BAB 2
LANDASAN TEORI

2.1 *Automated Essay Scoring* (AES)

Automated Essay Scoring (AES) didefinisikan sebagai sistem berbasis komputer yang memanfaatkan teknologi *Natural Language Processing* (NLP) dan *Machine Learning* (ML) untuk mengevaluasi esai secara otomatis sehingga menyerupai penilaian manusia [5]. Sistem ini menawarkan peningkatan objektivitas dan konsistensi penilaian, sehingga dapat meminimalkan bias subjektif yang kerap terjadi pada penilai manusia [16].

2.2 *Natural Language Processing* (NLP)

Natural Language Processing (NLP) didefinisikan sebagai serangkaian teknik komputasi untuk menganalisis dan merepresentasikan bahasa alami dengan tujuan agar komputer mampu memproses bahasa menyerupai kemampuan manusia [17]. Secara fundamental, NLP dikategorikan ke dalam dua komponen utama, yaitu *Natural Language Understanding* (NLU) yang berorientasi pada pemahaman makna teks, dan *Natural Language Generation* (NLG) yang berfokus pada pembangkitan teks baru [17]. Konsep ini menjadi landasan arsitektur sistem dalam penelitian ini, yaitu NLU diterapkan untuk mengevaluasi jawaban siswa, sedangkan NLG digunakan untuk menyusun umpan balik (*feedback*) yang adaptif.

2.2.1 Natural Language Understanding (NLU)

Natural Language Understanding (NLU) merupakan fokus utama dalam pengembangan model representasi bahasa yang bertujuan untuk memprediksi hubungan antar teks melalui analisis secara holistik pada tingkat kalimat, maupun menghasilkan *output* yang mendetail pada tingkat token [18]. Kemajuan signifikan dalam tugas-tugas NLU dicapai melalui penggunaan pre-training representasi bahasa yang bersifat dua arah (bidirectional) secara mendalam, yang memungkinkan model untuk menyatukan konteks dari sisi kiri dan kanan secara bersamaan di seluruh lapisan[18]. Dalam domain Automated Essay Scoring (AES), komponen NLU memegang peranan vital untuk menangkap semantik jawaban siswa yang kompleks guna meningkatkan akurasi penilaian dibandingkan metode

ekstraksi fitur tradisional [19].

Dalam penelitian ini, NLU diimplementasikan menggunakan IndoBERT, sebuah model yang dilatih secara khusus untuk Bahasa Indonesia [10]. Model ini digunakan untuk mengekstraksi representasi kontekstual yang mendalam guna menyelesaikan tiga tugas evaluasi sekaligus (*multi-task learning*): prediksi skor, klasifikasi label kebenaran, dan klasifikasi tingkat penalaran (*Level of Reasoning*).

2.2.2 *Natural Language Understanding* (NLU)

Natural Language Understanding (NLU) merupakan fokus utama dalam pengembangan model representasi bahasa yang bertujuan memprediksi hubungan antarteks melalui analisis holistik pada tingkat kalimat maupun menghasilkan *output* terperinci pada tingkat token [18]. Kemajuan signifikan pada tugas NLU dicapai melalui penggunaan *pre-training* representasi bahasa dua arah (*bidirectional*) secara mendalam, yang memungkinkan model menyatukan konteks dari sisi kiri dan kanan secara bersamaan pada seluruh lapisan [18]. Dalam domain AES, komponen NLU memegang peranan penting untuk menangkap semantik jawaban siswa yang kompleks guna meningkatkan akurasi penilaian dibandingkan metode ekstraksi fitur tradisional [19].

Dalam penelitian ini, NLU diimplementasikan menggunakan IndoBERT, yaitu model yang dilatih secara khusus untuk Bahasa Indonesia [10]. Model ini digunakan untuk mengekstraksi representasi kontekstual yang mendalam guna menyelesaikan tiga tugas evaluasi sekaligus (*multi-task learning*), yaitu prediksi skor, klasifikasi label kebenaran, dan klasifikasi tingkat penalaran (*Level of Reasoning*/LOR).

A Arsitektur BERT

IndoBERT dibangun di atas arsitektur *Transformer Encoder* yang diperkenalkan oleh Vaswani et al. Arsitektur *Transformer* sepenuhnya didasarkan pada mekanisme *self-attention* dan memungkinkan pemrosesan paralel serta waktu pelatihan yang lebih singkat [20].

Kemampuan BERT dalam memahami konteks dua arah (*bidirectional*) diperoleh melalui dua objektif pelatihan utama yang diperkenalkan oleh Devlin et al. [18]:

1. *Masked Language Model* (MLM): Model dilatih untuk memprediksi 15%

token yang disembunyikan (*masked*) secara acak dari *input*. Tujuannya adalah mendorong model mempelajari representasi kata berdasarkan konteks kiri dan kanan secara bersamaan.

2. *Next Sentence Prediction* (NSP): Model dilatih untuk memprediksi apakah kalimat B merupakan kelanjutan logis dari kalimat A. Kemampuan ini relevan untuk tugas AES guna memahami koherensi antarkalimat dalam esai siswa.

Dalam menjalankan fungsinya, arsitektur ini didukung oleh beberapa komponen komputasi utama:

1. *Scaled Dot-Product Attention*
   Fondasi matematis utama model ini adalah fungsi *attention* yang memetakan sebuah *query* dan sekumpulan pasangan *key-value* menjadi sebuah *output*. Dalam praktiknya, komputasi ini dilakukan secara serentak menggunakan operasi matriks. Bobot *attention* dihitung menggunakan fungsi *softmax* dari hasil perkalian titik (*dot product*) yang diskalakan, sebagaimana ditunjukkan pada rumus 2.1:

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

   Komponen-komponen dalam rumus tersebut didefinisikan sebagai berikut:

   (a) Q (*query*): matriks yang berisi representasi token yang sedang dievaluasi kecocokannya.

   (b) K (*key*): matriks yang berisi representasi kunci dari token pembanding dalam sekuens.

   (c) V (*value*): matriks yang berisi informasi yang akan diberi bobot.

   (d) d_k: dimensi vektor *query* dan *key*.

   Pada prosesnya, matriks Q dikalikan dengan transposisi matriks K (K^T) untuk memperoleh skor kompatibilitas awal. Skor tersebut kemudian dibagi dengan faktor skala $\sqrt{d_k}$. Langkah penskalaan ini penting karena nilai d_k yang besar dapat menyebabkan hasil perkalian titik menjadi terlalu besar. Nilai yang terlalu besar akan mendorong fungsi *softmax* ke daerah jenuh sehingga nilai gradien menjadi sangat kecil. Setelah diskalakan dan dinormalisasi menggunakan *softmax*, bobot atensi tersebut dikalikan dengan matriks V untuk menghasilkan representasi akhir.

2. *Multi-Head Attention*

Untuk memungkinkan model menangkap informasi secara bersamaan dari berbagai subruang representasi pada posisi yang berbeda, digunakan mekanisme *Multi-Head Attention* [20]. Daripada menggunakan satu fungsi atensi tunggal, model memproyeksikan matriks *query*, *key*, dan *value* secara linear sebanyak h kali. Setiap proyeksi diproses melalui fungsi atensi secara paralel (disebut *head*). Hasil dari seluruh *head* kemudian digabungkan (*concatenated*) dan diproyeksikan kembali secara linear untuk menghasilkan representasi akhir.

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

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

Komponen-komponen dalam rumus di atas didefinisikan sebagai berikut:

(a) W_i^Q, W_i^K, W_i^V: matriks bobot parameter (*parameter matrices*) yang dapat dilatih (*learnable*).

(b) W^O: matriks bobot keluaran (*output matrix*) yang digunakan untuk memproyeksikan kembali hasil gabungan seluruh *head* ke dimensi model awal (d_{model}).

(c) Concat: operasi penggabungan (*concatenation*) vektor keluaran dari $head_1$ hingga $head_h$ menjadi satu matriks tunggal.

3. *Attention Pooling*

Dalam NLP, tidak semua kata atau token memberikan kontribusi yang sama terhadap makna keseluruhan teks esai [21]. Komputasi *Attention Pooling* dilakukan melalui tiga tahapan matematis berikut:

(a) *Hidden Projection*: Vektor h_t dilewatkan ke sebuah *Multilayer Perceptron* (MLP) satu lapis untuk memperoleh representasi tersembunyi (u_t).

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

(b) Normalisasi *Attention Weights*: Tingkat kepentingan setiap token diukur melalui tingkat kesamaan (*similarity*) antara representasi u_t dan

vektor konteks tingkat kata (u_w). Nilai kesamaan tersebut kemudian dinormalisasi menggunakan fungsi *softmax* untuk menghasilkan bobot atensi (α_t).

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

(c) Agregasi Berbobot (*Weighted Sum*): Vektor representasi akhir (s) yang merangkum seluruh informasi sekuens teks dihitung sebagai penjumlahan berbobot dari seluruh *hidden state* asli (h_t) berdasarkan bobot 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* token ke-t yang dihasilkan oleh lapisan *encoder* IndoBERT.
- W_w dan b_w: matriks bobot (*weight*) dan vektor bias yang dipelajari selama proses *fine-tuning*.
- u_t: representasi tersembunyi token ke-t setelah melewati fungsi aktivasi nonlinier *tanh*.
- u_w: vektor konteks (*context vector*) yang diinisialisasi secara acak dan dipelajari selama pelatihan model.
- α_t: bobot atensi ternormalisasi yang menunjukkan kontribusi token ke-t terhadap makna teks esai.
- s: vektor keluaran tunggal (*document embedding*) yang merangkum informasi penting dari sekuens untuk diteruskan ke lapisan prediksi multitugas.

4. *Layer Normalization*
Setiap sublapisan dalam arsitektur *Transformer* diikuti oleh mekanisme *Layer Normalization* untuk menstabilkan dinamika status tersembunyi (*hidden state dynamics*) dan mempercepat konvergensi pelatihan [22]. Berbeda dengan *Batch Normalization*, mekanisme ini menghitung rata-rata dan varians secara

langsung dari seluruh representasi *input* pada satu lapisan yang sama dalam satu kasus pelatihan tunggal [22].

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

Komponen-komponen dalam rumus di atas didefinisikan sebagai berikut:

- x: Vektor representasi input dari suatu sub-lapisan yang akan dinormalisasi.
- μ: Skalar nilai rata-rata (*mean*) yang dihitung dari seluruh elemen di dalam vektor input x pada lapisan tersebut.
- σ: Skalar nilai standar deviasi dari elemen-elemen di dalam vektor input x.
- β: Vektor parameter bias (*shift*) yang dapat dilatih.

Meskipun formulasi awal *Layer Normalization* yang diperkenalkan oleh Ba *et al*. [22] menggunakan notasi parameter skala g dan bias b, implementasi matematis pada arsitektur berbasis *Transformer* secara luas telah mengadopsi notasi γ dan β sebagai parameter *affine* yang dapat dilatih [23].

B IndoBERT: Adaptasi Kontekstual Bahasa Indonesia

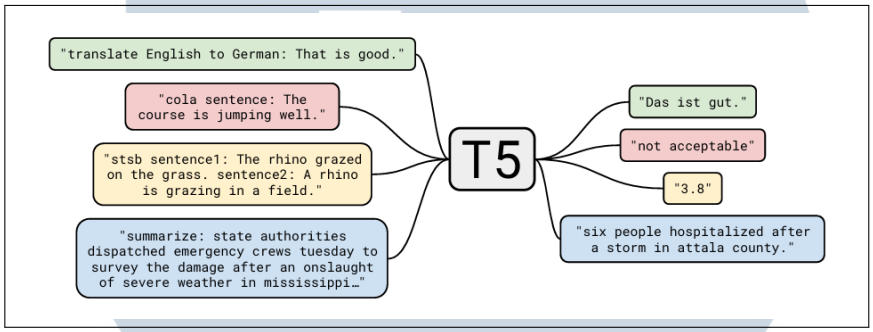
IndoBERT dilatih menggunakan dataset Indo4B yang dikumpulkan dari media sosial, blog, berita, dan situs web, dengan total sekitar 4 miliar kata dan 250 juta kalimat. Evaluasi pada *benchmark IndoNLU* menunjukkan bahwa IndoBERT secara konsisten mengungguli model multibahasa (mBERT) dalam berbagai tugas pemahaman bahasa, termasuk analisis sentimen dan penandaan urutan (*sequence tagging*) [24]. Keunggulan ini menjadikannya fondasi yang tepat untuk sistem penilaian esai otomatis dalam konteks Bahasa Indonesia.

2.2.3 *Natural Language Generation* (NLG)

Natural Language Generation (NLG) didefinisikan sebagai subbidang dari NLP yang berfokus pada pembuatan teks bahasa alami yang logis dan bermakna [25]. Dalam penelitian ini, NLG bertugas menerjemahkan hasil evaluasi, seperti skor dan label, menjadi teks umpan balik (*feedback*) yang adaptif menggunakan model idT5.

A Arsitektur *Text-to-Text Transfer Transformer* (T5)

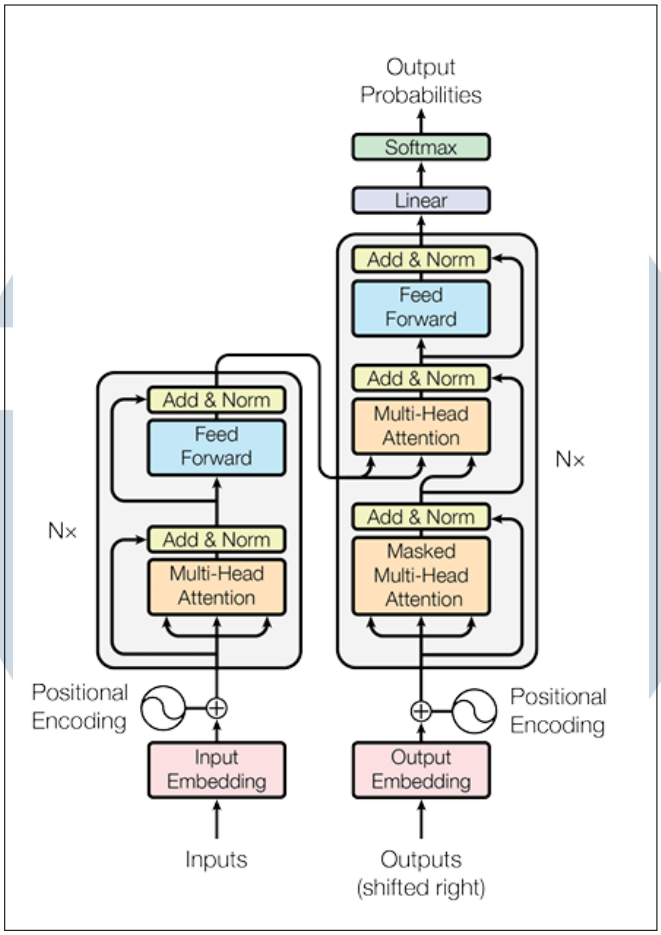
Text-to-Text Transfer Transformer (T5) mengusulkan pendekatan *text-to-text*, yaitu seluruh tugas NLP, baik klasifikasi maupun generasi, dipandang sebagai proses pemetaan sekuens teks *input* ke sekuens teks *output* [26]. Gambar 2.1 menggambarkan kerangka kerja T5, seluruh permasalahan pemrosesan teks diformulasikan sebagai transformasi teks *input* menjadi teks *output* [26].



Gambar 2.1. Kerangka kerja *text-to-text* pada model T5
Sumber: [26]

Untuk membedakan satu tugas dengan tugas lainnya, T5 memanfaatkan penambahan *task prefix* (awalan tugas) pada teks *input*. Model kemudian dilatih untuk memprediksi teks target yang sesuai menggunakan fungsi objektif *maximum likelihood* [26].

Meskipun T5 memperkenalkan paradigma pelatihan baru, arsitektur internalnya pada dasarnya mengadopsi struktur *Transformer* standar berbasis *encoder-decoder* yang pertama kali diusulkan oleh Vaswani et al. [26].

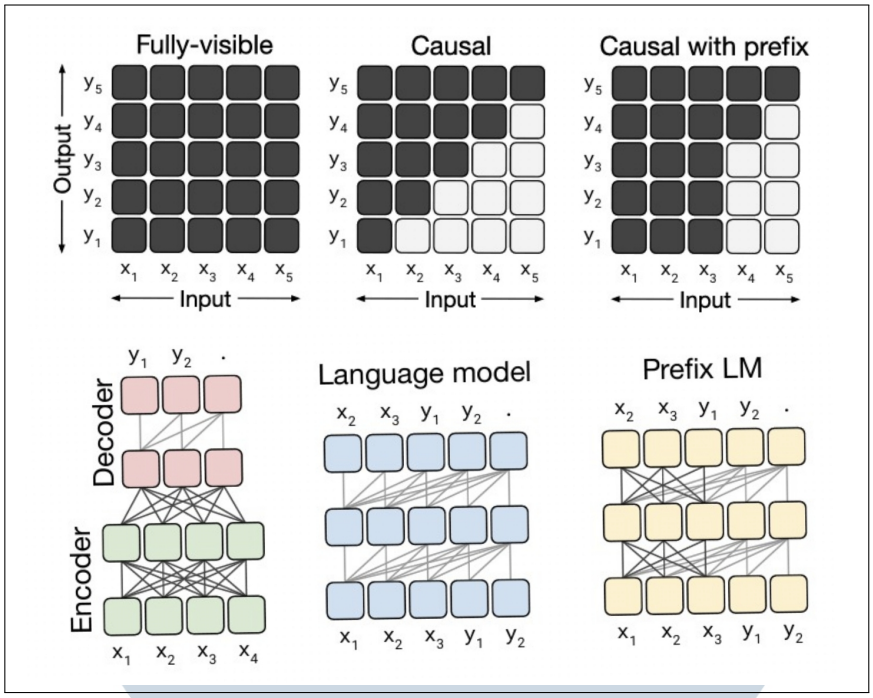


Gambar 2.2. Arsitektur dasar *encoder-decoder* pada *Transformer*
Sumber: [20]

Gambar 2.2 menggambarkan alur pemrosesan data pada arsitektur ini terbagi menjadi dua bagian utama, yaitu:

1. *Encoder*: bertugas menghasilkan representasi vektor yang merangkum makna dan konteks dari seluruh teks *input*.
2. *Decoder*: bertugas menghasilkan teks *output*. Lapisan ini juga berfungsi menyelaraskan token yang sedang dibangkitkan dengan representasi konteks yang telah dihasilkan oleh *encoder*.

Keunggulan utama arsitektur *encoder-decoder* pada T5 terletak pada visibilitas informasi saat memproses teks, yang diatur oleh mekanisme *attention mask* [26].



Gambar 2.3. Perbandingan skema *attention mask* dan struktur model
Sumber: [26]

Gambar 2.3 menunjukkan bahwa pada encoder digunakan fully-visible mask, sedangkan pada decoder digunakan causal mask untuk menjaga sifat autoregressive pada proses generasi teks. Secara spesifik, alur kerja T5 diatur sebagai berikut:

1. *Fully-visible Mask* (pada *encoder*): saat membaca teks *input*, model diizinkan melihat seluruh token dalam sekuens secara bersamaan. Hal ini memungkinkan model memahami konteks kalimat secara utuh, baik dari kata sebelum maupun sesudah token yang sedang diproses.

2. *Causal Mask* (pada *decoder*): pada tahap pembangkitan teks *output*, *decoder* menggunakan *causal masking*. Artinya, saat model memprediksi token ke-*i*, model hanya diizinkan melihat token yang telah dihasilkan sebelumnya dan tidak dapat melihat token berikutnya. Hal ini penting untuk menjaga integritas pada tugas generatif.

Untuk menangani konteks Bahasa Indonesia, penelitian ini menggunakan idT5. Model ini merupakan hasil adaptasi dari mT5 (*multilingual T5*) yang telah mengalami pemangkasan kosakata (*vocabulary pruning*) dan pelatihan ulang, sehingga memiliki ukuran 58% lebih efisien namun tetap mempertahankan kinerja generatif yang tinggi pada korpus Bahasa Indonesia [14].

B Metrik Evaluasi Generasi Teks

Kualitas teks umpan balik yang dihasilkan oleh model dievaluasi menggunakan metrik standar yang mengukur kesesuaian antara teks *output model* dan teks referensi (kunci jawaban ahli).

1. BLEU (*Bilingual Evaluation Understudy*)
   Metrik BLEU mengukur kualitas teks keluaran (*output*) dengan menghitung presisi *n*-gram yang telah dimodifikasi (*modified n-gram precision*) terhadap teks referensi. Berdasarkan rumusan perhitungannya, metrik ini melibatkan dua komponen utama [27]:

   (a) *Brevity Penalty* (BP): Faktor penalti multiplikatif yang diterapkan untuk mencegah model memperoleh skor tinggi hanya dengan menghasilkan teks keluaran yang sangat singkat. Penalti ini bernilai 1,0 jika panjang kandidat sama dengan atau lebih besar daripada referensi, dan akan menurun secara eksponensial jika lebih pendek.

   $$BP = \begin{cases} 1 & \text{jika } c > r \\ \exp(1 - r/c) & \text{jika } c \leq r \end{cases} \tag{2.7}$$

   Komponen pembentuk *Brevity Penalty* adalah sebagai berikut:

   i. c: panjang total teks kandidat (jumlah kata atau token) yang dievaluasi.
   ii. r: panjang referensi efektif (*effective reference length*), yaitu panjang teks referensi yang paling mendekati panjang teks kandidat.

   (b) Skor Akhir BLEU: Dihitung menggunakan rata-rata ukur (*geometric mean*) dari presisi *n*-gram yang telah dimodifikasi, kemudian dikalikan dengan faktor *Brevity Penalty*.

   $$\text{BLEU} = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{2.8}$$

   Pada rumus BLEU, komponen-komponen yang digunakan didefinisikan sebagai berikut:

i. N: panjang maksimal urutan kata (n-gram) yang digunakan dalam evaluasi.

ii. w_n: bobot positif seragam untuk setiap tingkat n-gram, dengan total bobot berjumlah satu ($\sum w_n = 1$). Umumnya digunakan bobot $w_n = 1/N$.

iii. p_n: nilai presisi n-gram yang dimodifikasi.

2. ROUGE (*Recall-Oriented Understudy for Gisting Evaluation*)
Berbeda dengan BLEU yang merupakan metrik berbasis presisi, ROUGE adalah metrik yang berfokus pada perhitungan *recall* [28]. ROUGE-N mengukur nilai *recall* n-gram antara ringkasan kandidat yang dievaluasi dan sekumpulan ringkasan referensi [28]. Perhitungan ini didefinisikan melalui rumus berikut:

$$\text{ROUGE-N} = \frac{\sum_{S \in \{\text{Reference Summaries}\}} \sum_{\text{gram}_n \in S} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{S \in \{\text{Reference Summaries}\}} \sum_{\text{gram}_n \in S} \text{Count}(\text{gram}_n)} \tag{2.9}$$

Komponen-komponen dalam rumus di atas didefinisikan sebagai berikut [28]:

(a) n: menunjukkan panjang unit n-gram.

(b) {Reference Summaries}: himpunan ringkasan referensi (atau kunci jawaban ideal dari pakar) yang digunakan sebagai acuan.

(c) gram_n: unit n-gram yang sedang dievaluasi.

(d) $\text{Count}_{\text{match}}(\text{gram}_n)$: jumlah maksimum n-gram yang muncul secara bersamaan pada ringkasan kandidat dan himpunan ringkasan referensi.

(e) $\text{Count}(\text{gram}_n)$: jumlah total kemunculan n-gram pada sisi referensi, yang menjadi penyebut dalam sifat *recall* metrik ini.

Selain menggunakan nilai *recall* murni, evaluasi sistem juga memanfaatkan F-measure untuk menyeimbangkan nilai presisi (P) dan *recall* (R) [28]. *F-measure* dihitung menggunakan parameter β yang mengatur tingkat kepentingan antara presisi dan *recall*. Dalam implementasi ekuivalen ketika presisi dan *recall* dianggap sama penting ($\beta = 1$), rumus ini menjadi rata-rata harmonik yang dikenal sebagai F1-score.

$$F = \frac{(1+\beta^2)RP}{R+\beta^2 P} \quad \xrightarrow{\beta=1} \quad F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.10}$$

2.3 Penelitian Terdahulu

Pada penelitian ini, kajian penelitian terdahulu difokuskan pada bidang *Automated Essay Scoring* (AES), *Natural Language Processing* (NLP), penerapan model berbasis *Transformer*, serta sistem pembangkitan umpan balik otomatis. Beberapa penelitian yang relevan akan dibahas pada Tabel 2.1 untuk menunjukkan posisi penelitian ini dibandingkan studi sebelumnya, sekaligus menegaskan kontribusi dan kebaruan yang ditawarkan.

Tabel 2.1. Ringkasan Penelitian Terdahulu

Judul Penelitian	**Peneliti & Tahun**	**Metode/Model**	**Dataset & Hasil Utama**	**Kelebihan & Kelemahan**
Automated Indonesian Essay Scoring and Holistic Feedback Using Bidirectional Encoder Representations for Transformers	Amalia et al. (2024) [11]	AES Bahasa Indonesia dengan feedback holistik berbasis kategori skor. Fine tuned IndoBERT.	Dataset: ASAP (B. Inggris) diterjemahkan ke B. Indonesia (Google API). 3.684 esai. Hasil: QWK: 0.90. Akurasi Training: 100%.	Kelebihan: Performa scoring sangat tinggi berkat IndoBERT. Kekurangan: Feedback bersifat statis/rule-based (template berdasarkan rentang nilai), bukan generative, dan dataset hasil terjemahan.
FABRIC: Automated Scoring and Feedback Generation for Essays	Han et al. (2023) [29]	BERT untuk Scoring + ChatGPT untuk Feedback (EssayCoT pipeline).	Dataset: DRESS + ASAP. Hasil: QWK meningkat 26,37%. Feedback dinilai sangat membantu ahli.	Kelebihan: Dataset asli, feedback sangat mendetail. Kelemahan: Bergantung API ChatGPT (komersial).
Distilling ChatGPT for Explainable Automated Student Answer Assessment	Li et al. (2023) [15]	Distilasi ChatGPT ke LongT5 untuk feedback yang dapat dijelaskan (rationales).	Dataset: ASAP-SAS (Short Answer). Hasil: QWK naik 11% dibanding ChatGPT.	Kelebihan: Model efisien, rationale jelas. Kelemahan: Hanya jawaban pendek, domain sains.

Bersambung ke halaman berikutnya

Tabel 2.1Ringkasan Penelitian Terdahulu (Lanjutan)

Judul Penelitian	**Peneliti & Tahun**	**Metode/Model**	**Dataset & Hasil Utama**	**Kelebihan & Kelemahan**
Implementasi LSTM untuk Generasi Feedback Berbahasa Indonesia	Semba (2024) [30]	Word2Vec + LSTM untuk Scoring dan Feedback.	Dataset: ASAP diterjemahkan, 960 data. Hasil: MSE rendah, dan BLEU < 0.3 (feedback buruk).	Kelebihan: Coba feedback generatif. Kelemahan: Dataset terjemahan, kualitas buruk, dan LSTM lemah untuk konteks panjang.
A Multitask Learning System for Trait-based Automated Short Answer Scoring	Ramesh & Sanampudi (2023) [31]	SBERT + LSTM (multitask learning) untuk trait scoring.	Dataset: ASAP Prompt 2 + OS. Hasil: QWK 0.691–0.672.	Kelebihan: Skor per-aspek. Kelemahan: Tidak menghasilkan feedback teks.
Automated Essay Scoring Incorporating Annotations from Automated Feedback Systems	Ormerod (2025) [32]	ModernBERT + T5 (annotator + scorer).	Dataset: PERSUADE. Hasil: QWK 0.866, melampaui baseline manusia (0.745).	Kelebihan: Anotasi argumentatif + grammar, arsitektur modern. Kelemahan: Bias subgrup, dan hanya Bahasa Inggris.

Bersambung ke halaman berikutnya

Tabel 2.1Ringkasan Penelitian Terdahulu (Lanjutan)

Judul Penelitian	**Peneliti & Tahun**	**Metode/Model**	**Dataset & Hasil Utama**	**Kelebihan & Kelemahan**
A Trait-based Deep Learning Automated Essay Scoring System with Adaptive Feedback	Hussein et al. (2020) [33]	LSTM.	Dataset: ASAP. Hasil: QWK 0.858.	Kelebihan: Skor berdasarkan trait. Kelemahan: Feedback adaptif terbatas, dan menggunakan model lama (LSTM).