BAB 2
LANDASAN TEORI

Bab ini menyajikan landasan teori yang memuat teori dan konsep dasar yang digunakan sebagai landasan penelitian.

2.1 Sintaksis Bahasa Indonesia

2.1.1 Definisi dan Karakteristik Sintaksis

Sintaksis merupakan cabang linguistik yang mempelajari hubungan antarkata dalam pembentukan satuan bahasa yang lebih besar, seperti frasa, klausa, dan kalimat [20]. Bahasa Indonesia memiliki karakteristik sintaksis yang sangat dipengaruhi oleh urutan kata (*word order*) dan penggunaan kata tugas, seperti konjungsi serta preposisi, dalam membentuk hubungan antarunsur kalimat [21]. Oleh karena itu, perubahan susunan kata atau penggunaan kata penghubung yang tidak tepat dapat menyebabkan perubahan makna maupun ketidakgramatikalan struktur kalimat. Karakteristik ini menjadikan analisis sintaksis penting dalam proses identifikasi kesalahan bahasa, khususnya kesalahan yang melibatkan relasi antarfrasa dan antarklausa.

2.1.2 Unit Sintaktis

Pembahasan mengenai unit sintaktis pada bagian ini mengacu pada Tata Bahasa Baku Bahasa Indonesia oleh Moeliono dkk. [22], kecuali jika disebutkan lain.

A Frasa

Frasa merupakan gabungan dua kata atau lebih yang bersifat nonpredikatif dan menempati satu fungsi sintaktis dalam klausa atau kalimat. Berbeda dengan klausa, frasa tidak mengandung hubungan subjek dan predikat di dalamnya. Hubungan antarfrasa berperan penting dalam membentuk struktur kalimat yang gramatikal dan mudah dipahami.

A.1 Jenis-Jenis Frasa

Berdasarkan unsur pusatnya, frasa dalam bahasa Indonesia diklasifikasikan menjadi beberapa jenis, yaitu:

1. **Frasa Nominal (Noun Phrase / NP):** berpusat pada nomina dan umumnya berfungsi sebagai subjek atau objek. Contoh: *tim penyelamat gabungan* pada kalimat ”Tim penyelamat gabungan dikerahkan ke lokasi kebakaran sejak dini hari”, yang berfungsi sebagai subjek kalimat.

2. **Frasa Verbal (Verb Phrase / VP):** berfokus pada verba sebagai inti dan berperan sebagai predikat yang menyatakan tindakan, proses, atau keadaan. Contoh: *sedang menyelidiki* pada kalimat ”Kepolisian sedang menyelidiki penyebab kebakaran yang melanda gudang tersebut”, yang berfungsi sebagai predikat kalimat.

3. **Frasa Preposisional (Prepositional Phrase / PP):** diawali preposisi dan biasanya berfungsi sebagai keterangan yang menunjukkan hubungan makna, seperti tempat, waktu, arah, atau sebab. Contoh: *di kawasan industri* pada kalimat ”Kebakaran terjadi di kawasan industri pada pukul 03.00 WIB”, yang berfungsi sebagai keterangan tempat.

4. **Frasa Adjektival (Adjective Phrase / ADJP):** menerangkan nomina dengan memberikan informasi mengenai sifat atau keadaan, baik dalam fungsi atributif di dalam frasa nominal maupun dalam fungsi predikat pada klausa. Contoh: *sangat parah* pada kalimat ”Kerugian akibat kebakaran itu diperkirakan sangat parah.”, yang berfungsi sebagai predikat untuk menerangkan keadaan subjek *kerugian*.

5. **Frasa Adverbial (Adverbial Phrase / ADVP):** berfungsi menerangkan verba, adjektiva, atau keseluruhan kalimat. Contoh: *dengan cepat* pada kalimat ”Petugas pemadam kebakaran merespons laporan warga dengan cepat.”, yang menerangkan cara berlangsungnya tindakan pada verba *merespons*.

B Klausa

Klausa merupakan satuan sintaksis yang memiliki unsur predikatif dan menjadi komponen pembentuk kalimat. Berbeda dengan frasa, klausa

mengandung hubungan predikatif antara unsur-unsur penyusunnya sehingga berpotensi membentuk sebuah kalimat yang utuh.

B.1 Jenis-Jenis Klausa

Berdasarkan kemampuannya untuk berdiri sendiri, klausa dibedakan menjadi klausa bebas dan klausa terikat. Klausa bebas dapat berdiri sendiri sebagai kalimat yang lengkap, seperti *"Warga sekitar segera mengungsi."*, sedangkan klausa terikat memerlukan klausa lain untuk membentuk makna yang utuh, seperti *"karena api menjalar dengan cepat"* yang baru berterima secara gramatikal apabila digabungkan dengan klausa bebas, misalnya "Warga sekitar segera mengungsi karena api menjalar dengan cepat.".

Berdasarkan hubungan antarklausa, klausa juga dibedakan menjadi klausa koordinatif dan klausa subordinatif. Klausa koordinatif memiliki kedudukan yang setara dan umumnya dihubungkan oleh konjungsi seperti *dan*, *atau*, dan *tetapi*, seperti pada kalimat "Petugas memadamkan api dan warga membantu evakuasi korban.", di mana kedua klausa dapat berdiri sendiri sebagai kalimat yang lengkap. Sebaliknya, klausa subordinatif memiliki kedudukan yang tidak setara dengan klausa utama dan biasanya ditandai oleh konjungsi subordinatif, seperti *karena*, *agar*, *jika*, dan *sehingga*, seperti pada kalimat "Proses evakuasi berjalan lambat karena akses jalan menuju lokasi tertutup material longsor",

C Hubungan Antarunsur

Hubungan antarunsur dalam sintaksis merujuk pada keterkaitan gramatikal antara unsur-unsur pembentuk kalimat. Kejelasan hubungan antarunsur penting untuk membentuk kalimat yang gramatikal dan mudah dipahami [23]. Ketidaktepatan hubungan antarunsur, seperti hilangnya unsur inti kalimat atau penggunaan kata penghubung yang tidak sesuai, dapat menyebabkan struktur kalimat menjadi tidak efektif dan menimbulkan ambiguitas makna.

2.1.3 Konjungsi dan Relasi Antarunsur

Bagian ini membahas klasifikasi konjungsi dalam bahasa Indonesia berdasarkan kedudukan gramatikal unsur yang dihubungkannya, mengacu pada [22].

A Konjungsi Koordinatif

Konjungsi koordinatif berfungsi menghubungkan dua unsur atau lebih yang memiliki kedudukan gramatikal setara, baik pada tataran kata, frasa, maupun klausa. Dalam kalimat majemuk setara, konjungsi koordinatif menghubungkan klausa-klausa yang masing-masing dapat berdiri sendiri sebagai kalimat tunggal. Konjungsi koordinatif yang umum digunakan dalam bahasa Indonesia antara lain *dan*, *atau*, *tetapi*, *namun*, *melainkan*, dan *sedangkan*.

Konjungsi koordinatif memiliki peran penting dalam pembentukan kalimat majemuk karena menentukan hubungan logis antarunsur yang dihubungkan. Ketidaktepatan penggunaannya dapat menimbulkan kesalahan sintaksis pada struktur kalimat.

B Konjungsi Subordinatif

Konjungsi subordinatif digunakan untuk menghubungkan klausa subordinatif (anak kalimat) dengan klausa utama (induk kalimat) sehingga membentuk hubungan yang tidak setara. Klausa yang diawali konjungsi subordinatif umumnya tidak dapat berdiri sendiri dan memerlukan klausa utama untuk membentuk kalimat yang lengkap.

Berdasarkan makna hubungan yang dinyatakan, konjungsi subordinatif dalam bahasa Indonesia dapat diklasifikasikan sebagai berikut:

1. **Waktu**: *ketika, saat, sewaktu, setelah, sebelum, sesudah, sejak, selama*
2. **Syarat**: *jika, kalau, apabila, bila, seandainya*
3. **Sebab**: *karena, sebab*
4. **Tujuan**: *agar, supaya, untuk*
5. **Konsesif (pengakuan)**: *walaupun, meskipun, biarpun, sekalipun*
6. **Hasil**: *sehingga, sampai*
7. **Cara**: *dengan, tanpa*

Pemahaman terhadap konjungsi subordinatif penting dalam analisis sintaksis karena menentukan hubungan hierarkis antarklausa dalam kalimat majemuk bertingkat. Ketidaktepatan penggunaannya dapat menyebabkan struktur kalimat menjadi tidak lengkap atau tidak sesuai dengan kaidah bahasa.

C Konjungsi Korelatif dan Paralelisme

Konjungsi korelatif merupakan pasangan konjungsi yang digunakan untuk menghubungkan unsur-unsur yang memiliki kedudukan setara dalam kalimat. Penggunaan konjungsi korelatif menuntut adanya kesejajaran struktur (paralelisme), yaitu kesamaan bentuk gramatikal pada unsur-unsur yang dihubungkan.

Beberapa contoh pasangan konjungsi korelatif dalam bahasa Indonesia antara lain:

1. *bukan...tetapi* dan *bukan...melainkan*
2. *baik...maupun* dan *baik...ataupun*

2.2 Natural Language Processing

Natural Language Processing (NLP) merupakan cabang kecerdasan buatan yang memadukan ilmu komputer dan linguistik untuk memungkinkan sistem komputer memahami, menganalisis, dan menghasilkan bahasa manusia [24]. NLP mencakup berbagai tingkat analisis bahasa, mulai dari morfologi, sintaksis, hingga semantik. Salah satu aspek penting dalam NLP adalah analisis sintaksis, yang berfokus pada struktur kalimat dan hubungan antarunsur bahasa. Tugas seperti *Part-of-Speech* (POS) tagging dan *phrase chunking* banyak digunakan untuk merepresentasikan informasi sintaktis dalam bentuk yang dapat diproses oleh model komputasional.

Perkembangan NLP telah beralih dari pendekatan berbasis aturan (*rule-based*) menuju pendekatan berbasis pembelajaran mesin (*machine learning*) dan *deep learning*, sehingga memungkinkan sistem mempelajari pola bahasa dari data secara lebih adaptif [24].

2.3 Sequence Labeling

2.3.1 Part-of-Speech (POS) Tagging

Part-of-Speech (POS) tagging merupakan tugas pelabelan sekuens (*sequence labeling*) yang bertujuan untuk menetapkan kategori kelas kata pada setiap token dalam suatu kalimat.

Secara formal, diberikan suatu urutan token hasil proses tokenisasi $X = (x_1, x_2, \ldots, x_n)$, maka tugas POS tagging adalah menentukan urutan label $Y = (y_1, y_2, \ldots, y_n)$ dengan panjang yang sama, di mana setiap label y_i berkorespondensi dengan token x_i. Dengan demikian, proses ini dapat dipandang sebagai fungsi pemetaan:

$$f : X \rightarrow Y \tag{2.1}$$

POS tagging termasuk tugas *sequence labeling* karena penentuan label suatu token dipengaruhi oleh konteks token di sekitarnya. Informasi kelas kata yang dihasilkan banyak digunakan dalam berbagai tugas NLP, seperti analisis sintaksis, *phrase chunking*, dan pelabelan sekuens lainnya [25].

2.3.2 Phrase Chunking

Phrase chunking atau *light parsing* merupakan teknik dalam pemrosesan bahasa alami yang digunakan untuk mengelompokkan token ke dalam unit-unit frasa secara non-rekursif [26]. Berbeda dengan *full parsing* yang menghasilkan struktur sintaksis lengkap, *chunking* hanya mengidentifikasi batas-batas frasa, seperti frasa nominal (*Noun Phrase*/NP) dan frasa verbal (*Verb Phrase*/VP), tanpa memodelkan hubungan hierarkis antarfrasa secara mendalam.

Dalam linguistik komputasi, *chunking* digunakan untuk merepresentasikan struktur sintaktis kalimat pada tingkat frasa sehingga memudahkan analisis hubungan antarunsur bahasa [26]. Informasi batas frasa juga berperan penting dalam berbagai tugas NLP yang berkaitan dengan analisis sintaksis dan pelabelan sekuens.

2.3.3 Skema IOB/BIO Tagging

Dalam tugas pelabelan sekuens (*sequence labeling*), diperlukan skema pelabelan untuk merepresentasikan batas-batas unit bahasa yang terdiri atas satu kata atau lebih. Salah satu skema yang umum digunakan adalah IOB (*Inside, Outside, Beginning*) atau BIO tagging [25]. Skema ini memberikan prefiks pada setiap label untuk menunjukkan posisi token dalam suatu unit bahasa.

1. **B (*Beginning*)**: menandai token pertama suatu unit.

2. **I (*Inside*)**: menandai token yang masih berada dalam unit yang sama.

3. **O (*Outside*)**: menandai token yang tidak termasuk ke dalam unit yang didefinisikan.

Skema BIO memungkinkan representasi batas-batas frasa secara eksplisit sehingga banyak digunakan dalam *phrase chunking* dan berbagai tugas pelabelan sekuens lainnya [25].

2.4 Conditional Random Field (CRF)

Conditional Random Field (CRF) merupakan model probabilistik yang digunakan untuk pelabelan dan segmentasi data sekuensial. Model ini termasuk dalam kategori *discriminative model*, yang secara langsung memodelkan probabilitas bersyarat suatu urutan label terhadap urutan observasi. Keunggulan utama CRF terletak pada kemampuannya memanfaatkan fitur-fitur yang saling tumpang tindih (*overlapping features*) serta mempertimbangkan dependensi antar-label dalam suatu sekuens. Dengan memodelkan keterkaitan antar-label secara eksplisit, CRF mampu menghasilkan prediksi yang lebih konsisten dibandingkan metode yang mengklasifikasikan setiap token secara independen.

Bentuk CRF yang paling umum digunakan adalah *Linear-Chain CRF*, yaitu model yang merepresentasikan data sekuensial sebagai rangkaian observasi $X = (x_1, x_2, \ldots, x_T)$ dan label $Y = (y_1, y_2, \ldots, y_T)$. Pada model ini, prediksi suatu label dipengaruhi oleh observasi yang diberikan serta label-label di sekitarnya, sehingga hubungan kontekstual antar-token dapat dimodelkan secara lebih baik. Pendekatan ini banyak digunakan pada berbagai tugas NLP seperti *Part-of-Speech Tagging*, *Named Entity Recognition*, dan *Phrase Chunking*.

Secara formal, probabilitas urutan label Y terhadap observasi X didefinisikan sebagai:

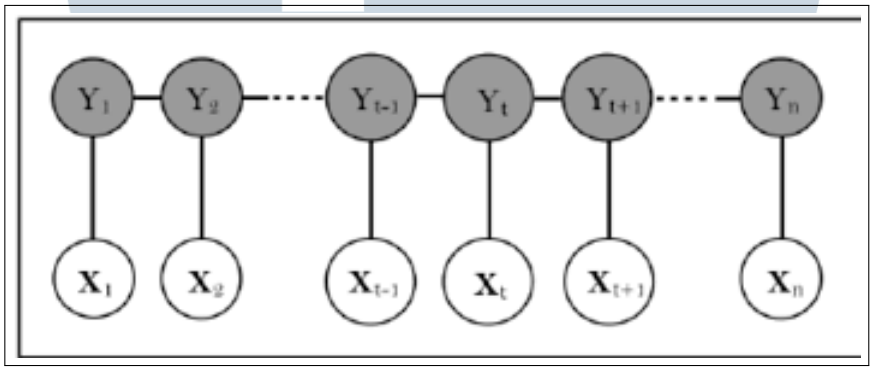
$$P(Y|X) = \frac{1}{Z(X)} \exp\left(\sum_{t=1}^{T}\sum_{k} \lambda_k f_k(y_{t-1}, y_t, X, t)\right) \tag{2.2}$$

dengan $f_k(y_{t-1}, y_t, X, t)$ sebagai fungsi fitur, λ_k sebagai bobot fitur, dan $Z(X)$ sebagai faktor normalisasi (*partition function*) yang dihitung dari seluruh kemungkinan urutan label sehingga total probabilitas bernilai satu.

Dalam *Linear-Chain CRF* terdapat dua jenis informasi utama yang digunakan dalam proses prediksi, yaitu *state feature* dan *transition feature*. *State*

feature merepresentasikan hubungan antara observasi dan label pada posisi yang sama, misalnya hubungan antara token, POS tag, atau chunk dengan label yang diprediksi. Sementara itu, *transition feature* merepresentasikan hubungan antar-label yang berurutan sehingga model dapat mempelajari pola transisi label yang sering muncul dalam data.

Struktur *Linear-Chain CRF* ditunjukkan pada Gambar 2.1. Node observasi $X_1, X_2, \ldots, X_n$ merepresentasikan data masukan, sedangkan node label $Y_1, Y_2, \ldots, Y_n$ merepresentasikan label yang diprediksi. Hubungan vertikal antara observasi dan label menunjukkan pemanfaatan fitur masukan pada proses prediksi, sementara hubungan horizontal antar-label merepresentasikan dependensi antar-label yang berurutan.



Gambar 2.1. Struktur Linear-Chain CRF
Sumber: [27]

Proses pelatihan CRF bertujuan untuk memperoleh bobot fitur λ yang menghasilkan probabilitas tertinggi bagi urutan label yang benar pada data latih. Untuk tujuan tersebut, CRF memaksimalkan fungsi *log-likelihood* yang diturunkan dari probabilitas bersyarat $P(Y|X)$.

$$\mathscr{L}(\lambda) = -\sum_{i=1}^{N} \log P(Y^{(i)} \mid X^{(i)}) \tag{2.3}$$

Karena proses optimasi lebih mudah dilakukan dalam bentuk minimisasi, fungsi objektif tersebut umumnya dinyatakan sebagai *negative log-likelihood*. Selama proses pelatihan, model secara iteratif menghitung probabilitas urutan label, mengevaluasi nilai *log-likelihood*, menghitung gradien terhadap parameter model, dan memperbarui bobot fitur hingga mencapai kondisi konvergen.

Tahapan umum pelatihan *Linear-Chain CRF* ditunjukkan pada Algoritma 1.

Algorithm 1 Pelatihan Linear-Chain CRF

Require: Data latih (X, Y)
Ensure: Bobot fitur optimal λ
1: Inisialisasi bobot fitur λ
2: **repeat**
3: Hitung probabilitas $P(Y|X)$
4: Hitung nilai *log-likelihood*
5: Hitung gradien $\nabla_\lambda \mathscr{L}(\lambda)$
6: Perbarui bobot fitur λ
7: **until** konvergen
8: **return** λ

2.5 Text-to-Text Transfer Transformer (T5)

Text-to-Text Transfer Transformer (T5) merupakan model berbasis *transformer* yang memperlakukan seluruh tugas pemrosesan bahasa alami sebagai permasalahan *text-to-text*, di mana baik input maupun output direpresentasikan dalam bentuk teks [19]. Pendekatan ini memungkinkan satu model tunggal digunakan untuk berbagai tugas, seperti penerjemahan, peringkasan, hingga pelabelan sekuens, dengan format yang seragam.

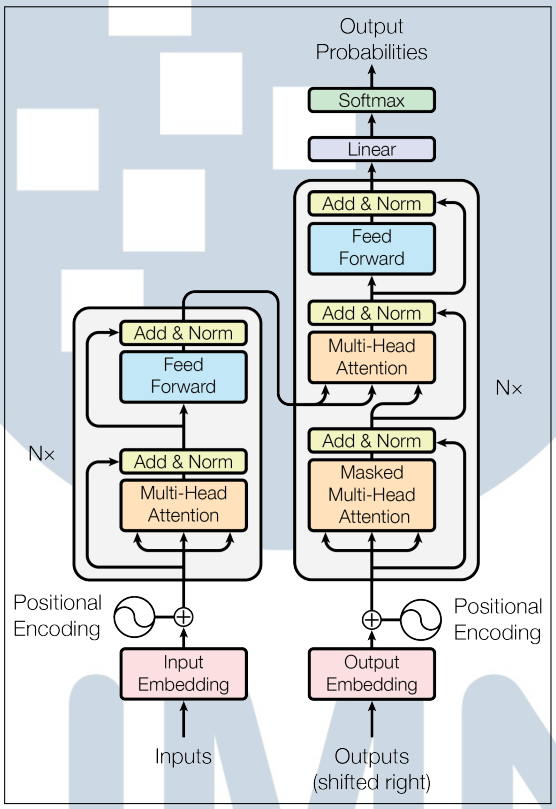
Model T5 umumnya digunakan melalui pendekatan *transfer learning*, yaitu memanfaatkan model yang telah melalui proses *pretraining* pada korpus berskala besar kemudian menyesuaikannya (*fine-tuning*) untuk tugas tertentu menggunakan dataset yang lebih spesifik. Pada tahap *fine-tuning*, parameter model yang telah dipelajari selama *pretraining* diperbarui kembali berdasarkan data pada tugas target sehingga model dapat beradaptasi terhadap karakteristik permasalahan yang dihadapi. Dalam penelitian ini, pendekatan tersebut diterapkan menggunakan model *Wikidepia/IndoT5-base* yang telah dipra-latih pada korpus berbahasa Indonesia dan kemudian di-*fine-tune* untuk tugas koreksi kesalahan sintaksis.

T5 dibangun di atas arsitektur *Transformer encoder-decoder*, di mana *encoder* bertugas memproses input menjadi representasi kontekstual, sedangkan *decoder* menghasilkan output secara autoregresif berdasarkan representasi tersebut. Berbeda dengan pendekatan tradisional yang merancang model khusus untuk setiap tugas, T5 menggunakan *task-specific prefix* pada input untuk memberi informasi mengenai jenis tugas yang akan dilakukan, sehingga seluruh proses dapat diformulasikan dalam satu kerangka kerja yang sama.

Selain itu, T5 mengadopsi beberapa penyesuaian pada arsitektur

transformer, seperti penggunaan *pre-layer normalization* dan *relative positional embeddings* untuk meningkatkan stabilitas pelatihan serta kemampuan model dalam menangkap hubungan posisi antar token.

Arsitektur yang digunakan dalam T5 merupakan arsitektur *Transformer encoder-decoder*, yang secara umum terdiri atas komponen encoder dan decoder sebagaimana ditunjukkan pada Gambar 2.2.



Gambar 2.2. Arsitektur Transformer Encoder-Decoder yang digunakan dalam T5
Sumber: [28]

2.6 Analisis Kesalahan Tata Bahasa

Analisis kesalahan tata bahasa (*grammatical error analysis*) merupakan bidang kajian dalam pemrosesan bahasa alami yang berfokus pada identifikasi dan penanganan penyimpangan kaidah bahasa dalam suatu teks. Secara umum, bidang ini terbagi menjadi dua tugas utama yang saling berkaitan namun memiliki tujuan berbeda, yaitu *Grammatical Error Detection* (GED) dan *Grammatical Error Correction* (GEC). GED berfokus pada tahap identifikasi keberadaan kesalahan, sedangkan GEC berfokus pada tahap perbaikan kesalahan yang telah teridentifikasi. Kedua tugas ini dapat diterapkan secara terpisah maupun secara berurutan sebagai

satu alur kerja (*pipeline*), di mana keluaran dari GED menjadi masukan bagi proses GEC.

2.6.1 Grammatical Error Detection (GED)

Grammatical Error Detection (GED) merupakan tugas dalam pemrosesan bahasa alami yang bertujuan mengidentifikasi keberadaan dan lokasi kesalahan tata bahasa pada suatu teks tanpa melakukan perbaikan secara langsung [29]. GED umumnya diformulasikan sebagai tugas pelabelan sekuens (*sequence labeling*), di mana setiap token diberi label berdasarkan keberadaan kesalahan, misalnya label *correct* (C) untuk token yang sudah sesuai kaidah dan *incorrect* (I) untuk token yang menyimpang dari kaidah bahasa. Sebagai contoh, pada kalimat ”Saya membeli buku itu untuk adik saya membaca”, kata *membaca* menyimpang dari struktur baku karena seharusnya diawali konjungsi tujuan seperti *agar*, sehingga sistem GED akan menandai token tersebut, atau frasa di sekitarnya, sebagai bagian yang mengandung kesalahan tanpa memberikan bentuk perbaikannya. Evaluasi kinerjanya biasanya menggunakan metrik *precision*, *recall*, dan *F1-score*, yang dihitung berdasarkan kesesuaian antara lokasi kesalahan yang diprediksi sistem dengan lokasi kesalahan yang sebenarnya.

2.6.2 Grammatical Error Correction (GEC)

Grammatical Error Correction (GEC) merupakan tugas dalam pemrosesan bahasa alami yang bertujuan untuk menghasilkan teks yang lebih gramatikal dengan memperbaiki kesalahan yang terdapat pada teks masukan [29]. Berbeda dengan *Grammatical Error Detection* (GED) yang hanya mengidentifikasi keberadaan kesalahan, GEC melakukan transformasi dari teks sumber menjadi teks hasil koreksi yang sesuai dengan kaidah bahasa. Melanjutkan contoh sebelumnya, sistem GEC akan mengubah kalimat ”Saya membeli buku itu untuk adik saya membaca” menjadi ”Saya membeli buku itu agar adik saya membaca”, yaitu dengan menyisipkan konjungsi tujuan yang sesuai pada posisi yang telah diidentifikasi mengandung kesalahan. Evaluasi sistem GEC umumnya dilakukan dengan membandingkan hasil koreksi yang dihasilkan model terhadap kalimat referensi menggunakan metrik yang mengukur tingkat kesesuaian hasil koreksi [29].

Kedua tugas tersebut dapat dipandang sebagai dua tahap yang saling melengkapi dalam satu alur kerja penanganan kesalahan tata bahasa. Pada

tahap deteksi, sistem berperan sebagai penyaring awal yang memisahkan kalimat atau segmen teks yang diduga mengandung kesalahan dari kalimat yang sudah gramatikal, tanpa mengubah isi teks. Hasil deteksi tersebut, yang umumnya berupa label posisi atau rentang kesalahan, selanjutnya digunakan sebagai informasi masukan pada tahap koreksi. Pada tahap koreksi, sistem memanfaatkan informasi lokasi kesalahan tersebut untuk menghasilkan bentuk perbaikan yang sesuai dengan konteks kalimat, sehingga proses koreksi dapat difokuskan pada bagian yang memang teridentifikasi bermasalah alih-alih memproses seluruh kalimat secara menyeluruh. Pendekatan bertahap semacam ini umum diterapkan dalam sistem penanganan kesalahan tata bahasa karena memungkinkan setiap tahap dioptimalkan sesuai dengan karakteristik tugasnya masing-masing, yaitu tugas klasifikasi/pelabelan pada tahap deteksi dan tugas generasi teks pada tahap koreksi.

2.7 Metrik Evaluasi

Bagian ini membahas metrik evaluasi klasifikasi yang meliputi *precision*, *recall*, *F1-score*, dan *accuracy* berdasarkan Daniel (2026) [25], kecuali jika disebutkan lain.

2.7.1 Confusion Matrix

Confusion Matrix merupakan tabel evaluasi yang digunakan untuk membandingkan hasil prediksi model dengan label aktual pada data uji [30]. Matriks ini memberikan informasi mengenai jumlah prediksi yang benar maupun salah untuk setiap kelas, sehingga dapat digunakan sebagai dasar perhitungan berbagai metrik evaluasi klasifikasi.

Struktur umum *confusion matrix* untuk klasifikasi biner ditunjukkan pada Tabel 2.1.

Tabel 2.1. Confusion Matrix

	Predicted Positive	**Predicted Negative**
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

Komponen *True Positive* (TP) dan *True Negative* (TN) menunjukkan jumlah prediksi yang sesuai dengan label aktual. Sebaliknya, *False Positive* (FP) dan *False Negative* (FN) menunjukkan jumlah prediksi yang tidak sesuai dengan label aktual.

Keempat komponen tersebut menjadi dasar dalam perhitungan metrik evaluasi seperti *accuracy*, *precision*, *recall*, dan *F1-score*.

2.7.2 Precision

Precision (ketepatan) merupakan metrik evaluasi yang mengukur proporsi prediksi positif yang benar dibandingkan dengan seluruh prediksi positif yang dihasilkan oleh model. Metrik ini menunjukkan tingkat keakuratan model ketika memberikan prediksi positif.

Secara matematis, *precision* dirumuskan sebagai:

$$Precision = \frac{TP}{TP + FP} \tag{2.4}$$

Nilai *precision* yang tinggi menunjukkan bahwa model memiliki tingkat *false positive* yang rendah.

2.7.3 Recall

Recall (sensitivitas) merupakan metrik evaluasi yang mengukur kemampuan model dalam menemukan seluruh instance positif yang terdapat pada data. Metrik ini menunjukkan seberapa banyak instance positif yang berhasil dikenali dibandingkan dengan seluruh instance positif yang sebenarnya ada.

Secara matematis, *recall* dirumuskan sebagai:

$$Recall = \frac{TP}{TP + FN} \tag{2.5}$$

Nilai *recall* yang tinggi menunjukkan bahwa model mampu mendeteksi sebagian besar instance positif yang terdapat pada data.

2.7.4 F1-Score

F1-score merupakan metrik evaluasi yang menggabungkan *precision* dan *recall* ke dalam satu nilai tunggal melalui rata-rata harmonik. Metrik ini digunakan untuk mengukur keseimbangan antara kemampuan model dalam menghasilkan prediksi positif yang tepat dan kemampuannya dalam menemukan seluruh instance positif yang relevan.

Secara matematis, *F1-score* dirumuskan sebagai:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.6}$$

atau dapat dinyatakan dalam bentuk komponen *confusion matrix* sebagai:

$$F1 = \frac{2TP}{2TP + FP + FN} \tag{2.7}$$

Nilai *F1-score* yang tinggi menunjukkan bahwa model memiliki keseimbangan yang baik antara *precision* dan *recall*. Oleh karena itu, metrik ini banyak digunakan pada tugas klasifikasi dan pelabelan sekuens, terutama ketika distribusi kelas tidak seimbang (*class imbalance*).

2.7.5 Accuracy

Accuracy (akurasi) merupakan metrik evaluasi yang mengukur proporsi prediksi yang benar terhadap seluruh prediksi yang dihasilkan oleh model. Metrik ini menunjukkan tingkat ketepatan model secara keseluruhan pada data uji.

Secara matematis, *accuracy* dirumuskan sebagai:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.8}$$

Nilai *accuracy* yang tinggi menunjukkan bahwa sebagian besar prediksi model sesuai dengan label aktual. Dalam tugas pelabelan sekuens, seperti *Part-of-Speech Tagging* dan deteksi kesalahan sintaksis, *accuracy* digunakan untuk mengukur proporsi label yang berhasil diprediksi dengan benar terhadap seluruh label pada data uji. Namun, pada dataset dengan distribusi kelas yang tidak seimbang (*class imbalance*), metrik ini perlu diinterpretasikan bersama *precision*, *recall*, dan *F1-score* agar diperoleh gambaran performa model yang lebih representatif.

2.7.6 Loss Function

Loss function merupakan komponen penting dalam proses pelatihan model pembelajaran mesin yang digunakan untuk mengukur perbedaan antara

prediksi model dan label yang sebenarnya. Nilai loss digunakan sebagai dasar untuk memperbarui parameter model sehingga prediksi yang dihasilkan semakin mendekati target.

Dalam berbagai model klasifikasi dan pelabelan sekuens, loss function berperan sebagai fungsi objektif yang dioptimasi selama proses pelatihan. Salah satu pendekatan yang umum digunakan adalah memaksimalkan likelihood dari data, yang dalam praktiknya sering diubah menjadi bentuk negative log-likelihood untuk mempermudah proses optimasi.

A Regularisasi

Regularisasi merupakan teknik yang digunakan dalam pembelajaran mesin untuk mencegah terjadinya overfitting, yaitu kondisi ketika model terlalu menyesuaikan diri dengan data latih sehingga performanya menurun pada data baru. Teknik ini dilakukan dengan menambahkan komponen penalti terhadap kompleksitas model pada fungsi objektif.

Secara umum, fungsi objektif dengan regularisasi dapat dinyatakan sebagai:

$$\mathscr{L}_{\text{reg}}(\theta) = -\sum_{i=1}^{N} \log P(y^{(i)} \mid x^{(i)}) + \lambda R(\theta) \tag{2.9}$$

di mana $R(\theta)$ merupakan fungsi regularisasi dan λ adalah parameter yang mengontrol kekuatan regularisasi.

Terdapat dua jenis regularisasi yang umum digunakan, yaitu regularisasi L1 dan L2. Regularisasi L1 mendorong bobot fitur yang tidak relevan menjadi nol sehingga menghasilkan model yang lebih *sparse*. Sementara itu, regularisasi L2 mengecilkan bobot semua fitur tanpa memaksanya menjadi nol, sehingga menghasilkan model yang lebih stabil terhadap variasi data. Pemilihan jenis regularisasi serta nilai parameter λ berpengaruh terhadap keseimbangan antara kompleksitas model dan kemampuan generalisasi.

B Loss Function pada Model Generatif

Dalam tugas generasi teks, model generatif berbasis Transformer digunakan untuk menghasilkan sekuens output berdasarkan sekuens input yang diberikan. Model ini umumnya dilatih menggunakan pendekatan autoregresif, di mana setiap token diprediksi berdasarkan token-token sebelumnya dalam sekuens.

Fungsi loss yang umum digunakan dalam pelatihan model generatif adalah Cross-Entropy loss, yang mengukur perbedaan antara distribusi probabilitas yang diprediksi oleh model dengan distribusi target yang sebenarnya. Secara matematis, fungsi loss tersebut dapat dirumuskan sebagai:

$$\mathscr{L} = -\sum_{t=1}^{T} \log P(y_t \mid y_{<t}, x) \tag{2.10}$$

di mana x merupakan sekuens input, y_t adalah token target pada posisi ke-t, dan $y_{<t}$ adalah seluruh token sebelum posisi tersebut. Model mempelajari distribusi probabilitas token berikutnya secara kondisional terhadap konteks sebelumnya, sehingga mampu menghasilkan teks yang koheren dan sesuai dengan input.

2.7.7 Validasi Pakar

Validasi pakar (*human evaluation*) merupakan pendekatan evaluasi yang melibatkan penilai manusia untuk menilai kualitas dan kewajaran keluaran suatu sistem pemrosesan bahasa alami [31]. Pendekatan ini sering digunakan sebagai pelengkap evaluasi otomatis, seperti metrik berbasis confusion matrix, terutama pada tugas yang melibatkan aspek linguistik dan interpretatif yang sulit diukur secara kuantitatif. Dalam penelitian yang melibatkan data beranotasi, validasi pakar berperan penting untuk meningkatkan kualitas dan konsistensi *ground truth* yang digunakan sebagai acuan pelatihan maupun evaluasi model.

Proses validasi dilakukan dengan membandingkan hasil anotasi terhadap interpretasi pakar bahasa, sehingga kesalahan pelabelan atau ketidakkonsistenan anotasi dapat diidentifikasi dan diperbaiki sebelum dataset digunakan dalam proses pelatihan model. Dengan demikian, validasi pakar membantu memastikan bahwa label yang digunakan benar-benar merepresentasikan fenomena kebahasaan yang menjadi objek penelitian.

2.7.8 Evaluasi Model Generatif

Evaluasi model generatif bertujuan untuk mengukur kualitas teks yang dihasilkan dengan membandingkannya terhadap teks referensi yang dianggap benar. Berbeda dengan evaluasi klasifikasi yang berfokus pada label, evaluasi model generatif dilakukan berdasarkan tingkat kesamaan antara teks hasil prediksi dan

teks referensi. Pada penelitian ini, kualitas hasil koreksi model T5 dievaluasi menggunakan metrik BLEU, ROUGE-L, dan Exact Match.

A BLEU Score

BLEU (*Bilingual Evaluation Understudy*) merupakan metrik evaluasi yang diperkenalkan oleh Papineni et al. [32] untuk mengukur tingkat kesamaan antara teks hasil prediksi dan teks referensi. Metrik ini bekerja dengan menghitung kecocokan n-gram antara kedua teks dan mengombinasikannya dengan *brevity penalty* untuk mengurangi kecenderungan model menghasilkan teks yang terlalu pendek.

Secara formal, BLEU dihitung sebagai berikut:

$$\text{BLEU} = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{2.11}$$

di mana BP adalah *brevity penalty*, w_n adalah bobot untuk setiap n-gram, dan p_n adalah precision n-gram ke-n.

Nilai BLEU yang lebih tinggi menunjukkan tingkat kemiripan yang lebih besar antara teks hasil prediksi dan teks referensi.

B ROUGE-L

ROUGE (*Recall-Oriented Understudy for Gisting Evaluation*) merupakan metrik evaluasi yang digunakan untuk mengukur tingkat kemiripan antara teks hasil prediksi dan teks referensi [33] [34]. Berbeda dengan BLEU yang berorientasi pada precision n-gram, ROUGE lebih berfokus pada seberapa banyak informasi dari teks referensi yang berhasil dipertahankan pada hasil prediksi. Terdapat beberapa varian ROUGE yang umum digunakan, yaitu:

- **ROUGE-1**, mengukur kesamaan unigram (1-gram) antara teks prediksi dan referensi.
- **ROUGE-2**, mengukur kesamaan bigram (2-gram) antara teks prediksi dan referensi.
- **ROUGE-L**, mengukur kesamaan berdasarkan *Longest Common Subsequence* (LCS), yaitu urutan token terpanjang yang muncul pada kedua teks tanpa harus berurutan secara langsung [33, 34].

Secara umum, nilai ROUGE-L dihitung berdasarkan nilai precision dan recall dari Longest Common Subsequence (LCS) sebagai berikut:

$$P_{LCS} = \frac{LCS(X,Y)}{m} \tag{2.12}$$

$$R_{LCS} = \frac{LCS(X,Y)}{n} \tag{2.13}$$

dengan:

- $LCS(X,Y)$ adalah panjang *Longest Common Subsequence* antara teks prediksi dan referensi,
- m adalah jumlah token pada teks prediksi,
- n adalah jumlah token pada teks referensi.

Nilai ROUGE-L kemudian diperoleh menggunakan rata-rata harmonik (F-measure) [33]:

$$ROUGE\text{-}L = \frac{(1+\beta^2)R_{LCS}P_{LCS}}{R_{LCS}+\beta^2 P_{LCS}} \tag{2.14}$$

di mana β digunakan untuk mengatur keseimbangan antara precision dan recall. Nilai ROUGE-L berada pada rentang 0 hingga 1. Semakin tinggi nilai ROUGE-L menunjukkan semakin besar kesamaan struktur antara teks hasil prediksi dan teks referensi.

C Exact Match

Exact Match (EM) digunakan untuk mengukur persentase prediksi yang identik secara keseluruhan dengan teks referensi [35]. Suatu prediksi dianggap benar apabila seluruh token pada hasil prediksi sama persis dengan token pada teks referensi. Exact Match dihitung menggunakan persamaan berikut:

$$EM = \frac{Jumlah\ Prediksi\ Benar}{Jumlah\ Seluruh\ Data} \tag{2.15}$$

Nilai EM berada pada rentang 0 sampai 1 atau 0% sampai 100%. Semakin tinggi nilai EM menunjukkan semakin banyak hasil koreksi yang identik dengan referensi.