

BAB II

KAJIAN LITERATUR

2.1 *Natural Language Processing (NLP)*

Natural Language Processing (NLP) merupakan cabang dari *artificial intelligence* yang berfokus pada perancangan dan implementasi sistem serta algoritma yang mampu berinteraksi menggunakan bahasa manusia [19]. NLP telah menjadi bagian dari kehidupan sehari-hari karena telah banyak diterapkan dalam perangkat lunak komputer maupun pada ponsel pintar [20].

NLP dapat diterapkan dalam berbagai bidang, seperti penerjemahan mesin, deteksi email spam, ekstraksi informasi, peringkasan teks, dan sistem tanya jawab [21].

1. *Machine Translation: Machine Translation* bertujuan mengatasi hambatan bahasa dengan menerjemahkan teks dari satu bahasa ke bahasa lain tanpa menghilangkan makna, tata bahasa, dan konteks kalimat. Perkembangannya beralih dari pendekatan statistik ke model berbasis jaringan saraf dan *deep learning*, seperti yang digunakan Google sejak 2016. Kualitas terjemahan dievaluasi menggunakan metrik otomatis seperti BLEU, NIST, dan *Word Error Rate*, yang dirancang untuk mendekati penilaian manusia terhadap kelancaran dan kesesuaian makna.
2. *Text Categorization: Text categorization* adalah proses mengelompokkan dokumen teks ke dalam kategori yang telah ditentukan, seperti berita, dokumen resmi, atau email, guna meningkatkan efisiensi pengelolaan informasi. Aplikasi utamanya mencakup pengindeksan dokumen, pengelolaan tiket keluhan, dan penyaringan *email spam*. Sistem ini bekerja dengan menganalisis isi teks dan metadata menggunakan berbagai aturan atau model pembelajaran mesin untuk menentukan kategori yang paling sesuai.
3. *Spam Filtering: Spam filtering* merupakan penerapan khusus dari *text categorization* yang menggunakan teknik *machine learning* untuk membedakan *email spam* dan *non-spam* secara otomatis. Berbagai algoritma seperti *Naïve Bayes*, *Support Vector Machines*, *Decision Trees*, dan *Maximum Entropy* digunakan karena lebih efektif dibandingkan aturan manual. Model *Multivariate Bernoulli* dan *Multinomial* umum digunakan untuk merepresentasikan dokumen, dengan *Naïve Bayes* menjadi pilihan populer karena kesederhanaan dan performanya yang baik.
4. *Information Extraction: Information Extraction (IE)* berfokus pada pengambilan informasi penting seperti entitas, relasi, dan fakta dari teks tidak terstruktur untuk membangun representasi terstruktur. Teknik IE banyak digunakan dalam mesin pencari,

sistem periklanan, dan pengelolaan basis data, dengan dukungan metode seperti HMM, POS *tagging*, *stemming*, dan *word sense disambiguation*. Meskipun ekstraksi entitas relatif berhasil, pengambilan relasi antar entitas masih menjadi tantangan utama dalam pengembangan sistem IE.

5. *Summarization: Text summarization* bertujuan merangkum informasi penting dari satu atau banyak dokumen tanpa menghilangkan makna utama, sebagai solusi atas ledakan informasi digital. Pendekatan peringkasan dapat bersifat *single* atau *multi-document*, *generic* atau *query-focused*, serta *supervised* atau *unsupervised*. Berbagai model berbasis topik dan probabilistik dikembangkan untuk menghasilkan ringkasan yang informatif dan relevan, serta banyak dimanfaatkan dalam analisis sentimen dan pengambilan keputusan.
6. *Dialogue System: Dialogue system* memungkinkan interaksi alami antara manusia dan komputer, baik untuk memberikan informasi maupun menjalankan perintah tertentu. Sistem modern memanfaatkan pemahaman bahasa pada berbagai level, dari fonem hingga wacana, sehingga mampu mempertahankan konteks percakapan. Perkembangan ini melahirkan asisten virtual seperti *Siri*, *Google Assistant*, dan *Alexa* yang dapat berinteraksi secara natural dengan pengguna.
7. *Medicine*: Dalam bidang medis, NLP digunakan untuk mengekstraksi, menganalisis, dan merangkum informasi klinis dari teks bebas seperti rekam medis dan laporan kesehatan. Sistem seperti LSP-MLP, MEDLEE, dan RECIT membantu mengubah narasi medis menjadi data terstruktur untuk mendukung diagnosis, analisis obat, dan basis pengetahuan biomedis. Pengembangan NLP medis juga menekankan kemampuan multibahasa dan representasi pengetahuan yang independen dari bahasa.

NLP memiliki peran penting dalam pengolahan data berbasis bahasa alami, khususnya ketika tujuan analisis berfokus pada pemahaman opini dan sikap yang terkandung dalam teks [22]. Dalam konteks tersebut, NLP digunakan dalam analisis sentimen karena kemampuannya memproses teks tidak terstruktur untuk mengidentifikasi opini, emosi, dan sikap yang diekspresikan melalui bahasa alami [22]. Dalam konteks tersebut, analisis sentimen memanfaatkan teknik NLP untuk mengekstraksi dan menganalisis opini, sentimen, serta emosi manusia yang terdapat dalam data teks tidak terstruktur [23].

Dalam proses pengolahannya, NLP dilakukan melalui sejumlah tahapan yang saling berurutan dan saling mendukung, yaitu [24]:

1. *Analisis Leksikal (Lexical Analysis)*: Analisis leksikal merupakan tahap pertama dalam NLP yang berfungsi untuk memindai teks masukan dan memecahnya menjadi unit-unit

dasar yang bermakna (*lexeme* atau *token*), sehingga teks dapat diproses pada tahap selanjutnya.

2. Analisis Morfologis (*Morphological Analysis*): Analisis morfologis adalah tahap yang berfokus pada struktur kata dan proses pembentukan kata berdasarkan morfem, termasuk identifikasi kata dasar dan bentuk turunannya.
3. Analisis Sintaktik (*Syntactic Analysis / Parsing*): Analisis sintaktik merupakan tahap ketiga dalam NLP yang bertujuan memeriksa tata bahasa dan struktur kalimat, serta memastikan susunan kata sesuai dengan aturan gramatikal bahasa.
4. Analisis Semantik (*Semantic Analysis*): Analisis semantik berkaitan dengan pemahaman makna kata dan kalimat. Pada tahap ini, sistem NLP menafsirkan arti dari teks yang telah dianalisis secara struktural.
5. Analisis Pragmatik (*Pragmatic Analysis*): Analisis pragmatik merupakan tahap akhir NLP yang menangani interpretasi bahasa berdasarkan konteks penggunaan dan situasi nyata, sehingga makna yang dihasilkan menjadi lebih sesuai dengan maksud sebenarnya.

Perkembangan NLP mengalami perubahan signifikan dengan diperkenalkannya *foundation models* seperti BERT, GPT, dan T5, yang mendorong pergeseran paradigma dalam pengembangan dan penerapan model bahasa alami [25]. Model-model ini dilatih pada data berskala besar dan mampu menyelesaikan beragam tugas bahasa alami dengan kebutuhan pelatihan ulang yang minimal melalui pendekatan *transfer learning* dan *prompting* [25]. Inovasi tersebut kemudian dikembangkan lebih lanjut melalui GPT-4, yang mengintegrasikan teks dan gambar dalam satu sistem *multimodal*, sehingga memungkinkan pemrosesan bahasa yang lebih kontekstual dan mendekati cara manusia memahami makna [26]. Upaya peningkatan efisiensi juga dilakukan melalui pengembangan model terbuka seperti LLaMA, yang menawarkan performa tinggi dengan kebutuhan komputasi lebih rendah [27]. Di sisi lain, model PaLM memperluas kemampuan NLP melalui peningkatan kapasitas penalaran dalam skenario *few-shot* dan *zero-shot learning* [28]. Rangkaian kemajuan tersebut memperkuat posisi NLP tidak hanya sebagai teknologi pemrosesan bahasa, tetapi juga sebagai sarana untuk memahami opini, emosi, dan sikap yang tersirat dalam teks.

2.2 Text Mining

Text mining adalah proses untuk mengekstraksi informasi yang berguna dari kumpulan data teks tidak terstruktur [29]. Pengembangan *text mining* didorong oleh pertumbuhan pesat data tekstual digital, yang tidak dapat dianalisis secara efektif

menggunakan teknik *data mining* tradisional [30]. Dalam konteks tersebut, *text mining* mencakup teknik untuk mengekstraksi, menganalisis, dan menemukan informasi bermakna dari dokumen teks [31]. Secara lebih spesifik, *text mining* berfokus pada pengubahan teks bahasa alami menjadi bentuk terstruktur agar dapat diproses oleh mesin [30].

Text mining pada dasarnya dilakukan melalui serangkaian tahapan sistematis yang bertujuan mengubah data teks mentah menjadi informasi yang dapat dianalisis secara komputasional. Berikut adalah tahapannya [32]:

1. Pengumpulan data teks (*corpus collection*): Tahap awal *text mining* adalah membangun korpus, yaitu kumpulan dokumen teks bahasa alami yang menjadi dasar analisis selanjutnya.
2. Pra-pemrosesan teks (*text preprocessing*): Setelah korpus tersedia, teks harus melalui tahap pra-pemrosesan, yaitu proses mentransformasi teks sebelum analisis untuk menentukan unit analisis, menghilangkan informasi yang tidak relevan, serta meningkatkan kualitas representasi teks.
3. Ekstraksi fitur teks (*feature extraction*): Tahap berikutnya adalah ekstraksi fitur, yaitu proses pengubahan teks yang telah dilakukan *preprocessing* diubah menjadi representasi numerik menggunakan pendekatan *closed vocabulary* atau *open vocabulary*.
4. Transformasi dan pembobotan fitur: Fitur yang telah diekstraksi dapat ditransformasikan lebih lanjut, misalnya melalui pembobotan frekuensi atau pemodelan topik, untuk meningkatkan kualitas representasi dokumen sebelum analisis.
5. Analisis teks (*text analysis*): Fitur teks kemudian digunakan dalam tahap analisis, baik untuk keperluan visualisasi, *clustering*, maupun prediksi menggunakan pendekatan statistik atau *machine learning*.
6. Interpretasi hasil: Tahap akhir *text mining* adalah interpretasi hasil, yaitu memahami dan mengaitkan hasil analisis teks dengan konteks penelitian agar menghasilkan kesimpulan yang bermakna.

Tujuan utama *text mining* adalah mengekstraksi pengetahuan yang bernilai dari kumpulan data teks dalam jumlah besar [31]. Untuk mencapai tujuan tersebut, *text mining* bertujuan menemukan pola tersembunyi, tren, dan hubungan konseptual dalam data teks tidak terstruktur [31]. Hasil dari proses ini digunakan untuk mendukung pengambilan keputusan berbasis data melalui analisis otomatis dokumen teks [29]. Dalam penggunaannya, *text mining* mencakup tugas seperti *text classification*, *clustering*, *information extraction*, dan *text summarization* [29].

Selain tujuan dan penggunaannya, *text mining* juga memiliki keterkaitan erat dengan NLP, meskipun keduanya merupakan konsep yang tidak sepenuhnya identik [29]. *Text mining* berfokus pada ekstraksi pengetahuan dan pola dari data teks, sementara itu NLP berfokus pada pemrosesan dan pemahaman bahasa manusia secara komputasional [32,33]. Dalam konteks penerapannya, teknik-teknik NLP digunakan sebagai bagian dari proses *text mining* untuk memproses bahasa alami [29]. Dengan demikian, NLP berperan sebagai komponen pendukung dalam *text mining* [29].

2.3 Analisis Sentimen

Analisis sentimen merupakan subbidang dari *text mining* dan NLP yang berfokus pada identifikasi, ekstraksi, dan klasifikasi opini atau sikap subjektif yang diekspresikan dalam teks, dengan tujuan utama menentukan polaritas sentimen (positif, negatif, atau netral) [33].

Tujuan dari analisis sentimen adalah untuk mengembangkan alat otomatis yang mampu mengekstraksi informasi subjektif dari teks dalam bahasa alami, seperti opini dan sentimen, sehingga dapat menghasilkan pengetahuan yang terstruktur dan dapat ditindaklanjuti untuk digunakan baik oleh sistem pendukung keputusan maupun oleh pengambil keputusan [34].

Analisis sentimen diklasifikasikan menjadi tiga kategori utama, yaitu sentimen positif, sentimen negatif, dan sentimen netral [35]:

1. Sentimen Positif: Sentimen positif adalah opini atau sikap subjektif yang menunjukkan evaluasi, perasaan, atau penilaian yang menguntungkan terhadap suatu entitas. Sebuah teks dikategorikan sebagai sentimen positif apabila mengandung ekspresi yang merefleksikan kepuasan, kesukaan, atau apresiasi.
2. Sentimen Negatif: Sentimen negatif adalah opini atau sikap subjektif yang mengekspresikan ketidakpuasan, ketidaksukaan, kritik, atau evaluasi yang merugikan terhadap suatu entitas. Sentimen negatif umumnya memuat kata, frasa, atau struktur yang menunjukkan penilaian buruk atau perasaan tidak menyenangkan terhadap objek tertentu.
3. Sentimen Netral: Sentimen netral mengacu pada teks yang tidak secara jelas mengekspresikan sentimen positif maupun negatif, melainkan bersifat informatif, faktual, atau objektif. Sentimen netral muncul ketika suatu pernyataan tidak mengandung muatan evaluatif atau emosional yang kuat, sehingga tidak dapat dikategorikan sebagai opini positif atau negatif.

Dalam analisis sentimen, pembagian kelas sentimen dapat dilakukan dalam bentuk klasifikasi biner maupun klasifikasi tiga kelas, tergantung pada cara penanganan opini yang bersifat netral atau ambigu [36]. Klasifikasi sentimen biner membagi data hanya ke dalam dua kelas, yaitu sentimen positif dan sentimen negatif, dan sering digunakan pada penelitian yang berfokus pada polaritas sentimen yang jelas [36]. Namun, beberapa penelitian menunjukkan bahwa dalam data opini nyata terdapat ulasan dengan polaritas yang tidak pasti, sehingga pendekatan biner berpotensi mengalami kesalahan klasifikasi apabila sentimen netral dipaksakan ke dalam kelas positif atau negatif [37]. Untuk mengatasi keterbatasan tersebut, klasifikasi sentimen tiga kelas mempertahankan kategori netral agar pernyataan yang bersifat informatif atau ambigu tidak memengaruhi akurasi klasifikasi polaritas utama [37]. Pendekatan tiga kelas ini banyak diterapkan pada data ulasan daring dan media sosial, seperti pada penelitian analisis ulasan Yelp dan TikTok, yang secara eksplisit menggunakan label positif, negatif, dan netral untuk merepresentasikan variasi opini pengguna secara lebih realistis [38,39]. Dengan demikian, pembagian sentimen ke dalam tiga kelas memungkinkan pemodelan sentimen yang lebih sesuai dengan karakteristik data opini publik dibandingkan pendekatan biner pada kasus tertentu [39].

Analisis sentimen banyak digunakan saat ini seiring dengan meningkatnya volume data opini yang dihasilkan melalui media sosial dan platform digital, sehingga diperlukan metode otomatis untuk mengekstraksi dan memahami sikap serta pandangan publik secara efisien [40]. Keunggulan analisis sentimen terletak pada kemampuannya mengubah teks tidak terstruktur menjadi informasi terstruktur yang mendukung pengambilan keputusan lintas domain, dan pendekatan ini tetap relevan karena opini manusia terus berkembang serta masih menghadapi tantangan linguistik yang mendorong penelitian lanjutan [40].

2.4 Lexicon-Based

Dalam analisis sentimen, terdapat dua pendekatan utama yang digunakan untuk mengklasifikasikan polaritas teks, yaitu pendekatan berbasis *machine learning* dan pendekatan berbasis leksikon (*lexicon-based*) [35]. Pendekatan berbasis *machine learning* membangun model klasifikasi dengan melatih algoritma pada data berlabel untuk mempelajari pola sentimen secara otomatis [35]. Sebaliknya, pendekatan *lexicon-based* menggunakan kamus kata (*sentiment lexicon*) yang telah diberi nilai polaritas untuk menentukan orientasi sentimen suatu teks tanpa memerlukan data pelatihan berlabel [35]. Pendekatan *lexicon-based* bekerja dengan cara mencocokkan setiap kata dalam teks dengan entri pada kamus sentimen, kemudian mengakumulasi skor polaritas seluruh kata untuk

menghasilkan skor sentimen keseluruhan dari teks tersebut [41]. Keunggulan utama pendekatan ini terletak pada kemampuannya untuk diterapkan tanpa memerlukan data pelatihan dalam jumlah besar, sehingga cocok digunakan pada kondisi data berlabel yang terbatas [35].

Pembangunan *sentiment lexicon* dapat dilakukan melalui tiga metode utama, yaitu metode manual, metode berbasis kamus (*dictionary-based*), dan metode berbasis korpus (*corpus-based*) [35]. Metode manual melibatkan pakar bahasa atau peneliti yang secara langsung mengidentifikasi dan memberikan skor polaritas terhadap kata-kata opini, sehingga menghasilkan leksikon yang akurat namun membutuhkan waktu dan tenaga yang besar [35]. Metode berbasis kamus memanfaatkan sumber daya linguistik seperti *WordNet* atau tesaurus daring untuk memperluas daftar kata sentimen awal (*seed words*) secara otomatis melalui pencarian sinonim dan antonim [35]. Sementara itu, metode berbasis korpus mengekstraksi kata-kata sentimen baru dari koleksi teks berskala besar dengan memanfaatkan pola statistik ko-kemunculan (*co-occurrence*) atau pola sintaktik untuk menentukan polaritas kata berdasarkan konteks penggunaannya [35]. Dibandingkan metode berbasis kamus yang cenderung menghasilkan leksikon bersifat umum, metode berbasis korpus memiliki keunggulan dalam menghasilkan leksikon yang lebih spesifik terhadap domain tertentu [35].

Salah satu *sentiment lexicon* yang dikembangkan khusus untuk bahasa Indonesia adalah InSet (*Indonesian Sentiment Lexicon*), yang dibangun oleh Koto dan Rahmaningtyas (2017) dengan tujuan mengidentifikasi opini tertulis dan mengkategorikannya ke dalam opini positif atau negatif pada konteks *microblog* berbahasa Indonesia [42]. InSet Lexicon disusun menggunakan koleksi kata yang diekstraksi dari data *tweet* berbahasa Indonesia, kemudian setiap kata diberi skor secara manual oleh anotator manusia dengan rentang nilai dari -5 (sangat negatif) hingga +5 (sangat positif) [42]. Proses konstruksi InSet mengikuti pendekatan yang serupa dengan leksikon AFINN untuk bahasa Inggris, dengan menambahkan tahap *stemming* menggunakan pustaka Sastrawi dan perluasan melalui kumpulan sinonim (*synonym set*) untuk memperkaya cakupan kata [42]. Hasil akhir InSet terdiri dari 3.609 kata positif dan 6.609 kata negatif, dan berdasarkan evaluasi eksperimental, InSet menunjukkan performa yang lebih baik dibandingkan leksikon sentimen bahasa Indonesia lainnya yang digunakan sebagai *baseline* [42].

Dalam penerapannya, pendekatan *lexicon-based* menentukan label sentimen suatu teks berdasarkan akumulasi skor polaritas dari kata-kata yang terdeteksi dalam leksikon [43]. Setiap kata pada teks yang termasuk dalam *opinion words* di leksikon akan diambil bobotnya, kemudian seluruh bobot tersebut dijumlahkan untuk menghasilkan nilai *polarity*

score pada teks tersebut [43]. Penentuan kelas sentimen didasarkan pada hasil perhitungan *polarity score*: teks dengan *polarity score* bernilai positif dikategorikan sebagai sentimen positif, teks dengan *polarity score* bernilai negatif dikategorikan sebagai sentimen negatif, dan teks dengan *polarity score* sama dengan nol dikategorikan sebagai sentimen netral [43]. Pendekatan ini telah digunakan dalam berbagai penelitian analisis sentimen berbahasa Indonesia, termasuk analisis sentimen terhadap perkuliahan daring yang mengklasifikasikan 5.811 *tweet* dan memperoleh akurasi sebesar 79,2% [43]. Selain itu, dalam studi perbandingan metode pelabelan, pendekatan InSet Lexicon menghasilkan akurasi 89,7% yang lebih tinggi dibandingkan metode pelabelan berbasis *rating* pada klasifikasi sentimen ulasan aplikasi [44].

Meskipun pendekatan *lexicon-based* memiliki keunggulan dalam hal kesederhanaan dan tidak memerlukan data pelatihan berlabel, pendekatan ini juga memiliki keterbatasan [35]. Pendekatan *lexicon-based* tidak memperhitungkan konteks penggunaan kata dalam kalimat, sehingga rentan terhadap kesalahan klasifikasi pada kalimat yang mengandung sarkasme, negasi, atau ekspresi ambigu [35]. Selain itu, performa *lexicon-based* sangat bergantung pada kelengkapan dan kualitas kamus sentimen yang digunakan, karena kata-kata yang tidak tercakup dalam leksikon tidak akan berkontribusi pada perhitungan skor sentimen [41]. Oleh karena itu, beberapa penelitian mengombinasikan pendekatan *lexicon-based* dengan metode *machine learning* atau *deep learning* untuk meningkatkan akurasi klasifikasi sentimen secara keseluruhan [35].

2.5 Oversampling Berbasis Augmentasi Teks

Ketidakseimbangan data pada klasifikasi teks terjadi ketika jumlah data pada setiap kelas tidak terdistribusi secara proporsional [45]. Kondisi tersebut dapat menyebabkan data pada kelas minoritas lebih sering diklasifikasikan secara tidak tepat oleh model [45]. Untuk mengatasi permasalahan tersebut, salah satu pendekatan yang digunakan adalah *oversampling*, yaitu penambahan jumlah data pada kelas minoritas untuk menyeimbangkan *dataset* [45]. Pada data teks, penerapan *oversampling* dapat dilakukan melalui *text augmentation*, yaitu teknik pembentukan variasi teks baru dari data yang telah tersedia [45].

Text augmentation digunakan untuk meningkatkan keragaman data latih tanpa harus melakukan pengumpulan data baru secara langsung [46]. Teknik ini bekerja dengan membuat data tambahan melalui modifikasi terhadap teks asli, sehingga model memperoleh variasi pola bahasa yang lebih beragam selama proses pelatihan [46]. Dalam konteks klasifikasi teks, augmentasi dapat dilakukan pada level token atau kata, yaitu dengan

memodifikasi unsur kata dalam kalimat tanpa mengubah keseluruhan struktur *dataset* [46]. Salah satu metode augmentasi teks yang banyak digunakan pada tugas klasifikasi teks adalah *Easy Data Augmentation* (EDA) [47].

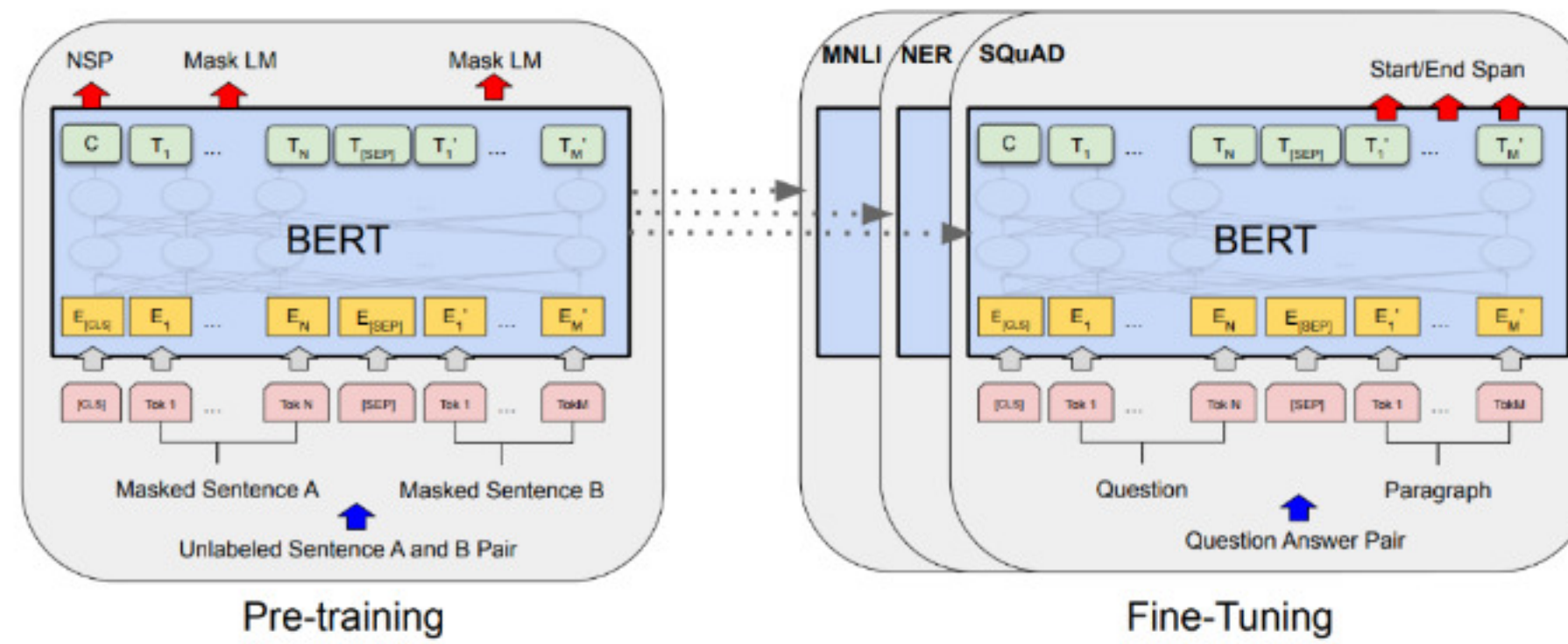
EDA merupakan metode augmentasi teks sederhana yang terdiri dari empat operasi utama, yaitu *synonym replacement*, *random insertion*, *random swap*, dan *random deletion* [47]. *Random swap* dilakukan dengan menukar posisi kata secara acak dalam sebuah kalimat untuk menghasilkan variasi susunan kata [47]. *Random deletion* dilakukan dengan menghapus kata secara acak dalam sebuah kalimat berdasarkan probabilitas tertentu [47]. Kedua teknik tersebut termasuk teknik augmentasi berbasis modifikasi kata yang dapat digunakan untuk memperbanyak variasi data teks pada proses pelatihan model klasifikasi [46].

Penggunaan EDA pada data tidak seimbang telah diterapkan dalam penelitian klasifikasi teks berbahasa Indonesia [48]. Penelitian Azizah dkk. menggunakan EDA untuk menangani data *imbalance* pada teks konsultasi kesehatan daring berbahasa Indonesia [48]. Teknik EDA yang digunakan dalam penelitian tersebut mencakup *random swap* dan *random deletion* sebagai bagian dari proses augmentasi teks [48]. Dengan demikian, *oversampling* berbasis augmentasi teks dapat dipahami sebagai pendekatan penambahan data teks minoritas melalui pembentukan variasi teks baru, bukan hanya melalui penggandaan data yang sama [45].

2.6 IndoBERT

IndoBERT merupakan model BERT berbahasa Indonesia yang dilatih menggunakan *dataset* Indo4B [49]. Model ini merupakan salah satu model BERT monolingual pertama untuk bahasa Indonesia yang dilatih sesuai dengan praktik terbaik yang berlaku dalam penelitian dan pengembangan model bahasa pada bidang NLP [50].

Pada penelitian yang dilakukan oleh Koto dkk., IndoBERT diperkenalkan sebagai model bahasa monolingual berbasis arsitektur transformer bergaya BERT yang dilatih menggunakan skema *masked language modeling* dengan konfigurasi standar *BERT-Base (uncased)*. Dalam penelitian tersebut, IndoBERT menggunakan kosakata WordPiece bahasa Indonesia berukuran 31.923 dan dilatih pada korpus berjumlah 220 juta kata yang bersumber dari Wikipedia Indonesia, artikel berita, serta korpus web Indonesia. Proses pelatihan menghasilkan lebih dari satu juta data latih dan dilakukan menggunakan infrastruktur GPU berskala besar hingga mencapai nilai *perplexity* yang sebanding dengan BERT-base berbahasa Inggris [50].



Gambar 2. 1 Arsitektur umum IndoBERT berbasis *encoder* BERT [51]

Secara arsitektural, IndoBERT mengadopsi model *Bidirectional Encoder Representations from Transformers* (BERT), yaitu model bahasa berbasis *encoder* transformer yang dirancang untuk mempelajari representasi bahasa secara kontekstual dan dua arah (*bidirectional*), sebagaimana ditunjukkan pada Gambar 2.1 [51]. Teks masukan pada IndoBERT terlebih dahulu diproses melalui tahap *tokenization* menggunakan skema WordPiece yang dikembangkan khusus untuk bahasa Indonesia, sehingga mampu menangani variasi morfologi dan kosakata lokal secara lebih efektif [50]. Setiap sekuens *input* pada IndoBERT diawali oleh token khusus [CLS] (*classification*) yang ditempatkan pada posisi pertama. Representasi akhir token [CLS] digunakan sebagai ringkasan seluruh sekuens untuk keperluan tugas klasifikasi [51]. Setiap token kemudian direpresentasikan dalam bentuk *embedding* vektor yang merupakan hasil penjumlahan dari *token embedding*, *positional embedding*, dan *segment embedding*, sebagaimana diterapkan pada arsitektur BERT standar [51]. *Embedding* tersebut selanjutnya diproses oleh sejumlah lapisan *encoder transformer* yang masing-masing terdiri atas mekanisme *multi-head self-attention* dan *position-wise feed-forward network*, yang memungkinkan model menangkap hubungan semantik dan sintaktik antarkata dalam satu kalimat maupun lintas kalimat secara menyeluruh [51]. Pada tahap *pre-training*, IndoBERT dilatih menggunakan skema *Masked Language Modeling* (MLM) dengan cara menyembunyikan sebagian token pada teks masukan secara acak dan model ditugaskan untuk memprediksi token asli berdasarkan konteks sekitarnya, sehingga menghasilkan representasi bahasa yang kaya akan informasi konteks [50]. Representasi hasil pra-pelatihan ini selanjutnya dapat disesuaikan (*fine-tuning*) untuk berbagai tugas pemrosesan bahasa alami berbahasa Indonesia, seperti klasifikasi teks, analisis sentimen, dan pemahaman teks, dengan menambahkan lapisan tugas-spesifik di atas *encoder* IndoBERT [51,52].

Dalam penelitian IndoBERT yang dilakukan oleh Koto dkk., model bahasa pra-latih IndoBERT digunakan sebagai *encoder* utama yang kemudian dilakukan proses *fine-tuning* untuk berbagai tugas pemrosesan bahasa alami berbahasa Indonesia dengan menambahkan lapisan keluaran yang sederhana sesuai dengan tugas yang dikerjakan. Proses *fine-tuning* ini tidak mengubah arsitektur dasar IndoBERT, melainkan menyesuaikan bobot model melalui sejumlah parameter pelatihan (*training hyperparameters*) yang telah tersedia dalam kerangka kerja pembelajaran mendalam. Parameter-parameter tersebut berperan penting dalam mengendalikan proses optimisasi, stabilitas pembelajaran, serta kemampuan generalisasi model terhadap data baru. Berikut adalah parameter pelatihan IndoBERT [50]:

1. *Batch size*: *Batch size* adalah parameter yang menentukan jumlah data latih yang diproses secara bersamaan dalam satu iterasi pelatihan sebelum model melakukan pembaruan bobot. Parameter ini memengaruhi stabilitas gradien, penggunaan memori, dan efisiensi proses pelatihan.
2. *Learning rate*: *Learning rate* adalah parameter yang mengatur besar langkah pembaruan bobot model pada setiap iterasi pelatihan. Nilai *learning rate* menentukan seberapa cepat model menyesuaikan bobotnya terhadap kesalahan prediksi selama proses pembelajaran.
3. *Optimizer* (AdamW): *Optimizer* merupakan komponen yang berfungsi untuk mengatur mekanisme pembaruan bobot model berdasarkan nilai gradien yang dihasilkan selama pelatihan. *Optimizer* menentukan bagaimana dan kapan bobot model diperbarui agar proses pembelajaran berjalan optimal.
4. *Epoch*: *Epoch* mengacu pada satu siklus penuh pelatihan saat seluruh data latih telah diproses oleh model satu kali. Jumlah *epoch* atau langkah pelatihan digunakan untuk menentukan durasi proses pembelajaran model.
5. *Max sequence length*: *Max sequence length* adalah parameter yang menentukan jumlah maksimum token yang dapat diproses oleh model dalam satu masukan. Parameter ini digunakan untuk membatasi panjang teks agar sesuai dengan kapasitas model dan efisiensi komputasi.
6. *Early stopping*: *Early stopping* adalah mekanisme untuk menghentikan pelatihan ketika performa model tidak lagi meningkat, guna mencegah overfitting dan pemborosan sumber daya komputasi.

Mekanisme *self-attention* pada arsitektur *transformer* yang digunakan IndoBERT dihitung menggunakan *scaled dot-product attention* [51]. Pada proses ini, setiap token diproyeksikan secara linear menjadi tiga vektor utama: *Query* (Q) yang merepresentasikan informasi token saat ini, *Key* (K) sebagai representasi dari seluruh token, dan *Value* (V) yang

berisi nilai informasinya [17]. Hasil interaksi *dot product* antara Q dan K kemudian diskalakan dimensi *key* (d_k) agar nilainya tidak terlalu besar sebelum diproses oleh fungsi *softmax* [17]. Fungsi *softmax* yang digunakan pada mekanisme *self-attention* merupakan fungsi aktivasi yang mengubah vektor berisi nilai riil menjadi distribusi probabilitas, sehingga seluruh elemen bernilai di antara 0 dan 1 dengan jumlah total sama dengan 1 [19]. Fungsi ini dirumuskan pada Persamaan (1), yaitu Z_i menyatakan nilai elemen ke- i pada vektor *input*, dan penyebut merupakan jumlah eksponensial dari seluruh elemen vektor.

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^n e^{z_j}} \dots\dots\dots (1)$$

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V \dots\dots\dots (2)$$

Untuk memungkinkan model menangkap berbagai aspek hubungan antar token secara bersamaan, *transformer* menggunakan mekanisme *Multi-Head Attention* yang menjalankan beberapa fungsi *attention* secara parallel [17]. *Output* dari seluruh *attention head* kemudian digabungkan melalui operasi *concatenation* dan diproyeksikan kembali menggunakan matriks bobot W^O , sebagaimana dirumuskan pada Persamaan (3).

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_i, \dots, \text{head}_h)W^O \dots\dots\dots (3)$$

Setelah melewati sub-lapisan *multi-head attention*, setiap representasi token diproses lebih lanjut oleh *feed-forward network* (FFN) yang diterapkan secara independen pada setiap posisi token [51]. FFN terdiri atas dua transformasi linear yang dipisahkan oleh fungsi aktivasi non-linear, yaitu perluasan dimensi dari *hidden size* (768) menjadi *intermediate size* (3072) dan proyeksi kembali menjadi 768 dimensi. Perumusan FFN disajikan pada Persamaan 4, yaitu $W_1 \in \mathbb{R}^{768 \times 3072}$ dan $W_2 \in \mathbb{R}^{3072 \times 768}$ merupakan matriks bobot, sedangkan b_1 dan b_2 merupakan vektor bias.

$$\text{FFN}(x) = \text{GELU}(xW_1 + b_1)W_2 + b_2 \dots\dots\dots (4)$$

Fungsi aktivasi *Gaussian Error Linear Unit* (GELU) digunakan pada FFN sebagai pengganti ReLU [51]. GELU memberikan transisi yang lebih halus dibandingkan ReLU karena bobot suatu *input* ditentukan secara probabilistik berdasarkan distribusi normal, sehingga model dapat menangkap pola non-linear yang lebih kompleks [19]. GELU dirumuskan pada Persamaan 5.

$$\text{GELU}(x) = x \cdot \phi(x) \dots\dots\dots (5)$$

Pada Persamaan 5, $\phi(x)$ menyatakan fungsi distribusi kumulatif dari distribusi normal standar.

Setiap sub-lapisan dalam *encoder transformer*, baik *multi-head attention* maupun *feed-forward network*, dilengkapi dengan *residual connection* yang diikuti oleh *layer normalization* [51]. *Residual connection* bertujuan memudahkan aliran gradien selama proses pelatihan pada jaringan dalam dengan menambahkan *input* sub-lapisan kembali ke *outputnya*, sedangkan *layer normalization* berfungsi menstabilkan distribusi aktivasi antar dimensi untuk mempercepat konvergensi. Kombinasi kedua operasi ini dirumuskan pada Persamaan 6.

$$x_{out} = \text{LayerNorm}(x + \text{Sublayer}(x)) \dots\dots\dots (6)$$

Pada Persamaan 6, x merupakan *input* sub-lapisan dan $\text{Sublayer}(x)$ merupakan keluaran dari *multi-head attention* atau FFN.

Layer normalization menormalisasi aktivasi pada setiap token secara independen berdasarkan rata-rata dan variansi dari seluruh dimensi vektor representasi token tersebut [51]. Perhitungan rata-rata (μ) dan variansi (σ^2) dilakukan terhadap n dimensi dari vektor *input* x , seperti yang ditunjukkan pada Persamaan 7 dan Persamaan 8.

$$\mu = \frac{1}{n} \sum_{i=1}^n x_i \dots\dots\dots (7)$$

$$\sigma^2 = \frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2 \dots\dots\dots (8)$$

Nilai yang telah dinormalisasi kemudian diskalakan kembali menggunakan dua parameter yang dapat dipelajari selama pelatihan, yaitu γ (*gain*) dan β (*bias*). Perumusan akhir *layer normalization* disajikan pada Persamaan 9, yaitu ε merupakan konstanta kecil (1×10^{-12}) yang ditambahkan untuk mencegah pembagian dengan nol.

$$\text{LayerNorm}(x) = \frac{x - \mu}{\sqrt{\sigma^2 + \varepsilon}} \cdot \gamma + \beta \dots\dots\dots (9)$$

Representasi *input* pada IndoBERT dibentuk dari penjumlahan tiga jenis *embedding*, yaitu *token embedding*, *segment embedding*, dan *position embedding* [51]. Proses ini ditunjukkan pada Persamaan 10. Pada Persamaan 3, E_{token} merepresentasikan identitas

$$E_{input} = E_{token} + E_{segment} + E_{position} \dots\dots\dots (10)$$

leksikal token berdasarkan kosakata *WordPiece*, $E_{segment}$ menunjukkan segmen kalimat (A atau B), dan $E_{position}$ menyandikan informasi posisi urutan token dalam sekuens [51]. IndoBERT menggunakan *learned positional embeddings* yang dipelajari selama proses *pre-training*, sehingga vektor posisi disesuaikan secara optimal berdasarkan data pelatihan [51].

IndoBERT memiliki beberapa versi yang dikembangkan dengan perbedaan ukuran dan kompleksitas model guna menyesuaikan kebutuhan performa dan efisiensi komputasi, yaitu [52]:

1. **IndoBERTBASE:** IndoBERTBASE merupakan varian standar dari IndoBERT yang menggunakan konfigurasi *BERT-Base (uncased)*. Model ini terdiri atas 12 *transformer layers*, 12 *self-attention heads* pada setiap *layer*, serta *hidden size* sebesar 768, dengan total parameter sekitar 124,5 juta parameter. Varian ini dirancang untuk memberikan keseimbangan antara performa dan efisiensi komputasi, sehingga banyak digunakan sebagai model dasar dalam berbagai tugas NLP berbahasa Indonesia, termasuk klasifikasi teks dan analisis sentimen.
2. **IndoBERTLARGE:** IndoBERTLARGE merupakan varian IndoBERT dengan kapasitas model yang lebih besar yang mengacu pada konfigurasi *BERT-Large*. Model ini umumnya memiliki 24 *transformer layers*, 16 *self-attention heads*, serta *hidden size* sebesar 1024, dengan jumlah parameter yang secara signifikan lebih besar dibandingkan IndoBERTBASE. Peningkatan kapasitas model ini memungkinkan IndoBERTLARGE untuk menangkap representasi semantik yang lebih kompleks, namun memerlukan sumber daya komputasi dan waktu pelatihan yang lebih besar.
3. **IndoBERT-liteBASE:** IndoBERT-liteBASE dikembangkan sebagai versi ringan dari IndoBERTBASE dengan tujuan meningkatkan efisiensi komputasi. Varian ini memiliki jumlah parameter yang lebih kecil dibandingkan IndoBERTBASE, sehingga membutuhkan memori dan waktu *inference* yang lebih rendah. Meskipun demikian, IndoBERT-liteBASE tetap mempertahankan kemampuan pemodelan konteks bahasa Indonesia dan dapat digunakan secara efektif pada berbagai tugas NLP dengan keterbatasan sumber daya komputasi.
4. **IndoBERT-liteLARGE:** IndoBERT-liteLARGE merupakan varian yang mengombinasikan pendekatan *lightweight model* dengan kapasitas yang relatif lebih besar dibandingkan IndoBERT-liteBASE. Model ini dirancang untuk memberikan performa yang lebih baik daripada versi *lite* lainnya, namun tetap lebih efisien dibandingkan IndoBERTLARGE. Oleh karena itu, varian ini dapat dipandang sebagai kompromi antara akurasi model dan efisiensi komputasi.

Dalam penelitian Isa dkk., pendekatan klasifikasi teks tradisional yang digunakan sebagai pembanding meliputi metode ekstraksi fitur *Term Frequency–Inverse Document Frequency* (TF-IDF) yang dikombinasikan dengan algoritma *Support Vector Machine* (SVM) dan *Naïve Bayes*. Metode-metode ini mengandalkan representasi statistik berbasis frekuensi kemunculan kata tanpa mempertimbangkan konteks semantik secara mendalam, sehingga meskipun mampu memberikan performa yang cukup baik, hasil klasifikasinya

masih berada di bawah model *transformer* seperti IndoBERT yang memanfaatkan mekanisme *self-attention* untuk memahami konteks teks secara menyeluruh [49].

2.7 BiGRU

Bidirectional Gated Recurrent Unit (BiGRU) merupakan pengembangan dari *Gated Recurrent Unit* (GRU), yaitu salah satu varian *Recurrent Neural Network* (RNN) yang dirancang untuk menyederhanakan struktur *Long Short-Term Memory* (LSTM) [53]. GRU hanya menggunakan dua gerbang utama, yaitu *reset gate* dan *update gate*, sehingga mampu mengurangi kompleksitas komputasi tanpa mengorbankan kemampuan dalam menangkap dependensi temporal [53].

Mekanisme gerbang pada GRU memanfaatkan dua fungsi aktivasi non-linear, yaitu *sigmoid* dan *tanh* [53]. Fungsi *sigmoid* (σ) digunakan pada *update gate* dan *reset gate* untuk menghasilkan nilai antara 0 dan 1 yang merepresentasikan proporsi informasi yang diteruskan atau disaring, sedangkan fungsi *tanh* digunakan pada pembentukan kandidat *hidden state* untuk menghasilkan nilai dalam rentang -1 hingga 1 [53]. Kedua fungsi ini dirumuskan pada Persamaan 11 dan Persamaan 12.

$$\sigma(x) = \frac{1}{1+e^{-x}} \dots\dots\dots (11)$$

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \dots\dots\dots (12)$$

Secara matematis, mekanisme gerbang pada GRU dapat dirumuskan sebagai berikut. *Update gate* (z_t) mengontrol sejauh mana *hidden state* sebelumnya diteruskan ke langkah waktu berikutnya [53]. Nilai *update gate* diperoleh melalui fungsi aktivasi *sigmoid* terhadap kombinasi linear dari *input* saat ini (x_t) dan *hidden state* sebelumnya (h_{t-1}), dengan W_z sebagai matriks bobot dan b_z sebagai bias [53]. Perumusan *update gate* disajikan pada Persamaan (13).

$$z_t = \sigma(W_z[h_{t-1}, x_t] + b_z) \dots\dots\dots (13)$$

Reset gate (r_t) berperan dalam mengontrol seberapa besar informasi dari *hidden state* sebelumnya yang relevan untuk pembentukan kandidat *hidden state* baru. Perhitungannya menggunakan fungsi *sigmoid* dengan matriks bobot W_r dan bias b_r [53], sebagaimana ditunjukkan pada Persamaan (14).

$$r_t = \sigma(W_r[h_{t-1}, x_t] + b_r) \dots\dots\dots (14)$$

Berdasarkan nilai *reset gate* yang diperoleh, kandidat *hidden state* baru ($\tilde{h}_t$) dibentuk menggunakan fungsi aktivasi *tanh* terhadap *input* saat ini dan informasi *hidden state* sebelumnya yang telah disaring melalui perkalian elemen per elemen ($\odot$) dengan *reset gate*.

W_h merupakan matriks bobot dan b_h adalah bias [53]. Perhitungan kandidat *hidden state* baru dirumuskan pada Persamaan (15).

$$\tilde{h}_t = \tanh(W_h[r_t \odot h_{t-1}, x_t] + b_h) \dots\dots\dots (15)$$

Hidden state akhir (h_t) diperoleh dengan menggabungkan *hidden state* sebelumnya dan kandidat *hidden state* baru, dengan proporsi yang ditentukan oleh *update gate*. Semakin besar nilai z_t , semakin besar porsi *hidden state* sebelumnya yang dipertahankan, sedangkan jika nilainya kecil maka kandidat *hidden state* baru akan lebih mendominasi [53]. Perumusan *hidden state* akhir disajikan pada Persamaan (16).

$$h_t = (1 - z_t)h_{t-1} + z_t\tilde{h}_t \dots\dots\dots (16)$$

Pengembangan lebih lanjut dari GRU menghasilkan BiGRU, yang menggabungkan dua jaringan GRU dengan arah pemrosesan maju (*forward*) dan mundur (*backward*) untuk memanfaatkan informasi dari konteks masa lalu dan masa depan secara simultan [53]. Pendekatan dua arah ini memungkinkan setiap representasi keadaan tersembunyi pada BiGRU dipengaruhi oleh keseluruhan urutan data, sehingga menghasilkan pemodelan sekuens yang lebih komprehensif dibandingkan model satu arah [54].

Secara arsitektural, BiGRU terdiri dari dua jaringan GRU yang bekerja secara paralel untuk memproses urutan data dari dua arah yang berlawanan, yaitu arah maju dan arah mundur, sehingga memungkinkan model memanfaatkan informasi dari konteks sebelumnya dan sesudahnya secara bersamaan [53]. Pada proses ini, setiap elemen dalam sekuens dipetakan ke dalam representasi keadaan tersembunyi yang dihasilkan dari penggabungan keluaran GRU dua arah, sehingga informasi temporal dari keseluruhan urutan dapat direpresentasikan secara lebih menyeluruh [54]. Pendekatan pemrosesan dua arah ini memungkinkan BiGRU menangkap hubungan sekuensial yang lebih kaya dibandingkan model satu arah, khususnya dalam pemodelan teks yang bergantung pada konteks global dan urutan kata [55]. Oleh karena itu, BiGRU banyak digunakan pada pemodelan data sekuens kompleks karena mampu meningkatkan kualitas ekstraksi fitur temporal dengan struktur yang relatif efisien dan stabil [54].

Pada BiGRU, *forward* GRU menghasilkan *hidden state* maju ($\vec{h}_t$) berdasarkan *input* saat ini dan *hidden state* pada langkah waktu sebelumnya, sedangkan *backward* GRU menghasilkan *hidden state* mundur ($\overleftarrow{h}_t$) berdasarkan *input* saat ini dan *hidden state* pada langkah waktu sesudahnya. Fungsi $\text{GRU}(\cdot)$ merepresentasikan transformasi non-linear melalui mekanisme *gate* [54,56]. Perumusan *hidden state* arah maju disajikan pada Persamaan (17) dan *hidden state* arah mundur disajikan pada Persamaan (18).

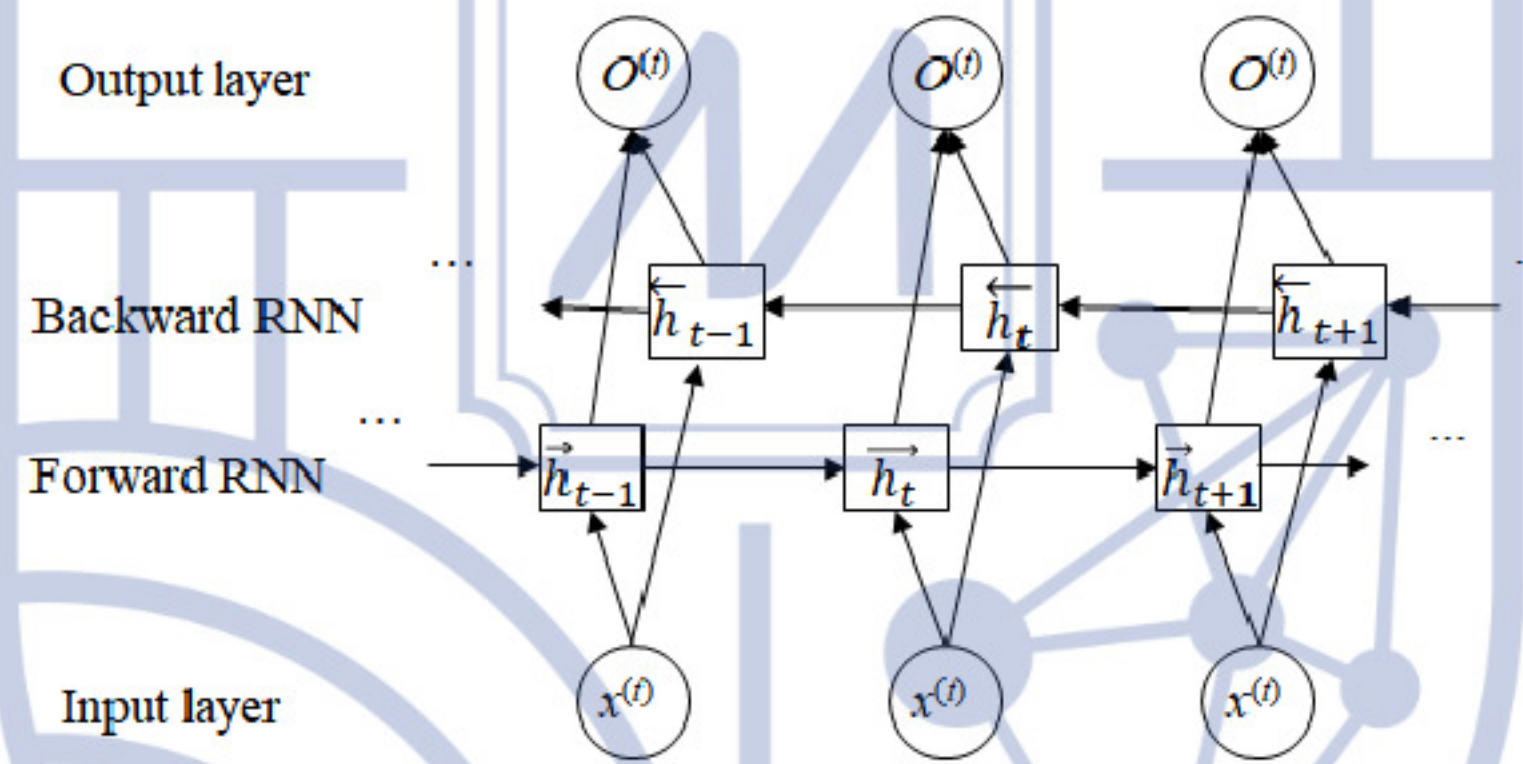
$$\vec{h}_t = GRU(x_t, \vec{h}_{t-1}) \dots\dots\dots (17)$$

$$\overleftarrow{h}_t = GRU(x_t, \overleftarrow{h}_{t+1}) \dots\dots\dots (18)$$

Representasi akhir setiap token diperoleh melalui konkatenasi *hidden state* dari kedua arah, dengan simbol $[\cdot]$ menunjukkan operasi konkatenasi. Dengan konfigurasi *hidden size* 256 per arah, *output* BiGRU untuk setiap token berdimensi 512 (256 + 256) [55]. Operasi konkatenasi ini dirumuskan pada Persamaan (19).

$$h_t = [\vec{h}_t; \overleftarrow{h}_t] \dots\dots\dots (19)$$

Cara kerja BiGRU didasarkan pada pemrosesan sekuens menggunakan dua jaringan GRU yang bekerja secara paralel dalam arah maju (*forward*) dan arah mundur (*backward*) untuk memodelkan konteks sekuens secara menyeluruh seperti pada Gambar 2.2 [56]. Pada arah maju, GRU memproses token berdasarkan urutan sebelumnya, sedangkan pada arah mundur GRU memproses token berdasarkan urutan sesudahnya, sehingga informasi historis dan prospektif dapat dimanfaatkan secara simultan [57]. Untuk setiap token *input* x_i , GRU



Gambar 2. 2 Struktur BiGRU [56]

menghasilkan representasi keadaan tersembunyi (*hidden state*) yang diperbarui berdasarkan *input* saat ini dan keadaan tersembunyi pada langkah waktu sebelumnya atau sesudahnya, tergantung arah pemrosesan [58]. Representasi keadaan tersembunyi dari arah maju dan arah mundur kemudian digabungkan menggunakan operasi konkatenasi untuk membentuk representasi kontekstual yang lebih kaya bagi setiap token dalam sekuens [57]. Dengan mekanisme pemrosesan dua arah tersebut, setiap token dalam sekuens dipengaruhi oleh keseluruhan konteks urutan, yang memungkinkan BiGRU menangkap dependensi sekuensial secara lebih efektif dibandingkan model GRU satu arah [58].

Dalam tugas klasifikasi teks, BiGRU berfungsi sebagai lapisan pemodelan sekuens yang mengekstraksi representasi fitur kontekstual dua arah dari urutan kata, yang selanjutnya digunakan sebagai masukan ke lapisan klasifikasi untuk menentukan label kelas [15]. Lapisan klasifikasi pada umumnya terdiri atas *dropout* sebagai mekanisme regularisasi, *fully*

connected layer sebagai transformasi linear menuju dimensi jumlah kelas, dan fungsi *softmax* untuk menghasilkan distribusi probabilitas antar kelas [59]. *Dropout* merupakan teknik regularisasi yang menonaktifkan sebagian neuron secara acak pada saat pelatihan untuk mencegah *overfitting* [60]. Neuron yang dinonaktifkan tidak berkontribusi pada *forward pass* maupun pembaruan bobot, sehingga model tidak terlalu bergantung pada fitur tertentu, sedangkan pada saat *inference* seluruh neuron digunakan secara penuh. Setelah *dropout*, vektor representasi kalimat dilewatkan melalui *fully connected layer* yang memetakan vektor tersebut ke dimensi jumlah kelas. Operasi ini dirumuskan pada Persamaan 20, yaitu h merupakan vektor representasi kalimat, W merupakan matriks bobot, dan b merupakan vektor bias. Keluaran dari operasi ini disebut *logit*, yaitu skor mentah yang belum dinormalisasi untuk setiap kelas.

$$z = hW + b \dots\dots\dots (20)$$

Nilai *logit* kemudian diubah menjadi distribusi probabilitas menggunakan fungsi *softmax*, sehingga diperoleh probabilitas untuk masing-masing kelas. Kelas dengan nilai probabilitas tertinggi dipilih sebagai hasil prediksi akhir [59].

Pada teks pendek, BiGRU efektif dalam meningkatkan kualitas fitur karena mampu memanfaatkan konteks sebelum dan sesudah token secara simultan, sehingga menghasilkan representasi yang lebih diskriminatif untuk proses klasifikasi [15]. Selain digunakan sebagai model klasifikasi mandiri, BiGRU juga sering dikombinasikan dengan *embedding* berbasis *transformer* seperti BERT yang hasilnya diproses lebih lanjut oleh BiGRU untuk memodelkan dependensi sekuensial sebelum dilakukan klasifikasi akhir [60].

2.8 Hybrid Model

Hybrid model merupakan pendekatan pemodelan yang menggabungkan dua atau lebih metode pembelajaran untuk memanfaatkan keunggulan masing-masing metode dalam satu sistem terpadu [61]. Pendekatan penggabungan ini diperkenalkan sebagai solusi untuk menutupi keterbatasan yang dimiliki oleh metode individual ketika digunakan secara terpisah [61]. Studi survei menunjukkan bahwa pengembangan *hybrid deep learning model* didorong oleh kenyataan bahwa setiap model *deep learning* memiliki kekuatan dan kelemahan yang berbeda dalam proses pembelajaran [62]. Oleh karena itu, *hybrid model* dirancang untuk mengompensasi kelemahan tersebut dengan menggabungkan beberapa pendekatan dalam satu kerangka pemodelan [62]. Selain itu, *hybrid model* tidak dibatasi pada kombinasi arsitektur tertentu, melainkan pada prinsip integrasi berbagai metode pembelajaran untuk meningkatkan performa sistem [62].

Prinsip kerja *hybrid model* didasarkan pada penggabungan beberapa pendekatan pembelajaran yang memiliki fungsi dan karakteristik berbeda dalam satu sistem [63]. *Hybrid model* dapat mengintegrasikan teknik deep learning dengan metode pembelajaran mesin tradisional untuk memanfaatkan kekuatan representasi dan kesederhanaan model secara bersamaan [63]. Pendekatan ini memungkinkan sistem mengekstraksi fitur yang lebih kaya dibandingkan penggunaan satu model tunggal [62]. Selain itu, *hybrid model* dirancang untuk memanfaatkan keunggulan setiap komponen model dalam alur pemrosesan yang saling melengkapi [63]. Dengan prinsip tersebut, *hybrid model* mampu menghasilkan performa yang lebih stabil dalam pemrosesan bahasa alami [62].

Hybrid model telah banyak digunakan dalam penelitian NLP, khususnya pada tugas klasifikasi teks dan analisis sentimen [61]. Penelitian empiris menunjukkan bahwa *hybrid model* mampu mencapai akurasi yang lebih tinggi dibandingkan model deep learning tunggal pada berbagai jenis *dataset* [61]. Tan dkk. menunjukkan bahwa pendekatan *ensemble-hybrid* dapat mengungguli metode *state-of-the-art* dalam tugas analisis sentimen [64]. Selain itu, *hybrid model* juga diterapkan dalam konteks multibahasa untuk meningkatkan kemampuan generalisasi model [65]. Penggunaan *hybrid model* juga meluas pada aplikasi NLP yang melibatkan data media sosial dan teks tidak terstruktur [66].

Penggunaan *hybrid model* dilatarbelakangi oleh fakta bahwa setiap pendekatan pembelajaran memiliki keterbatasan ketika diterapkan pada data teks yang kompleks [61]. Model deep learning tunggal sering kali tidak mampu menangkap seluruh karakteristik sintaktik dan semantik teks secara optimal [62]. Kondisi ini mendorong peneliti untuk mengombinasikan beberapa teknik pembelajaran agar dapat meningkatkan kualitas representasi teks [62]. Raj dkk. menegaskan bahwa penggabungan metode pembelajaran mesin dan NLP dilakukan untuk meningkatkan kinerja sistem dibandingkan pendekatan tunggal. Dengan demikian, *hybrid model* dikembangkan sebagai solusi praktis untuk mengatasi keterbatasan model tunggal dalam tugas NLP [66].

Hybrid model memiliki relevansi tinggi dalam pengembangan sistem NLP modern karena mampu meningkatkan representasi teks dan kinerja klasifikasi [67]. Model tunggal sering mengalami keterbatasan dalam menangani variasi dan kompleksitas data teks [67]. Oleh karena itu, *hybrid model* dikembangkan untuk menghasilkan representasi teks yang lebih baik melalui penggabungan beberapa pendekatan pembelajaran [67]. Tinjauan komprehensif menunjukkan bahwa *hybrid model* merupakan salah satu arah penelitian utama dalam NLP berbasis deep learning saat ini [68]. Dengan demikian, *hybrid model*

dipandang sebagai pendekatan strategis dan relevan untuk pengembangan sistem NLP yang lebih efektif dan adaptif [63].

2.9 Confusion Matrix

Pengujian model klasifikasi merupakan proses untuk menilai performa suatu model klasifikasi [69]. Pengujian dilakukan dengan menggunakan data uji yang kelas sebenarnya telah diketahui [69]. Evaluasi performa model dilakukan setelah model menghasilkan hasil klasifikasi terhadap data uji [59]. Melalui proses ini, performa model dinilai berdasarkan hasil klasifikasi terhadap data uji [70]. Salah satu metode yang digunakan untuk menguji kinerja model klasifikasi adalah *confusion matrix* [69].

Confusion matrix digunakan untuk mengevaluasi hasil prediksi model terhadap data uji yang memiliki label sebenarnya [69]. *Confusion matrix* merupakan tabel berbentuk matriks yang menunjukkan hubungan antara kelas aktual dan kelas hasil prediksi model [69]. *Confusion matrix* digunakan untuk mengevaluasi hasil prediksi model terhadap data uji yang memiliki label sebenarnya [71]. Metode ini banyak diterapkan pada penelitian klasifikasi teks berbasis *machine learning* dan *deep learning* [59]. Pada permasalahan klasifikasi dengan n kelas, *confusion matrix* memiliki n baris dan n kolom [69]. *Confusion matrix* dibentuk dengan membandingkan kelas aktual dan kelas hasil prediksi untuk setiap data uji [70]. Pada klasifikasi biner, *confusion matrix* menghasilkan empat kemungkinan hasil klasifikasi yaitu, *true positive* (TP), *true negative* (TN), *false positive* (FP), dan *false negative* (FN) [69]. *True Positive* (TP) merupakan kondisi ketika model memprediksi suatu data ke dalam kelas positif dan hasil prediksi tersebut sesuai dengan kelas aktualnya, *True Negative* (TN) adalah kondisi ketika model memprediksi data sebagai kelas negatif dan prediksi tersebut sesuai dengan kelas aktual, *False Positive* (FP) terjadi ketika model memprediksi data sebagai kelas positif, namun pada kenyataannya data tersebut termasuk ke dalam kelas negatif, *False Negative* (FN) merupakan kondisi ketika model memprediksi data sebagai kelas negatif, padahal kelas aktualnya adalah positif [69].

Confusion matrix dibentuk dengan membandingkan kelas aktual dan kelas hasil prediksi untuk setiap data uji [70]. Setiap hasil perbandingan antara kelas aktual dan kelas prediksi dicatat ke dalam sel matriks yang sesuai [70]. Pada klasifikasi biner, *confusion matrix* disusun dalam bentuk matriks berukuran 2×2 [69]. Baris pada *confusion matrix* merepresentasikan kelas aktual, sedangkan kolom merepresentasikan kelas hasil prediksi [69]. Nilai TP dan TN terletak pada diagonal utama matriks karena menunjukkan prediksi yang benar [69]. Sebaliknya, nilai FP dan FN berada pada sel non-diagonal karena

menunjukkan hasil prediksi yang salah [69]. *Confusion matrix* menyimpan hasil dasar klasifikasi berupa *true positive*, *true negative*, *false positive*, dan *false negative*, yang digunakan untuk membentuk berbagai metrik evaluasi seperti accuracy, precision, recall, dan F1-score [70,71].

1. *Accuracy*: *Accuracy* merupakan metrik evaluasi yang digunakan untuk mengukur proporsi prediksi yang benar terhadap seluruh prediksi yang dihasilkan oleh model. Nilai accuracy dihitung berdasarkan jumlah *true positive* dan *true negative* dibandingkan dengan total seluruh hasil klasifikasi yang tercatat dalam *confusion matrix*.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \dots\dots\dots (21)$$

2. *Precision*: *Precision* digunakan untuk mengukur ketepatan prediksi positif yang dihasilkan oleh model. Metrik ini menunjukkan proporsi data yang diprediksi sebagai kelas positif dan benar-benar termasuk ke dalam kelas positif, yang dihitung dari nilai *true positive* dan *false positive* pada *confusion matrix*.

$$Precision = \frac{TP}{TP+FP} \dots\dots\dots (22)$$

3. *Recall*: *Recall* merupakan metrik yang digunakan untuk mengukur kemampuan model dalam mendeteksi seluruh data yang benar-benar termasuk ke dalam kelas positif. *Recall* dihitung berdasarkan perbandingan antara *true positive* dan jumlah keseluruhan data positif, yaitu *true positive* dan *false negative*, yang tersimpan dalam *confusion matrix*.

$$Recall = \frac{TP}{TP+FN} \dots\dots\dots (23)$$

4. *F1-score*: *F1-score* merupakan metrik evaluasi yang menggabungkan *precision* dan *recall* dalam satu nilai harmonik. Metrik ini digunakan untuk memberikan keseimbangan antara *precision* dan *recall*, terutama pada *dataset* dengan distribusi kelas yang tidak seimbang. *F1-score* didefinisikan pada tingkat kelas dan dibentuk dari nilai *precision* dan *recall* kelas tersebut, yang keduanya diturunkan dari komponen *confusion matrix*, sehingga memberikan representasi performa model yang lebih stabil dibandingkan *accuracy* saja.

$$F1_{class} = \frac{2 \times Precision_{class} \times Recall_{class}}{Precision_{class} + Recall_{class}} \dots\dots\dots (24)$$

Pada klasifikasi dengan tiga kelas, *confusion matrix* diperluas menjadi matriks berukuran 3×3, yaitu setiap kelas diperlakukan secara bergantian sebagai kelas positif sementara dua kelas lainnya dianggap sebagai kelas negatif [72]. Dengan pendekatan ini, nilai TP suatu kelas merupakan elemen diagonal matriks pada perpotongan baris dan kolom kelas tersebut, nilai TN merupakan penjumlahan empat sel yang tersisa setelah baris dan

kolom kelas tersebut dihilangkan, nilai FP diperoleh dari penjumlahan sel-sel pada kolom kelas tersebut di luar nilai TP, dan FN diperoleh dari penjumlahan sel-sel pada baris kelas tersebut di luar nilai TP [72]. Akurasi keseluruhan model pada klasifikasi tiga kelas dihitung berdasarkan penjumlahan elemen diagonal matriks dibagi total sampel uji [72]:

$$Accuracy = \frac{TP_A + TP_B + TP_C}{N} \dots\dots\dots (25)$$

Untuk menghasilkan metrik evaluasi tunggal yang merepresentasikan performa model secara keseluruhan, digunakan pendekatan *macro-average* yang menghitung rata-rata aritmatika *precision*, *recall*, dan *F1-score* dari ketiga kelas tanpa mempertimbangkan jumlah sampel per kelas, sehingga setiap kelas diperlakukan dengan bobot yang sama [72].

$$Precision_{macro} = \frac{Precision_A + Precision_B + Precision_C}{3} \dots\dots\dots (26)$$

$$Recall_{macro} = \frac{Recall_A + Recall_B + Recall_C}{3} \dots\dots\dots (27)$$

$$F1_{macro} = \frac{2 \times Precision_{macro} \times Recall_{macro}}{Precision_{macro} + Recall_{macro}} \dots\dots\dots (28)$$

Selain *macro-average*, pendekatan *weighted-average* juga digunakan dengan memperhitungkan proporsi jumlah sampel aktual setiap kelas terhadap total sampel, sehingga lebih representatif pada kondisi distribusi kelas yang tidak seimbang [72]:

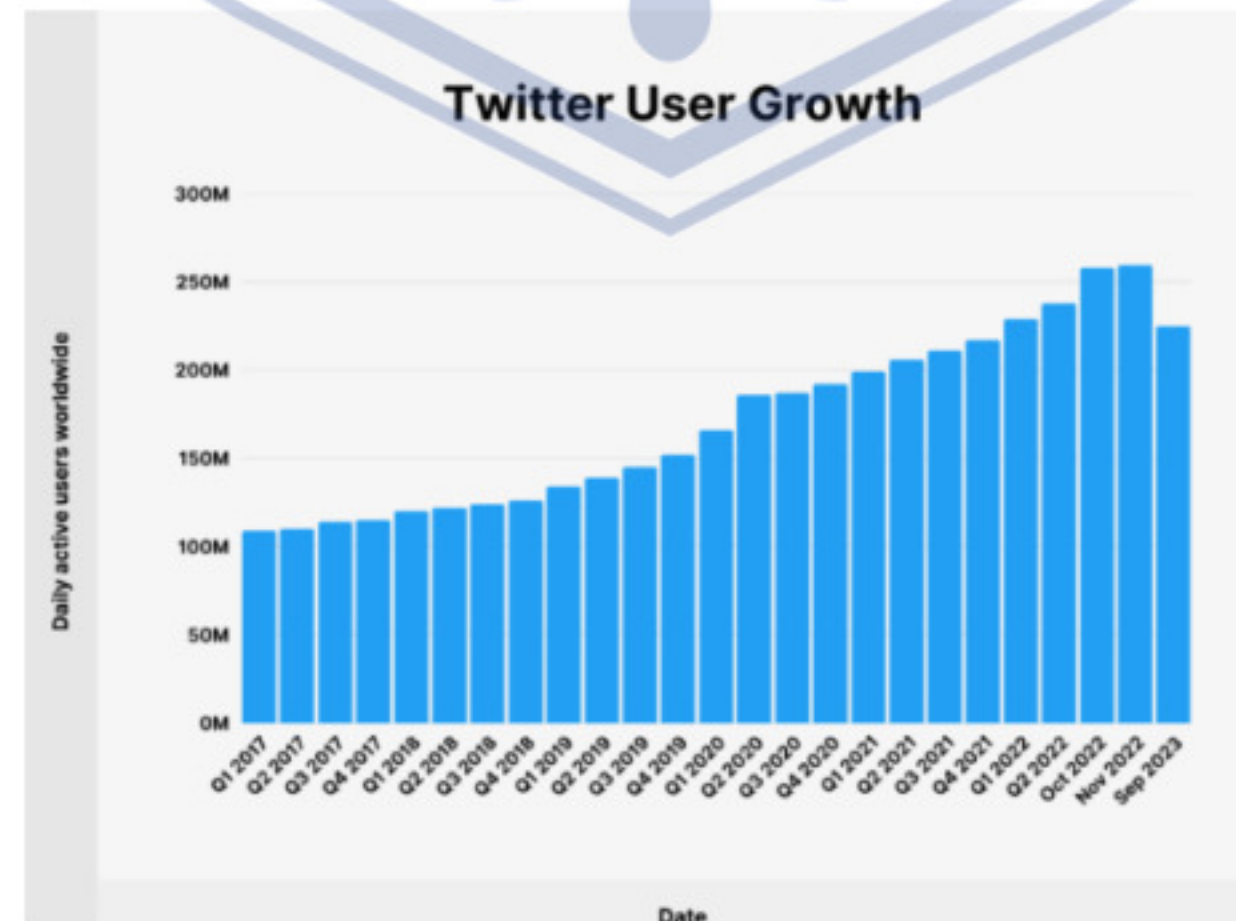
$$Precision_{weighted} = \frac{N_A \times Precision_A + N_B \times Precision_B + N_C \times Precision_C}{N} \dots\dots\dots (29)$$

$$Recall_{weighted} = \frac{N_A \times Recall_A + N_B \times Recall_B + N_C \times Recall_C}{N} \dots\dots\dots (30)$$

$$F1_{weighted} = \frac{2 \times Precision_{weighted} \times Recall_{weighted}}{Precision_{weighted} + Recall_{weighted}} \dots\dots\dots (31)$$

2.10 X

X atau yang sebelumnya dikenal sebagai Twitter adalah layanan *microblogging* populer yang memungkinkan pengguna untuk berbagi, mengirim, dan menginterpretasikan konten secara *real-time*, singkat dan mudah [73]. X memungkinkan pengguna untuk melihat



Gambar 2. 3 Pengguna Aktif Twitter 2017-2023 [76]

tweets tanpa harus mendaftar akun [74]. Pengguna dapat berkomunikasi dengan publik dengan menggunakan simbol *at* (@) untuk menandai percakapan dengan seseorang dan dapat menyertakan simbol *hashtag* (#) untuk membahas topik tertentu [75].

Pertumbuhan jumlah pengguna aktif harian X secara global dari tahun 2017 hingga 2023 menunjukkan tren peningkatan yang cenderung stabil, dari 109 juta pengguna pada Q1 2017 hingga mencapai puncaknya sebesar 259,4 juta pengguna pada November 2022, sebelum mengalami penurunan menjadi 245 juta pada September 2023 seperti pada Gambar 2.3 [76]. Berdasarkan data statistik dari Statista, pada awal tahun 2024 X memiliki sekitar 429 juta pengguna di seluruh dunia, dengan lebih dari 106 juta pengguna berasal dari Amerika Serikat, serta rata-rata 67 juta pengguna aktif yang masuk (*logged-in*) di wilayah Uni Eropa per Juli 2024. Platform ini juga menunjukkan ketimpangan demografis, yang menunjukkan pengguna laki-laki mendominasi hampir 61 persen dari total basis pengguna [77]. Berdasarkan survei global yang dilakukan pada tahun 2025, hampir setengah pengguna X (sebelumnya Twitter) memperhatikan berita dari media konvensional dan jurnalis, sementara proporsi yang relatif sama juga terlibat dengan konten dari kreator dan figur publik seperti selebritas dan *influencer*, yang menunjukkan adanya keseimbangan perhatian antara media tradisional dan suara individu di platform tersebut [78].

2.11 Makan Bergizi Gratis

Program Makan Bergizi Gratis (MBG) merupakan kebijakan intervensi gizi yang dirancang oleh pemerintah untuk menyediakan asupan makanan sehat dan seimbang secara gratis kepada kelompok usia sekolah, mulai dari pra sekolah hingga sekolah menengah, termasuk santri di pesantren [7]. Program ini menekankan pemenuhan kebutuhan gizi harian sebagai bagian dari strategi pembangunan sumber daya manusia (SDM) jangka panjang [7].

Berdasarkan Peraturan Presiden Republik Indonesia Nomor 115 Tahun 2025, Program Makan Bergizi Gratis diselenggarakan sebagai upaya pemenuhan gizi dalam rangka membangun generasi yang sehat, cerdas, dan produktif guna mendukung pertumbuhan ekonomi, pengentasan kemiskinan, serta pengembangan sumber daya manusia yang berkualitas. Penyelenggaraan program ini menekankan pentingnya penguatan ekosistem pendukung pemberian makan bergizi melalui tata kelola yang terarah, terintegrasi, dan berkelanjutan. Oleh karena itu, pengaturan mengenai tata kelola penyelenggaraan Program Makan Bergizi Gratis diperlukan untuk memberikan arah kebijakan, landasan pelaksanaan, serta kepastian hukum agar program dapat dilaksanakan secara efektif dan akuntabel [79].

Seiring dengan implementasinya sebagai kebijakan strategis nasional, Program Makan Bergizi Gratis (MBG) juga memunculkan sejumlah dinamika dan kontroversi dalam pelaksanaannya. Berdasarkan hasil survei yang dikutip oleh ANTARA, MBG dinilai memberikan manfaat nyata, khususnya dalam meringankan beban ekonomi keluarga serta menciptakan peluang kerja baru di tingkat lokal [80]. Survei tersebut menunjukkan bahwa 37,1% responden memahami tujuan utama MBG sebagai upaya memperbaiki kondisi gizi anak-anak Indonesia, yang sekaligus dipandang sebagai investasi bagi masa depan generasi mendatang [80]. Selain itu, 25,7% responden menilai bahwa penyediaan makanan bergizi di sekolah melalui MBG berpotensi mengurangi beban keluarga dalam memenuhi kebutuhan gizi anak sehari-hari [80].

Meskipun demikian, di tengah apresiasi publik tersebut, pelaksanaan Program Makan Bergizi Gratis juga menghadapi berbagai tantangan serius, terutama terkait aspek keamanan pangan. Jaringan Pemantau Pendidikan Indonesia (JPPI) mencatat sebanyak 1.242 kasus keracunan makanan yang berkaitan dengan MBG selama Januari 2026 [81]. Secara kumulatif, jumlah korban keracunan yang dilaporkan sejak awal pelaksanaan program pada tahun 2025 hingga awal 2026 mencapai 21.254 orang [81]. Temuan tersebut sejalan dengan laporan Badan Gizi Nasional (BGN) yang menunjukkan bahwa kasus keracunan pangan yang berkaitan dengan MBG menyumbang hingga 46% dari total kejadian keracunan pangan di Indonesia [8,9,10]. Kondisi ini menimbulkan perhatian publik terhadap efektivitas tata kelola, pengawasan mutu, serta kesiapan ekosistem pendukung dalam penyelenggaraan program di berbagai daerah.

Di sisi lain, komitmen pemerintah terhadap keberlanjutan program ini semakin menguat dengan disahkannya RAPBN 2026 yang menetapkan MBG sebagai program prioritas nasional dengan alokasi anggaran sebesar Rp335 triliun [10]. Besarnya skala anggaran dan luasnya jangkauan sasaran program berimplikasi pada meningkatnya perhatian dan respons masyarakat. Hal ini tercermin dari laporan analitik yang menunjukkan bahwa isu MBG telah menjadi topik perbincangan yang intens di ruang publik digital, dengan lebih dari 172 ribu diskusi tercatat [11].