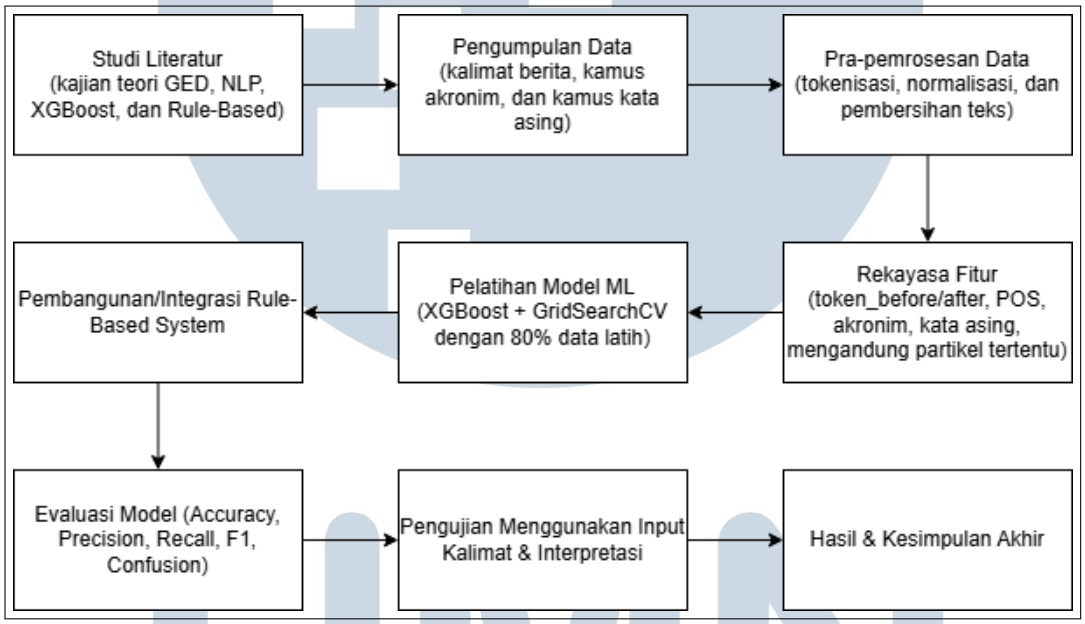
BAB 3
METODOLOGI PENELITIAN

3.1 Tahapan Penelitian

Penelitian ini dilakukan melalui beberapa tahapan, yaitu studi literatur, persiapan dan pengumpulan dataset, pre-processing, feature engineering, pelatihan model, integrasi *rule-based system*, dan evaluasi hasil. Alur tahapan penelitian dapat dilihat pada Gambar 3.1.



Gambar 3.1. Pipeline Penelitian

3.2 Studi Literatur

Tahap studi literatur dilakukan untuk memperoleh pemahaman yang jelas mengenai konsep-konsep dasar yang digunakan dalam penelitian ini. Hal yang ditinjau adalah teori mengenai *natural language processing*, deteksi kesalahan tata bahasa, algoritma klasifikasi, serta *rule-based system*.

3.3 Pengumpulan Data

Tahap pengumpulan data dilakukan menggunakan *script* Python untuk membangun korpus kalimat Bahasa Indonesia yang memuat berbagai variasi

penggunaan partikel, baik yang benar maupun yang salah. Korpus utama diperoleh dari artikel berita daring berbahasa Indonesia (Tribunnews). Dataset dibangun menggunakan pendekatan berbasis aturan (*rule-based dataset construction*) untuk menghasilkan dataset terlabel yang menggambarkan sembilan kategori kesalahan penulisan partikel, yaitu:

1. Penggunaan *pun* pada konjungsi.
2. Penggunaan *pun* pada kata umum.
3. Penggunaan *per* sebagai preposisi.
4. Penggunaan *per* sebagai awalan kata..
5. Penggunaan *-lah*/*-kah* pada kata umum.
6. Penggunaan *pun* pada akronim.
7. Penggunaan *-lah*/*-kah* pada akronim.
8. Penggunaan *pun* pada istilah asing.
9. Penggunaan *-lah*/*-kah* pada istilah asing.

3.3.1 Penandaan Bagian Kalimat yang Salah

Setelah dataset Rule 1-9 disiapkan, setiap baris diproses untuk mengekstraksi bagian kalimat yang mengandung kesalahan. Proses ini dilakukan dengan membandingkan kalimat asal dengan bentuk koreksi. Hasil proses ini adalah dua kolom tambahan:

1. *kata_salah*: potongan kata atau frasa dalam kalimat yang mengandung kesalahan penulisan partikel,
2. *koreksi*: bentuk penulisan yang benar sesuai PUEBI.

Variasi penulisan yang dipertimbangkan diantaranya adalah:

1. bentuk gabung-pisah (*kalaupun* dan *kalau pun*),
2. bentuk penyisipan atau penghilangan tanda hubung pada akronim dan istilah asing (*PNGpun* dan *PNG-pun*),

3. bentuk kata dasar yang digabung secara tidak tepat (*tigapertujuh* dan *tiga per tujuh*).

Baris data yang tidak memiliki pasangan *kata_salah* dan *koreksi* yang valid dihapus dari dataset kesalahan.

3.3.2 Penggabungan dan Pembersihan Dataset Kesalahan

Seluruh data yang valid dari sembilan kategori digabungkan menjadi satu himpunan data kesalahan partikel. Tahap berikutnya adalah pembersihan lanjutan untuk meningkatkan kualitas dataset, meliputi:

1. menghapus baris dengan *kata_salah* dan *koreksi* yang identik,
2. menghapus koreksi bertanda hubung yang tidak sesuai untuk kategori tertentu,
3. memastikan tidak ada nilai kosong pada kolom utama.

Setelah pembersihan, dataset kesalahan memiliki struktur kolom: *id*, kalimat, label (1–9), *kata_salah*, dan *koreksi*. Distribusi data untuk sembilan kategori kesalahan dapat dilihat pada Tabel 3.1:

Tabel 3.1. Distribusi Data untuk Sembilan Kategori Kesalahan

Label	Jumlah Data
1	100,000
2	103,567
3	112,772
4	100,000
5	109,623
6	103,190
7	103,179
8	103,493
9	103,482

3.3.3 Penyusunan Dataset Partikel Benar

Dataset partikel benar (label 0) dibentuk dari dua sumber:

1. Kalimat hasil koreksi otomatis. Untuk setiap kalimat pada dataset kesalahan, segmen *kata_salah* diganti dengan *koreksi* menggunakan pola dan aturan bahasa Indonesia yang sesuai. Hasilnya adalah kalimat baru dengan penggunaan partikel yang telah diperbaiki.

2. Kalimat dari data asli. Kalimat-kalimat tambahan diambil dari korpus berita yang sama (Tribunnews), dibersihkan, difilter, dan dihilangkan duplikatnya.

Kedua himpunan kalimat tersebut digabungkan dan diseleksi hingga mencapai jumlah target 500.000 kalimat kandidat benar.

3.3.4 Pembersihan Dataset Benar

Sebagai upaya memastikan bahwa kalimat berlabel 0 bebas dari kesalahan partikel, dilakukan tahap pemeriksaan pola secara otomatis. Beberapa pola yang diperiksa antara lain:

1. penulisan *pun* yang masih digabung dengan kata umum,

2. akhiran *-lah*/*-kah* yang masih ditulis terpisah,

3. penggunaan *per* yang tidak sesuai sebagai preposisi.

Kalimat yang masih memuat pola mencurigakan dihapus. Dari 500.000 kandidat, tersisa 121.035 kalimat yang lolos pemeriksaan dan dijadikan dataset partikel benar (label 0), dengan kolom *kata_salah* dan *koreksi* dibiarkan kosong.

3.3.5 Dataset Akhir

Dataset akhir diperoleh dengan menggabungkan dataset kesalahan partikel (label 1–9) dan dataset partikel benar (label 0). Seluruh baris kemudian diacak dan diberi penomoran ulang. Dataset akhir memiliki total sebanyak 1,060,341 data. Oleh karena itu, distribusi label akhir ditunjukkan pada Tabel 3.2:

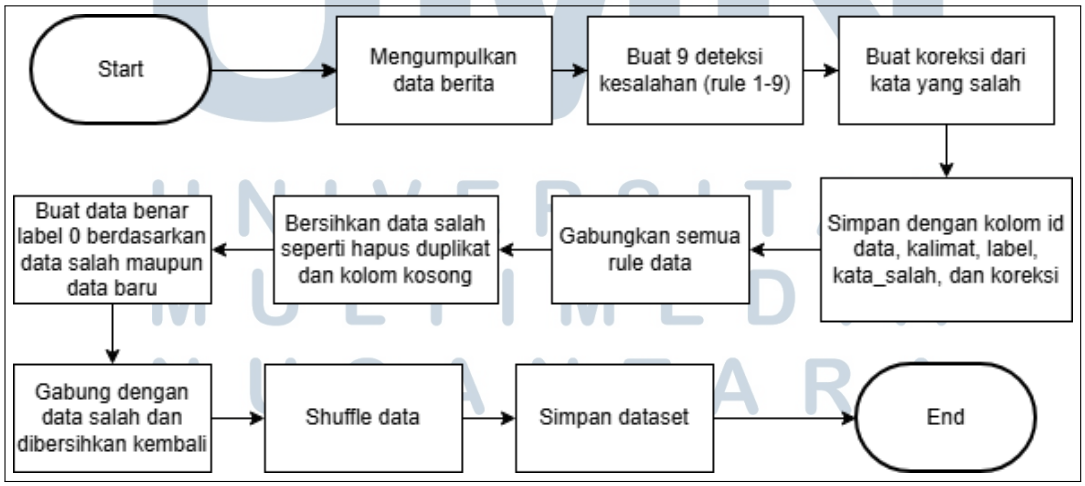
Tabel 3.2. Distribusi Label pada Dataset Akhir

Label	Jumlah Data
0	121,035
1	100,000
2	103,567
3	112,772
4	100,000
5	109,623
6	103,190
7	103,179
8	103,493
9	103,482

Dataset inilah yang digunakan pada tahap pra-pemrosesan, rekayasa fitur, dan pelatihan model pada bagian berikutnya. Proses pembentukan dataset secara singkat dapat dilihat pada *flowchart* di bawah. *Flowchart* tersebut merangkum alur *merge*, *labeling*, *shuffle*, hingga penomoran ulang data sebelum digunakan pada tahap pemodelan.

3.3.6 Flowchart Pembuatan Dataset

Proses pembuatan dataset dibagi menjadi beberapa tahap sehingga menghasilkan luaran data yang dapat diproses lebih lanjut.



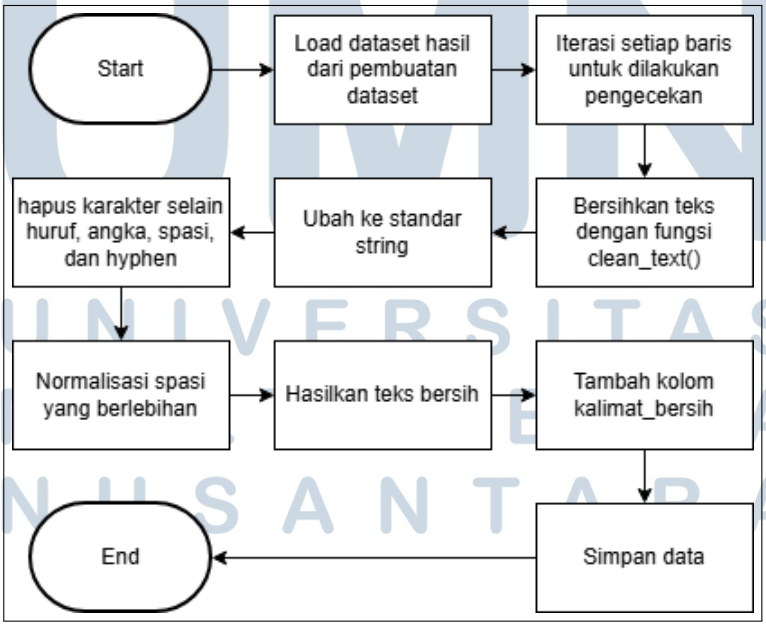
Gambar 3.2. Flowchart Pembuatan Dataset

Pada Gambar 3.2, proses penyusunan dataset dimulai dari pengumpulan korpus berita sebagai sumber utama kalimat yang digunakan untuk membuat sembilan kategori kesalahan partikel. Seluruh dataset setiap *rule* digabungkan dan dibersihkan. Hasilnya adalah satu kumpulan dataset kesalahan pada partikel bahasa Indonesia.

Selanjutnya, disusun dataset penggunaan partikel yang benar (label 0) yang berasal dari dua sumber, yaitu hasil koreksi otomatis dari dataset salah dan kalimat baru di luar korpus, baik mengandung partikel maupun tidak. Dataset benar ini juga melalui proses pembersihan untuk memastikan tidak ada pola kesalahan yang tersisa. Data salah dan data benar digabungkan menjadi satu dan disimpan untuk melalui *pre-processing* lanjutan.

3.4 Pra-Pemrosesan Data

Tahap pra-pemrosesan data bertujuan untuk menyiapkan teks agar lebih seragam dan mudah diolah pada tahap rekayasa fitur dan pelatihan model. Pada tahap ini, fokus utama adalah membersihan kalimat dari karakter yang tidak diperlukan serta membentuk representasi teks yang konsisten tanpa menghilangkan informasi penting terkait partikel. Hasil pra-pemrosesan tersebut diterapkan pada *dataset_final_clean.csv* dan menghasilkan *file* baru bernama *dataset_preprocessed.csv*.



Gambar 3.3. Flowchart Pra-Pemrosesan

Gambar 3.3 merangkum proses pra-pemrosesan yang diterapkan untuk menghasilkan bentuk teks yang bersih dan konsisten. Tahapan yang dimaksud adalah sebagai berikut.

1. Pembersihan Tanda Baca dan Karakter Khusus
   Setiap kalimat terlebih dahulu diubah ke bentuk yang seragam dengan menghapus tanda baca atau karakter khusus yang tidak diperlukan. Hanya karakter huruf, angka, spasi, dan tanda hubung yang dipertahankan.

2. Pelestarian Tanda Hubung dan Akronim
   Dalam proses pembersihan, tanda hubung sengaja dipertahankan karena memiliki peran penting untuk akronim dan istilah asing (misalnya *PNG-pun*, *KPU-pun*, atau *bulk-lah*.

3. Pembentukan Kolom kalimat_bersih
   Hasil pembersiham disimpan dalam kolom baru bernama *kalimat_bersih*, sementara kalimat asli tetap dipertahankan pada kolom *kalimat*. Sehingga, setiap baris data memiliki dua versi teks: versi asli yang merefleksikan konteks kalimat sebagaimana muncul pada sumber berita, dan versi yang telah dibersihkan yang digunakan sebagai dasar untuk proses tokenisasi dan pembentukan fitur.

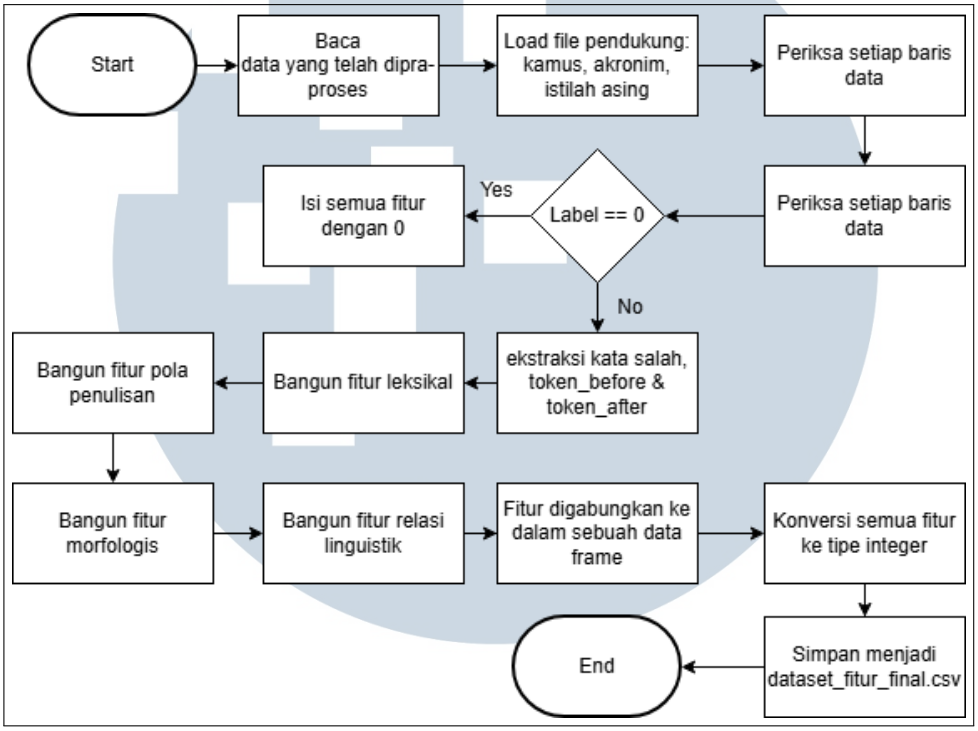
4. Persiapan untuk Rekayasa Fitur
   Dataset yang telah diperkaya dengan kolom *kalimat_bersih* kemudian disimpan ke dalam *dataset_preprocessed.csv*

3.5 Rekayasa Fitur

Tahap rekayasa fitur bertujuan mengubah teks hasil pra-pemrosesan menjadi representasi numerik yang mampu menggambarkan konteks linguistik partikel Bahasa Indonesia. Representasi ini menjadi fondasi bagi model *XGBoost* untuk membedakan pola kesalahan penulisan pada sembilan kategori partikel. Fitur-fitur dibangun secara manual berdasarkan analisis linguistik dan didukung oleh tiga sumber, yaitu: *kamus(in)_clean.csv*, *daftar_akronim.csv*, dan *nounlist_bersih.csv*. Tahap ini menghasilkan file akhir *dataset_fitur_final.csv* yang digunakan untuk pelatihan.

Secara garis besar, rekayasa fitur mencakup empat kelompok utama: *(1) fitur konteks token*, *(2) fitur leksikal*, *(3) fitur pola penulisan*, *(4) fitur morfologis dan*

POS, serta *(5) fitur relasi linguistik*. Seluruh fitur hanya dihasilkan untuk data yang berlabel selain 0 (data kesalahan), sementara data berlabel 0 akan menerima nilai 0 pada seluruh fitur. Alur rekayasa fitur secara singkat dapat dilihat pada gambar berikut.



Gambar 3.4. Flowchart Rekayasa Fitur

Gambar 3.4 menjelaskan bahwa tahap pertama adalah mengidentifikasi token yang berada sebelum dan sesudah segmen *kata_salah*. Informasi ini penting untuk membangun fitur kontekstual, khususnya untuk membedakan apakah sebuah partikel mengikuti konjungsi, pronomina, atau satuan numerik. Ekstraksi konteks dilakukan dengan mencari kata yang muncul tepat sebelum dan sesudah segmen *kata_salah* dalam sebuah kalimat. Proses ini tidak hanya membandingkan kata secara langsung, tetapi juga mempertimbangkan variasi penulisan seperti kata yang digabung, dipisah, atau ditulis dengan tanda hubung. Dengan cara ini, sistem dapat menentukan dua nilai konteks utama, yaitu `token_before` dan `token_after`.

Sebagai contoh, pada kalimat "*Walau pun itu sulit*", segmen kesalahan adalah "*pun*", sehingga `token_before` = "walau" dan `token_after` = "itu". Pada contoh lain, seperti "*KPU-pun hadir di lokasi*", sistem tetap dapat mengenali bahwa `token_before` = "KPU" dan `token_after` = "hadir" meskipun penulisannya melibatkan tanda hubung.

3.5.1 Fitur Leksikal

Fitur leksikal menggambarkan sifat dasar suatu token yang berpotensi mengandung kesalahan penulisan partikel. Beberapa fitur yang dibentuk adalah sebagai berikut:

- `len_error`: panjang karakter dari *kata_salah*.
- `is_base_acronym`: bernilai 1 apabila kata dasar termasuk ke daftar akronim.
- `is_base_foreign`: bernilai 1 apabila kata dasar terdaftar sebagai istilah asing.
- `is_base_uppercase`: mengidentifikasi apakah kata dasar seluruhnya huruf kapital.
- `is_not_in_main_dictionary`: menandai kata dasar yang tidak muncul dalam kamus utama.
- `digit_count`: jumlah digit angka di dalam kata dasar (misalnya pada akronim seperti *P3K*).

3.5.2 Fitur Pola Penulisan

Fitur pola penulisan menangkap bentuk struktural dari kata atau frasa yang digunakan. Fitur ini penting karena banyak kesalahan partikel berhubungan langsung dengan cara kata tersebut ditulis.

- `contains_space`: mendeteksi pemisahan yang tidak tepat (mis. *meski pun*).
- `contains_hyphen`: mendeteksi penulisan dengan tanda hubung (mis. *PNG-pun*).
- `starts_with_per`: menandai apakah kata dimulai dengan “per”.
- `contains_pun`, `contains_lah`, `contains_kah`: mendeteksi keberadaan partikel.
- `contains_per`: mendeteksi kemunculan token “per” sebagai unsur terpisah.

3.5.3 Fitur Morfologis dan POS

Informasi morfologis dan kelas kata diperoleh dari `kamus(in)_clean.csv`, yang berisi penandaan POS untuk ribuan kata bahasa Indonesia. Fitur ini membantu model dalam memahami fungsi kata dasar dalam struktur kalimat. Fitur yang dibangun antara lain:

- `base_word_pos_GANTI`: kata dasar merupakan pronomina.
- `base_word_pos_BENDA`: kata dasar merupakan nomina.
- `base_word_pos_HUBUNG`: kata dasar merupakan konjungsi.
- `base_word_pos_LAINNYA`: tidak memiliki tanda POS dalam kamus.

POS diposisikan sebagai indikator penting karena banyak kesalahan partikel melibatkan pasangan kata yang secara alami bukan pasangan morfologis yang valid.

3.5.4 Fitur Relasi Linguistik Partikel

Fitur relasi linguistik menangkap hubungan antara partikel dengan kata di sekitarnya. Fitur-fitur ini secara langsung berkaitan dengan kaidah PUEBI dan pola sintaktis yang umum dalam bahasa Indonesia.

- `is_before_pun_a_pronoun`: mendeteksi kasus seperti *dia pun*, *aku pun*.
- `is_before_pun_a_conjunction`: mendeteksi pola *walau pun*, *meski pun*, dan bentuk sejenis.
- `is_base_a_pun_conjunction`: mengenali kata dasar seperti *meskipun*, *adapun*, dan *walaupun*.
- `is_after_per_a_unit`: memvalidasi apakah unsur setelah "per" merupakan satuan umum (mis. *per meter*, *per hari*).
- `is_per_a_valid_prefix`: membedakan penggunaan "per-" sebagai awalan sah (mis. *perkotaan*).

Fitur-fitur ini sangat penting untuk membedakan kategori kesalahan yang tampak mirip tetapi berasal dari konteks linguistik yang berbeda.

3.5.5 Pembentukan Dataset Fitur Akhir

Seluruh fitur yang telah dibentuk terdiri dari fitur leksikal, pola penulisan, morfologis/POS, dan relasi linguistik yang kemudian digabungkan ke dalam sebuah *DataFrame* dengan kolom *label* sebagai kolom pertama. Baris dengan label 0 diisi nilai 0 pada seluruh fitur, sedangkan baris selain 0 berisi nilai hasil perhitungan fitur. Seluruh kolom fitur kemudian dikonversi ke tipe numerik *int8*, dan apabila tersedia informasi POS sebelum atau sesudah token, dilakukan proses *one-hot encoding*. Tahap ini menghasilkan berkas akhir *dataset_fitur_final.csv*.

Gambaran konkret mengenai nilai fitur yang terbentuk dapat dilihat pada dua contoh kalimat berikut:

1. Kalimat 1: "*Walau pun ia lelah, ia tetap melanjutkan perjalanan.*"
2. Kalimat 2: "*Diskon diberlakukan per orang pada hari tertentu.*"

Nilai fitur berikut merupakan hasil ekstraksi terhadap `kata_salah`, `token_before`, dan `token_after` pada kedua kalimat tersebut.

Tabel 3.3. Fitur Leksikal untuk Dua Kalimat

Fitur	**Contoh 1**	**Contoh 2**
len_error	9	9
is_base_acronym	0	0
is_base_foreign	0	0
is_not_in_main_dictionary	0	1
digit_count	0	0

Tabel 3.3 memperlihatkan fitur-fitur dasar yang menggambarkan karakteristik kata, seperti panjang karakter, keberadaan dalam kamus, serta apakah kata tersebut merupakan akronim atau istilah asing.

Tabel 3.4. Fitur Pola Penulisan untuk Dua Kalimat

Fitur	**Contoh 1**	**Contoh 2**
contains_space	1	1
contains_hyphen	0	0
contains_pun	1	0
contains_per	0	1
starts_with_per	0	1

Tabel 3.4 menampilkan fitur-fitur yang memeriksa pola penulisan, seperti adanya spasi, tanda hubung, serta kemunculan partikel tertentu di dalam kata.

Tabel 3.5. Fitur Morfologis dan POS untuk Dua Kalimat

Fitur	**Contoh 1**	**Contoh 2**
base_word_pos_GANTI	0	0
base_word_pos_BENDA	0	0
base_word_pos_HUBUNG	1	0
base_word_pos_LAINNYA	0	1

Tabel 3.5 menunjukkan fitur-fitur yang berasal dari informasi kelas kata (POS), seperti apakah kata dasar merupakan nomina, pronomina, atau konjungsi.

Tabel 3.6. Fitur Relasi Linguistik untuk Dua Kalimat

Fitur	**Contoh 1**	**Contoh 2**
is_before_pun_a_pronoun	0	0
is_before_pun_a_conjunction	1	0
is_base_a_pun_conjunction	1	0
is_after_per_a_unit	0	1
is_per_a_valid_prefix	0	0

Tabel 3.6 memperlihatkan fitur yang memeriksa hubungan antara partikel dan kata-kata di sekitarnya, misalnya apakah "pun" didahului konjungsi atau apakah "per" diikuti satuan ukuran.

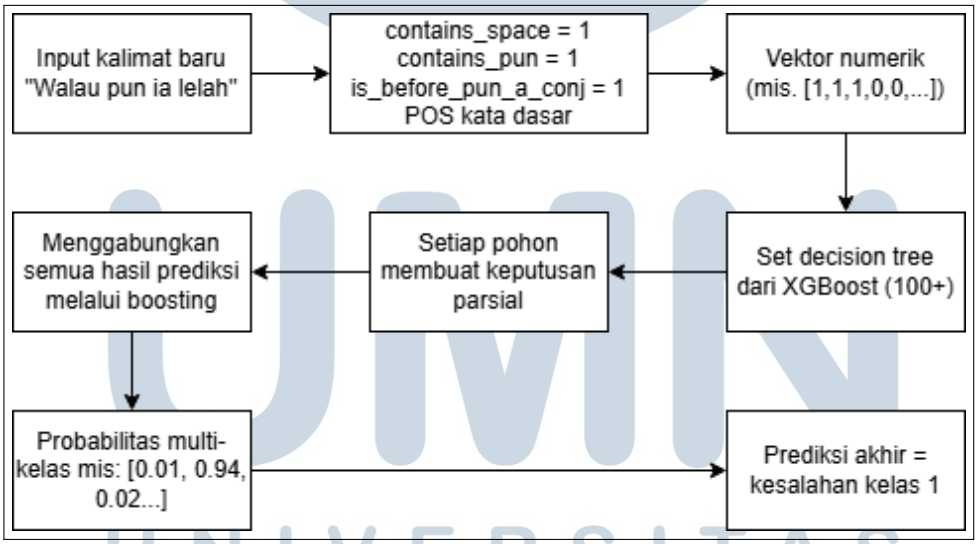
Dataset fitur ini kemudian menjadi masukan utama pada proses pelatihan model XGBoost.

3.6 Pelatihan Model (Training)

Tahap pelatihan model dilakukan menggunakan algoritma *XGBoostClassifier* dengan masukan berupa dataset fitur akhir *dataaset_fitur_final.csv*. Pada tahap awal, seluruh kolom fitur selain *label* dikonversi ke tipe numerik yang seragam (*float32*) agar stabil dan efisien ketika diproses oleh XGBoost. Selanjutnya, data dibagi menjadi dua bagian, yaitu 80% data latih dan 20% data uji.

Model dasar XGBoost disusun dengan beberaa parameter yang telah ditetapkan, misalnya *objective=multi:softprob* (klasifikasi multikelas), jumlah kelas *num_class* = 10, serta pengaturan *subsample* dan *colsample_bytree* sebesar 0,9 untuk mengurangi risiko *overfitting*. Selain itu, dilakukan juga *hyperparameter tuning* dengan *GridSearchCV*. Kombinasi parameter yang dicoba meliputi kedalaman pohon (*max_depth*), laju pembelajaran (*learning_rate*), dan jumlah pohon (*n_estimators*). Setiap kombinasi diuji menggunakan *Stratified K-Fold* dengan 3 lipatan dan tiga skema penilaian dengan fokus model terbaik berdasarkan skor F1-Macro.

Setelah proses *grid search* selesai, model dengan nilai F1-Macro rata-rata tertinggi pada data latih dipilih sebagai *best estimator*. Model terbaik ini kemudian dievaluasi menggunakan 20% data uji yang tidak pernah dilibatkan dalam proses pelatihan. Metrik yang dihitung meliputi akurasi, presisi, *recall*, dan F1-Score per kelas, serta *confusion_matrix* untuk melihat pola kesalahan antar kelas. Hasil pelatihan yang telah selesai akan disimpan dalam *xgb_manual_features_best.joblib* untuk digunakan pada tahap pengujian dan implementasi sistem.



Gambar 3.5. Contoh Proses XGBoost

Sebagai ilustrasi, Gambar 3.5 memberikan contoh kalimat "*Walau pun ia lelah*". Fitur yang diekstrasi mencakup pola penulisan (misalnya `contains_space = 1` dan `contains_pun = 1`), konteks linguistik (`is_before_pun_a_conjunction = 1`), serta informasi morfologis kata dasar "walau". Berdasarkan kombinasi fitur ini, model dapat memahami bahwa pasangan kata tersebut mengikuti pola kesalahan kategori *pun* setelah konjungsi (label 1), sehingga menghasilkan prediksi yang

sesuai.

Secara umum, proses kerja model XGBoost adalah pembentukan banyak pohon keputusan yang masing-masing mempelajari pola tertenu dari fitur yang diberikan. Setiap pohon menangkap aturan dalam bentuk *branches*, misalnya ”apakah kata sebelum *pun* merupakan konjungsi?”, ”apakah terdapat spasi antara kata dasar dan partikel?”, atau ”apakah kata setelah *per* merupakan satuan?”. XGBoost kemudian menggabungkan kontribusi seluruh hasil tersebut, memperbaikinya dengan *boosting*, hingga mencapai model akhir yang mampu merasakan pola-pola kesalahan partikel secara konsisten.

3.7 Integrasi Rule-Based System

Pendekatan *rule-based* digunakan untuk melengkapi hasil prediksi model pembelajaran dengan menambahkan aturan linguistik yang disusun berdasarkan kaidah Bahasa Indonesia sesuai *Pedoman Umum Ejaan Bahasa Indonesia (PUEBI)*. Tujuannya adalah memperbaiki hasil klasifikasi pada kasus tertentu, seperti akronim, istilah asing, atau partikel konjungsi yang sulit dikenali secara akurat oleh model berbasis data. *Rule-based system* berfungsi sebagai lapisan validasi linguistik (*post-filter*) untuk memastikan hasil deteksi tetap sesuai dengan kaidah bahasa yang berlaku.

3.8 Evaluasi Model

Tahap evaluasi dilakukan terhadap 20% data uji menggunakan metrik performa yaitu akurasi (*accuracy*), presisi (*precision*), recall, dan F1-Score. Metrik ini digunakan untuk menilai kemampuan model dalam mengklasifikasikan penggunaan partikel secara tepat serta membandingkan performa antar kelas kesalahan. Hasil dari evaluasi ini akan dianalisis untuk menarik kesimpulan mengenai efektivitas XGBoost dan *rule-based system* dalam mendeteksi kesalahan penggunaan partikel pada teks berita. Pengujian deteksi kalimat juga dilakukan untuk memastikan model yang telah dilatih benar-benar mampu mendeteksi kesalahan penggunaan partikel.

3.9 Pembuatan API

Tahap ini bertujuan untuk membungkus model U-TAPIS ke dalam sebuah layanan berbasis web sehingga dapat diakses oleh antarmuka web maupun

aplikasi lain. Implementasi API dilakukan menggunakan *framework* Flask dengan memanfaatkan modul deteksi yang telah dibangun pada tahap sebelumnya. *Request* dari API tersebut berisi teks yang akan digunakan untuk deteksi.

Gambar 3.6. API *Request*

Pada Gambar 3.6, dalam *request* tersebut sistem masih berjalan secara lokal dan untuk melakukan deteksi dengan metode POST pada *endpoint* "/detect-utapis" dengan `paragraph` sebagai *body* berisi teks untuk melakukan pengecekan.

3.10 Spesifikasi Sistem

Pada penelitian ini digunakan perangkat keras dan perangkat lunak untuk menjalankan serta mengimplementasikan algoritma XGBoost dan sistem *rule-based* dalam klasifikasi kesalahan penulisan partikel Bahasa Indonesia. Spesifikasi sistem perangkat keras dan perangkat lunak yang digunakan dapat dilihat pada Tabel 3.7:

Tabel 3.7. Spesifikasi sistem perangkat keras dan perangkat lunak penelitian

Komponen	**Spesifikasi**
Processor	9th Gen Intel(R) Core(TM) i5-9300H @ 2.40GHz
Memory	8GB DDR4-2666
GPU	NVIDIA GeForce GTX 1650
Hard Disk	512GB SSD PCIe NVMe
Operating System	Windows 11 Home 64-bit
Bahasa Pemrograman	Python 3.10.11
IDE	Visual Studio Code