BAB 2
LANDASAN TEORI

2.1 Penelitian Terkait

Penelitian terkait disusun sebagai pembanding antara penelitian terdahulu dan penelitian ini, bukan sebagai daftar ringkasan pustaka. Rujukan dipilih berdasarkan kedekatan dengan empat aspek utama, yaitu domain ulasan produk dan PRDECT-ID, tugas klasifikasi emosi teks, penggunaan BERT atau IndoBERT, serta penerapan *parameter-efficient fine-tuning* (PEFT) terutama LoRA. Karena itu, setiap penelitian dibaca dari persamaan, perbedaan, kontribusi, dan batas relevansinya terhadap penerapan LoRA pada IndoBERT untuk klasifikasi emosi ulasan produk berbahasa Indonesia.

Tabel 2.1 merangkum penelitian yang paling dekat dengan domain data, tugas klasifikasi emosi, dan model klasifikasi ulasan produk Indonesia. Tabel 2.2 merangkum penelitian yang berkaitan dengan PEFT dan LoRA. Pemisahan ini diperlukan karena penelitian pada domain ulasan produk belum tentu mengevaluasi efisiensi parameter, sedangkan penelitian LoRA belum tentu berada pada domain ulasan *e-commerce* Indonesia. Dengan pemisahan tersebut, perbandingan dapat menunjukkan bagian mana yang sudah dijawab oleh penelitian terdahulu dan bagian mana yang masih menjadi celah penelitian.

Tabel 2.1. Penelitian terkait pada domain, dataset, dan model klasifikasi emosi

Peneliti dan Tahun	**Topik, Dataset, dan Metode**	**Hasil Utama**	**Keterbatasan, Relevansi, dan Celah**
Sutoyo et al. [8]	Penyusunan PRDECT-ID sebagai dataset ulasan produk Tokopedia berbahasa Indonesia untuk klasifikasi emosi dan sentimen.	Dataset berisi 5.400 ulasan dari 29 kategori produk Tokopedia dengan label sentimen dan lima label emosi. Kontribusi utamanya adalah penyediaan dataset yang relevan untuk klasifikasi emosi ulasan produk Indonesia.	Persamaannya dengan skripsi ini terletak pada domain ulasan produk Indonesia, sumber data PRDECT-ID, dan skema label emosi. Perbedaannya, artikel ini merupakan artikel data sehingga tidak mengevaluasi IndoBERT, LoRA, atau efisiensi parameter. Rujukan ini menjadi dasar dataset, celah yang diisi adalah pengujian adaptasi IndoBERT yang efisien pada dataset tersebut.
Lanjut pada halaman berikutnya			

Tabel 2.1 Penelitian terkait pada domain, dataset, dan model klasifikasi emosi (lanjutan)

Peneliti dan Tahun	**Topik, Dataset, dan Metode**	**Hasil Utama**	**Keterbatasan, Relevansi, dan Celah**
Raharko dan Yamasari [15]	Klasifikasi emosi ulasan *e-commerce* menggunakan Support Vector Machine dengan beberapa fungsi *kernel*.	Pada data tanpa *resampling*, akurasi tertinggi dicapai oleh kernel linear dan *sigmoid* sebesar 66,30%. Nilai F1-score tertinggi dicapai oleh kernel *sigmoid* sebesar 62,91%.	Persamaannya terletak pada tugas klasifikasi emosi ulasan *e-commerce*. Perbedaannya, penelitian ini menggunakan SVM sehingga belum memanfaatkan representasi kontekstual IndoBERT dan belum membahas PEFT. Rujukan ini menjadi pembanding non-Transformer, celah yang diisi adalah pengujian Transformer Indonesia dengan adaptasi parameter efisien.
Lanjut pada halaman berikutnya			

Tabel 2.1 Penelitian terkait pada domain, dataset, dan model klasifikasi emosi (lanjutan)

Peneliti dan Tahun	**Topik, Dataset, dan Metode**	**Hasil Utama**	**Keterbatasan, Relevansi, dan Celah**
Pramana et al. [11]	Klasifikasi emosi ulasan produk pada PRDECT-ID menggunakan BiLSTM, BERT, dan *ensemble*.	Evaluasi menunjukkan bahwa model berbasis BERT dan *ensemble* kompetitif untuk PRDECT-ID. *Unweighted stacked generalization* memperoleh F1-score 74%, sedangkan *weighted stacked ensemble* memperoleh F1-score 75%.	Persamaannya terletak pada PRDECT-ID dan tugas klasifikasi emosi. Perbedaannya, penelitian ini berfokus pada BiLSTM, BERT, dan *ensemble*, bukan LoRA atau efisiensi *fine-tuning*. Rujukan ini menjadi pembanding performa berbasis BERT/*ensemble*, celah yang diisi adalah evaluasi performa dan biaya pelatihan IndoBERT+LoRA terhadap *full fine-tuning*.
Lanjut pada halaman berikutnya			

Tabel 2.1 Penelitian terkait pada domain, dataset, dan model klasifikasi emosi (lanjutan)

Peneliti dan Tahun	**Topik, Dataset, dan Metode**	**Hasil Utama**	**Keterbatasan, Relevansi, dan Celah**
Christian et al. [12]	Klasifikasi emosi ulasan *e-commerce* Indonesia menggunakan IndoBERT, DistilBERT, augmentasi data, dan penyesuaian *hyperparameter*.	IndoBERT mencapai akurasi 80,00% setelah penyesuaian *hyperparameter*. Percobaan *bagging* lima model IndoBERT mencapai akurasi 79,77% dan F1-score 79,79%, sehingga tidak melampaui konfigurasi IndoBERT terbaik.	Persamaannya kuat pada domain ulasan *e-commerce* Indonesia dan penggunaan IndoBERT. Perbedaannya, penelitian tersebut tidak berfokus pada LoRA, jumlah parameter latih, waktu pelatihan, atau penggunaan memori. Rujukan ini menjadi pembanding terdekat dari sisi model dan domain, celah yang diisi adalah analisis konfigurasi LoRA dan perbandingannya dengan *full fine-tuning*.

Tabel 2.2. Penelitian terkait pada PEFT dan LoRA

Peneliti dan Tahun	**Topik, Dataset, dan Metode**	**Hasil Utama**	**Keterbatasan, Relevansi, dan Celah**
Hu et al. [13]	Pengembangan LoRA sebagai metode adaptasi model bahasa pra-latih melalui pembaruan *low-rank*.	LoRA membekukan bobot utama model dan melatih matriks adaptasi berperingkat rendah. Pada GPT-3 175B, LoRA dilaporkan dapat mengurangi jumlah parameter latih hingga 10.000 kali dan kebutuhan memori GPU hingga 3 kali dibandingkan *full fine-tuning*.	Persamaannya dengan skripsi ini terbatas pada metode LoRA. Perbedaannya, penelitian ini tidak membahas IndoBERT, bahasa Indonesia, atau ulasan produk Tokopedia. Rujukan ini menjadi dasar metodologis, celah yang diisi adalah penerapan dan pengujian LoRA pada IndoBERT untuk klasifikasi emosi ulasan Indonesia.
Lanjut pada halaman berikutnya			

Tabel 2.2 Penelitian terkait pada PEFT dan LoRA (lanjutan)

Peneliti dan Tahun	**Topik, Dataset, dan Metode**	**Hasil Utama**	**Keterbatasan, Relevansi, dan Celah**
Wang et al. [14]	Survei PEFT pada model bahasa besar, mencakup taksonomi metode, penggunaan, serta isu evaluasi efisiensi dan performa.	Survei ini menegaskan bahwa PEFT perlu dievaluasi tidak hanya berdasarkan performa prediktif, tetapi juga berdasarkan biaya adaptasi seperti jumlah parameter latih dan kebutuhan komputasi.	Persamaannya terletak pada isu PEFT dan evaluasi efisiensi. Perbedaannya, penelitian ini bersifat survei umum dan tidak mengevaluasi PRDECT-ID, IndoBERT, atau klasifikasi emosi ulasan produk. Rujukan ini memperkuat alasan penggunaan metrik efisiensi, celah yang diisi adalah pembuktian empiris pada domain ulasan *e-commerce* Indonesia.
Lanjut pada halaman berikutnya			

Tabel 2.2 Penelitian terkait pada PEFT dan LoRA (lanjutan)

Peneliti dan Tahun	**Topik, Dataset, dan Metode**	**Hasil Utama**	**Keterbatasan, Relevansi, dan Celah**
Nwaiwu [19]	Perbandingan LoRA, IA^3, ReFT, dan *full fine-tuning* untuk klasifikasi teks *low-resource* pada AG News dan Amazon Reviews menggunakan DistilBERT.	LoRA memperoleh F1-score tertinggi pada Amazon Reviews, sedangkan ReFT menggunakan parameter dan memori yang lebih kecil. Hasil ini menunjukkan adanya kompromi antara performa dan efisiensi pada PEFT.	Persamaannya terletak pada perbandingan PEFT dengan *full fine-tuning* dan perhatian pada efisiensi. Perbedaannya, dataset, bahasa, dan model berbeda dari IndoBERT serta ulasan Tokopedia Indonesia. Rujukan ini menjadi pembanding metodologis, celah yang diisi adalah pengujian LoRA pada model dan data Indonesia.
Lanjut pada halaman berikutnya			

Tabel 2.2 Penelitian terkait pada PEFT dan LoRA (lanjutan)

Peneliti dan Tahun	**Topik, Dataset, dan Metode**	**Hasil Utama**	**Keterbatasan, Relevansi, dan Celah**
Wang et al. [17]	Klasifikasi emosi dan empati pada WASSA 2023 menggunakan LoRA pada DeBERTa dan RoBERTa.	Sistem berbasis DeBERTa+LoRA dilaporkan memperoleh Macro-F1 0,486 pada *development set* dan 0,514 pada *test set* untuk Track3-EMO WASSA 2023.	Persamaannya terletak pada penggunaan LoRA untuk tugas emosi. Perbedaannya, domain, bahasa, dataset, dan model berbeda dari ulasan produk Indonesia. Rujukan ini menjadi pembanding tugas emosi berbasis LoRA, celah yang diisi adalah penerapan pada IndoBERT dan ulasan *e-commerce* berbahasa Indonesia.
Lanjut pada halaman berikutnya			

Tabel 2.2 Penelitian terkait pada PEFT dan LoRA (lanjutan)

Peneliti dan Tahun	**Topik, Dataset, dan Metode**	**Hasil Utama**	**Keterbatasan, Relevansi, dan Celah**
Hidayatullah [18]	LoRA untuk klasifikasi emosi teks Indonesia menggunakan pendekatan PEFT.	Penelitian ini menunjukkan bahwa PEFT berbasis LoRA dapat dipakai sebagai alternatif terhadap *full-parameter fine-tuning* pada klasifikasi emosi teks Indonesia.	Persamaannya terletak pada LoRA dan klasifikasi emosi berbahasa Indonesia. Perbedaannya, penelitian tersebut tidak secara khusus mengevaluasi PRDECT-ID, ulasan Tokopedia, atau IndoBERT Base dalam rancangan skripsi ini. Rujukan ini dekat dari sisi tugas dan bahasa, celah yang diisi adalah evaluasi LoRA pada domain ulasan produk dan konfigurasi IndoBERT.
Lanjut pada halaman berikutnya			

Tabel 2.2 Penelitian terkait pada PEFT dan LoRA (lanjutan)

Peneliti dan Tahun	**Topik, Dataset, dan Metode**	**Hasil Utama**	**Keterbatasan, Relevansi, dan Celah**
Arthana et al. [21]	Fine-tuning IndoBERT dengan LoRA untuk klasifikasi kesantunan teks Indonesia.	Penelitian ini mengevaluasi penggunaan LoRA pada IndoBERT untuk tugas klasifikasi teks Indonesia di luar domain emosi.	Persamaannya terletak pada penggunaan LoRA dan IndoBERT untuk klasifikasi teks Indonesia. Perbedaannya, tugas yang dikaji adalah kesantunan, bukan emosi ulasan produk. Rujukan ini menjadi pembanding dari sisi model dan bahasa, celah yang diisi adalah evaluasi LoRA untuk lima emosi pada ulasan *e-commerce*.
Lanjut pada halaman berikutnya			

Tabel 2.2 Penelitian terkait pada PEFT dan LoRA (lanjutan)

Peneliti dan Tahun	**Topik, Dataset, dan Metode**	**Hasil Utama**	**Keterbatasan, Relevansi, dan Celah**
Nugroho et al. [20]	LoRA pada model BERT untuk deteksi distorsi kognitif berbahasa Indonesia menggunakan Multilingual-BERT, IndoNLU-IndoBERT, dan IndoLEM-IndoBERT.	IndoLEM-IndoBERT dengan LoRA memperoleh hasil terbaik dengan *accuracy* 0,9657 dan F1-score 0,9676 pada 100% data. Penelitian tersebut juga menilai efisiensi inferensi.	Persamaannya terletak pada penerapan LoRA pada BERT/IndoBERT untuk teks Indonesia serta perhatian pada efisiensi. Perbedaannya, domainnya adalah deteksi distorsi kognitif, bukan ulasan *e-commerce*. Rujukan ini hanya menjadi pembanding metodologis, celah yang diisi adalah pengujian pada klasifikasi emosi ulasan produk.

Kajian pada Tabel 2.1 menunjukkan perkembangan penelitian pada sisi domain dan model. PRDECT-ID memberi dasar data dan skema lima emosi untuk ulasan produk Indonesia. Penelitian berikutnya membandingkan pendekatan klasik, BERT/IndoBERT, augmentasi data, dan *ensemble*. Persamaannya dengan skripsi ini terletak pada domain ulasan produk, tugas klasifikasi emosi, atau penggunaan model berbasis BERT. Perbedaannya, penelitian-penelitian tersebut masih berorientasi pada capaian performa klasifikasi dan belum menjadikan efisiensi parameter sebagai objek evaluasi utama. Oleh karena itu, bagian ini menunjukkan bahwa kebutuhan domain sudah jelas, tetapi aspek efisiensi adaptasi IndoBERT masih belum terjawab secara langsung.

Kajian pada Tabel 2.2 menunjukkan perkembangan dari sisi PEFT dan LoRA. LoRA telah digunakan sebagai strategi adaptasi model pada berbagai tugas klasifikasi teks, termasuk tugas emosi, kesantunan, dan distorsi kognitif. Namun, kedekatan setiap penelitian bersifat parsial: ada yang dekat pada metode LoRA, ada yang dekat pada tugas emosi, ada yang dekat pada IndoBERT, dan ada yang dekat pada evaluasi efisiensi. Tidak ada penelitian dalam tabel yang sekaligus mencakup PRDECT-ID, IndoBERT Base, LoRA, klasifikasi lima emosi ulasan produk, dan perbandingan efisiensi terhadap *full fine-tuning*. Dengan demikian, rujukan LoRA pada bagian ini digunakan sebagai pembanding metodologis, bukan sebagai bukti bahwa konteks penelitian ini sudah pernah dijawab sepenuhnya.

Berdasarkan perbandingan tersebut, skripsi ini tidak mengulang penelitian terdahulu, melainkan menggabungkan aspek yang sebelumnya masih terpisah. Celah penelitian yang diisi adalah evaluasi beberapa konfigurasi LoRA pada IndoBERT Base untuk klasifikasi lima emosi ulasan produk berbahasa Indonesia, dengan perbandingan langsung terhadap *full fine-tuning* pada ruang eksperimen yang sama. Celah tersebut mencakup tiga aspek. Pertama, domain dan skema label diarahkan pada ulasan produk Tokopedia dengan lima kelas emosi yang mengikuti PRDECT-ID. Kedua, strategi adaptasi difokuskan pada variasi konfigurasi LoRA, yaitu *rank*, *alpha*, dan *target modules*. Ketiga, evaluasi tidak hanya menggunakan *macro-F1*, tetapi juga metrik efisiensi berupa jumlah *trainable parameters*, waktu pelatihan, dan *peak GPU memory*. Posisi penelitian ini adalah sebagai studi yang menghubungkan klasifikasi emosi ulasan *e-commerce* Indonesia dengan evaluasi PEFT pada IndoBERT secara terukur dari sisi performa dan efisiensi.

2.2 Pemrosesan Bahasa Alami dan Klasifikasi Teks

Pemrosesan bahasa alami atau *Natural Language Processing* (NLP) merupakan bidang yang mengkaji pemrosesan bahasa manusia secara komputasional. Pada pembelajaran mesin, teks perlu diubah menjadi representasi numerik agar dapat diproses oleh model. Representasi tersebut kemudian digunakan untuk berbagai tugas, seperti klasifikasi teks, ekstraksi informasi, analisis sentimen, dan klasifikasi emosi [22, 23, 24].

Klasifikasi teks merupakan tugas untuk memetakan dokumen, kalimat, atau ulasan ke kategori tertentu berdasarkan isi teksnya. Pada tugas *supervised learning*, model mempelajari hubungan antara masukan berupa teks dan keluaran berupa label dari data berlabel. Dalam klasifikasi emosi ulasan produk, masukan berupa

teks ulasan berbahasa Indonesia, sedangkan keluaran berupa kelas emosi yang telah ditentukan.

Tugas klasifikasi emosi yang dibahas berbentuk *single-label multiclass classification*. Setiap ulasan memiliki satu label emosi dominan dari lima kelas, yaitu *anger*, *fear*, *happy*, *love*, dan *sadness*. Bentuk tugas ini berbeda dari *multi-label classification* karena satu ulasan tidak diberi lebih dari satu label emosi secara bersamaan. Dengan demikian, model menghasilkan distribusi probabilitas terhadap seluruh kelas, kemudian kelas dengan probabilitas tertinggi dipilih sebagai prediksi akhir [23, 25].

Model berbasis Transformer seperti BERT dan IndoBERT mendukung klasifikasi teks karena mampu membangun representasi kontekstual. Representasi kontekstual memungkinkan makna token dipengaruhi oleh token lain dalam kalimat. Kemampuan ini relevan pada ulasan *e-commerce* karena ekspresi emosi tidak selalu muncul melalui satu kata kunci, tetapi dapat terbentuk dari susunan kata, negasi, konteks produk, dan gaya bahasa pengguna [26, 27, 28].

2.3 Ulasan Produk pada E-Commerce

Ulasan produk merupakan bentuk teks opini yang ditulis konsumen mengenai produk atau pengalaman pembelian. Dalam kajian *opinion mining*, ulasan pelanggan dapat digunakan untuk mengekstraksi opini, sentimen, evaluasi, dan aspek produk yang dianggap penting oleh pengguna [29, 24]. Pada domain *e-commerce*, isi ulasan dapat berkaitan dengan kualitas produk, manfaat produk, kesesuaian barang, layanan penjual, pengiriman, dan pengalaman transaksi. Karena ulasan ditulis oleh pengguna, teksnya dapat bervariasi dari sisi panjang, kelengkapan informasi, serta kualitas kebahasaan [30].

Ulasan *e-commerce* berbahasa Indonesia termasuk data teks tidak terstruktur karena tidak selalu mengikuti pola kalimat baku dan tidak langsung tersedia dalam bentuk fitur numerik. Ulasan dapat berisi kalimat singkat, pengulangan huruf, campuran kata Indonesia dan asing, istilah produk, emotikon, serta kata tidak baku. Kondisi tersebut membuat model perlu memproses teks secara kontekstual agar label emosi tidak hanya ditentukan oleh kemunculan kata tertentu [24, 30, 31].

PRDECT-ID relevan sebagai dasar domain karena dataset tersebut berasal dari ulasan produk Tokopedia. Apabila data tambahan dikumpulkan dari domain yang sama, konsistensi domain perlu dijaga agar interpretasi hasil tidak diperluas menjadi klaim terhadap seluruh platform *e-commerce* atau seluruh bentuk ulasan

daring.

2.4 Analisis Sentimen dan Klasifikasi Emosi

Analisis sentimen dan *opinion mining* merupakan bidang yang mengkaji opini, sentimen, evaluasi, sikap, dan emosi yang diekspresikan dalam teks [32, 24]. Analisis sentimen umumnya berfokus pada pengelompokan opini berdasarkan polaritas, seperti positif, negatif, atau netral. Klasifikasi emosi lebih spesifik karena memetakan teks ke kategori emosi tertentu. Dua ulasan yang sama-sama negatif dapat merepresentasikan emosi berbeda, misalnya *anger* ketika pengguna menyampaikan protes keras dan *sadness* ketika pengguna menunjukkan kekecewaan.

Sentimen dan emosi sama-sama berkaitan dengan ekspresi afektif dalam teks, tetapi keduanya memiliki fokus yang berbeda. Sentimen biasanya merangkum orientasi penilaian terhadap objek, sedangkan emosi menjelaskan keadaan afektif yang lebih spesifik, seperti marah, takut, senang, suka, atau sedih. Perbedaan ini penting pada ulasan produk karena polaritas yang sama dapat mengandung bentuk emosi yang berbeda. Ulasan negatif dapat menunjukkan kemarahan, kekhawatiran, atau kesedihan, sedangkan ulasan positif dapat menunjukkan kepuasan biasa atau kesukaan yang kuat [24, 33].

Klasifikasi emosi merupakan bentuk khusus dari klasifikasi teks karena teks dipetakan ke kategori afektif yang telah ditentukan [22, 24, 33]. Pada ulasan *e-commerce*, label emosi dapat membantu menangkap respons pengguna secara lebih rinci dibandingkan sentimen berbasis polaritas. Oleh karena itu, skema label emosi perlu didefinisikan secara operasional agar setiap kelas memiliki batas makna yang jelas.

2.5 Dataset PRDECT-ID dan Skema Label Emosi

PRDECT-ID merupakan dataset ulasan produk Indonesia untuk tugas klasifikasi emosi dan sentimen [8]. Dataset ini berisi ulasan produk berbahasa Indonesia dari Tokopedia yang berasal dari 29 kategori produk. PRDECT-ID memuat 5.400 ulasan produk dan menyediakan label emosi untuk lima kelas, yaitu *anger*, *fear*, *happy*, *love*, dan *sadness* [8].

PRDECT-ID menjadi dataset rujukan domain karena menyediakan ulasan produk Tokopedia berbahasa Indonesia dengan skema label emosi yang sesuai

untuk tugas klasifikasi emosi. Penggunaan data tambahan pada domain Tokopedia perlu menjaga konsistensi dengan karakteristik tugas PRDECT-ID, terutama dalam hal label emosi, struktur data, dan pemisahan data evaluasi. Dengan demikian, landasan dataset tidak hanya didasarkan pada ketersediaan ulasan, tetapi juga pada kesesuaian domain dan skema label dengan penelitian terdahulu [8, 11].

Dasar konseptual label emosi mengacu pada pendekatan prototipe emosi dari Shaver et al. [33]. Pendekatan tersebut memandang pengetahuan emosi sebagai kategori yang tersusun secara hierarkis dan berbasis prototipe. Pada tingkat dasar, kategori seperti *love*, *joy*, *anger*, *sadness*, dan *fear* dapat digunakan untuk membedakan pengalaman emosi dalam bahasa sehari-hari [33]. Skema lima label yang digunakan, yaitu *anger*, *fear*, *happy*, *love*, dan *sadness*, mengikuti skema label PRDECT-ID. Label *happy* diposisikan sebagai padanan operasional dari *joy* sesuai penggunaan label pada dataset [8].

Kajian emosi dasar lain seperti Ekman [34] dan Plutchik [35] digunakan sebagai rujukan pendukung bahwa emosi dapat dikaji sebagai kategori-kategori utama. Namun, skema lima label tersebut tidak diposisikan sebagai skema universal untuk seluruh tugas klasifikasi emosi. Skema tersebut diperlakukan sebagai skema operasional yang sesuai dengan dataset, domain ulasan *e-commerce*, dan tujuan eksperimen. Definisi pada Tabel 2.3 disusun sebagai definisi operasional berbasis teori emosi dan karakteristik label PRDECT-ID [33, 8].

Tabel 2.3. Definisi konseptual dan operasional label emosi

Label	**Definisi Konseptual**	**Definisi Operasional pada Ulasan**
Anger	Emosi yang berhubungan dengan kemarahan, protes, atau penolakan terhadap kondisi yang dianggap tidak sesuai [33, 34].	Ulasan yang menunjukkan komplain keras, ketidakpuasan eksplisit, nada menyalahkan, atau ekspresi marah terhadap produk, pengiriman, maupun layanan.
Lanjut pada halaman berikutnya		

Tabel 2.3 Definisi konseptual dan operasional label emosi (lanjutan)

Label	**Definisi Konseptual**	**Definisi Operasional pada Ulasan**
Fear	Emosi yang berhubungan dengan rasa takut, khawatir, cemas, atau antisipasi terhadap risiko [33, 35].	Ulasan yang menunjukkan kekhawatiran terhadap keamanan, kerusakan, ketidakpastian kondisi barang, risiko penggunaan, atau pengalaman transaksi yang menimbulkan kecemasan.
Happy	Emosi positif yang berkaitan dengan kegembiraan, kepuasan, atau rasa senang dan secara operasional dipadankan dengan kategori *joy* pada teori emosi [33, 35].	Ulasan yang menunjukkan kepuasan, rasa senang, pengalaman pembelian yang sesuai harapan, atau apresiasi positif terhadap produk dan layanan.
Love	Emosi positif yang berkaitan dengan rasa suka, afeksi, kedekatan, atau penilaian sangat menyukai objek tertentu [33].	Ulasan yang menunjukkan kesukaan kuat terhadap produk, loyalitas, pujian intens, atau ungkapan sangat menyukai barang yang diterima.
Sadness	Emosi yang berkaitan dengan kesedihan, kekecewaan, kehilangan, atau rasa tidak puas yang bernada pasrah [33, 34].	Ulasan yang menunjukkan kekecewaan, penyesalan, pengalaman tidak sesuai harapan, atau ungkapan sedih karena kualitas produk, pengiriman, atau layanan.

2.6 Tantangan Teks Ulasan Berbahasa Indonesia

Pemrosesan bahasa Indonesia memiliki tantangan karena sumber daya, *benchmark*, dan korpus beranotasi untuk bahasa Indonesia tidak sebanyak bahasa bersumber daya tinggi. IndoLEM dan IndoNLU dikembangkan untuk menyediakan *benchmark* serta sumber daya evaluasi bagi NLP bahasa Indonesia, dan keduanya menegaskan pentingnya model serta dataset yang sesuai dengan karakteristik bahasa Indonesia [28, 10]. Dalam konteks ulasan produk, tantangan tersebut diperkuat oleh karakter teks pengguna yang tidak selalu mengikuti gaya bahasa formal.

Teks pengguna berbahasa Indonesia dapat mengandung bentuk kolokial, variasi ejaan, singkatan, kata tidak baku, kesalahan ketik, campuran bahasa, dan bentuk *out-of-vocabulary*. *Colloquial Indonesian Lexicon* digunakan untuk normalisasi kata kolokial Indonesia dan menunjukkan bahwa bentuk bahasa tidak baku perlu diperhatikan dalam pengolahan teks Indonesia [31]. Pada ulasan *e-commerce*, variasi semacam ini dapat muncul bersama istilah produk, nama merek, ekspresi emosional, repetisi huruf, emotikon, dan campuran istilah asing.

Ekspresi emosi pada ulasan juga dapat bersifat implisit. Ulasan seperti keluhan terhadap keterlambatan pengiriman, kekhawatiran terhadap keamanan produk, atau kekecewaan terhadap kondisi barang tidak selalu memuat kata emosi secara eksplisit. Oleh karena itu, model klasifikasi emosi perlu menangkap konteks kalimat dan hubungan antartoken. Model berbasis representasi kontekstual seperti IndoBERT relevan untuk tugas ini karena memproses teks berdasarkan konteks sekuens, bukan hanya daftar kata terpisah [28, 11, 12].

2.7 Pelabelan Otomatis dan Anotasi Berbantuan LLM

Label target otomatis adalah label yang dihasilkan melalui proses anotasi berbantuan sistem, bukan melalui penetapan label referensi manual final oleh anotator manusia. Dalam konteks LLM, anotasi otomatis dapat membantu mempercepat pembentukan label, tetapi hasilnya perlu diperlakukan secara hati-hati karena dapat dipengaruhi oleh instruksi, panduan label, domain teks, dan ambiguitas tugas [36, 37]. Oleh karena itu, label target otomatis lebih tepat diposisikan sebagai label berbantuan sistem atau label lemah, bukan sebagai *gold standard* manual final.

Dalam penelitian berbasis data teks, platform anotasi seperti Label

Studio dapat digunakan untuk mengelola data, menyusun skema label, dan mengintegrasikan bantuan model dalam proses pelabelan. Dokumentasi Label Studio menjelaskan bahwa ML backend dapat dihubungkan dengan Label Studio untuk mengotomasi tugas pelabelan dan mengembalikan prediksi kepada pengguna [38]. Ketika model bahasa besar atau sistem pelabelan otomatis digunakan sebagai alat bantu, label yang dihasilkan tetap perlu diposisikan sebagai hasil berbantuan sistem karena kualitasnya dapat dipengaruhi oleh karakteristik model, instruksi anotasi, dan ambiguitas domain teks [36].

Konsep label lemah berkaitan dengan *weak supervision*, yaitu penggunaan sumber label yang lebih murah atau lebih mudah diperoleh, tetapi berpotensi mengandung *noise* dan konflik [39]. Kontrol kualitas label otomatis tidak dimaksudkan untuk membuktikan bahwa seluruh label benar secara objektif menurut penilaian manusia. Kontrol kualitas diarahkan untuk memastikan konsistensi teknis dataset, seperti format label, distribusi kelas, duplikasi, dan potensi kebocoran data antarsplit.

Kelemahan utama pelabelan otomatis menggunakan GPT tanpa pemeriksaan manusia yang terstruktur dan terukur adalah ketepatan makna label belum dapat dinilai menggunakan pembanding di luar sistem GPT. Pemeriksaan terhadap format label, distribusi kelas, data duplikat, dan konsistensi nama kelas hanya menunjukkan bahwa dataset telah tersusun dengan baik secara teknis, tetapi belum membuktikan bahwa setiap ulasan telah diberi kelas emosi yang tepat. Ketika hanya melibatkan penulis sebagai anotator manusia yang melakukan pengecekan secara acak, penelitian ini tidak memiliki ukuran kesesuaian antara label GPT dan penilaian manusia (*inter-annotator agreement*), tingkat kesalahan pada setiap kelas, maupun hasil penyelesaian (*adjudication*) untuk ulasan yang bermakna ambigu. Kelemahan tersebut merupakan konsekuensi langsung dari posisi label sebagai *weak supervision* atau *machine-annotated label*, yang secara teoretis tetap sah digunakan sebagai sinyal supervisi selama sumber label, protokol anotasi, dan prosedur pengendalian kualitasnya dilaporkan secara transparan [39, 36]. Perlu ditekankan bahwa yang diotomasi dalam penelitian ini adalah eksekusi pelabelan, bukan perancangan kriteria, yaitu skema lima emosi, *decision protocol*, *ambiguity resolution rules*, serta definisi operasional setiap kelas disusun dan dikendalikan oleh peneliti berdasarkan teori emosi prototipe [33], sehingga pelabelan tidak berlangsung sebagai keluaran model yang bebas kendali, melainkan sebagai penerapan protokol anotasi eksplisit oleh model.

Keluaran GPT juga dapat dipengaruhi oleh susunan instruksi, contoh

yang diberikan dalam *prompt*, konteks data, serta model atau versi model yang digunakan. Kondisi ini dapat menyebabkan kesalahan pelabelan terjadi secara sistematis sehingga hasilnya terlihat teratur, meskipun sebagian label belum tentu sesuai dengan penilaian manusia [36, 37]. Akan tetapi, sifat sistematis tersebut justru menjadi penjamin keadilan perbandingan dalam desain komparatif penelitian ini. Karena seluruh konfigurasi, yaitu *full fine-tuning* dan sepuluh konfigurasi LoRA, dilatih dan dievaluasi pada target label yang identik, noise label bersifat sama (*shared*) di seluruh metode. Dengan demikian, noise tersebut dapat memengaruhi tingkat absolut metrik, tetapi tidak memengaruhi validitas perbandingan relatif antarmetode, karena selisih performa yang teramati merefleksikan perbedaan strategi adaptasi parameter, bukan perbedaan kualitas label yang diterima masing-masing metode.

Sebagai pengganti validasi semantik oleh manusia, pengendalian kualitas teknis tetap dilakukan untuk membatasi ruang kesalahan label. Pengendalian tersebut mencakup validasi format label terhadap lima kelas yang sah, pemeriksaan distribusi kelas agar mendekati seimbang, deduplikasi berbasis representasi teks hasil prapemrosesan, audit *overlap exact-string* antarsplit untuk mencegah kebocoran data, dokumentasi perubahan label awal terhadap label target, serta pengecekan manual secara acak. Rangkaian pemeriksaan ini menjamin konsistensi teknis dataset dan integritas evaluasi lintas sumber, walaupun secara jujur tidak menggantikan penilaian kebenaran semantik oleh anotator manusia.

Dengan posisi tersebut, penggunaan label otomatis tetap konsisten dengan tujuan eksperimen komparatif selama sumber label, prosedur pembentukannya, dan ruang lingkup interpretasi hasil dilaporkan secara terbuka. Prinsip ini sejalan dengan kajian anotasi berbantuan LLM yang menekankan transparansi terhadap sumber label, instruksi anotasi, dan proses pengendalian kualitas [36, 37].

2.8 Prapemrosesan Teks

Prapemrosesan teks adalah tahap untuk menyiapkan data ulasan sebelum tokenisasi dan pelatihan model. Pada teks ulasan berbahasa Indonesia, prapemrosesan perlu dilakukan secara hati-hati karena teks pengguna dapat memuat variasi tidak baku, *slang*, bentuk kolokial, dan ekspresi opini yang menjadi bagian dari informasi yang akan dianalisis. Pembersihan yang terlalu agresif dapat mengubah atau menghilangkan ciri tekstual yang berguna untuk menafsirkan opini dan ekspresi emosi, sedangkan model tetap membutuhkan masukan yang cukup

konsisten agar proses tokenisasi dan deduplikasi dapat dilakukan dengan lebih stabil [24, 31, 22].

Prapemrosesan pada teks ulasan dapat mencakup penyeragaman huruf dan penyaringan karakter yang tidak digunakan dalam bentuk masukan penelitian. Stemming, lemmatisasi, penghapusan *stopword*, atau normalisasi *slang* perlu diperlakukan sebagai keputusan metodologis, bukan sebagai tahapan wajib, karena setiap proses dapat mengubah informasi tekstual yang digunakan model.

2.9 Transformer dan BERT

Transformer adalah arsitektur jaringan saraf yang memodelkan hubungan antartoken melalui mekanisme *attention*, tanpa bergantung pada rekuren sebagai operasi utama [26]. Arsitektur awal Transformer terdiri atas encoder dan decoder. BERT menggunakan tumpukan encoder untuk membangun representasi kontekstual dua arah. Setiap blok encoder memuat *multi-head self-attention*, jaringan *feed-forward*, koneksi residual, dan normalisasi lapisan.

Pemrosesan paralel pada Transformer memungkinkan seluruh posisi dalam sekuens dihitung secara bersamaan pada satu lapisan. Akan tetapi, urutan token tetap perlu direpresentasikan melalui informasi posisi. Pada BERT, informasi token, segmen, dan posisi digabungkan sebelum masuk ke tumpukan encoder [27].

2.9.1 Tokenisasi WordPiece dan Format *Input* BERT

Tokenisasi merupakan proses membagi teks menjadi unit-unit yang lebih kecil yang disebut token. Token tersebut dapat berupa kata, subkata, karakter, atau tanda baca, bergantung pada metode tokenisasi dan kosakata yang digunakan oleh model [40]. Dalam pemrosesan bahasa alami, tokenisasi berfungsi mengubah teks mentah menjadi rangkaian unit yang selanjutnya dapat dikonversi ke bentuk numerik dan diproses oleh model [40].

Tokenisasi berbasis kata memiliki keterbatasan ketika model menggunakan kosakata berukuran tetap. Kata yang jarang muncul atau tidak terdapat dalam kosakata dapat menjadi kata di luar kosakata atau *out-of-vocabulary*. Pendekatan tokenisasi subkata mengurangi permasalahan tersebut dengan merepresentasikan kata langka atau kata yang belum pernah ditemukan sebagai rangkaian unit yang lebih kecil daripada kata [41]. Penggunaan subkata memungkinkan model menggunakan kosakata simbol yang terbatas untuk merepresentasikan lebih banyak

variasi kata [41]. Namun, unit subkata yang dihasilkan tidak selalu sama dengan morfem, suku kata, atau unsur linguistik lain karena pembentukannya bergantung pada kosakata dan algoritma tokenisasi [41].

Salah satu metode tokenisasi subkata adalah WordPiece. WordPiece melakukan prasegmentasi teks menjadi kata, kemudian memecah setiap kata berdasarkan token yang tersedia dalam kosakata [40]. Untuk memecah suatu kata, WordPiece menggunakan strategi *greedy longest-match-first* atau *maximum matching*, yaitu memilih prefiks terpanjang dari bagian teks yang tersisa dan tersedia dalam kosakata [40]. Proses tersebut diulangi sampai seluruh bagian kata berhasil dipetakan menjadi token-token subkata [40]. Pada WordPiece yang digunakan BERT, subkata yang terletak di tengah kata umumnya ditandai dengan prefiks `##` untuk menunjukkan bahwa token tersebut merupakan lanjutan dari token sebelumnya [40].

Apabila suatu kata tidak dapat diuraikan secara lengkap menggunakan token yang tersedia dalam kosakata, WordPiece memetakan kata tersebut ke token khusus yang menyatakan token tidak dikenal, yaitu `[UNK]` [40]. Oleh karena itu, tokenisasi subkata dapat mengurangi kemunculan token tidak dikenal, tetapi tidak menjamin bahwa token `[UNK]` tidak pernah muncul [40]. Hasil tokenisasi juga harus dipahami sebagai segmentasi berdasarkan kosakata model dan bukan sebagai hasil analisis morfologis bahasa secara otomatis [41, 40].

BERT menggunakan tokenisasi WordPiece untuk membentuk sekuens masukannya [27]. Untuk masukan yang terdiri atas satu teks, bentuk konseptual sekuens token BERT dapat ditulis seperti Persamaan 2.1.

$$\texttt{[CLS]}\ t_1\ t_2\ \ldots\ t_n\ \texttt{[SEP]}, \tag{2.1}$$

Pada Persamaan 2.1, t_i menyatakan token ke-i yang dihasilkan oleh WordPiece. Token `[CLS]` ditempatkan pada awal sekuens. Setelah sekuens diproses oleh encoder BERT, representasi kontekstual pada posisi tersebut dapat digunakan oleh kepala klasifikasi sebagai representasi sekuens [27]. Token `[SEP]` digunakan sebagai pemisah antara dua sekuens dan sebagai penanda batas sekuens masukan [27]. Untuk masukan yang terdiri atas pasangan teks, bentuk konseptual sekuens BERT dapat ditulis sebagai berikut [27].

$$\texttt{[CLS]};t_1^A;\ldots;t_m^A;\texttt{[SEP]};t_1^B;\ldots;t_n^B;\texttt{[SEP]}, \tag{2.2}$$

Pada Persamaan 2.2, t_i^A menyatakan token pada segmen pertama dan t_i^B

menyatakan token pada segmen kedua. BERT membedakan kedua segmen tersebut melalui *segment embedding* yang menunjukkan apakah suatu token berasal dari segmen A atau segmen B [27].

Selain `[CLS]`, `[SEP]`, dan `[UNK]`, BERT menggunakan token khusus `[MASK]` dalam proses prapelatihan berbasis *masked language modeling*. Pada proses tersebut, sebagian posisi token dipilih untuk diprediksi berdasarkan konteks di sekitarnya [27]. Token `[MASK]` tidak ditambahkan secara rutin pada masukan tugas klasifikasi saat *fine-tuning* atau inferensi karena token tersebut digunakan sebagai bagian dari tujuan prapelatihan BERT [27].

Setelah proses segmentasi, setiap token dipetakan ke bilangan bulat berdasarkan indeks token tersebut dalam kosakata *tokenizer*. Rangkaian indeks tersebut disebut *input IDs* dan digunakan sebagai representasi numerik awal yang diberikan kepada model [42]. Nilai pada *input IDs* hanya menunjukkan identitas token dalam kosakata dan bukan nilai semantik atau ukuran kedekatan makna antartoken [42].

Selain *input IDs*, *input* BERT dapat memuat *token type IDs*. *Token type IDs*, yang juga disebut *segment IDs*, digunakan untuk membedakan keanggotaan token pada segmen masukan [27, 42]. Pada masukan satu teks, token-token berada pada segmen yang sama, sedangkan pada masukan pasangan teks, nilai segmen digunakan untuk membedakan token pada sekuens pertama dan sekuens kedua [27, 42].

Pemrosesan sejumlah sekuens dalam satu *batch* memerlukan panjang tensor yang seragam. Oleh karena itu, sekuens yang lebih pendek dapat ditambah dengan token `[PAD]` melalui proses *padding*, sedangkan sekuens yang melebihi panjang maksimum dapat dipotong melalui proses *truncation* [42]. Token `[PAD]` berfungsi sebagai pengisi untuk menyeragamkan panjang sekuens dan tidak diperlakukan sebagai bagian dari teks yang harus diperhatikan oleh model [42].

Posisi token hasil *padding* dibedakan menggunakan *attention mask*. Pada *input* BERT yang diproses menggunakan *tokenizer* Transformers, *attention mask* bernilai 1 pada posisi token yang diproses dan bernilai 0 pada posisi `[PAD]` [42]. *Attention mask* merupakan penanda masukan yang memberi tahu model posisi mana yang perlu diperhatikan dan bukan bobot perhatian yang dipelajari oleh model [42].

Secara ringkas, keluaran proses tokenisasi yang diberikan kepada BERT dapat dinyatakan seperti Persamaan 2.3.

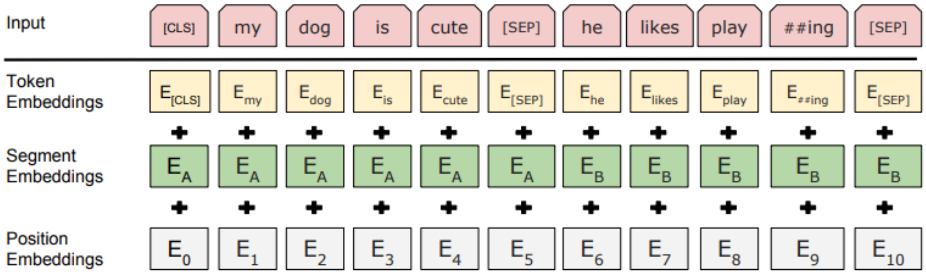
$$\mathscr{T}(x) = (\boldsymbol{i}, \boldsymbol{m}, \boldsymbol{s}), \tag{2.3}$$

dengan x menyatakan teks masukan, $\boldsymbol{i}$ menyatakan rangkaian *input IDs*, $\boldsymbol{m}$ menyatakan *attention mask*, dan $\boldsymbol{s}$ menyatakan *token type IDs* apabila diperlukan oleh model. Persamaan 2.3 merupakan notasi ringkas untuk menunjukkan komponen keluaran *tokenizer* dan bukan persamaan pembelajaran model.

Keluaran tokenisasi selanjutnya dipetakan ke representasi vektor melalui lapisan *embedding* BERT. BERT membentuk representasi masukan setiap token melalui penjumlahan *token embedding*, *position embedding*, dan *segment embedding* [27].

2.9.2 Representasi Input BERT

BERT membentuk representasi masukan melalui penjumlahan tiga jenis *embedding*, yaitu *token embedding*, *position embedding*, dan *segment embedding* [27], sebagaimana ditunjukkan pada Gambar 2.1.



Gambar 2.1. Visualisasi *embedding* pada BERT

Sumber: [27]

Token embedding menyatakan identitas token berdasarkan kosakata, *position embedding* menyatakan posisi token dalam sekuens, sedangkan *segment embedding* menyatakan segmen kalimat pada masukan berpasangan [27]. Representasi masukan untuk token ke-i dapat ditulis seperti Persamaan 2.4.

$$x_i = E_{tok}(t_i) + E_{pos}(i) + E_{seg}(s_i) \tag{2.4}$$

Pada Persamaan 2.4, x_i adalah representasi masukan token ke-i, t_i adalah token atau ID token ke-i, s_i adalah identitas segmen token ke-i, E_{tok} adalah matriks *token embedding*, E_{pos} adalah matriks *position embedding*, dan E_{seg} adalah matriks *segment embedding*. Penjumlahan ini menjadikan model memperoleh informasi identitas token, posisi token, dan segmen token secara bersamaan.

2.9.3 Self-Attention dan Multi-Head Attention

Pada *self-attention*, representasi masukan X diproyeksikan menjadi *query*, *key*, dan *value*. Proyeksi tersebut dapat ditulis seperti Persamaan 2.5.

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V \tag{2.5}$$

Pada Persamaan 2.5, X adalah matriks representasi masukan sekuens, W_Q, W_K, dan W_V adalah matriks bobot proyeksi, sedangkan Q, K, dan V masing-masing adalah matriks *query*, *key*, dan *value*. Proyeksi ini menjadi dasar bagi perhitungan skor perhatian antartoken [26].

Mekanisme *scaled dot-product attention* ditulis seperti Persamaan 2.6.

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.6}$$

Pada Persamaan 2.6, Q, K, dan V masing-masing merepresentasikan *query*, *key*, dan *value*, sedangkan d_k adalah dimensi *key*. Perkalian QK^T menghasilkan skor kesesuaian antartoken. Pembagian dengan $\sqrt{d_k}$ digunakan untuk menjaga skala skor, kemudian fungsi *softmax* mengubah skor tersebut menjadi bobot perhatian yang ternormalisasi [26].

Bobot perhatian untuk token ke-i terhadap token ke-j dapat ditulis sebagai a_{ij}. Berdasarkan bobot tersebut, *context vector* token ke-i merupakan penjumlahan berbobot terhadap vektor *value*, sebagaimana ditunjukkan pada Persamaan 2.7.

$$c_i = \sum_{j=1}^{n} a_{ij} v_j \tag{2.7}$$

Pada Persamaan 2.7, c_i adalah *context vector* untuk token ke-i, a_{ij} adalah bobot perhatian dari token ke-i terhadap token ke-j, v_j adalah vektor *value* token ke-j, dan n adalah panjang sekuens. Dengan demikian, representasi suatu token dibentuk dari kombinasi informasi token lain yang diberi bobot berbeda.

Transformer menggunakan *multi-head attention* agar model dapat mempelajari beberapa pola perhatian secara paralel. Bentuk ringkasnya ditunjukkan pada Persamaan 2.8.

$$\begin{aligned} \text{head}_m &= \text{Attention}(QW_m^Q, KW_m^K, VW_m^V), \\ \text{MultiHead}(Q,K,V) &= \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \end{aligned} \tag{2.8}$$

Pada Persamaan 2.8, head_m adalah kepala perhatian ke-m, h adalah jumlah kepala perhatian, W_m^Q, W_m^K, dan W_m^V adalah bobot proyeksi pada kepala ke-m, sedangkan W^O adalah bobot proyeksi keluaran. Setiap kepala dapat mempelajari pola hubungan token yang berbeda sebelum hasilnya digabungkan kembali [26].

2.9.4 Residual Connection, Layer Normalization, dan Feed-Forward Network

Selain *attention*, Transformer menggunakan *residual connection* dan *layer normalization* pada keluaran sublapisan. Bentuk konseptualnya dapat ditulis seperti Persamaan 2.9.

$$Y = \text{LayerNorm}(X + F(X)) \tag{2.9}$$

Pada Persamaan 2.9, X adalah masukan sublapisan, $F(X)$ adalah keluaran transformasi sublapisan seperti *attention* atau *feed-forward network*, dan Y adalah keluaran setelah penjumlahan residual dan normalisasi. *Residual connection* membantu mempertahankan informasi dari masukan sebelumnya, sedangkan *layer normalization* menstabilkan representasi pada lapisan model [26, 27].

Setiap lapisan Transformer juga memiliki *position-wise feed-forward network* (FFN) yang diterapkan pada setiap posisi token secara terpisah. Rumus FFN pada Transformer ditunjukkan pada Persamaan 2.10.

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2.10}$$

Pada Persamaan 2.10, x adalah representasi token pada satu posisi, W_1 dan W_2 adalah matriks bobot, sedangkan b_1 dan b_2 adalah bias. Persamaan tersebut mengikuti rumus FFN pada Transformer dengan aktivasi ReLU [26]. BERT tetap menggunakan struktur encoder Transformer, tetapi menggunakan aktivasi GELU pada lapisan transformasinya [27]. Oleh karena itu, bentuk konseptual FFN pada

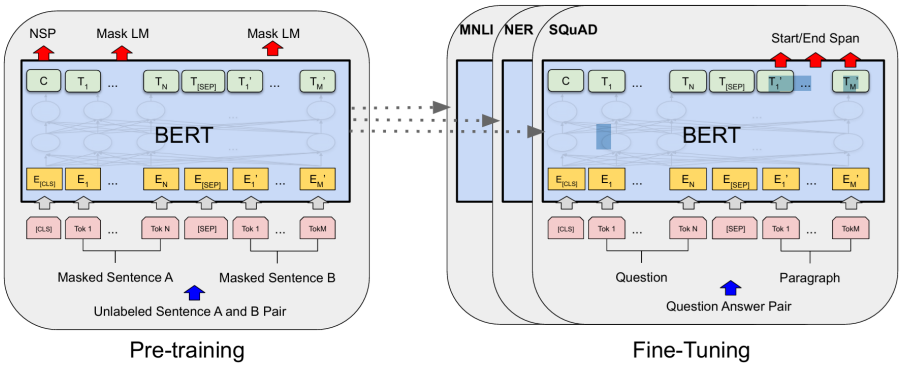
BERT dapat ditulis seperti Persamaan 2.11.

$$\text{FFN}_{BERT}(x) = \text{GELU}(xW_1 + b_1)W_2 + b_2 \tag{2.11}$$

Pada Persamaan 2.11, x adalah representasi token pada satu posisi, W_1 dan W_2 adalah matriks bobot, b_1 dan b_2 adalah bias, sedangkan $\text{GELU}(\cdot)$ adalah fungsi aktivasi yang digunakan pada BERT.

2.9.5 BERT untuk Klasifikasi Teks

BERT atau *Bidirectional Encoder Representations from Transformers* adalah model bahasa pra-latih berbasis encoder Transformer yang dirancang untuk mempelajari representasi bidireksional dari teks [27]. BERT dilatih menggunakan *masked language modeling* dan *next sentence prediction* pada korpus besar, kemudian dapat diadaptasi ke tugas hilir dengan menambahkan lapisan keluaran yang sesuai [27]. Untuk tugas klasifikasi sekuens atau *sequence classification*, representasi token `[CLS]` umum digunakan sebagai representasi sekuens yang masuk ke lapisan klasifikasi.



Gambar 2.2. Arsitektur model BERT yang terdiri dari dua tahap: *pre-training* dan *fine-tuning*

Sumber: [27]

Gambar 2.2 menunjukkan alur umum BERT dari tahap *pre-training* menuju *fine-tuning*. Pada tahap *pre-training*, BERT mempelajari representasi bahasa dari korpus besar melalui tugas *masked language modeling* dan *next sentence prediction*. Setelah itu, model dapat disesuaikan pada tugas hilir seperti klasifikasi teks dengan menambahkan lapisan keluaran dan melatih model pada data berlabel [27].

Pada klasifikasi multikelas, representasi [CLS] dapat diproyeksikan ke *logit* setiap kelas melalui lapisan klasifikasi. Bentuk umumnya ditunjukkan pada Persamaan 2.12.

$$z = h_{[CLS]} W_{cls} + b_{cls} \tag{2.12}$$

Pada Persamaan 2.12, $h_{[CLS]}$ adalah representasi token [CLS], W_{cls} adalah matriks bobot lapisan klasifikasi, b_{cls} adalah bias, dan z adalah vektor *logit* untuk seluruh kelas. Probabilitas prediksi kelas dihitung menggunakan *softmax*, seperti ditunjukkan pada Persamaan 2.13.

$$p_c = \frac{\exp(z_c)}{\sum_{j=1}^{C} \exp(z_j)} \tag{2.13}$$

Pada Persamaan 2.13, p_c adalah probabilitas prediksi kelas ke-c, z_c adalah *logit* kelas ke-c, C adalah jumlah kelas, dan z_j adalah *logit* untuk kelas ke-j. Kelas dengan nilai probabilitas terbesar dapat dipilih sebagai prediksi akhir pada tugas *single-label multiclass classification* [23, 27].

Penggunaan BERT pada klasifikasi emosi ulasan produk didasarkan pada kemampuannya membangun representasi kontekstual. Berbeda dari representasi berbasis kata statis, representasi kontekstual dapat berubah sesuai konteks kemunculan token dalam kalimat. Pada domain ulasan produk, kemampuan ini penting karena makna ekspresi emosi dapat bergantung pada kombinasi kata, negasi, konteks produk, dan pola ungkapan pengguna [27, 11].

2.10 IndoBERT

IndoBERT adalah model bahasa pra-latih untuk bahasa Indonesia yang diperkenalkan bersama IndoLEM sebagai *benchmark* NLP bahasa Indonesia [28]. IndoBERT dikembangkan untuk menyediakan representasi kontekstual yang lebih sesuai dengan bahasa Indonesia dibandingkan penggunaan model umum yang tidak secara khusus dilatih pada korpus Indonesia. IndoLEM mencakup beberapa tugas yang merepresentasikan aspek morfosintaksis, semantik, dan wacana dalam bahasa Indonesia [28].

Selain IndoLEM, IndoNLU menyediakan *benchmark* dan sumber daya untuk evaluasi pemahaman bahasa alami Indonesia [10]. Keberadaan IndoLEM dan

IndoNLU menunjukkan bahwa evaluasi NLP Indonesia membutuhkan sumber daya yang sesuai dengan bahasa dan domainnya. IndoBERT relevan untuk klasifikasi emosi ulasan *e-commerce* karena tugas yang dibahas menggunakan teks berbahasa Indonesia dengan variasi bahasa pengguna.

Dalam konteks klasifikasi teks Indonesia, IndoBERT dapat digunakan sebagai model dasar yang kemudian diadaptasi pada tugas hilir. Penggunaan model dasar yang sama pada beberapa strategi adaptasi memungkinkan perbandingan diarahkan pada cara adaptasi model, bukan pada perbedaan arsitektur dasar. Dengan demikian, perbandingan antara *full fine-tuning* dan LoRA dapat dipusatkan pada strategi pembaruan parameter.

2.11 Fine-Tuning dan Full Fine-Tuning

Fine-tuning adalah proses mengadaptasi model pra-latih ke tugas hilir dengan melatih model pada dataset tugas tersebut. Pada BERT, proses ini dapat dilakukan dengan menambahkan lapisan keluaran khusus untuk tugas seperti klasifikasi teks, kemudian memperbarui parameter model berdasarkan fungsi kerugian tugas hilir [27]. Strategi ini memanfaatkan pengetahuan bahasa yang telah dipelajari pada tahap pra-latih dan menyesuaikannya dengan label tugas tertentu.

Full fine-tuning adalah bentuk *fine-tuning* yang memperbarui seluruh atau hampir seluruh parameter model pra-latih selama pelatihan. Pendekatan ini sering menjadi *baseline* pembanding karena semua parameter model dapat beradaptasi terhadap tugas hilir [27, 43]. Namun, pembaruan seluruh parameter membutuhkan penyimpanan gradien dan status *optimizer* untuk banyak parameter, sehingga biaya memori dan waktu pelatihan dapat menjadi lebih tinggi dibandingkan pendekatan yang hanya melatih sebagian kecil parameter [13, 14].

Dalam penelitian empiris, *full fine-tuning* lazim digunakan sebagai *baseline* ketika metode adaptasi lain dievaluasi. Perbandingan terhadap *baseline* diperlukan agar performa metode hemat parameter tidak dibaca secara terpisah dari biaya adaptasinya. Dengan demikian, *baseline* tidak diposisikan sebagai metode yang harus selalu dikalahkan, tetapi sebagai acuan untuk membaca *trade-off* antara performa dan efisiensi [13, 19].

2.12 Parameter-Efficient Fine-Tuning dan LoRA

Parameter-Efficient Fine-Tuning (PEFT) adalah kelompok metode adaptasi model pra-latih yang bertujuan menyesuaikan model pada tugas hilir dengan melatih sebagian kecil parameter atau menambahkan modul adaptasi yang relatif kecil [14]. Berbeda dari *full fine-tuning*, PEFT tidak menjadikan pembaruan seluruh parameter sebagai kebutuhan utama. Pendekatan ini relevan ketika sumber daya komputasi terbatas, eksperimen membutuhkan banyak konfigurasi, atau model perlu diadaptasi untuk beberapa tugas [14, 19].

PEFT tidak selalu harus menghasilkan performa lebih tinggi daripada *full fine-tuning*. Nilai utamanya terletak pada kemungkinan memperoleh performa yang kompetitif dengan biaya adaptasi yang lebih rendah. Karena itu, evaluasi PEFT perlu mencakup performa klasifikasi dan efisiensi sumber daya. Nwaiwu mengevaluasi LoRA, IA^3, dan ReFT pada klasifikasi teks sumber daya rendah dengan memperhatikan akurasi, F1-score, jumlah parameter latih, dan penggunaan memori GPU [19].

LoRA merupakan salah satu metode PEFT. Posisi LoRA dalam PEFT perlu dijelaskan agar pembahasan tidak mencampuradukkan istilah umum dan metode spesifik. PEFT mengacu pada kelompok pendekatan adaptasi hemat parameter, sedangkan LoRA merujuk pada teknik tertentu yang menggunakan matriks adaptasi berperingkat rendah pada bobot model [13, 14].

2.12.1 Low-Rank Adaptation

Low-Rank Adaptation (LoRA) adalah metode PEFT yang mengadaptasi model pra-latih dengan membekukan bobot utama model dan menambahkan pembaruan berbasis dekomposisi *low-rank* pada lapisan tertentu [13]. Jika bobot pra-latih dinyatakan sebagai W_0, maka pembaruan bobot yang biasanya dipelajari pada *full fine-tuning* dinyatakan sebagai ΔW. LoRA membatasi pembaruan tersebut melalui dua matriks berperingkat rendah, sehingga ΔW direpresentasikan sebagai BA [13].

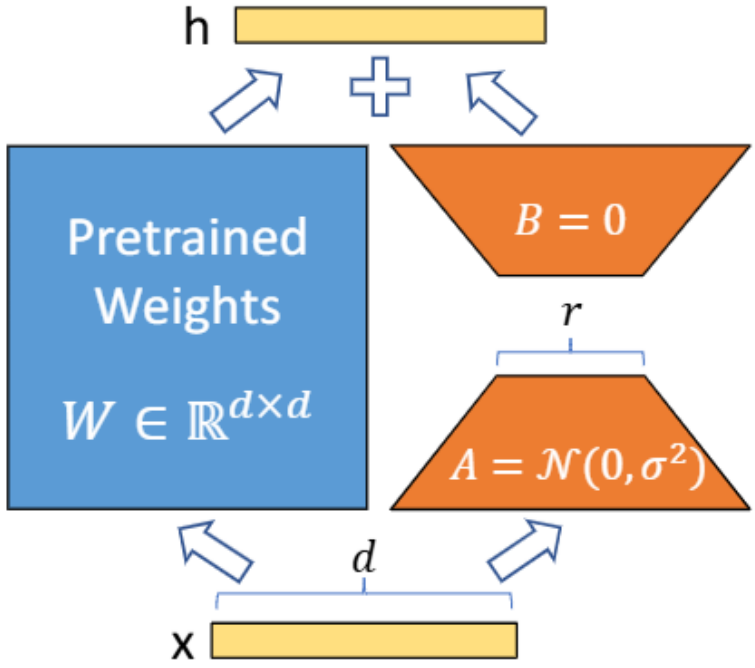
Secara umum, jika $W_0 \in \mathbb{R}^{d\times k}$, LoRA menggunakan $B \in \mathbb{R}^{d\times r}$ dan $A \in \mathbb{R}^{r\times k}$ dengan $r \ll \min(d,k)$. Nilai r disebut *rank* LoRA dan menentukan ukuran ruang adaptasi yang dilatih. Bobot efektif setelah pembaruan LoRA dapat ditulis seperti Persamaan 2.14.

$$W' = W_0 + \Delta W, \quad \Delta W = \frac{\alpha}{r} BA \tag{2.14}$$

Pada Persamaan 2.14, W' adalah bobot efektif setelah penambahan LoRA, W_0 adalah bobot pra-latih yang dibekukan, ΔW adalah pembaruan LoRA, A dan B adalah matriks adaptasi berperingkat rendah, α adalah faktor skala, dan r adalah *rank*. Bentuk *forward pass* LoRA pada suatu lapisan linear ditunjukkan pada Persamaan 2.15.

$$h = W_0 x + \Delta W x = W_0 x + \frac{\alpha}{r} BAx \tag{2.15}$$

Pada Persamaan 2.15, x adalah masukan lapisan, h adalah keluaran, $W_0 x$ adalah keluaran dari jalur bobot pra-latih, dan BAx adalah pembaruan dari jalur LoRA. Faktor α/r mengatur besarnya kontribusi pembaruan LoRA terhadap keluaran lapisan [13].



Gambar 2.3. Ilustrasi reparameterisasi pembaruan bobot pada metode LoRA
Sumber: [13]

Gambar 2.3 menunjukkan reparameterisasi pembaruan bobot pada LoRA. Jalur utama merepresentasikan bobot pra-latih W_0 yang dibekukan, sedangkan jalur

tambahan merepresentasikan pembaruan melalui matriks A dan B. Matriks A memproyeksikan representasi ke ruang berdimensi lebih kecil dengan ukuran *rank* r, sedangkan matriks B mengembalikannya ke dimensi keluaran. Selama pelatihan, hanya matriks adaptasi tersebut yang diperbarui, sedangkan bobot utama model tetap dibekukan [13].

Pada formulasi LoRA, salah satu matriks adaptasi diinisialisasi sehingga pembaruan awal bernilai nol. Akibatnya, pada awal pelatihan keluaran model tetap sama dengan keluaran model pra-latih sebelum adapter mempelajari pembaruan tugas hilir [13]. Setelah pelatihan, pembaruan LoRA dapat digabungkan kembali ke bobot utama untuk inferensi, sehingga LoRA tidak menambahkan latensi inferensi dibandingkan model yang telah digabungkan [13].

Dalam LoRA, *rank* menentukan kapasitas matriks adaptasi, *alpha* berperan sebagai faktor penskalaan, sedangkan *target module* menentukan bagian proyeksi model yang diberi adapter. Pada model berbasis Transformer/BERT, adapter LoRA dapat diterapkan secara konseptual pada lapisan linear yang membentuk proyeksi perhatian seperti `query`, `key`, dan `value`, maupun pada lapisan linear lain seperti `dense`. Jika adapter dipasang pada bobot proyeksi tertentu, proyeksi tersebut menggunakan bobot efektif W' sebagaimana Persamaan 2.14, misalnya proyeksi *query* atau *value* dapat dihitung dengan bobot yang telah memperoleh pembaruan LoRA, sedangkan modul yang tidak dipasangi adapter tetap memakai bobot pra-latihnya. Penyebutan modul tersebut tidak berarti bahwa semua modul selalu memberikan hasil terbaik. Pilihan *target modules* tetap perlu diuji secara empiris pada model dan dataset tertentu [26, 27, 13].

Perbandingan antara *full fine-tuning* dan LoRA perlu dibaca sebagai perbandingan performa sekaligus efisiensi. *Full fine-tuning* memperbarui seluruh parameter model sehingga kapasitas adaptasinya besar, tetapi biaya pelatihannya juga lebih tinggi. LoRA memperbarui parameter tambahan berukuran lebih kecil, sehingga jumlah *trainable parameters*, waktu pelatihan, dan *peak GPU memory* dapat lebih rendah pada konfigurasi tertentu. Oleh karena itu, analisis LoRA perlu menyajikan performa klasifikasi bersama indikator efisiensi agar *trade-off* antara performa dan penggunaan sumber daya dapat ditafsirkan secara jelas [13, 14, 19].

2.13 Pembagian Data, Validation Split, dan Test Split

Pada eksperimen klasifikasi, data berlabel umumnya dibagi menjadi data latih, validasi, dan uji. Data latih digunakan untuk memperbarui parameter model,

data validasi digunakan untuk memantau performa selama pengembangan dan memilih konfigurasi, sedangkan data uji digunakan untuk evaluasi akhir setelah konfigurasi dipilih [23, 44].

Validation split juga dapat digunakan untuk *early stopping*, yaitu menghentikan pelatihan ketika performa validasi menunjukkan tanda tidak lagi membaik atau mulai mengalami *overfitting* [44]. *Test split* tidak digunakan dalam proses pemilihan model agar evaluasi akhir tetap objektif dan tidak mengalami *test-set selection bias* [23, 44].

Pada klasifikasi emosi multikelas, pembagian data juga perlu mempertimbangkan distribusi label agar setiap *split* tetap merepresentasikan kelas yang dievaluasi. Jika distribusi kelas berubah terlalu jauh antarsplit, perbandingan performa dapat menjadi sulit ditafsirkan. Oleh sebab itu, proses pembagian data perlu dijelaskan bersama distribusi kelas dan audit potensi kebocoran data pada bagian metodologi [23, 45, 46].

2.14 Fungsi Kerugian Cross-Entropy dan Label Smoothing

Fungsi kerugian yang umum digunakan untuk klasifikasi multikelas adalah *cross-entropy loss*. Fungsi ini mengukur perbedaan antara distribusi label target dan distribusi probabilitas prediksi model [23]. Untuk satu sampel, *cross-entropy loss* dapat ditulis seperti Persamaan 2.16.

$$\mathscr{L}_{CE} = -\sum_{c=1}^{C} y_c \log(p_c) \tag{2.16}$$

Pada Persamaan 2.16, C adalah jumlah kelas, y_c adalah target untuk kelas ke-c, dan p_c adalah probabilitas prediksi model untuk kelas ke-c. Pada klasifikasi dengan label *one-hot*, nilai y_c bernilai 1 untuk kelas benar dan 0 untuk kelas lain, sehingga kerugian terutama dipengaruhi oleh probabilitas yang diberikan model kepada kelas benar [23].

Label smoothing adalah teknik regularisasi yang melunakkan distribusi target agar model tidak terlalu percaya diri terhadap satu kelas. *Label smoothing* dapat membantu generalisasi dan kalibrasi pada beberapa skenario, meskipun efeknya tetap bergantung pada konteks penggunaan [47]. Jika parameter *smoothing* dinyatakan sebagai ε, target kelas dapat diubah sebagaimana ditunjukkan pada Persamaan 2.17.

$$y_c^{LS} = (1-\varepsilon)y_c + \frac{\varepsilon}{C} \tag{2.17}$$

Pada Persamaan 2.17, y_c^{LS} adalah target setelah *label smoothing*, y_c adalah target *one-hot*, ε adalah nilai *smoothing*, dan C adalah jumlah kelas. Untuk kelas benar, target menjadi $1-\varepsilon+\varepsilon/C$, sedangkan untuk kelas selain kelas benar target menjadi ε/C. Fungsi kerugian dan regularisasi perlu dilaporkan secara eksplisit agar konfigurasi pelatihan dapat ditafsirkan bersama hasil performa model.

2.15 Metrik Evaluasi dan Analisis Kesalahan

Evaluasi klasifikasi multikelas dapat dilakukan menggunakan *confusion matrix*, *accuracy*, *precision*, *recall*, dan F1-score [25, 48]. *Confusion matrix* menunjukkan hubungan antara label aktual dan label prediksi. Baris dan kolom pada matriks tersebut dapat digunakan untuk melihat kelas yang sering benar diprediksi serta pasangan kelas yang sering tertukar [25].

Accuracy mengukur proporsi prediksi benar terhadap seluruh data. Rumus *accuracy* ditunjukkan pada Persamaan 2.18. Rumus ini digunakan untuk klasifikasi multikelas karena menghitung jumlah prediksi benar dari seluruh sampel [25].

$$Accuracy = \frac{\sum_{i=1}^{N} \mathbf{1}(y_i = \hat{y}_i)}{N} \tag{2.18}$$

Pada Persamaan 2.18, N adalah jumlah sampel, y_i adalah label aktual sampel ke-i, $\hat{y}_i$ adalah label prediksi sampel ke-i, dan $\mathbf{1}(\cdot)$ adalah fungsi indikator yang bernilai 1 jika kondisi di dalam kurung benar.

Untuk setiap kelas, *precision*, *recall*, dan F1-score dapat dihitung menggunakan skema satu-kelas-terhadap-kelas-lain. *Precision* mengukur proporsi prediksi positif yang benar pada kelas tertentu, sedangkan *recall* mengukur proporsi data aktual kelas tersebut yang berhasil ditemukan oleh model. F1-score merupakan rata-rata harmonik antara *precision* dan *recall* [25]. Rumus ketiga metrik tersebut ditunjukkan pada Persamaan 2.19, Persamaan 2.20, dan Persamaan 2.21.

$$Precision_c = \frac{TP_c}{TP_c + FP_c} \tag{2.19}$$

$$Recall_c = \frac{TP_c}{TP_c + FN_c} \tag{2.20}$$

$$F1_c = 2 \times \frac{Precision_c \times Recall_c}{Precision_c + Recall_c} \tag{2.21}$$

Pada Persamaan 2.19, Persamaan 2.20, dan Persamaan 2.21, TP_c, FP_c, dan FN_c masing-masing menunjukkan *true positive*, *false positive*, dan *false negative* untuk kelas ke-c. Dalam skema satu-kelas-terhadap-kelas-lain, TP_c adalah jumlah data kelas c yang diprediksi sebagai kelas c, FP_c adalah jumlah data bukan kelas c yang diprediksi sebagai kelas c, sedangkan FN_c adalah jumlah data kelas c yang diprediksi sebagai kelas lain. TN_c menunjukkan jumlah data bukan kelas c yang juga diprediksi bukan kelas c.

Macro average menghitung rata-rata metrik setiap kelas dengan bobot yang sama. Penggunaan metrik *macro* perlu dijelaskan secara eksplisit karena pemilihan metrik dapat memengaruhi interpretasi hasil dan peringkat sistem [48]. Persamaan 2.22, Persamaan 2.23, dan Persamaan 2.24 menunjukkan bentuk *macro-precision*, *macro-recall*, dan *macro-F1*.

$$Macro\text{-}Precision = \frac{1}{C}\sum_{c=1}^{C} Precision_c \tag{2.22}$$

$$Macro\text{-}Recall = \frac{1}{C}\sum_{c=1}^{C} Recall_c \tag{2.23}$$

$$Macro\text{-}F1 = \frac{1}{C}\sum_{c=1}^{C} F1_c \tag{2.24}$$

Pada Persamaan 2.22, Persamaan 2.23, dan Persamaan 2.24, C menunjukkan jumlah kelas. *Macro-F1* memberi kontribusi yang sama kepada setiap kelas terhadap skor akhir. Hal ini penting untuk mengevaluasi performa setiap kelas secara lebih seimbang, terutama ketika distribusi kelas tidak sepenuhnya identik [25, 48].

Weighted average menghitung rata-rata metrik setiap kelas dengan bobot

berdasarkan jumlah sampel atau *support* pada kelas tersebut. *Support* menunjukkan jumlah data aktual pada suatu kelas dalam data evaluasi. Pada *classification report*, *macro average* dihitung sebagai rata-rata tidak berbobot, sedangkan *weighted average* dihitung dengan mempertimbangkan *support* setiap label [49]. Rumus *weighted F1-score* ditunjukkan pada Persamaan 2.25.

$$Weighted\text{-}F1 = \sum_{c=1}^{C} \frac{n_c}{N} F1_c \tag{2.25}$$

Pada Persamaan 2.25, n_c adalah *support* kelas ke-c dan N adalah jumlah seluruh sampel evaluasi. Dalam klasifikasi emosi multikelas, *confusion matrix* melengkapi metrik agregat karena dapat menunjukkan pasangan emosi yang sering tertukar. Informasi ini penting untuk membaca pola kesalahan model, terutama pada kelas emosi yang secara semantik berdekatan.

Confusion matrix tidak hanya digunakan untuk menghitung jumlah prediksi benar dan salah, tetapi juga dapat menjadi dasar analisis kesalahan. Dalam klasifikasi emosi, analisis kesalahan membantu mengidentifikasi pasangan kelas yang sering tertukar, kelas dengan *precision* atau *recall* rendah pada *classification report*, serta contoh ulasan yang salah diprediksi. Analisis ini tidak diperlakukan sebagai metode pelatihan baru, melainkan sebagai pembacaan pola kesalahan agar keterbatasan model dapat dipahami secara lebih substantif daripada hanya melihat nilai *accuracy* atau F1-score [25, 48, 49].

2.16 Efisiensi Parameter dan Sumber Daya Pelatihan

Efisiensi adaptasi model dapat dilihat dari jumlah parameter yang dilatih, persentase parameter yang dilatih, waktu pelatihan, dan penggunaan memori GPU maksimum selama pelatihan. LoRA dirancang untuk mengurangi jumlah parameter yang dilatih dengan membekukan bobot utama model dan melatih matriks adaptasi berperingkat rendah [13]. PEFT secara umum juga dibahas sebagai pendekatan untuk menurunkan kebutuhan parameter dan sumber daya ketika mengadaptasi model besar atau model pra-latih ke tugas hilir [14].

Persentase *trainable parameters* dapat dihitung menggunakan Persamaan 2.26. Ukuran ini menunjukkan proporsi parameter yang benar-benar diperbarui selama pelatihan dibandingkan total parameter model [13, 14].

$$\text{Trainable Percentage} = \frac{\text{Trainable Parameters}}{\text{Total Parameters}} \times 100\% \tag{2.26}$$

Pada Persamaan 2.26, Trainable Parameters adalah jumlah parameter yang diperbarui selama pelatihan, sedangkan Total Parameters adalah seluruh parameter pada model yang dievaluasi.

Selain efisiensi parameter, efisiensi pelatihan dapat diamati melalui waktu pelatihan dan *peak GPU memory*. Waktu pelatihan menunjukkan durasi proses adaptasi model sampai pelatihan selesai, sedangkan *peak GPU memory* menunjukkan penggunaan memori GPU maksimum selama pelatihan. Nwaiwu menggunakan parameter latih dan penggunaan memori GPU sebagai bagian dari evaluasi PEFT pada klasifikasi teks sumber daya rendah, sehingga indikator serupa relevan untuk menilai efisiensi LoRA [19].

Efisiensi tidak ditafsirkan sebagai pengganti performa klasifikasi, tetapi sebagai dimensi evaluasi tambahan. Pada kajian PEFT, efisiensi perlu dibaca bersama performa klasifikasi. Metode adaptasi hemat parameter tidak harus menghasilkan skor yang lebih tinggi daripada *full fine-tuning*, nilai utamanya terletak pada kemampuan mempertahankan performa yang kompetitif dengan jumlah parameter latih atau kebutuhan sumber daya yang lebih rendah. Oleh sebab itu, laporan evaluasi perlu menyajikan performa dan efisiensi secara berdampingan agar kompromi antara keduanya dapat ditafsirkan secara jelas [13, 14, 19].