BAB 2
LANDASAN TEORI

2.1 Makan Bergizi Gratis (MBG)

MBG merupakan kebijakan strategis pemerintah yang berfokus pada penguatan ketahanan pangan dan peningkatan kualitas nutrisi masyarakat demi mendukung target pembangunan jangka panjang [32]. Secara operasional, program ini dirancang untuk mengatasi masalah *stunting* dan malnutrisi pada anak sekolah dengan menyediakan asupan makanan dengan standar gizi yang terkontrol. Diskursus mengenai kebijakan ini di platform digital seringkali mencerminkan dinamika persepsi masyarakat terhadap efektivitas distribusi dan dampak ekonomi dari program tersebut [33].

2.2 Analisis Sentimen

Analisis sentimen merupakan teknik komputasi untuk mengidentifikasi dan mengklasifikasikan opini yang terkandung dalam teks digital untuk memahami kecenderungan emosional publik terhadap suatu isu [34]. Dalam beberapa tahun terakhir, analisis sentimen telah berkembang pesat dengan memanfaatkan model berbasis *deep learning* untuk menangani kompleksitas bahasa alami, termasuk bahasa non-formal dan slang yang sering ditemukan pada komentar media sosial di Indonesia [35].

2.3 RoBERTa

RoBERTa merupakan sebuah inovasi arsitektur berbasis *Transformer* yang dikembangkan sebagai optimalisasi dari model BERT. Secara mendasar, RoBERTa tidak mengubah struktur arsitektur asli dari BERT, melainkan melakukan modifikasi mendalam pada strategi pelatihan (*pre-training*). Model ini menghilangkan tugas *Next Sentence Prediction* (NSP) yang sebelumnya digunakan pada BERT, karena terbukti tidak terlalu berkontribusi pada performa model, dan menggantinya dengan fokus penuh pada *Masked Language Modeling* (MLM) [36]. Selain itu, RoBERTa dilatih menggunakan volume data yang jauh lebih besar, ukuran *batch* yang ditingkatkan, serta durasi pelatihan yang lebih panjang, sehingga mampu menghasilkan representasi bahasa yang jauh lebih kaya dan kontekstual [37].

Cara kerja RoBERTa berpusat pada mekanisme *Dynamic Masking* yang diterapkan pada proses *Masked Language Modeling*. Berbeda dengan BERT yang menggunakan *static masking* di mana token yang disembunyikan ditentukan sekali di awal dan tetap sama di setiap *epoch*, RoBERTa mengubah pola token yang di-*masking* setiap kali data dimasukkan ke dalam model selama proses pelatihan berjalan [38]. Proses ini memaksa *encoder* berbasis *self-attention* untuk terus belajar memahami konteks kata di sekitarnya secara lebih fleksibel dan adaptif [39]. Model ini membaca seluruh urutan teks secara dua arah (*bidirectional*), menghitung bobot hubungan antar-kata menggunakan mekanisme *Scaled Dot-Product Attention*.

2.4 BERT

Bidirectional Encoder Representations from Transformers (BERT) merupakan inovasi model representasi bahasa yang dirancang untuk mengekstraksi informasi kontekstual secara mendalam dari teks. Berbeda dengan model pendahulunya yang membaca teks secara sekuensial dari kiri ke kanan atau sebaliknya, BERT menerapkan pendekatan dua arah (*bidirectional*) sejati. Melalui pendekatan ini, representasi dari setiap token dibangun dengan mempertimbangkan seluruh kata di sekitarnya secara simultan, baik yang berada di sisi kiri maupun kanan.

Dalam siklus pengembangannya, BERT beroperasi melalui dua tahapan utama: *pre-training* (pra-pelatihan) dan *fine-tuning* (penyelarasan halus). Pada fase *pre-training*, model mempelajari karakteristik linguistik dasar, struktur kalimat, dan semantik bahasa menggunakan korpus data teks masif tanpa label secara *unsupervised*. Setelah model memiliki pemahaman bahasa yang generik, model memasuki fase *fine-tuning*. Pada tahap ini, parameter yang telah terlatih disesuaikan kembali menggunakan dataset spesifik yang berskala lebih kecil seperti korpus penilaian esai otomatis untuk menjalankan tugas akhir dengan akurasi yang tinggi.

2.4.1 Arsitektur IndoBERT

IndoBERT mengadopsi struktur *Transformer Encoder* yang merujuk pada kerangka kerja buatan Vaswani et al. Arsitektur ini sepenuhnya mengeliminasi penggunaan jaringan saraf berulang (*recurrent*) dan menggantinya dengan mekanisme *self-attention*. Perubahan ini memungkinkan seluruh elemen teks

diproses secara paralel, yang berdampak pada pemangkasan waktu komputasi saat pelatihan [40].

Karakteristik kontekstual dua arah pada model ini dibentuk melalui dua target optimalisasi utama selama masa pra-pelatihan [41]:

Untuk mengeksekusi fungsi-fungsi tersebut, kerangka arsitektur ini ditopang oleh beberapa komponen komputasi inti:

2.4.2 Scaled Dot-Product Attention

Fungsi atensi merupakan komponen matematis yang bertugas memetakan sebuah vektor *query* terhadap kumpulan pasangan *key-value* hingga menghasilkan suatu nilai keluaran. Agar komputasi berjalan efisien, proses ini diwujudkan melalui manipulasi operasi matriks. Bobot kedekatan antar-token diperoleh melalui fungsi *softmax* dari hasil perkalian titik (*dot product*) yang telah diskalakan, seperti yang dirumuskan pada persamaan 2.1:

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.1}$$

Variabel-variabel di dalam persamaan tersebut merepresentasikan komponen berikut:

1. Q (*query*): matriks yang merepresentasikan token yang sedang dianalisis tingkat relevansinya.
2. K (*key*): matriks penanda atau kunci dari seluruh kata pembanding yang ada dalam urutan teks.
3. V (*value*): matriks yang membawa nilai informasi aktual yang akan diberi bobot pembobotan.
4. d_k: parameter dimensi yang dimiliki oleh vektor *query* dan *key*.

Secara teknis, kalkulasi dimulai dengan mengalikan matriks Q dengan matriks K yang telah ditransposisikan (K^T) guna mencari skor kecocokan awal. Hasil perkalian tersebut kemudian dibagi dengan faktor penyeimbang $\sqrt{d_k}$. Langkah pembagian ini krusial untuk mengantisipasi nilai d_k yang tinggi; jika angka hasil perkalian terlalu besar, fungsi *softmax* akan bergeser ke area jenuh yang memicu fenomena *vanishing gradient* (gradien menjadi sangat kecil). Setelah

dinormalisasi lewat fungsi *softmax*, bobot yang dihasilkan dikalikan dengan matriks V untuk memperoleh representasi kontekstual akhir.

2.4.3 Multi-Head Attention

Agar model mampu mengekstraksi informasi dari berbagai sudut pandang representasi dan posisi yang berbeda secara simultan, diterapkan mekanisme *Multi-Head Attention* [40]. Alih-alih hanya mengandalkan satu fungsi atensi tunggal, model memproyeksikan matriks *query*, *key*, dan *value* secara linear sebanyak h kali. Setiap proyeksi linear ini dialirkan ke dalam fungsi atensi secara paralel yang disebut sebagai *head*. Hasil ekstraksi dari seluruh *head* kemudian disatukan (*concatenated*) dan diproyeksikan kembali secara linear menuju dimensi asli model.

$$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1,\ldots,\text{head}_h)W^O \tag{2.2}$$

$$\text{di mana} \quad \text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Elemen-elemen pada parameter didefinisikan sebagai:

1. W_i^Q, W_i^K, W_i^V: matriks bobot transformasi proyeksi yang bersifat dinamis dan dapat dipelajari selama proses *training*.
2. W^O: matriks bobot keluaran yang bertugas memetakan kembali gabungan representasi seluruh *head* menuju dimensi spasial awal model (d_{model}).
3. Concat: fungsi penggabungan elemen matriks dari $head_1$ hingga $head_h$ menjadi satu kesatuan linear.

2.4.4 Attention Pooling

Dalam pemrosesan bahasa alami, tingkat kontribusi setiap kata terhadap makna esai secara global bersifat variatif tidak semua token memiliki bobot urgensi yang setara [42]. Proses kalkulasi *Attention Pooling* dipecah menjadi tiga fase matematis:

1. Proyeksi Tersembunyi: Vektor status h_t ditransformasikan melalui *Multilayer Perceptron* (MLP) berlapisan tunggal guna memunculkan representasi tersembunyi (u_t).

$$u_t = \tanh(W_w h_t + b_w) \tag{2.3}$$

2. Normalisasi Bobot Atensi: Tingkat kontribusi token ditentukan dengan mengukur kesamaan (*similarity*) antara matriks representasi u_t dengan vektor konteks level kata (u_w). Skor kesamaan ini kemudian dinormalisasi menggunakan fungsi *softmax* untuk menghasilkan parameter bobot final (α_t).

$$\alpha_t = \frac{\exp(u_t^\top u_w)}{\sum_t \exp(u_t^\top u_w)} \tag{2.4}$$

3. Agregasi Berbobot (*Weighted Sum*): Vektor representasi akhir (s) yang merangkum esensi informatif dari keseluruhan teks dihitung melalui penjumlahan berbobot atas status tersembunyi (h_t) asli menggunakan koefisien α_t.

$$s = \sum_t \alpha_t h_t \tag{2.5}$$

Komponen-komponen dalam tahapan *Attention Pooling* didefinisikan sebagai berikut:

- h_t: vektor *hidden state* pada token ke-t yang diekstraksi dari lapisan *encoder* IndoBERT.
- W_w dan b_w: parameter matriks bobot beserta bias yang nilainya dioptimasi selama fase penyelarasan halus (*fine-tuning*).
- u_t: representasi tersembunyi dari token ke-t setelah melalui transformasi fungsi aktivasi nonlinier *tanh*.
- u_w: vektor konteks tingkat kata yang diinisialisasi secara acak dan diperbarui selama model dilatih.
- α_t: nilai bobot atensi ternormalisasi yang merepresentasikan tingkat signifikansi token ke-t terhadap esai.
- s: vektor luaran tunggal (*document embedding*) yang mengintegrasikan informasi penting dari teks untuk disalurkan ke lapisan prediksi multitugas.

2.4.5 Layer Normalization

Untuk menstabilkan pergeseran distribusi nilai pada status tersembunyi serta mempercepat durasi konvergensi saat pelatihan, tiap sublapisan di dalam struktur *Transformer* disusul oleh proses *Layer Normalization* [43]. Berbeda dengan *Batch Normalization* yang bekerja lintas sampel, metode ini mengkalkulasi nilai rata-rata dan varians secara internal dari seluruh representasi fitur *input* pada satu lapisan untuk satu sampel data tunggal [43].

$$\text{LayerNorm}(x) = \gamma \frac{x - \mu}{\sigma} + \beta \tag{2.6}$$

Notasi parameter dalam rumus didefinisikan sebagai berikut:

- x: Vektor fitur masukan dari sublapisan yang menjadi objek normalisasi.
- *mu*: Nilai rerata (*mean*) yang dihitung secara internal dari total elemen pada vektor masukan x.
- *sigma*: Nilai standar deviasi dari sebaran elemen di dalam vektor masukan x.
- *beta*: Vektor bias (*shift*) teroptimasi yang berfungsi menggeser distribusi aktivasi setelah normalisasi.
- *gamma*: Vektor skala (*gain*) teroptimasi untuk menyesuaikan varians aktivasi.

Meskipun formulasi awal *Layer Normalization* yang dirancang oleh Ba et al. [43] memakai simbol parameter skala g dan bias b, variasi implementasi pada arsitektur modern berbasis *Transformer* secara konsisten mengadopsi simbol γ dan β sebagai parameter *affine* yang dapat dilatih [44].

2.4.6 IndoBERT: Adaptasi Kontekstual Bahasa Indonesia

IndoBERT dikembangkan menggunakan korpus data Indo4B yang mencakup spektrum luas dari artikel berita, blog, percakapan media sosial, dan laman web, dengan volume mencapai 4 miliar kata yang tersebar dalam 250 juta kalimat. Pengujian empiris pada standarisasi *benchmark IndoNLU* membuktikan bahwa IndoBERT secara konsisten melampaui performa model multibahasa (mBERT) dalam menyelesaikan variasi tugas pemahaman bahasa alami, seperti analisis sentimen maupun *sequence tagging* [45]. Reputasi performa

ini menjadikannya arsitektur fondasi yang sangat ideal untuk membangun sistem penilaian esai otomatis berbasis Bahasa Indonesia.

2.5 YouTube API v3

YouTube saat ini menjadi salah satu sumber data terbesar opini publik karena volume komentarnya yang sangat tinggi dan mencakup berbagai lapisan masyarakat [46]. Pengambilan data dari YouTube dilakukan menggunakan *YouTube Data API v3*, yang memungkinkan peneliti untuk mengakses metadata video dan komentar secara terstruktur guna keperluan analisis teks lebih lanjut [47].

2.6 Text Pre-Processing

Sebelum data komentar masuk ke tahap klasifikasi, data tersebut perlu diproses terlebih dahulu melalui tahapan *pre-processing*. Tujuannya adalah untuk memperkecil dimensi ruang vektor serta membersihkan teks dari simbol atau karakter yang tidak diperlukan [48]. Dengan melakukan *pre-processing*, proses klasifikasi dapat berjalan lebih mudah, cepat, dan menghasilkan hasil yang lebih akurat. Tanpa tahapan ini, berbagai karakter yang tidak relevan seperti *emoji*, simbol, maupun kata tidak baku dapat mengganggu proses klasifikasi dan menurunkan performa model [48].

2.6.1 Data Cleaning

Data cleaning merupakan fase krusial dalam prapemrosesan teks yang bertujuan untuk mengeliminasi komponen atau informasi yang tidak relevan di dalam kumpulan data sebelum masuk ke tahap pemodelan [49]. Langkah ini memegang peranan penting dalam meningkatkan performa dan akurasi model analisis sentimen saat mengidentifikasi polaritas opini pada teks [50]. Selain membersihkan simbol, tanda baca, ataupun karakter khusus yang tidak memberikan kontribusi semantik, tahapan ini umumnya mencakup proses *case folding*. Prosedur *case folding* ini menyamakan seluruh huruf menjadi bentuk kecil (*lowercase*) agar kata yang sama tidak diidentifikasi sebagai entitas yang berbeda akibat variasi penggunaan huruf kapital [51].

2.6.2 Tokenization

Tokenisasi merupakan proses dekomposisi suatu urutan teks atau kalimat menjadi satuan-satuan modular yang lebih kecil dan bermakna, seperti kata, sub-kata, atau frasa, yang dikenal sebagai token [52]. Struktur token yang dihasilkan dari tahapan ini nantinya ditransformasikan menjadi representasi vektor masukan pada arsitektur analisis sentimen selanjutnya. Melalui segmentasi kalimat menjadi token-token penyusunnya, struktur teks menjadi lebih terorganisasi sehingga mempermudah model dalam mengekstrak fitur sintaktis maupun semantik secara lebih optimal [49].

2.6.3 Stemming

Dalam pemrosesan bahasa alami untuk Bahasa Indonesia, *stemming* merupakan tahapan krusial yang bertujuan untuk mereduksi kata berimbuhan menjadi bentuk kata dasarnya dengan memotong afiks secara sistematis [53]. Proses ini sangat penting untuk standarisasi teks karena mampu mengurangi variabilitas kosakata tanpa menghilangkan esensi semantiknya, sehingga dapat menurunkan dimensionalitas fitur dan meningkatkan efisiensi komputasi pada pemodelan lanjutan seperti pembelajaran mendalam atau klasifikasi teks [12]. Pustaka open-source PySastrawi menjadi salah satu alat yang paling sering diandalkan dalam riset komputasi terkini untuk menangani kompleksitas morfologi Bahasa Indonesia secara efektif sebelum data masuk ke tahap ekstraksi fitur [54].

2.6.4 Removing Stop Word

Stop word removal adalah proses eliminasi kata-kata umum yang sering muncul namun memiliki bobot rendah dalam menentukan sentimen, seperti kata sambung ”yang”, ”dan”, ”di”, serta ”dari”. Hal ini dilakukan untuk mengurangi dimensi data dan mempercepat proses komputasi [34].

2.6.5 Labeling

Labeling pada data merupakan tahapan penting dalam analisis sentimen, karena proses ini bertujuan untuk memberikan label pada setiap data teks sesuai dengan sentimen yang terkandung di dalamnya [55]. Secara tradisional, proses labeling dilakukan secara manual oleh penilai manusia, namun cara ini

membutuhkan waktu dan biaya yang cukup besar terutama ketika data yang diolah jumlahnya sangat banyak [55].

Untuk mengatasi keterbatasan tersebut, tahapan anotasi atau *labeling* data dapat dieksekusi secara otomatis memanfaatkan kapabilitas model berbasis *deep learning*. Dalam penelitian ini, proses pelabelan diimplementasikan menggunakan arsitektur RoBERTa yang telah dioptimalkan sebelumnya. Model ini secara mandiri memproses korpus teks dari setiap komentar dan memproyeksikan prediksi label kelas secara otomatis, yakni *positive* atau *negative*. Penentuan label akhir tersebut didasarkan pada kalkulasi nilai probabilitas tertinggi (*argmax*) yang dihasilkan pada lapisan output model [56].

2.7 Random Over Sampling (ROS)

Metode *Random Over-Sampling* (ROS) diterapkan sebagai strategi untuk mengatasi ketidakseimbangan distribusi kelas (*imbalanced datasets*) pada korpus data teks. Teknik ini beroperasi dengan cara menduplikasi sampel dari kategori minoritas secara acak melalui mekanisme penarikan sampel dengan pengembalian (*sampling with replacement*) hingga volumenya seimbang dengan kelas mayoritas [57, 58].

Dalam ranah pemrosesan bahasa alami (NLP), ROS menjadi pilihan utama dibandingkan metode sintetis seperti SMOTE karena karakteristiknya yang menjaga integritas data mentah. ROS diaplikasikan langsung pada tingkat teks sebelum fase tokenisasi atau ekstraksi fitur dijalankan. Pendekatan ini memastikan bahwa struktur sintaksis, gramatikal, dan semantik asli dari esai tidak mengalami distorsi. Sebaliknya, metode pembangkitan sintetis berbasis ruang vektor sering kali menghasilkan representasi numerik artifisial yang tidak memiliki padanan kata riil dalam struktur bahasa [59, 58].

Secara mekanis, penyetaraan jumlah sampel melalui ROS berfungsi untuk menyeimbangkan kontribusi gradien dalam fungsi kerugian (*loss function*) selama proses pelatihan. Hal ini mencegah algoritma tertuju secara bias hanya pada pola-pola dari kelas dominan [60, 57]. Namun, dari aspek performa, replikasi data yang identik secara berulang berpotensi menimbulkan risiko *overfitting*, di mana model cenderung menghafal sampel minoritas alih-alih mempelajari batas keputusan yang generik [57]. Oleh karena itu, pemilihan metode ini didasarkan pada kompromi untuk mempertahankan orisinalitas konteks bahasa teks esai siswa, dengan mitigasi risiko *overfitting* yang dikendalikan melalui arsitektur model yang

memiliki regularisasi kuat serta pengujian validasi yang ketat.

2.7.1 Rumus Random Over-Sampling

Secara matematis, proses penyeimbangan data menggunakan *Random Over-Sampling* dapat dirumuskan melalui persamaan berikut [57, 61]:

$$D_{\text{new}} = D_{\text{maj}} \cup D_{\text{min}} \cup D_{\text{rep}} \tag{2.7}$$

di mana ukuran atau kardinalitas dari himpunan data replika (D_{rep}) ditentukan secara eksplisit melalui rumus:

$$|D_{\text{rep}}| = |D_{\text{maj}}| - |D_{\text{min}}| \tag{2.8}$$

2.7.2 Penjelasan Rumus Random Over-Sampling

Berikut adalah penjelasan detail dari komponen-komponen rumus *Random Over-Sampling* pada Persamaan 2.7 dan 2.8.

1. D_{new}: Merepresentasikan keseluruhan *dataset* latih (*training dataset*) yang baru setelah melalui proses *over-sampling*. *Dataset* inilah yang nantinya diumpankan ke dalam model klasifikasi.
2. D_{maj}: Merupakan himpunan bagian (*subset*) data yang berisi seluruh sampel milik kelas mayoritas. Jumlah sampel pada kelas ini tetap utuh dan tidak mengalami modifikasi ($|D_{\text{maj}}|$).
3. D_{min}: Merupakan himpunan bagian data yang berisi sampel asli dari kelas minoritas sebelum dilakukan manipulasi jumlah.
4. D_{rep}: Merupakan himpunan sampel replika atau duplikat yang dipilih secara acak dari sampel-sampel yang ada di dalam D_{min}.
5. $\cup$: Simbol operasi gabungan (*union*) dalam teori himpunan, yang berarti *dataset* baru dibentuk dengan menggabungkan data mayoritas, data minoritas asli, dan data hasil duplikasi.

6. $|D_{\text{maj}}| - |D_{\text{min}}|$: Formulasi untuk menghitung selisih absolut antara jumlah data mayoritas dan minoritas. Selisih inilah yang menjadi parameter acuan bagi fungsi `RandomOverSampler` untuk menentukan seberapa banyak jumlah teks minoritas yang harus digandakan agar tercapai kondisi seimbang, yaitu $|D'_{\text{min}}| = |D_{\text{maj}}|$ [61, 58].

2.8 Confusion Matrix

Confusion matrix merupakan sebuah tabel yang digunakan untuk mengevaluasi performa model klasifikasi dengan cara membandingkan hasil prediksi model terhadap label aktual dari data uji [62]. Pada penelitian ini, *confusion matrix* yang digunakan adalah matriks berukuran 2×2 karena klasifikasi dilakukan terhadap dua kelas sentimen, yaitu Positif dan Negatif.

Tabel 2.1. *Confusion Matrix* Analisis Sentimen 2 Kelas

Label Aktual	**Label Prediksi**	
	Positif	**Negatif**
Positif	TP	FN
Negatif	FP	TN

Setiap sel pada Tabel 2.1 memiliki makna sebagai berikut.

1. *True Positive* (TP): Jumlah data yang berlabel aktual Positif dan diprediksi Positif oleh model (prediksi benar).
2. *True Negative* (TN): Jumlah data yang berlabel aktual Negatif dan diprediksi Negatif oleh model (prediksi benar).
3. *False Positive* (FP): Jumlah data yang berlabel aktual Negatif namun diprediksi Positif oleh model (prediksi salah).
4. *False Negative* (FN): Jumlah data yang berlabel aktual Positif namun diprediksi Negatif oleh model (prediksi salah).

Berdasarkan *confusion matrix* tersebut, metrik evaluasi yang digunakan dalam penelitian ini adalah sebagai berikut.

Accuracy mengukur proporsi prediksi yang benar dari keseluruhan data uji [63]:

$$textAccuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.9}$$

Precision mengukur seberapa tepat model dalam memprediksi suatu kelas, yaitu dari semua data yang diprediksi sebagai kelas tertentu, berapa banyak yang benar-benar merupakan kelas tersebut [63]:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.10}$$

Recall (juga dikenal sebagai *Sensitivity*) mengukur kemampuan model dalam menemukan seluruh data yang relevan dari suatu kelas, yaitu dari semua data yang sebenarnya merupakan kelas tertentu, berapa banyak yang berhasil diidentifikasi dengan benar [63]:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.11}$$

F1-Score merupakan rata-rata harmonik dari *Precision* dan *Recall*. Metrik ini sangat berguna ketika distribusi kelas tidak seimbang (*imbalanced*), karena memberikan gambaran performa yang lebih komprehensif dibandingkan hanya mengandalkan salah satu metrik saja [62]:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.12}$$

2.9 LDA (*Latent Dirichlet Allocation*)

LDA adalah model probabilistik generatif yang digunakan untuk melakukan *topic modeling* atau ekstraksi topik otomatis dari sekumpulan teks [64]. Dalam penelitian ini, LDA membantu mengelompokkan komentar YouTube mengenai program MBG ke dalam beberapa klaster topik (misalnya: anggaran, gizi, atau distribusi) tanpa perlu label manual pada topiknya. LDA bekerja berdasarkan asumsi bahwa setiap dokumen merupakan campuran dari berbagai topik dan setiap

topik merupakan campuran dari berbagai kata [65].

Secara matematis, LDA mendefinisikan dua distribusi utama yang digunakan dalam proses pemodelan topik [66, 67]. Pertama, distribusi topik untuk setiap dokumen d, yang dinotasikan sebagai θ_d, diambil dari distribusi *Dirichlet* dengan parameter α:

$$\theta_d \sim \text{Dir}(\alpha) \tag{2.13}$$

Kedua, distribusi kata untuk setiap topik k, yang dinotasikan sebagai ϕ_k, juga diambil dari distribusi *Dirichlet* dengan parameter β:

$$\phi_k \sim \text{Dir}(\beta) \tag{2.14}$$

di mana nilai α mengatur seberapa merata distribusi topik dalam satu dokumen, dan β mengatur seberapa beragam kosakata yang merepresentasikan satu topik. Nilai α dan β yang kecil mendorong distribusi yang lebih terfokus, sementara nilai yang besar menghasilkan distribusi yang lebih tersebar [68].

Inti dari proses generatif LDA adalah cara setiap kata $w_{d,n}$ dalam dokumen d dibangkitkan. Pertama, model memilih topik $z_{d,n}$ berdasarkan distribusi topik dokumen tersebut, kemudian memilih kata berdasarkan distribusi kata dari topik yang terpilih [66]:

$$z_{d,n} \sim \text{Multinomial}(\theta_d) \tag{2.15}$$

$$w_{d,n} \sim \text{Multinomial}(\phi_{z_{d,n}}) \tag{2.16}$$

Dari proses tersebut, probabilitas bersama (*joint probability*) dari seluruh kata yang teramati dan variabel tersembunyi dalam LDA diformulasikan sebagai [67]:

$$p(\mathbf{w}, \mathbf{z}, \theta, \phi \mid \alpha, \beta) = \prod_{k=1}^{K} p(\phi_k \mid \beta) \prod_{d=1}^{D} p(\theta_d \mid \alpha) \prod_{n=1}^{N_d} p(z_{d,n} \mid \theta_d)\, p(w_{d,n} \mid \phi_{z_{d,n}}) \tag{2.17}$$

di mana K adalah jumlah topik, D adalah jumlah dokumen, dan N_d adalah jumlah kata dalam dokumen d. Karena distribusi *posterior* dari variabel tersembunyi

tidak dapat dihitung secara langsung, inferensi dilakukan menggunakan metode *Collapsed Gibbs Sampling* [68].

Kualitas model LDA yang dihasilkan dievaluasi menggunakan dua metrik utama [66, 64]. ***Perplexity*** mengukur seberapa baik model memprediksi data, di mana semakin rendah nilainya, semakin baik modelnya:

$$\text{Perplexity}(D_{test}) = \exp\left(-\frac{\sum_{d=1}^{D} \log p(\mathbf{w}_d)}{\sum_{d=1}^{D} N_d}\right) \tag{2.18}$$

Coherence Score mengukur seberapa bermakna dan koheren topik yang dihasilkan secara semantik. Berbeda dengan *perplexity* yang bersifat matematis, *coherence score* lebih mencerminkan kualitas topik dari sudut pandang keterbacaan manusia, sehingga kedua metrik ini idealnya digunakan secara bersamaan dalam menentukan jumlah topik K yang optimal [68].

2.10 Penelitian Terdahulu

Pada penelitian ini, kajian penelitian terdahulu difokuskan pada bidang Analisis Sentimen, IndoBERT, dan Makan Bergizi Gratis. Beberapa penelitian yang relevan akan dibahas pada Tabel 2.2 dan Tabel 2.3 untuk menunjukkan posisi penelitian ini dibandingkan studi sebelumnya.

Tabel 2.2. Matriks Tinjauan Literatur Empiris Berdasarkan Metrik Akurasi (Bagian 1)

No	Referensi	Judul Penelitian	Metode / Algoritma	Metrik / Akurasi	Kelemahan / Kontribusi
1	septiana2025	Analyzing Public Sentiment on YouTube Comments Regarding the Free Lunch Policy Using the SVM Algorithm	Support Vector Machine (SVM)	Akurasi: 78.45% F1-Score: 0.76	Masih menggunakan machine learning tradisional (SVM) yang kurang menangkap konteks semantik kontekstual.
2	sitanggang2024	Analisis Sentimen Masyarakat terhadap Program Makan Siang Gratis pada Media Sosial X Menggunakan Algoritma Naïve Bayes	Naïve Bayes	Akurasi: 72.10% Precision: 0.70	Akurasi Naïve Bayes cenderung rendah pada struktur kalimat tidak baku dan sarkasme.
3	permana2025	Analisis Sentimen Debat Publik Pilpres 2024 Menggunakan LSTM dan IndoBERT pada Komentar YouTube	LSTM dan IndoBERT	IndoBERT Akurasi: 88.90% LSTM Akurasi: 76.20%	Objek penelitian adalah Debat Pilpres secara umum, bukan berfokus secara mendalam pada kebijakan spesifik MBG.
4	firdaus2025	Analisis Sentimen Publik terhadap Program Tabungan Perumahan Rakyat Menggunakan Model IndoBERT Lite pada Komentar YouTube	IndoBERT Lite	Akurasi: 86.15% Macro F1: 0.84	Topik spesifik tentang perumahan rakyat (Tapera), bukan kebijakan pangan atau MBG.
5	fatkhurrohman2025	Analisis Sentimen Program Makan Bergizi Gratis Pemerintah RI Melalui Twitter Menggunakan Metode SVM	Support Vector Machine (SVM)	Akurasi: 76.80% Recall: 0.75	Terbatas pada platform Twitter yang karakteristik teksnya pendek, berbeda dengan YouTube yang lebih deskriptif.

Tabel 2.3. Matriks Tinjauan Literatur Empiris Berdasarkan Metrik Akurasi (Bagian 2)

No	Referensi	Judul Penelitian	Metode / Algoritma	Metrik / Akurasi	Kelemahan / Kontribusi
6	ridwan2026	Analisis Sentimen Publik terhadap Program Makan Bergizi Gratis di Sosial Media TikTok Menggunakan Support Vector Machine	Support Vector Machine (SVM)	Akurasi: 79.12% F1-Score: 0.78	Menggunakan SVM konvensional dan berfokus pada ekosistem audiens video pendek TikTok.
7	mukarom2025	Analisis Sentimen Berbasis Aspek terhadap Program Makan Bergizi Gratis (MBG) di Twitter/X dengan Pendekatan Efektivitas Duncan	Aspect-Based Sentiment Analysis	Akurasi Aspek: 81.30%	Platform terbatas pada X dan tidak memanfaatkan arsitektur deep learning modern seperti IndoBERT.
8	pangestu_2026	Implementasi Analisis Sentimen Komentar YouTube Berbasis IndoBERT dengan Evaluasi Confidence Score	IndoBERT + Confidence Score	Akurasi: 89.40% Confidence Threshold > 0.8	Tidak diaplikasikan pada kasus kebijakan pangan strategis nasional seperti program MBG.
9	go_2026	Analisis Sentimen Publik Program Makan Bergizi Gratis (MBG) di YouTube: Perbandingan Kinerja Algoritma SVM dan Random Forest	SVM vs Random Forest	Random Forest: 79.50% SVM: 77.20%	Belum menggunakan model berbasis Transformer/Deep Learning Bahasa Indonesia (IndoBERT).