalimat. IndoBERT dilatih untuk menebak apakah kalimat B benar-benar mengikuti kalimat A atau hanya kalimat acak.

2. *Arsitektur bidirectional* (dua arah)

Ini adalah "senjata rahasia" IndoBERT. Tidak seperti model lama yang membaca teks dari kiri ke kanan saja, IndoBERT melihat seluruh kalimat secara bersamaan. Konteks: Dalam kalimat "Bank itu ada di pinggir sungai," IndoBERT tahu bahwa "Bank" di sini adalah lokasi geografis (tepi), bukan lembaga keuangan, karena ia melihat kata "sungai" di akhir kalimat secara serentak.

3. *Embedding* (representasi vektor)

Setelah belajar, IndoBERT mengubah setiap kata menjadi deretan angka rumit yang disebut Vektor. Kata-kata yang maknanya mirip (misal: "kereta" dan "sepur") akan memiliki posisi vektor yang berdekatan dalam ruang digitalnya. Hal ini memungkinkan komputer "merasakan" kemiripan makna, bukan sekadar mencocokkan huruf.

4. *Fine-tuning* (penyesuaian khusus)

Setelah IndoBERT "pintar" bahasa Indonesia secara umum, kita bisa melakukan *fine-tuning*. Jika diberikan data kecil yang sudah diberi label (misalnya 1.000 komentar YouTube yang sudah ditandai "positif" atau "negatif"). Karena sudah punya dasar bahasa yang kuat, IndoBERT hanya butuh sedikit waktu untuk belajar menjadi ahli analisis sentimen atau klasifikasi berita [40].

IndoBERT dipilih karena merupakan model bahasa yang dilatih menggunakan korpus berbahasa Indonesia dan mampu membentuk representasi kata berdasarkan konteks kalimat. Karakteristik tersebut membuat IndoBERT relevan untuk diuji dalam klasifikasi ulasan aplikasi berbahasa Indonesia yang mengandung variasi bahasa formal, informal, singkatan, dan kesalahan penulisan.

2.1.10 Uji Reliabilitas Antar Penilai (*Inter-Rater Reliability*) dengan *Cohen's Kappa*

Pengukuran reliabilitas instrumen yang melibatkan penilaian atau kategorisasi manusia memerlukan pendekatan statistik khusus yang dikenal sebagai uji kesepakatan antar-penilai (*Inter-Rater Reliability* / IRR). IRR merupakan metode evaluasi yang digunakan untuk mengukur sejauh mana dua atau lebih penilai (*raters*) independen memiliki kesepahaman yang konsisten dalam mengevaluasi, memberikan skor, atau mengklasifikasikan sekumpulan data yang sama. Tingkat kesepakatan yang tinggi menunjukkan bahwa pedoman kategorisasi bersifat objektif dan dapat direplikasi, sedangkan kesepakatan yang rendah mengindikasikan adanya bias subjektivitas.

Pada kasus pengujian instrumen di mana data yang dihasilkan berskala nominal (mengklasifikasikan objek ke dalam kategori yang saling lepas), metode statistik yang paling direkomendasikan adalah Cohen's Kappa (κ). Cohen's Kappa secara keilmuan jauh lebih tangguh (*robust*) dibandingkan dengan sekadar menghitung persentase persetujuan biasa (*simple percent agreement*). Keunggulan fundamental dari matriks ini terletak pada kemampuannya dalam mengoreksi probabilitas "kesepakatan kebetulan" (*chance agreement*). Artinya, algoritma ini memperhitungkan kemungkinan bahwa kedua penilai secara kebetulan memberikan label kategori yang sama hanya karena tebakan acak.

Secara matematis, koefisien *Cohen's Kappa* dihitung dengan membandingkan proporsi kesepakatan aktual dengan proporsi kesepakatan yang diharapkan secara kebetulan, menggunakan persamaan berikut:

$$K = \frac{Po - Pe}{1 - Pe} \dots\dots\dots (1)$$

Keterangan:

- a. K: Nilai koefisien *Kappa*
- b. Po: *Observed agreement* (proporsi kesepakatan aktual yang terjadi antara dua penilai).

c. *Pe: Expected agreement* (Proporsi kesepakatan yang diharapkan terjadi murni secara kebetulan) [41].

Rentang nilai yang dihasilkan oleh koefisien Kappa berada di antara -1 hingga 1. Nilai 1 merepresentasikan kesepakatan yang sempurna, sedangkan nilai Kappa sebesar 0 menunjukkan bahwa tingkat kesepakatan yang diamati tidak lebih tinggi daripada kesepakatan yang diharapkan terjadi secara kebetulan. Untuk menginterpretasikan tingkat keandalan pelabelan, salah satu pedoman interpretasi yang banyak digunakan di ranah akademis dapat dilihat pada Tabel 2.1

Tabel 2.1 Standar Interpretasi Nilai *Cohen's Kappa* [41]

Nilai Koefisien (Kappa)	Tingkat Kesepakatan (Level of Agreement)
< 0,00	Kesepakatan sangat buruk (Poor)
0,00 – 0,20	Kesepakatan ringan (Slight)
0,21 – 0,40	Kesepakatan cukup (Fair)
0,41 – 0,60	Kesepakatan sedang (Moderate)
0,61 – 0,80	Kesepakatan substansial (Substantial)
0,81 – 1,00	Kesepakatan hampir sempurna (Almost Perfect)

Penerapan Cohen's Kappa memberikan bukti mengenai tingkat konsistensi penerapan pedoman pelabelan oleh dua penilai independen pada sampel yang diuji. Dalam penelitian ini, nilai Kappa tidak digunakan untuk membuktikan bahwa setiap label pada seluruh dataset pasti benar, melainkan untuk menilai apakah pedoman anotasi dapat diterapkan secara konsisten pada sampel evaluasi. Oleh karena itu, interpretasi Kappa dibatasi pada reliabilitas pedoman dan tingkat kesepakatan antarpemilai pada sampel yang benar-benar diperiksa oleh kedua penilai.

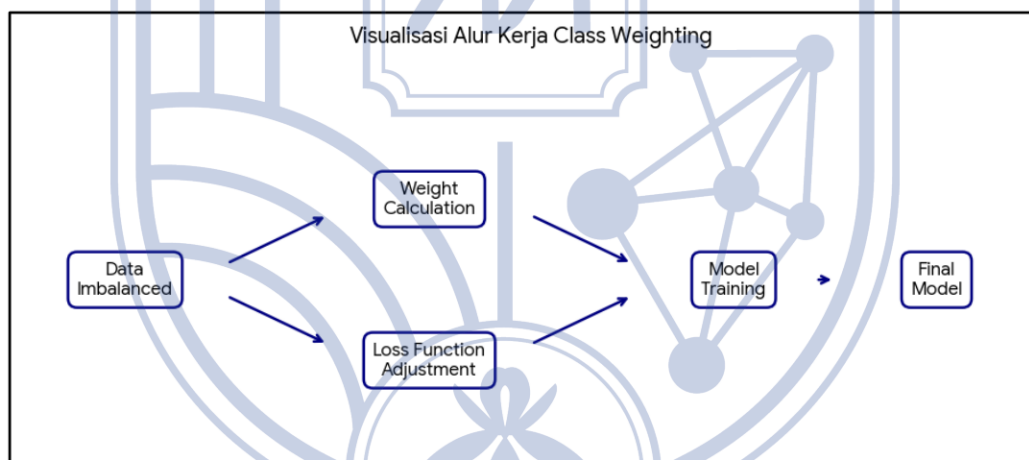
2.1.11 Penanganan Ketidakseimbangan Data dengan *Class Weighting*

Class weighting (pembobotan kelas) adalah teknik dalam *machine learning* yang digunakan untuk mengatasi masalah ketidakseimbangan data (*class imbalance*). Teknik ini bekerja dengan memberikan bobot yang lebih tinggi pada kelas minoritas (jumlah data sedikit) dan bobot lebih rendah pada kelas mayoritas (jumlah data banyak) selama proses pelatihan model [42].

Berikut adalah poin-poin penting mengenai *class weighting* terdiri dari:

1. Tujuan utama: mencegah model menjadi bias terhadap kelas mayoritas dan meningkatkan kemampuan model untuk mendeteksi kelas minoritas secara lebih akurat.
2. Cara kerja: menyesuaikan fungsi *loss* (kerugian) pada algoritma pembelajaran. Kesalahan prediksi pada kelas minoritas akan dianggap "lebih mahal" (penalti lebih tinggi) dibandingkan kesalahan pada kelas mayoritas.
3. Implementasi: banyak digunakan pada algoritma seperti *Logistic Regression*, *Random Forest*, *SVM*, dan *Neural Networks*.
4. Otomatisasi: opsi *class weight 'balanced'* dapat secara otomatis menyesuaikan bobot secara berbanding terbalik dengan frekuensi kelas dalam data masukan [42].

Dengan menggunakan *class weighting*, model akan memberikan perhatian lebih pada contoh data yang jarang, sehingga menghasilkan prediksi yang lebih adil dan tangguh. Kerangka kerja *class weighting* bekerja pada tingkat algoritma untuk menyeimbangkan perhatian model tanpa mengubah jumlah data asli. Berikut adalah alur kerangka kerja *class weighting* dapat dilihat pada Gambar 2.5.



Gambar 2.5 Alur Kerja Kerangka *Class Weight* [43]

Penjelasan:

1. Analisis distribusi data: mengidentifikasi jumlah sampel pada kelas mayoritas dan minoritas.
2. Penghitungan bobot (*weight calculation*): Menentukan nilai bobot untuk setiap kelas. Penentuan nilai bobot (W_c) biasanya berbanding terbalik dengan frekuensi kemunculan kelas tersebut dalam *dataset*. Salah satu metode yang paling sering digunakan adalah pembobotan berdasarkan frekuensi invers (*inverse class frequency*), yang dihitung dengan rumus:

$$W_c = \frac{N}{C \times N_c} \dots\dots\dots (2)$$

Di mana:

W_c = Bobot (*weight*) untuk kelas ke- c

N adalah total seluruh sampel dalam *dataset*

C adalah jumlah kelas

N_c adalah jumlah sampel pada kelas c

Dengan formula ini, kelas dengan jumlah sampel sedikit memperoleh nilai W_c yang lebih besar. Dalam implementasi penelitian ini, N dan N_c dihitung hanya dari data *training* pada setiap pembagian 80:20, bukan dari seluruh 10.000 data dan bukan dari data *testing*. Bobot tersebut kemudian dimasukkan ke fungsi *Weighted CrossEntropyLoss*, sehingga *loss* dari observasi kelas minoritas dikalikan dengan faktor yang lebih besar dan memberikan kontribusi gradien yang lebih kuat ketika prediksi salah. Data *testing* tidak digunakan untuk menentukan bobot kelas maupun memperbarui parameter model.

3. Modifikasi fungsi kerugian (*loss function adjustment*) : mengintegrasikan bobot ke dalam fungsi biaya (*cost function*). kesalahan prediksi pada kelas minoritas akan dikalikan dengan bobot yang lebih besar ($W_{\text{minoritas}} > W_{\text{mayoritas}}$).
 4. Proses pelatihan model: model memperbarui parameter (seperti *weights* dan bias) dengan memprioritaskan minimalisasi kesalahan pada kelas yang berbobot tinggi.
- Evaluasi akhir: mengukur performa menggunakan metrik yang relevan untuk data tidak seimbang, seperti *F1-score* untuk memastikan kelas minoritas terdeteksi dengan baik [43].

Dalam kajian Pemrosesan Bahasa Alami (NLP), penanganan ketidakseimbangan kelas menuntut pendekatan metodologis yang spesifik. Pendekatan manipulasi tingkat data (*data-level approach*) tradisional seperti *undersampling* berisiko membuang variasi kosakata yang krusial bagi pemahaman konteks model. Sementara itu, teknik *oversampling* sintetis berbasis interpolasi jarak seperti SMOTE (*Synthetic Minority Over-sampling Technique*), meski terbukti unggul pada data numerik (*tabular*), rentan menghasilkan representasi teks yang tidak koheren secara semantik ketika diterapkan pada ruang vektor dinamis milik arsitektur *transformer*. Sebagai alternatif di ranah NLP, teknik augmentasi data teks seperti *Back-Translation* atau *Easy Data Augmentation* (EDA) sering digunakan.

Namun, pada konteks ulasan aplikasi yang sangat bising (*noisy*), penuh dengan singkatan, dan bahasa informal, modifikasi kata secara paksa berisiko merusak niat linguistik asli (*linguistic intent*) dari pengguna, selain menuntut beban komputasi server yang sangat tinggi. Berdasarkan kelemahan pendekatan manipulasi data tersebut, penelitian ini menjustifikasi penggunaan pendekatan tingkat algoritma (*algorithm-level approach*) melalui teknik pembobotan kelas (*class weighting*). Modifikasi bobot komputasional secara langsung pada fungsi kerugian (*loss function*) memungkinkan model untuk memberikan penalti kesalahan (*cost*) yang lebih besar terhadap kelas minoritas (seperti laporan *bug* dan saran) tanpa perlu mendistorsi integritas *dataset* asli. Pendekatan ini memastikan IndoBERT tetap dapat menyerap konteks semantik bahasa Indonesia secara natural, dengan efisiensi komputasi yang jauh lebih superior dibandingkan metode augmentasi tingkat data.

2.1.12 Uji Ablasi (*Ablation Study*)

Uji ablasi (*ablation study*) adalah metode eksperimen dengan menghapus, mengganti, atau mengubah komponen, fitur, maupun bagian arsitektur model secara sistematis untuk mengetahui kontribusi setiap bagian terhadap kinerja keseluruhan model. Dengan membandingkan model lengkap dan model yang telah “diablasi”, peneliti dapat menentukan komponen yang benar-benar memberikan peningkatan performa [44].

Dalam disiplin rekayasa perangkat lunak dan *deep learning* modern, uji ablasi (*ablation study*) didefinisikan sebagai metodologi eksperimental yang dirancang untuk mengevaluasi kinerja suatu arsitektur *machine learning* dengan cara menghilangkan, mengganti, atau menonaktifkan komponen spesifik dari model secara sistematis.

Dalam penelitian NLP dan *deep learning*, uji ablasi digunakan untuk mengisolasi kontribusi suatu komponen atau intervensi dengan membandingkan kondisi kontrol dan kondisi yang dimodifikasi sambil mempertahankan faktor lain tetap sama. Pada penelitian ini, istilah uji ablasi digunakan dalam bentuk *controlled comparison* antara IndoBERT *baseline* dan IndoBERT + *class weighting*. Arsitektur model, *tokenizer*, parameter *fine-tuning*, ukuran *train/test*, serta data *split* pada setiap *seed* dipertahankan sama; satu-satunya intervensi yang dibedakan adalah penggunaan bobot kelas pada fungsi kerugian.

Dengan desain tersebut, IndoBERT *baseline* berfungsi sebagai kontrol menggunakan *CrossEntropyLoss* standar, sedangkan IndoBERT + *class weighting* menjadi kondisi intervensi menggunakan *Weighted CrossEntropyLoss*. Perbedaan metrik di antara kedua skenario pada *split* yang sama digunakan untuk mengukur dampak empiris pembobotan

kelas. Desain ini tidak mengklaim ablasi terhadap seluruh komponen arsitektur IndoBERT; yang diuji secara terkontrol hanya komponen *class weighting* [45].

2.1.13 Evaluasi Kinerja Klasifikasi Teks

Penerapan klasifikasi kelas secara nyata pasti dihadapkan pada distribusi data yang tidak seimbang (*imbalanced data*). Di mana kelas minoritas yang jumlahnya tertutup oleh dominasi ulasan sentimen positif atau negatif. Pemberian kondisi ini akan mengakibatkan model mengalami bias dan cenderung memprediksi kelas mayoritas demi mendapatkan nilai akurasi yang tinggi secara artifisial, sambil mengabaikan kelas minoritas. Metode evaluasi kinerja yang komprehensif mutlak diperlukan [38].

Pengujian keandalan model IndoBERT dilakukan menggunakan instrumen *Confusion Matrix*. *Confusion Matrix* merupakan salah satu cara dalam melakukan evaluasi performa pada suatu masalah klasifikasi. *Confusion Matrix* meringkas kinerja dari klasifikasi yang berhubungan dengan data uji. *Confusion Matrix* terdiri dari matriks dua dimensi di mana *indeks* pertama menyatakan kelas yang sebenarnya dari suatu objek atau kelas aktual dan *indeks* lainnya menyatakan kelas yang ditugaskan oleh pengklasifikasi atau kelas prediksi. *Confusion Matrix* ini menghasilkan empat nilai dasar klasifikasi yang dapat dilihat pada Tabel 2.2.

Tabel 2.2 *Confusion Matrix* [39]

		Actual Class	
		Positive (1)	Negative (1)
	Predicted Class		
	Positive (1)	TP	FP
	Negative (1)	FN	TN

Keterangan:

1. *True Positive (TP)*: model memprediksi kelas target, dan tebakannya benar.
2. *True Negative (TN)*: model memprediksi bukan kelas target, dan tebakannya benar.
3. *False Positive (FP)*: model memprediksi kelas target, namun tebakannya salah (*false alarm*).
4. *False Negative (FN)*: model memprediksi bukan kelas target, namun tebakannya salah (Model gagal mendeteksi target) [39].

Kinerja model dievaluasi secara matematis melalui empat metrik utama:

1. Akurasi (*Accuracy*)

Akurasi adalah rasio keseluruhan tebakan yang benar terhadap total populasi data yang diuji. Metrik ini mengukur seberapa sering model menebak dengan benar secara umum.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \dots\dots\dots (3)$$

Accuracy dapat memberikan gambaran yang kurang representatif pada *dataset* tidak seimbang karena nilainya dapat didominasi oleh kinerja kelas mayoritas.

2. Presisi (*precision*)

Presisi mengukur tingkat ketepatan spesifik, yaitu rasio data yang diprediksi sebagai target positif yang memang benar-benar merupakan positif aktual.

$$Precision = \frac{TP}{TP + FP} \dots\dots\dots (4)$$

3. Sensitivitas (*Recall*)

Sensitivitas mengukur tingkat sensitivitas model, yaitu kemampuan algoritma dalam mendeteksi dan menemukan keseluruhan data positif yang sebenarnya ada di dalam populasi uji.

$$Recall = \frac{TP}{TP + FN} \dots\dots\dots (5)$$

4. *F1-score* adalah rata-rata harmonik *precision* dan *recall*. Pada klasifikasi multi-kelas, *F1-score* dapat dirata-ratakan dengan beberapa cara. *Macro F1-score* menghitung *F1-score* setiap kelas secara terpisah lalu mengambil rata-rata sederhana, sehingga setiap kelas memperoleh kontribusi evaluasi yang sama tanpa dipengaruhi besar-kecilnya *support* kelas. Karena distribusi kelas penelitian tidak seimbang, *Macro F1-score* ditetapkan sebagai metrik utama untuk menilai keseimbangan kinerja antarkelas. *Accuracy* tetap dilaporkan sebagai ukuran ketepatan keseluruhan, tetapi tidak digunakan sendirian untuk menentukan skenario terbaik karena dapat didominasi kelas mayoritas.

$$F1\text{-score} = \frac{Precision \times Recall}{Precision + Recall} \dots\dots\dots (6)$$

Justifikasi penggunaan optimasi metrik *F1-score* yang dipadukan dengan penerapan pembobotan kelas (*class weighting*) sangat kuat dalam penelitian ini, karena sistem dipaksa agar lebih sensitif mengenali masalah fungsional aplikasi terlepas dari dominasi opini positif [39].

2.1.14 Analisis Ketidaksesuaian Peringkat Bintang dan Narasi Ulasan

Peringkat bintang (*star rating*) pada *platform* distribusi aplikasi seperti Google Play Store merupakan metadata kuantitatif yang diberikan pengguna untuk menilai kualitas aplikasi secara ringkas. Peringkat ini umumnya berada pada skala 1 sampai 5, di mana nilai rendah cenderung menunjukkan ketidakpuasan, sedangkan nilai tinggi cenderung menunjukkan kepuasan pengguna. Meskipun demikian, peringkat bintang tidak selalu mampu merepresentasikan isi semantik narasi ulasan secara utuh, karena pengguna dapat menyampaikan keluhan teknis, laporan *bug*, saran fitur, maupun pengalaman penggunaan yang lebih kompleks melalui teks ulasan [46].

Dalam literatur analisis ulasan daring, fenomena ketidaksesuaian antara teks ulasan dan *rating* numerik dikenal dengan istilah *Text-Rating Review Discrepancy* (TRRD). TRRD menggambarkan kondisi ketika isi tekstual ulasan tidak sepenuhnya selaras dengan *rating* yang diberikan pengguna. Fenomena ini penting diperhatikan karena sebagian penelitian terdahulu cenderung mengasumsikan bahwa *rating* dan teks ulasan memiliki hubungan yang sejalan, padahal konsistensi antara keduanya tidak selalu terjadi.

Dalam konteks ulasan aplikasi, ketidaksesuaian ini dapat muncul ketika pengguna memberikan *rating* tinggi tetapi tetap menuliskan keluhan mengenai *bug*, kegagalan fitur, performa aplikasi, atau kendala penggunaan tertentu. Sebaliknya, pengguna juga dapat memberikan *rating* rendah meskipun isi ulasannya lebih bersifat masukan atau saran pengembangan. Kondisi tersebut menunjukkan bahwa *rating* bintang tidak cukup digunakan sebagai indikator tunggal untuk memahami persepsi dan kebutuhan pengguna aplikasi [47].

Analisis ketidaksesuaian peringkat bintang dan narasi ulasan bertujuan untuk mengidentifikasi pola anomali antara metadata kuantitatif dan makna semantik teks. Dalam penelitian ini, analisis tersebut digunakan untuk melihat apakah hasil klasifikasi teks ulasan, memiliki keselarasan atau ketidaksesuaian dengan peringkat bintang aktual yang diberikan pengguna. Dengan demikian, analisis ini dapat membantu pengembang dalam menemukan

masalah teknis atau kebutuhan fitur yang berpotensi tersembunyi di balik *rating* bintang yang tinggi.

Pada *Natural Language Processing* (NLP) dan klasifikasi teks, teknik tabulasi memiliki peran yang melampaui metrik evaluasi komputasi tradisional. Jika evaluasi performa model (*machine learning evaluation*) seperti *Confusion Matrix* berfokus pada perbandingan antara nilai prediksi mesin (model IndoBERT) dengan nilai aktual (label kebenaran dasar/ *ground truth*) guna mencari akurasi, tabulasi silang digunakan untuk keperluan analisis perilaku (*behavioral analysis*). Dalam konteks penambangan opini pengguna, matriks tabulasi silang memetakan hasil akhir klasifikasi algoritma dengan metrik penilaian pengguna lainnya (seperti peringkat bintang). Pendekatan ini secara teoritis berfungsi untuk mengekstraksi wawasan operasional (*actionable insights*) dan mendeteksi distribusi respons pengguna terhadap suatu entitas atau layanan. Dengan memanfaatkan teknik tabulasi silang, varians anomali tersebut dapat dihitung proporsinya secara matematis. Analisis kuantitatif dari matriks silang ini memberikan validasi empiris bagi pengembang untuk tidak hanya bergantung pada agregasi rata-rata peringkat bintang, melainkan memahami konteks permasalahan teknis spesifik yang bersembunyi di balik distribusi metrik tersebut.

Sebagai ilustrasi konseptual, tabulasi silang dapat digunakan untuk memetakan distribusi kelas sentimen terhadap peringkat bintang pengguna. Dalam ilustrasi sederhana, kelas sentimen dibagi menjadi positif, netral, dan negatif, sedangkan peringkat bintang terdiri atas bintang 1 sampai dengan bintang 5. Bentuk matriks tabulasi silang tersebut disajikan pada Tabel 2.3.

Tabel 2.3 Matriks Tabulasi Silang Kelas Ulasan dan Peringkat Bintang

Kelas Sentimen	Bintang 1	Bintang 2	Bintang 3	Bintang 4	Bintang 5	Total Data
L1: Positif	L1 B1	L1 B2	L1 B3	L1 B4	L1 B5	Total L1
L2: Netral	L2 B1	L2 B2	L2 B3	L2 B4	L2 B5	Total L2
L3: Negatif	L3 B1	L3 B2	L3 B3	L3 B4	L3 B5	Total L3

Keterangan:

- Li : Kelas Label ke- i berdasarkan prediksi model IndoBERT dengan $i \in \{1, 2, 3\}$
- Bj : Peringkat Bintang ke- j yang diberikan oleh pengguna dengan $j \in \{1, 2, 3, 4, 5\}$
- $LiBj$: Merupakan variabel yang merepresentasikan frekuensi (jumlah) ulasan yang diklasifikasikan masuk ke dalam Label i dan memiliki peringkat Bintang j
- Total Li : Total kumulatif data ulasan untuk masing-masing kelas Label

Berdasarkan matriks 3 label di atas, anomali terjadi ketika label prediksi teks ulasan bertolak belakang dengan metrik peringkat bintang. Berikut adalah formulasi matematis untuk menghitung persentase anomali pada tiap kelas sentimen:

1. Rasio anomali sentimen positif (pujian dengan *rating* rendah) secara normatif, ulasan berlabel positif selaras dengan peringkat bintang tinggi. Anomali terjadi apabila teks ulasan terdeteksi positif (pujian, kepuasan), namun pengguna memberikan peringkat sangat rendah (Bintang 1 atau 2). Hal ini dapat mengindikasikan kekeliruan *input* pengguna atau gaya bahasa sarkasme.

$$Rasio_Anomali_Positif = \left(\frac{LiB1 + LiB2}{TotalLi} \right) \times 100\% \dots\dots\dots (7)$$

2. Rasio anomali sentimen negatif (keluhan dengan *rating* tinggi) ulasan berlabel negatif umumnya berasosiasi dengan ketidakpuasan dan peringkat bintang rendah. Anomali dicatat apabila teks berisi sentimen negatif (keluhan, kekecewaan), namun metrik peringkat menunjukkan nilai tinggi (Bintang 4 atau 5).

$$Rasio_Anomali_Positif = \left(\frac{LiB1 + LiB2}{TotalLi} \right) \times 100\% \dots\dots\dots (8)$$

3. Rasio anomali sentimen netral (peringkat bintang ekstrem) Label sentimen netral merepresentasikan teks yang objektif, informatif, atau ketiadaan emosi yang kuat, sehingga metrik yang paling ideal adalah Bintang 3. Anomali pada kelas ini dihitung dari ulasan netral yang justru mendapatkan penilaian ekstrem, baik ekstrem rendah (Bintang 1 dan 2) maupun ekstrem tinggi (Bintang 4 dan 5).

$$Rasio_Anomali_Netral = \left(\frac{L2B1 + L2B2 + L2B4 + L2B5}{TotalL2} \right) \times 100\% \quad \dots\dots\dots (9)$$

Uji Kruskal-Wallis diperkenalkan oleh W.H. Kruskal dan Wallis pada tahun 1952, yang merupakan pengujian dengan kategori lebih dari dua kelompok sampel yang saling bebas. Pengujian yang dapat digunakan pada analisis perbandingan untuk menguji lebih dari dua kelompok sampel yang saling bebas. Asumsi yang digunakan untuk menerapkan analisis Kruskal-Wallis antara lain sebagai berikut:

1. Data merupakan sampel acak.
2. Populasi (sampel) tidak mengikuti suatu berdistribusi tertentu.
3. Jumlah sampel tidak sama besar
4. Jenis skala data skala ordinal
5. Masing – masing sampel tidak saling berkaitan/mempengaruhi
6. Variabel acak berlanjut [48].

Untuk menguji apakah terdapat perbedaan distribusi peringkat bintang antarkelas ulasan, penelitian ini menggunakan Uji Kruskal-Wallis H. Uji ini mengurutkan seluruh *rating* bintang dari semua ulasan, kemudian membandingkan rata-rata peringkat (*mean rank*) antarkelas ulasan. Rumus Kruskal-Wallis adalah:

$$H = \frac{12}{N(N+1)} \sum_{i=1}^k \frac{R_i^2}{n_i} - 3(N+1) \quad \dots\dots\dots (10)$$

Keterangan:

H = Nilai statistik Kruskal-Wallis.

N = Total seluruh sampel ulasan Picsart.

k = Jumlah kelompok kelas (dalam hal ini k = 5).

n_i = Jumlah ulasan pada kelas ke-i.

R_i = Jumlah total peringkat (rangking) bintang pada kelas ke-i

Jika nilai $p\text{-value} < 0,05$, maka dapat disimpulkan bahwa terdapat perbedaan distribusi peringkat bintang yang signifikan antarkelas ulasan (*bug*, saran, negatif, positif, netral).

Penerapan analisis untuk menguji anomali antara metrik peringkat bintang (*star rating*) dengan kelas sentimen umpan balik memerlukan pendekatan statistik yang bertahap,

dimulai dari pemetaan deskriptif hingga pembuktian inferensial. Sebagai langkah awal deskriptif, penelitian ini menggunakan Matriks Tabulasi Silang (*Cross-Tabulation*). Pemilihan metode ini didasarkan pada kebutuhan teknis untuk memetakan koordinat distribusi frekuensi secara granular. Matriks tabulasi silang mampu menyajikan interaksi antar-variabel secara eksplisit pada setiap kombinasi sel. Pendekatan ini sangat krusial untuk melokalisasi dan mendeteksi fenomena anomali metrik secara konkret, seperti mengidentifikasi jumlah ulasan berkategori laporan *bug* dan saran fitur yang secara kontradiktif justru diberikan peringkat bintang lima oleh pengguna. Setelah anomali tersebut terpetakan secara deskriptif, tahap selanjutnya adalah melakukan pengujian inferensial untuk menguji apakah terdapat perbedaan distribusi. Peringkat bintang pada *platform* distribusi aplikasi dikategorikan sebagai data berskala ordinal, yang memiliki urutan jenjang namun tanpa jarak interval matematis yang pasti. Mengingat distribusi ulasan empiris umumnya bersifat asimetris dan tidak berdistribusi normal, penggunaan uji statistik parametrik seperti ANOVA (*Analysis of Variance*) menjadi tidak valid. Oleh karena itu, sebagai justifikasi metodologis, penelitian ini mengadopsi Uji Kruskal-Wallis. Uji non-parametrik ini dipilih secara spesifik karena ketangguhannya dalam mengevaluasi perbedaan median dan distribusi peringkat dari lebih dari dua kelompok independen (dalam hal ini, lima kelas ulasan) terhadap variabel terikat berskala ordinal. Pengujian ini memberikan landasan konfirmasi matematis yang solid bahwa perbedaan peringkat bintang antar-kategori kebutuhan perangkat lunak tersebut benar-benar signifikan secara statistik, bukan sekadar variasi kebetulan.

2.2 Penelitian Terdahulu (*State-of-the-Art*)

Tinjauan terhadap penelitian terdahulu bertujuan untuk memetakan pengetahuan yang telah tersedia, keterbatasan yang masih tersisa, dan posisi penelitian ini terhadap celah tersebut. Perbandingan difokuskan pada empat aspek yang berkaitan langsung dengan rancangan tesis, yaitu cakupan kelas target, penggunaan IndoBERT atau model *transformer*, penanganan *class imbalance*, serta analisis hubungan antara isi tekstual ulasan dan peringkat bintang. Perbandingan penelitian terdahulu disajikan pada Tabel 2.4.

Tabel 2.4 Perbandingan Penelitian Terdahulu (*State-of-the-Art*)

No	Peneliti & Tahun	Judul Penelitian	Algoritma / Metode	Hasil Penelitian	Keterbatasan terhadap Fokus Penelitian Ini
1	Daqiqil ID dkk., 2024	<i>Sentiment analysis of student evaluation feedback using transformer-based language models [2]</i>	<i>mBERT</i> , <i>IndoBERT</i> , <i>RoBERTa</i> Indonesia, <i>GPT-2</i> Indonesia, <i>Support Vector Machine (SVM)</i> , dan <i>Naive Bayes (NB)</i>	Penggunaan model bahasa berbasis <i>transformer</i> sangat efektif untuk analisis sentimen evaluasi mahasiswa karena kemampuannya memahami konteks bahasa Indonesia yang kompleks. <i>IndoBERT</i> adalah model yang paling direkomendasikan untuk penelitian ini	a. Penelitian ini hanya melabeli data ke dalam dua kelas sentimen (biner), yaitu positif dan negatif b. Belum menggunakan sistem <i>class weight</i> untuk menangani data yang tidak seimbang (<i>imbalanced data</i>). c. Tidak ditemukan analisis ketidaksesuaian antara hasil prediksi teks dengan peringkat bintang aktual.
2	Samudera dkk., 2024	<i>Sentiment Analysis of User Reviews on BSI Mobile and Action Mobile Applications on the Google Play Store Using Multinomial Naive Bayes Algorithm [3]</i>	<i>Multinomial Naive Bayes</i>	Algoritma <i>Multinomial Naive Bayes</i> terbukti sangat efektif untuk mengklasifikasi sentimen ulasan pengguna pada kedua aplikasi <i>mobile banking</i> tersebut, dibuktikan dengan nilai evaluasi (akurasi, presisi, <i>recall</i> , dan <i>F1-score</i>) yang semuanya berada di atas 70%.	a. Penelitian ini hanya melabeli data ke dalam dua kelas sentimen (biner), yaitu positif dan negatif b. Belum menggunakan sistem <i>class weight</i> untuk menangani data yang tidak seimbang (<i>imbalanced data</i>). c. Belum melakukan analisis ketidaksesuaian antara teks ulasan dan peringkat bintang, meskipun data penelitian bersumber dari Google Play Store. d. Analisis sentimen tidak ditujukan untuk proses pemeliharaan perangkat lunak selanjutnya.

3	Mursidah dkk., 2025	Klasifikasi sentimen Google Play Store aplikasi <i>ChatGPT</i> berbahasa Indonesia berbasis IndoBERT [4]	IndoBERT	Membuktikan bahwa IndoBERT mampu memproses ulasan pada ekosistem aplikasi generatif AI secara spesifik.	a. Sentimen hanya label positif, negatif, netral. b. Belum menggunakan sistem <i>class weight</i> untuk menangani data yang tidak seimbang (<i>imbalanced data</i>). c. Belum melakukan analisis ketidaksesuaian antara hasil prediksi teks dengan peringkat bintang aktual padahal data diambil dari Google Play Store. d. Analisis sentimen tidak ditujukan untuk proses pemeliharaan perangkat lunak selanjutnya.
4	Supriyadi dan Makatita, 2025	<i>Sentiment Analysis of TikTok User Comments on QRIS Adoption in Indonesia Using IndoBERT</i> [5]	IndoBERT	Model terbukti sangat stabil dan mampu membaca konteks bahasa tidak baku (<i>slang</i>) yang sangat umum digunakan di TikTok, membuktikan ketangguhan IndoBERT di ranah media sosial.	a. Sentimen hanya label positif, negatif, netral. b. Belum menggunakan sistem <i>class weight</i> untuk menangani data yang tidak seimbang (<i>imbalanced data</i>). c. Penelitian menggunakan komentar TikTok yang tidak memiliki metadata peringkat bintang, sehingga tidak melakukan analisis ketidaksesuaian antara teks dan <i>rating</i>. d. Analisis sentimen tidak ditujukan untuk proses pemeliharaan perangkat lunak selanjutnya

5	William dkk., 2024	<i>Emotion Recognition Indonesian Language from Twitter using IndoBERT and BI-LSTM [6]</i>	IndoBERT vs Bi-LSTM	Klasifikasi sentimen label emosi seperti senang, sedih, marah, terkejut, jijik, takut, netral. Eksperimen empiris ini membuktikan keunggulan arsitektur <i>transformer</i> secara telak. IndoBERT (akurasi 92,3%) terbukti jauh lebih superior dalam mengenali teks bahasa Indonesia dibandingkan dengan model sekuensial mutakhir Bi-LSTM (84,0%).	a. Sentimen difokuskan pada label emosi. Jadi bukan label positif, negatif, laporan <i>bug</i> atau saran fitur. b. Belum menggunakan sistem <i>class weight</i> jika ada data yang tidak seimbang (<i>imbalanced data</i>). c. Karena sumber dari Twitter maka analisis ketidaksesuaian antara hasil prediksi teks dengan peringkat bintang otomatis tidak ada. d. Analisis sentimen tidak ditujukan untuk proses pemeliharaan perangkat lunak selanjutnya.
6	Yulianti dan Nissa, 2024	<i>ABSA of Indonesian customer reviews using IndoBERT: single-sentence and sentence-pair classification approaches [7]</i>	IndoBERT Single-Sentence Classification	Membuktikan bahwa model IndoBERT berbasis klasifikasi per kalimat (<i>single-sentence</i>) sangat tangguh dalam mengekstraksi sentimen berbasis aspek (ABSA) pada ulasan pelanggan Indonesia. Fokus penelitian adalah klasifikasi ke dalam kategori akurasi, kualitas, pelayanan, harga, pengemasan dan pengiriman.	a. Sentimen tidak pada label positif, negatif, netral, laporan <i>bug</i> dan saran fitur. b. Belum menggunakan sistem <i>class weight</i> untuk menangani data yang tidak seimbang (<i>imbalanced data</i>). c. Belum melakukan analisis ketidaksesuaian antara hasil prediksi teks dengan peringkat bintang aktual. d. Analisis sentimen tidak ditujukan untuk proses pemeliharaan perangkat lunak selanjutnya.

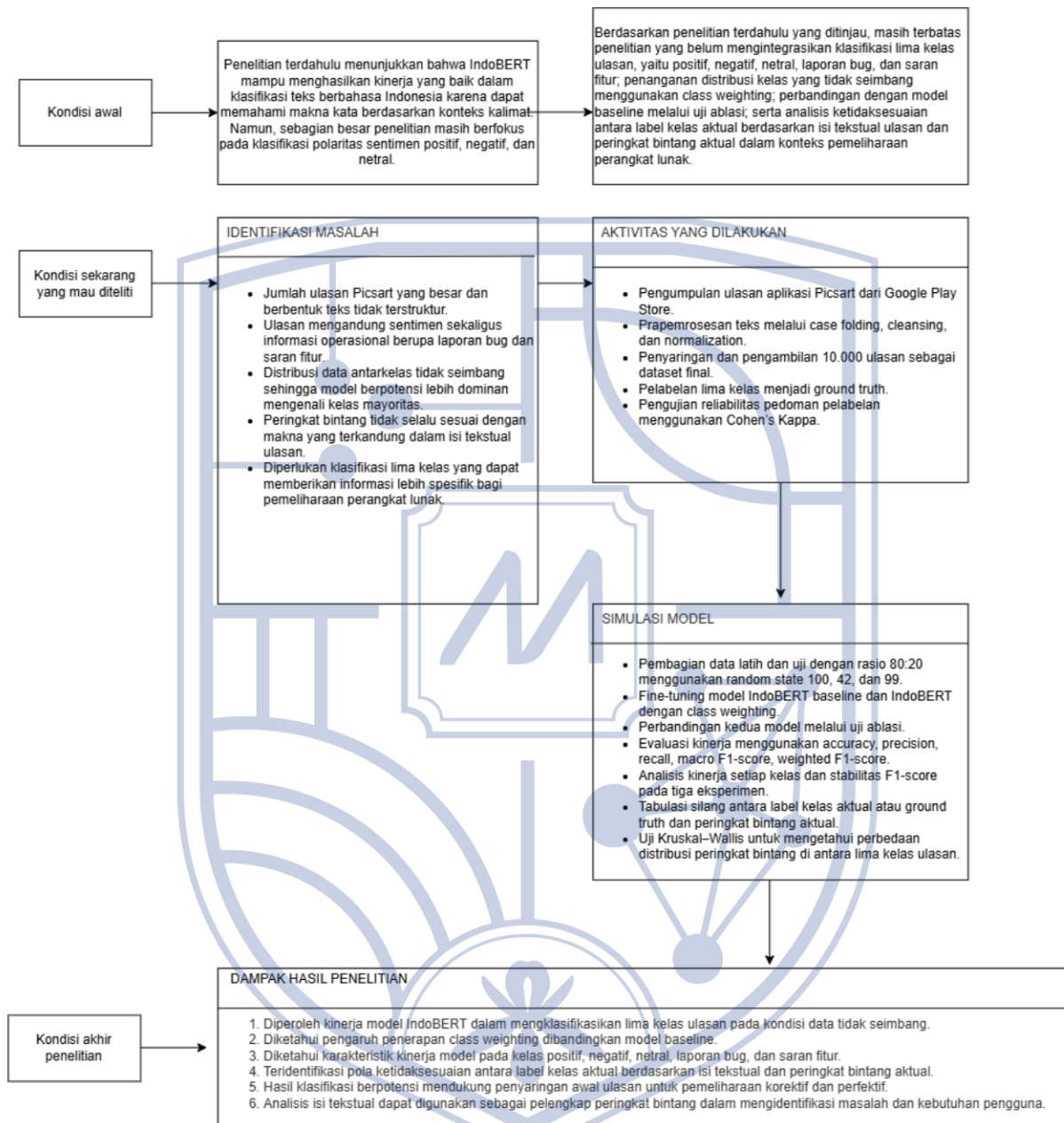
7	Sinaga dkk., 2025	<i>DynamicSentiment Analysis on the Emergence of Pre-Trained Generative Model-Based Applications in Indonesia</i> [8]	IndoBERT	Penelitian ini memperluas model IndoBERT dengan kerangka analisis sentimen dinamis yang berbutir halus (<i>fine-grained</i>) seperti positif, negatif, netral, sangat positif, sangat negatif.	a. Sentimen belum mencakup laporan <i>bug</i> dan saran fitur. b. Belum menggunakan sistem <i>class weight</i> untuk menangani data yang tidak seimbang (<i>imbalanced data</i>). c. Belum melakukan analisis ketidaksesuaian antara hasil prediksi teks dengan peringkat bintang aktual. d. Analisis sentimen tidak ditujukan untuk proses pemeliharaan perangkat lunak selanjutnya.
---	-------------------	---	----------	--	--

Sintesis penelitian terdahulu menunjukkan bahwa model berbasis *transformer*, khususnya IndoBERT, telah digunakan pada berbagai tugas klasifikasi teks berbahasa Indonesia, mulai dari sentimen biner dan tiga kelas, pengenalan emosi, hingga analisis sentimen berbasis aspek [2]–[8]. Temuan-temuan tersebut memberikan dasar bahwa IndoBERT relevan digunakan untuk memahami konteks bahasa Indonesia, tetapi fokus kelas dan tujuan analisis pada penelitian terdahulu berbeda dengan rancangan tesis ini.

2.3 Kerangka Konsep

Kerangka konseptual merupakan gambaran umum mengenai hubungan antara permasalahan penelitian, teori yang digunakan, metode yang diterapkan, dan hasil yang ingin dicapai. Dalam sebuah penelitian, kerangka konseptual berfungsi sebagai peta berpikir agar alur penelitian dapat dipahami secara sistematis, mulai dari kondisi awal yang menjadi dasar masalah, proses atau pendekatan yang digunakan untuk menyelesaikan masalah, hingga kondisi akhir yang diharapkan dari penelitian. Fungsi utama kerangka konseptual adalah membantu peneliti menjelaskan arah penelitian secara ringkas dan terstruktur. Melalui kerangka ini, pembaca dapat memahami mengapa suatu metode dipilih, bagaimana metode tersebut digunakan untuk menjawab rumusan masalah, serta bagaimana hasil penelitian diharapkan dapat memberikan kontribusi terhadap bidang kajian yang diteliti.

Alur kerangka konseptual dibagi menjadi kondisi awal, kondisi sekarang yang diteliti, dan kondisi akhir yang diharapkan sebagaimana ditunjukkan pada Gambar 2.6.



Gambar 2.6 Kerangka Konseptual

Secara struktural, kerangka konseptual penelitian terdiri dari:

1. Kondisi awal

Penelitian terdahulu menunjukkan bahwa IndoBERT telah digunakan untuk berbagai tugas klasifikasi teks berbahasa Indonesia, tetapi studi yang ditinjau masih memisahkan fokus polaritas sentimen dari *actionable feedback*, belum secara bersamaan mengevaluasi

class weighting terhadap *baseline*, dan belum mengintegrasikan analisis ketidaksesuaian isi tekstual dengan peringkat bintang dalam konteks pemeliharaan perangkat lunak.

2. Kondisi sekarang yang diteliti

Ulasan aplikasi Picsart tersedia dalam jumlah besar, berbentuk teks tidak terstruktur, serta memuat sentimen dan informasi operasional yang berbeda. Distribusi data antarkelas yang tidak seimbang dapat menyebabkan model lebih dominan mengenali kelas mayoritas, sedangkan peringkat bintang tidak selalu menggambarkan makna yang disampaikan dalam isi ulasan. Berdasarkan kondisi tersebut, penelitian menerapkan klasifikasi lima kelas menggunakan IndoBERT, membandingkan model *baseline* dengan model yang menggunakan *class weighting*, serta menganalisis ketidaksesuaian antara label kelas aktual berdasarkan isi tekstual ulasan dan peringkat bintang aktual.

3. Kondisi akhir penelitian

Diperoleh informasi mengenai kinerja IndoBERT dalam mengklasifikasikan ulasan positif, negatif, netral, laporan *bug*, dan saran fitur pada kondisi distribusi kelas yang tidak seimbang. Penelitian juga menghasilkan informasi mengenai pengaruh penerapan *class weighting*, karakteristik kinerja setiap kelas, serta pola ketidaksesuaian antara isi tekstual ulasan dan peringkat bintang. Hasil tersebut berpotensi digunakan sebagai alat bantu penyaringan awal ulasan untuk mendukung identifikasi masalah, kebutuhan pengguna, serta pemeliharaan perangkat lunak secara korektif dan perfektif.

Dalam penelitian ini, kerangka konseptual menggambarkan hubungan antara ulasan pengguna yang tidak terstruktur, kebutuhan klasifikasi hibrida sentimen dan intensi, masalah ketidakseimbangan kelas, penerapan IndoBERT dengan *class weighting*, serta analisis ketidaksesuaian antara narasi ulasan dan peringkat bintang. Kerangka ini menegaskan bahwa ulasan Picsart tidak hanya berfungsi sebagai indikator kepuasan, tetapi juga sebagai sumber informasi operasional yang berpotensi membantu penyaringan laporan *bug* dan saran fitur dalam konteks pemeliharaan perangkat lunak. Aspek yang belum diselesaikan secara bersamaan dalam tujuh penelitian yang ditinjau adalah penggabungan polaritas sentimen dengan intensi umpan balik berupa laporan *bug* dan saran fitur, evaluasi *class weighting* pada kondisi kelas tidak seimbang melalui perbandingan dengan *baseline*, serta analisis ketidaksesuaian antara isi tekstual dan *star rating*. Dengan demikian, *research gap* penelitian ini bersifat integratif dan metodologis, bukan sekadar perbedaan objek aplikasi yang digunakan. Posisi penelitian ini terhadap gap tersebut adalah mengevaluasi IndoBERT pada klasifikasi hibrida lima kelas dalam skema *single-label multi-class*, membandingkan *baseline* dengan *class weighting* melalui uji ablasi terkontrol, dan menganalisis *text-rating*

discrepancy pada dataset final ulasan Picsart. Posisi ini juga membatasi klaim kontribusi pada evaluasi empiris dan metodologis; penelitian tidak mengklaim menghasilkan arsitektur atau algoritma baru.

