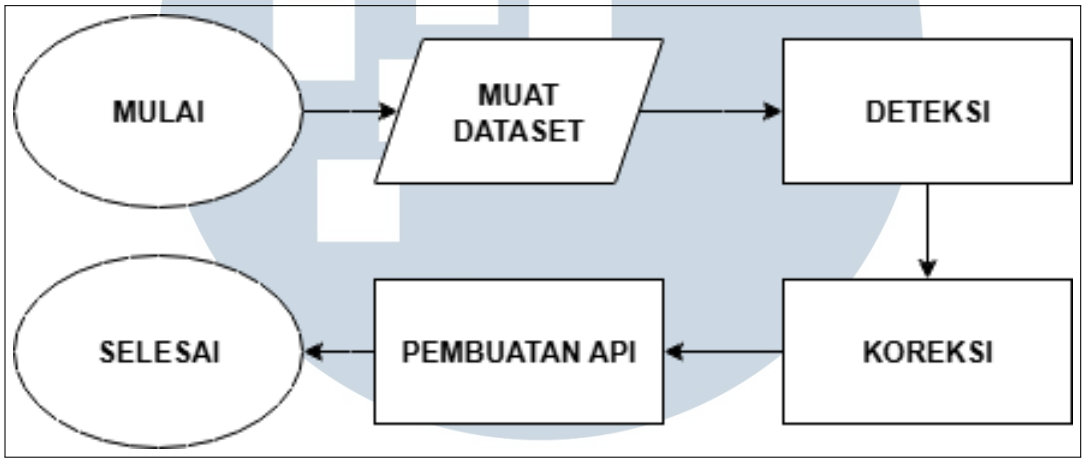
BAB 3
METODOLOGI PENELITIAN

3.1 Tahapan Penelitian

Penelitian ini dilakukan melalui tahapan-tahapan sebagaimana ditunjukkan pada Gambar 3.1. Proses ini bertujuan untuk menghasilkan model yang mampu mendeteksi kesalahan dalam penggunaan tanda baca.

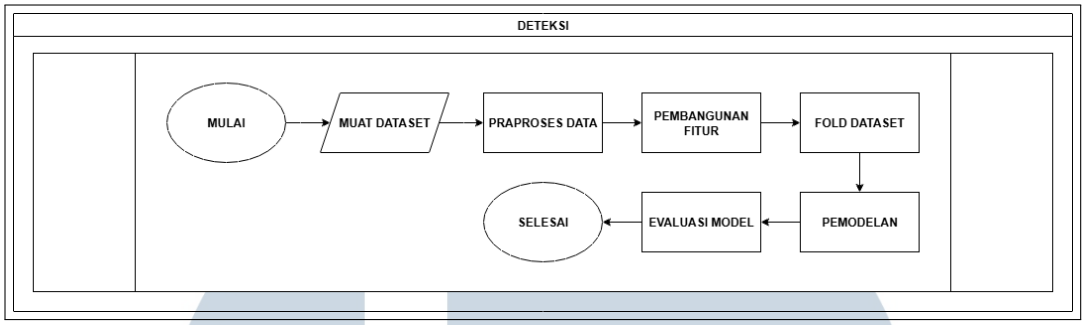


Gambar 3.1. Metodologi Penelitian

3.2 Deteksi

Proses deteksi dimulai dengan pengumpulan data dari sumber yang relevan. Data yang terkumpul kemudian diproses melalui tahap praproses untuk memastikan kualitas dan konsistensi. Selanjutnya, dilakukan pembangunan fitur atau penyusunan indikator untuk mengekstraksi elemen penting dari data. Setelah itu, data dibagi ke dalam beberapa bagian guna mendukung proses validasi atau pengujian. Tahap pemodelan atau analisis dilakukan untuk mengevaluasi hubungan antar variabel atau pola yang terdapat dalam data. Hasil yang diperoleh kemudian dievaluasi untuk menilai kesesuaian terhadap tujuan yang telah ditetapkan. Terakhir, proses penelitian ditutup dengan penyimpulan hasil dan penyusunan rekomendasi berdasarkan temuan yang ada.

Diagram ini menunjukkan tahapan utama dalam proses deteksi kesalahan tanda baca sesuai dengan Gambar 3.2. Proses dimulai dengan pengumpulan data yang merupakan tahap awal untuk menghimpun korpus yang akan digunakan.



Gambar 3.2. Deteksi

Selanjutnya, dilakukan pembersihan dan normalisasi data melalui tahap praproses dataset agar data siap diproses oleh model. Tahap berikutnya adalah pembangunan fitur, yaitu proses pembentukan fitur-fitur linguistik dan sintaksis yang akan digunakan untuk mendeteksi kesalahan. Setelah fitur dibentuk, data dibagi menjadi subset pelatihan, validasi, dan pengujian pada tahap *fold dataset*. Tahapan ini diikuti oleh proses pemodelan untuk membangun model deteksi berbasis RNN. Proses diakhiri dengan evaluasi model guna mengukur akurasi dan kinerja model yang dihasilkan.

3.2.1 Deteksi: Pengumpulan data

Pengumpulan dataset terdiri dari tiga bagian; Pengumpulan dataset utama, dua dataset pendukung, dan satu dataset sintesis. Seluruh dataset dikumpulkan ke dalam satu folder

A Dataset Utama

Dataset yang dikumpulkan berupa 10.000 dataset berita mentah dari Tribunnews.com dengan tiga format, `*.pdf`, `*.docx`, dan `*.txt`. Seluruh data yang mempunyai format `*.pdf` dan `*.docx` dikonversi menjadi data dengan format `*.txt`, kemudian seluruh data dikonversi menjadi `*.csv` untuk mempermudah pengolahan data.

B Dataset Pendukung

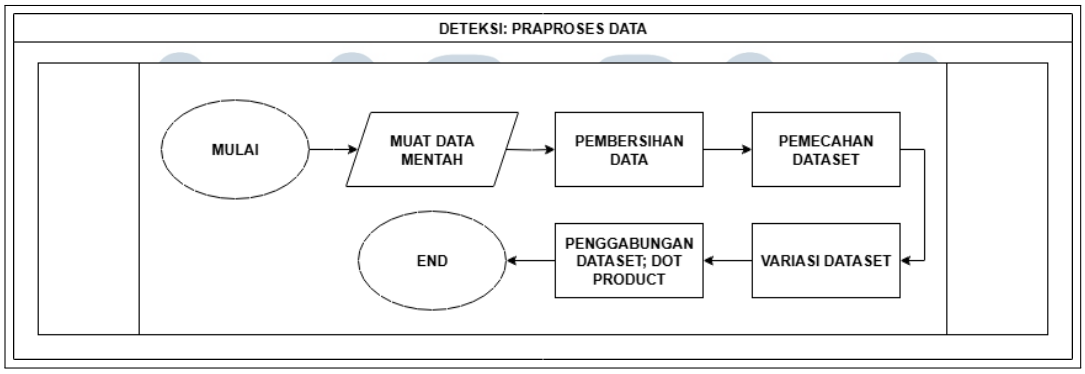
Dataset referensi yang terdiri dari 2.203 gelar akademik [33–46] dan 146.281 dataset kat baku dar KBBI V yang dimanfaatkan untuk mengidentifikasi kata-kata yang tidak boleh dihapus atau terpotong selama proses pemrosesan data,

karena mengandung tanda baca yang tidak menjadi fokus utama penelitian. Proses pengumpulan *dataset* ini dilakukan melalui dua pendekatan:

1. ***Scraping***: Digunakan ketika data tersedia dalam bentuk tabel. Pengambilan data dilakukan secara langsung dari situs web dengan menggunakan teknik *web scraping*.
2. **Input Manual**: Diterapkan pada sumber yang berbentuk naratif, terutama jika informasi tambahan perlu diperoleh dari situs lain untuk melengkapi data.

3.2.2 Deteksi: Praproses Data

Sebelum digunakan dalam tahap pelatihan model, data perlu diproses agar sesuai dengan format dan struktur yang dibutuhkan oleh sistem deteksi. Proses praproses ini bertujuan untuk menyiapkan data mentah menjadi representasi yang bersih, terorganisir, dan relevan dengan tujuan analisis. Tahapan ini mencakup segmentasi teks, penghapusan karakter yang tidak diperlukan, normalisasi huruf, serta persiapan struktur kalimat. Seluruh tahapan praproses dirancang untuk memastikan data dapat dipahami dan diproses secara optimal oleh algoritma pembelajaran mesin.



Gambar 3.3. Urutan *Praproses Data*

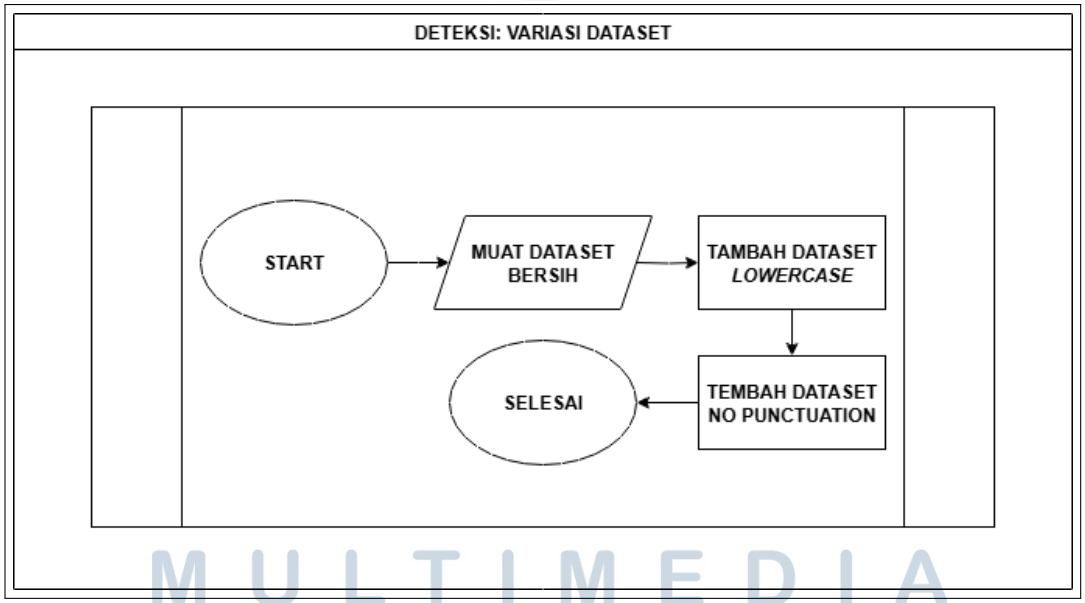
Dataset yang menjadi bahan pemodelan awalnya berada dalam kondisi tidak terstruktur. Oleh karena itu, sebelum fitur dapat diekstraksi dan diterapkan secara optimal, dilakukan serangkaian tahap praproses terhadap *dataset* utama yang telah dikumpulkan. Alur lengkap tahapan praproses ini diperlihatkan pada Gambar 3.3.

3.2.3 Pembersihan data

Langkah berikutnya adalah pembersihan data, yaitu proses pemurnian data dengan menghapus elemen-elemen yang tidak berkontribusi pada analisis selanjutnya. Elemen yang dikeluarkan meliputi judul artikel, *news header*, serta *news footer*, karena bagian tersebut merupakan informasi tambahan yang tidak termasuk ke dalam isi pokok berita.

A Variasi dan Pemecahan Dataset

Setelah proses pembersihan, langkah selanjutnya adalah membuat variasi data guna memperkaya representasi input yang akan digunakan dalam pelatihan model. Teknik variasi ini dilakukan dengan memecah setiap kalimat menjadi dua bagian, lalu mengubahnya ke dalam beberapa bentuk: huruf kecil dan tanpa tanda baca. Strategi ini tidak hanya membantu dalam menambah keragaman data, tetapi juga penting dalam mengatasi ketidakseimbangan distribusi data. Diagram pada Gambar 3.4 menggambarkan alur lengkap dari proses variasi dan pemecahan data tersebut.



Gambar 3.4. Urutan Variasi Data

Merujuk pada Gambar 3.4, *dataset* dibagi terlebih dahulu menjadi dua bagian sebelum dilakukan proses variasi, sehingga menghasilkan enam kolom: *Kalimat I*, *Kalimat II*, *lowercase I*, *lowercase II*, *no punctuation I*, dan *no*

punctuation II. Pembagian ini dilakukan untuk memperoleh penyebaran data yang lebih merata dan mengurangi risiko terjadinya *data imbalance* dalam proses pelatihan model.

B *Dot Product Pairing Dataset*

Penggabungan seluruh kolom dalam *dataset* secara bebas akan menghasilkan data yang bersifat *invalid*, karena kolom seperti *lowercase* dan *no punctuation* tidak merepresentasikan struktur kalimat yang benar. Untuk menghindari hal tersebut, dilakukan proses kombinasi silang (*cross-pairing*) antar dua klaster: klaster 1 (kolom dengan akhiran *I*) sebagai Kalimat 1, dan klaster 2 (kolom dengan akhiran *II*) sebagai Kalimat 2.

Kombinasi hanya diperbolehkan antara kolom dari kluster yang berbeda, dengan urutan tetap: Kalimat 1 berasal dari kluster 1 dan Kalimat 2 berasal dari kluster 2. Kombinasi antar kolom dalam kluster yang sama tidak diizinkan. Dengan masing-masing kluster terdiri atas tiga kolom.

Kombinasi ini mencakup pasangan data yang valid (misalnya: *Kalimat I* dan *Kalimat II*) maupun tidak valid (misalnya: *Kalimat I* dan *no punctuation II*), dan digunakan untuk melatih model dalam mengenali berbagai bentuk kesalahan tanda baca.

C Pemilihan acak Dataset

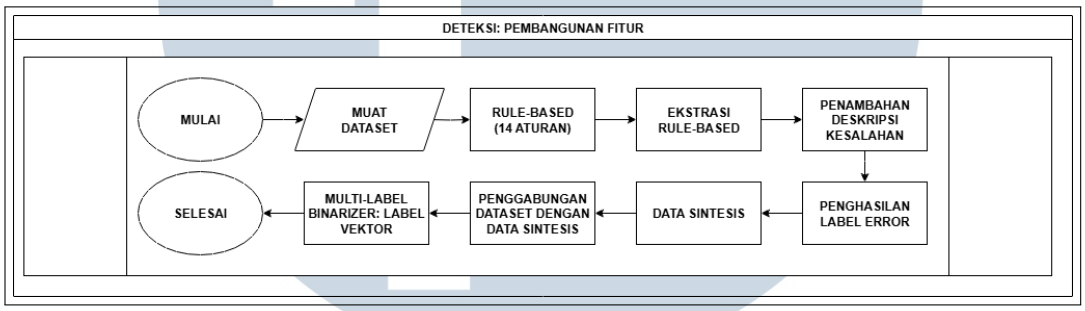
Dari sembilan kolom hasil *product*, dibuat sebuah kolom baru dengan nama *selected*. Kolom ini berisi satu pasangan yang dipilih secara acak dari sembilan kemungkinan pada setiap baris data. Pemilihan acak ini bertujuan untuk meningkatkan keragaman data yang digunakan dalam pelatihan serta mengurangi kompleksitas komputasi.

D Hapus Kolom

Langkah selanjutnya adalah menyederhanakan struktur *dataset* dengan menghapus seluruh kolom selain kolom *selected*. Kolom *selected* ini menjadi representasi akhir dari tahap *preprocessing*, dan akan digunakan sebagai masukan utama dalam proses pendeteksian kesalahan tanda baca.

3.2.4 *Deteksi: Pembangunan Fitur*

Sebelum data digunakan dalam pelatihan model, dilakukan proses pembangunan fitur untuk mengekstraksi informasi penting yang relevan dari data teks. Proses ini mencakup konversi kata menjadi representasi numerik, seperti tokenisasi dan transformasi menjadi vektor, yang memungkinkan model RNN memahami struktur dan makna kalimat. Tahapan ini bersifat krusial karena kualitas fitur yang dibangun secara langsung memengaruhi performa akhir dari model deteksi kesalahan tanda baca. Gambar 3.5 memperlihatkan alur lengkap pembangunan fitur dalam sistem yang dikembangkan.



Gambar 3.5. Urutan *Feature Engineering*

Pada langkah ini, data hasil dari praproses akan diproses lebih lanjut guna menunjang peningkatan akurasi dan efisiensi dari model RNN yang akan digunakan. Data ini kemudian dimanfaatkan dalam tahapan pelatihan dan prediksi terhadap kesalahan-kesalahan penggunaan tanda baca, sebagaimana diperlihatkan pada Gambar 3.5.

A *Rule-Based*

Pada tahap ini, dibuat sebanyak 14 aturan untuk mendeteksi kesalahan tanda baca. Aturan-aturan ini disusun berdasarkan pedoman EYD V [19–22] dan juga disesuaikan dengan kebutuhan jurnalis dalam menyunting tulisan. Daftar label untuk masing-masing aturan ditampilkan sebagai berikut:

1. Kalimat tidak diakhiri dengan tanda baca.,
2. Kalimat tanya tidak diakhiri dengan tanda tanya.,
3. Tanda baca tidak diikuti huruf kecil.,

4. Harus ada tanda baca sebelum huruf besar.,
5. Kalimat tidak diakhiri dengan lebih dari satu tanda seru.,
6. Tidak ada koma sebelum dan/atau.,
7. Koma setelah frasa kalau/karena/agar.,
8. Tidak ada koma sebelum tanda kutip.,
9. Tidak ada koma sebelum tetapi/melainkan/sedangkan.,
10. Tidak ada koma setelah Oleh karena itu/Jadi/Meskipun demikian.,
11. Tidak ada koma setelah nama Jalan/Jl.,
12. Tidak ada koma sebelum gelar., dan
13. Tidak ada koma setelah frasa Dalam/Atas.

B *Ektrasi Rule-Based*

Ekstraksi dari aturan dilakukan untuk menerapkan logika-logika tertentu yang bisa membantu dalam mengenali pola, menentukan keputusan, atau mengklasifikasikan data. Dengan cara ini, proses analisis bisa lebih mudah dalam menemukan hubungan atau pola yang terlihat jelas di dalam data.

C Pemasangan Label Deskripsi Kesalahan

Langkah ini bertujuan untuk memberikan penjelasan terkait kesalahan penggunaan tanda baca pada data yang telah melewati tahap praproses. Jika terdapat lebih dari satu jenis kesalahan dalam satu kalimat, maka semua penjelasannya akan digabung menjadi satu. Sebaliknya, jika tanda baca pada kalimat sudah benar, maka bagian penjelasan akan diberikan label ”Penggunaan tanda baca sudah benar”.

D Data Sintesis

Dalam penelitian ini, dilakukan proses sintesis data guna mengatasi ketidakseimbangan jumlah data pada masing-masing jenis kesalahan tanda baca. Data sintesis dibuat dengan menggunakan pola kalimat umum yang dimodifikasi secara sistematis untuk mencerminkan berbagai jenis kesalahan tanda baca yang

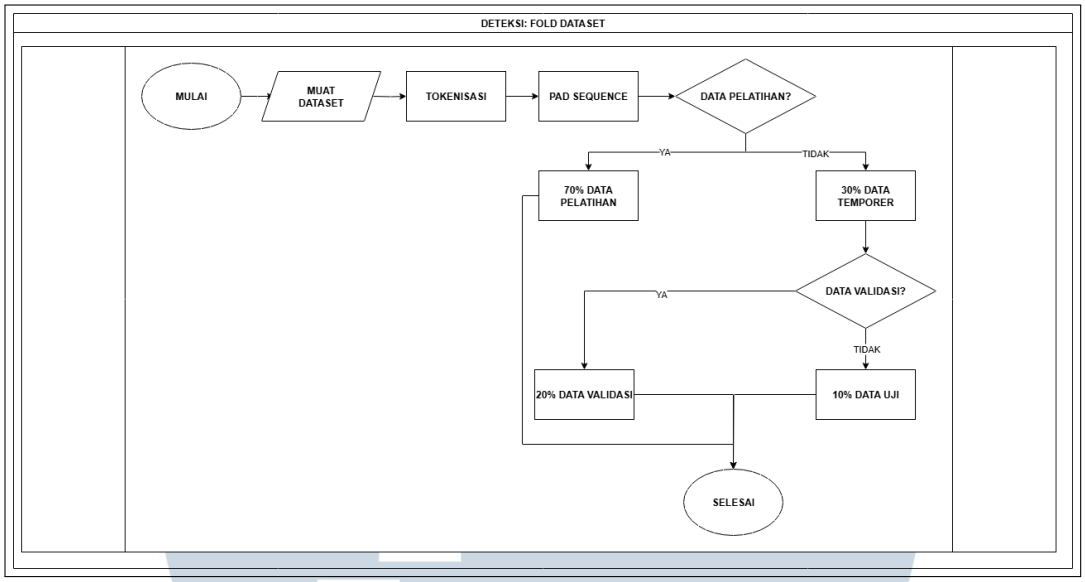
telah didefinisikan sebelumnya. Tujuan dari sintesis ini adalah untuk memastikan setiap label memiliki jumlah data yang memadai dan seimbang, sehingga model dapat belajar secara optimal tanpa bias terhadap label mayoritas.

E Multi-Label Binarizer (Label Vektor)

Setelah proses sintesis selesai, setiap kalimat dalam *dataset* dapat mengandung lebih dari satu jenis kesalahan tanda baca. Oleh karena itu, digunakan pendekatan multi-label untuk menangani skenario ini. Implementasi dilakukan menggunakan *MultiLabelBinarizer* dari pustaka `scikit-learn`. Proses ini mengubah daftar label dari setiap entri data menjadi vektor biner, di mana setiap posisi dalam vektor merepresentasikan keberadaan (1) atau ketiadaan (0) dari sebuah label kesalahan. Dengan format vektor biner ini, model dapat dilatih untuk melakukan klasifikasi multi-label secara efisien, memungkinkan prediksi lebih dari satu jenis kesalahan tanda baca pada sebuah kalimat.

3.2.5 *Deteksi: Fold Dataset*

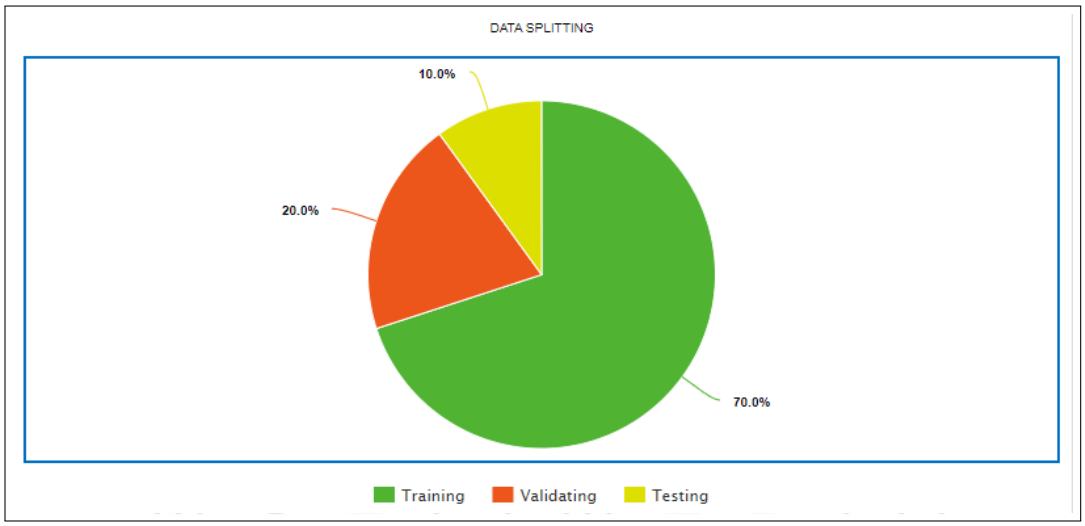
Tahapan *fold dataset* sebagaimana ditunjukkan pada Gambar 3.6 digunakan untuk mengevaluasi konsistensi kinerja dari model RNN yang dibangun. Proses ini bertujuan untuk mencegah terjadinya *overfitting* dan mengurangi bias terhadap kelas tertentu dalam data, sehingga model mampu melakukan generalisasi dengan lebih baik.



Gambar 3.6. Urutan *Fold dataset*

A Pemecahan Dataset

Proses ini bertujuan untuk membagi dataset ke dalam beberapa subset berdasarkan proporsi tertentu, yang masing-masing digunakan untuk proses pelatihan, validasi, dan pengujian model.



Gambar 3.7. *Data Splitting*

Merujuk pada Gambar 3.7, pembagian data dilakukan sebagai berikut:

1. 70% untuk data pelatihan,

2. 20% untuk data validasi, dan

3. 10% untuk data pengujian.

B Tokenisasi

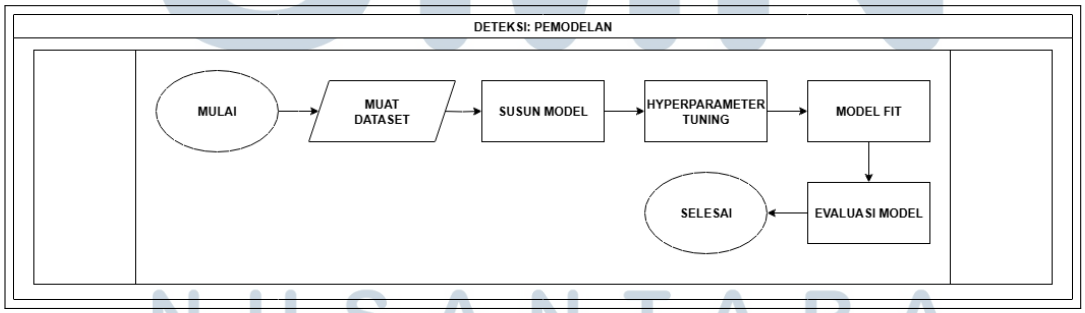
Untuk mempermudah proses pemodelan, kolom *selected* dalam dataset yang semula berupa teks diubah menjadi format numerik melalui proses tokenisasi. Dengan demikian, model dapat mengenali dan mempelajari pola-pola dalam data secara lebih efektif.

C *Pad Sequence*

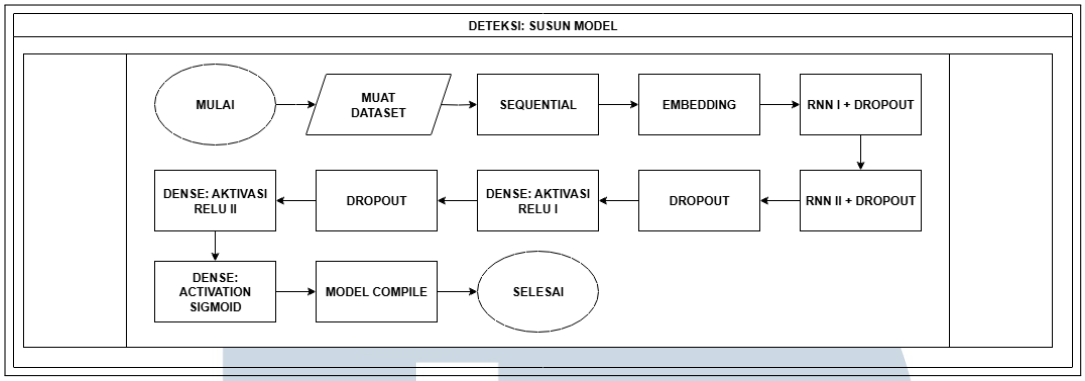
Setelah proses tokenisasi dilakukan, setiap urutan token yang dihasilkan kemudian disamakan panjangnya melalui penambahan nilai nol (*padding*). Langkah ini bertujuan untuk memastikan setiap input memiliki dimensi yang seragam, sehingga proses pelatihan model dalam bentuk *batch* dapat berjalan dengan efisien.

3.2.6 *Deteksi: Pemodelan*

Tahapan ini menjelaskan proses pembuatan dan pelatihan model untuk mendeteksi kesalahan penggunaan tanda baca sesuai dengan Gambar 3.8. Proses dimulai dari pemuatan dataset yang telah dibersihkan dan diproses sebelumnya. Selanjutnya, data disusun ke dalam model berarsitektur *sequential* seperti pada Gambar 3.9.



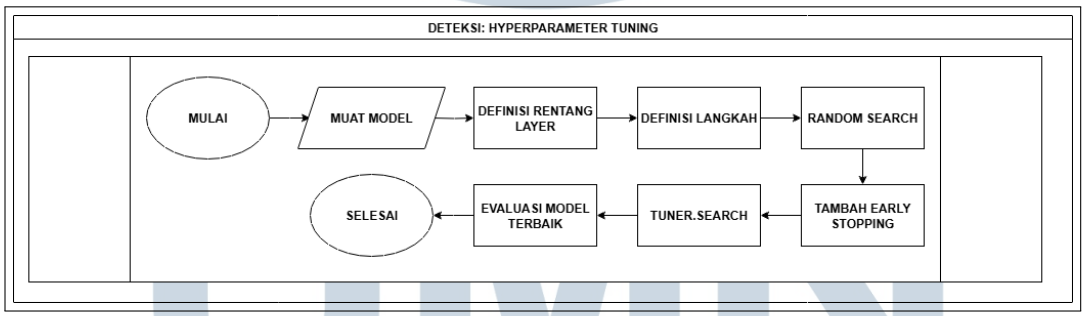
Gambar 3.8. Urutan *Modelling*



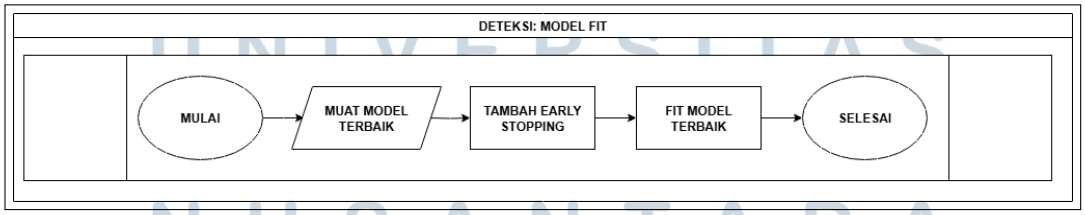
Gambar 3.9. Urutan *Stack Model*

Data yang masih berbentuk teks dikonversi ke bentuk vektor melalui proses *embedding*, yang memungkinkan representasi kata dalam dimensi numerik. Tahapan berikutnya adalah penerapan dua lapisan *SimpleRNN* secara berurutan, masing-masing dilengkapi dengan lapisan *Dropout* untuk mencegah *overfitting*.

Setelah itu, jaringan dilanjutkan dengan lapisan *Dense* beraktivasi *ReLU*, diselingi dengan *Dropout* tambahan, dan diakhiri dengan lapisan *Dense* beraktivasi *sigmoid* untuk melakukan klasifikasi biner.



Gambar 3.10. Urutan *Parameter Tuning*



Gambar 3.11. *Model Fitting*

Model dikompilasi dengan fungsi aktivasi dan parameter pelatihan yang sesuai menggunakan *Hyperparameter tuning* seperti pada Gambar 3.10.

Selanjutnya, diterapkan teknik *early stopping* untuk menghentikan pelatihan secara otomatis apabila tidak terjadi peningkatan performa pada data validasi. Untuk mengoptimalkan arsitektur, dilakukan proses pencarian hiperparameter menggunakan metode *Random Search*. Setelah konfigurasi terbaik diperoleh, model dilatih secara penuh menggunakan data yang telah disiapkan yang mengacu pada Gambar 3.11.

Diagram alur ini menggambarkan keseluruhan proses dari awal (*start*) hingga akhir (*end*) dalam membangun model deteksi berbasis jaringan saraf *Recurrent Neural Network (RNN)*.

A Deteksi: Evaluasi Model

Evaluasi terhadap performa model dilakukan melalui serangkaian tahapan penting yang bertujuan untuk mengukur efektivitas klasifikasi model. Langkah-langkah evaluasi dijabarkan sebagai berikut:

1. **Kurva ROC dan Penentuan Threshold Optimal:**
   - Kurva *Receiver Operating Characteristic (ROC)* digunakan untuk melihat kemampuan model dalam membedakan antara kelas positif dan negatif.
   - Probabilitas dari hasil prediksi terhadap data validasi (`y_val_probs`) diperoleh melalui fungsi `model.predict`.
   - Fungsi `roc_curve` dari `sklearn.metrics` menghasilkan tiga komponen utama:
     - `fpr` (*False Positive Rate*)
     - `tpr` (*True Positive Rate*)
     - `thresholds` (nilai ambang klasifikasi)
   - Threshold optimal ditentukan dengan memilih nilai ambang yang memaksimalkan selisih antara TPR dan FPR, menggunakan rumus berikut:

```
optimal_threshold = thresholds[np.argmax(tpr - fpr)]
```

2. **Pengujian Model pada Data Uji:**

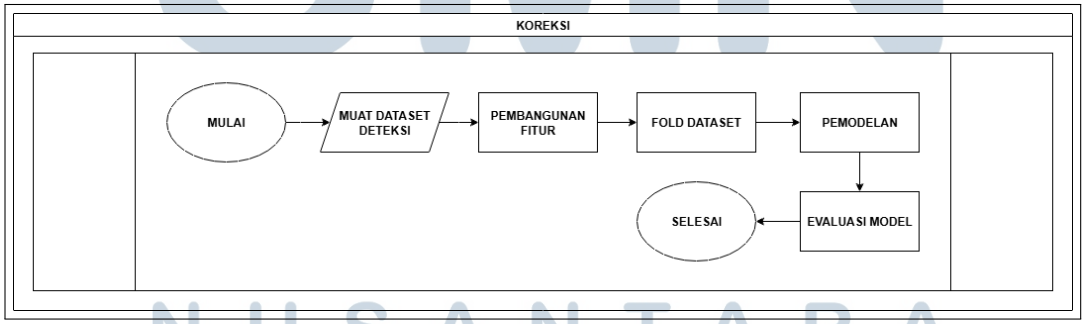
- Model digunakan untuk memprediksi kelas pada data uji (`X_test`) dengan menggunakan nilai threshold optimal:

  `y_pred_test = (model.predict(X_test) ≥ optimal_threshold).astype(int)`

- Hasil prediksi dibandingkan dengan label sebenarnya (`y_test`) melalui fungsi `classification_report`.
- Laporan tersebut mencakup berbagai metrik evaluasi:
  - **Precision:** Persentase prediksi positif yang benar.
  - **Recall:** Proporsi kelas positif yang berhasil diidentifikasi model.
  - **F1-Score:** Nilai rata-rata harmonis dari *precision* dan *recall*.
  - **Support:** Jumlah observasi pada masing-masing kelas.

3.3 Koreksi

Setelah proses deteksi berhasil mengidentifikasi keberadaan kesalahan tanda baca dalam kalimat, tahap selanjutnya adalah memberikan saran perbaikan atau koreksi yang sesuai. Proses koreksi ini tidak hanya bergantung pada hasil deteksi, tetapi juga memerlukan pemodelan tambahan yang dirancang khusus untuk memprediksi bentuk tanda baca yang tepat. Agar sistem koreksi dapat bekerja secara efektif, diperlukan pendekatan metodologis yang sistematis yang mencakup beberapa tahapan utama mulai dari pemrosesan ulang data hingga evaluasi akhir terhadap performa model. Alur lengkap metodologi koreksi ditampilkan pada Gambar 3.12.



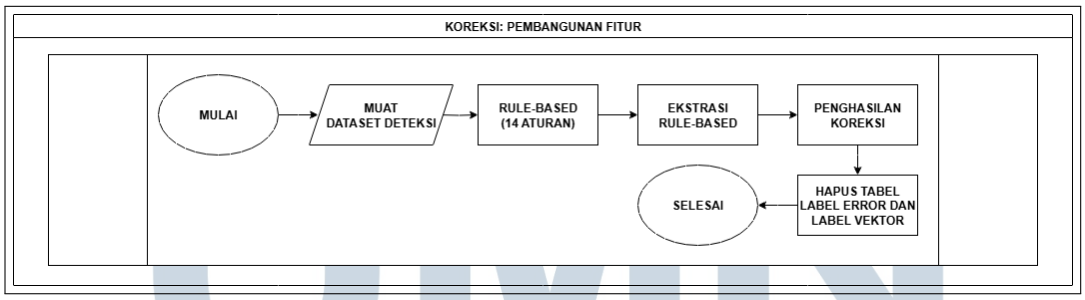
Gambar 3.12. Alur metodologi koreksi tanda baca

Diagram pada Gambar 3.12 menjelaskan alur metodologi untuk tahap pemberian saran perbaikan terhadap kesalahan tanda baca. Proses dimulai dengan

memuat data hasil dari *feature engineering* pada tahap *detection*. Kemudian dilakukan kembali proses *feature engineering* lanjutan untuk menyiapkan fitur tambahan yang dibutuhkan pada tahap saran. Setelah itu, dataset dibagi menjadi tiga subset (*training*, *validation*, dan *testing*) pada tahap *fold dataset*. Data yang telah dibagi digunakan dalam proses *modelling*, yaitu pelatihan model untuk menghasilkan saran koreksi. Tahap terakhir adalah *model evaluation* untuk mengukur performa dari model yang telah dibangun.

3.3.1 Koreksi: Pembangunan Fitur

Tahap pembangunan fitur merupakan langkah krusial dalam sistem koreksi kesalahan tanda baca karena mempengaruhi kualitas hasil yang akan dihasilkan oleh model. Pada tahap ini, data hasil deteksi diproses lebih lanjut untuk diekstraksi menjadi sejumlah fitur yang merepresentasikan pola-pola kesalahan berdasarkan aturan linguistik yang telah ditetapkan. Proses ini bertujuan agar sistem memiliki representasi data yang kaya dan relevan sebelum masuk ke proses koreksi otomatis. Diagram proses *feature engineering* untuk modul koreksi ditunjukkan pada Gambar 3.13.

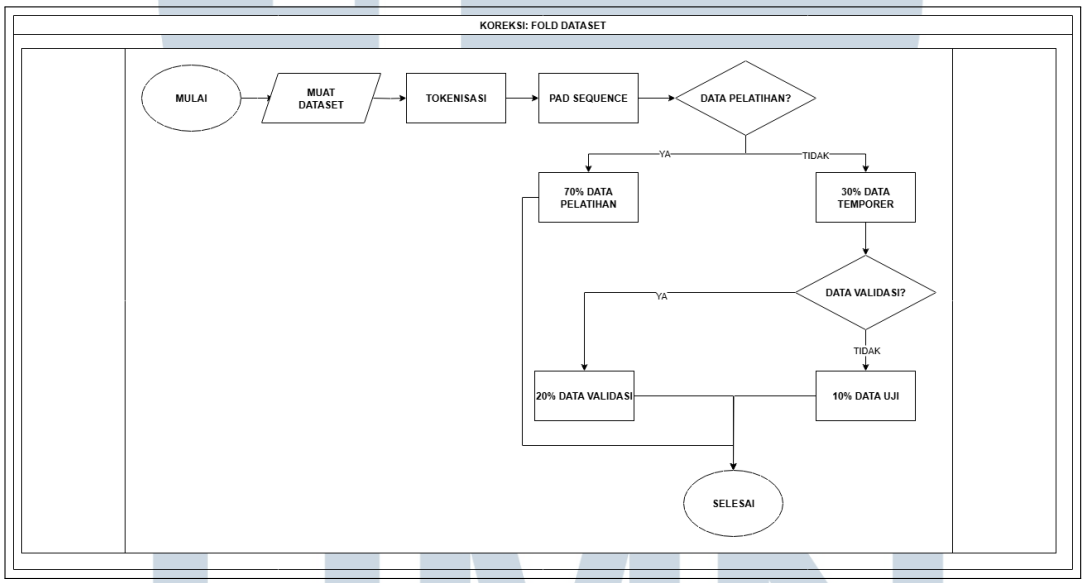


Gambar 3.13. *Feature Engineering* Koreksi

Mengacu pada Gambar 3.13, tahapan awal dalam proses *suggestion* adalah melakukan *feature engineering* terhadap data hasil deteksi. Proses ini diawali dengan memuat *dataset* dan menerapkan 14 aturan berbasis aturan (rule-based) yang telah dirancang sebelumnya untuk mengenali kesalahan tanda baca. Setiap aturan diterapkan untuk mengekstraksi fitur berbasis aturan dari kalimat yang dianalisis. Setelah fitur berhasil diekstraksi, sistem akan membangkitkan hasil koreksi berdasarkan fitur tersebut. Tahap ini diakhiri dengan menghapus kolom label kesalahan dan membentuk vektor label koreksi, yang akan digunakan pada tahap pemodelan selanjutnya.

3.3.2 Koreksi: Fold Dataset

Setelah fitur-fitur berhasil dibangun dari data hasil deteksi, proses selanjutnya adalah menyiapkan data tersebut untuk digunakan dalam pelatihan dan evaluasi model. Langkah ini dikenal sebagai proses *folding dataset* atau pembagian dataset, yang bertujuan untuk memastikan bahwa model dapat belajar secara efektif serta diuji secara adil dan terukur. Proses ini juga penting untuk menghindari *overfitting* dan memastikan bahwa performa model dapat digeneralisasi ke data yang belum pernah dilihat sebelumnya. Ilustrasi tahapan pembagian dataset dalam sistem koreksi ditunjukkan pada Gambar 3.14.

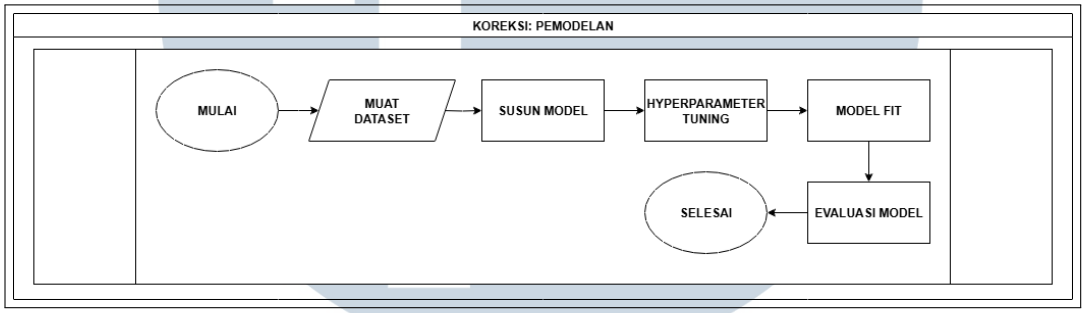


Gambar 3.14. *Fold Dataset* Koreksi

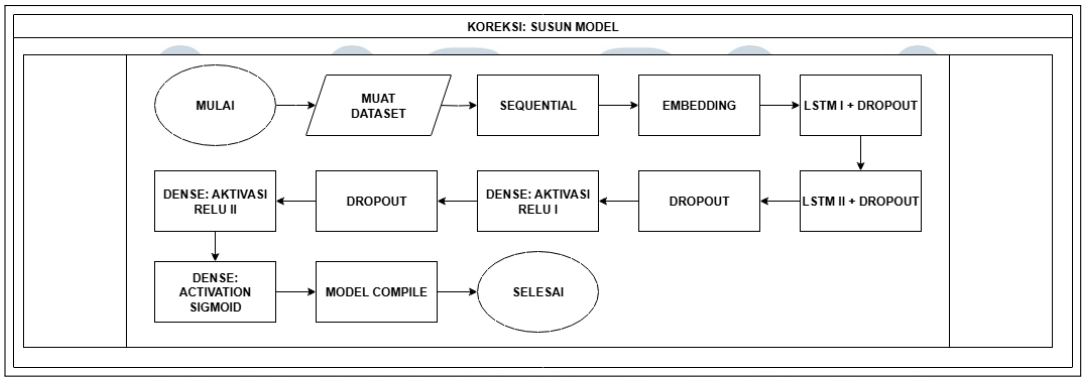
Setelah proses *feature engineering* selesai, langkah berikutnya adalah membagi *dataset* untuk keperluan pelatihan, validasi, dan pengujian model seperti pada Gambar 3.14. Dataset yang telah dibersihkan dan diberi label dibagi menjadi tiga bagian utama, yaitu 70% untuk pelatihan, 20% untuk validasi, dan 10% untuk pengujian. Dataset terlebih dahulu dimuat, kemudian dilakukan proses tokenisasi pada kolom *selected*, diikuti dengan proses *pad sequence* untuk memastikan panjang token seragam. Pembagian dataset dilakukan agar proses pelatihan dan evaluasi model dapat berjalan optimal dan adil.

3.3.3 Koreksi: Pemodelan

Tahap pemodelan merupakan inti dari sistem koreksi otomatis, di mana model pembelajaran mesin atau pembelajaran mendalam dilatih untuk memberikan prediksi koreksi berdasarkan fitur yang telah dibangun sebelumnya. Dalam konteks koreksi tanda baca, pemodelan dilakukan untuk mengidentifikasi dan mengoreksi kesalahan berdasarkan konteks linguistik dalam kalimat. Model yang dirancang harus mampu memahami urutan kata dan struktur kalimat agar dapat memberikan koreksi yang akurat. Visualisasi arsitektur umum dan struktur bertingkat dari model koreksi ditunjukkan pada Gambar 3.15 dan Gambar 3.16.



Gambar 3.15. *Modelling* Koreksi

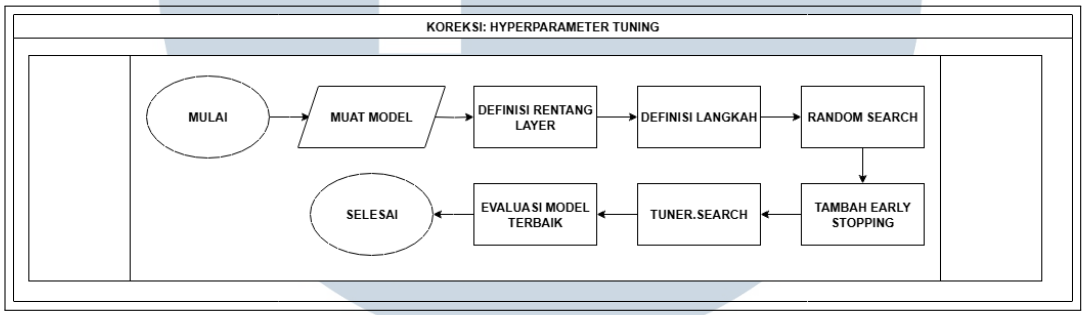


Gambar 3.16. *Stack Modelling* Koreksi

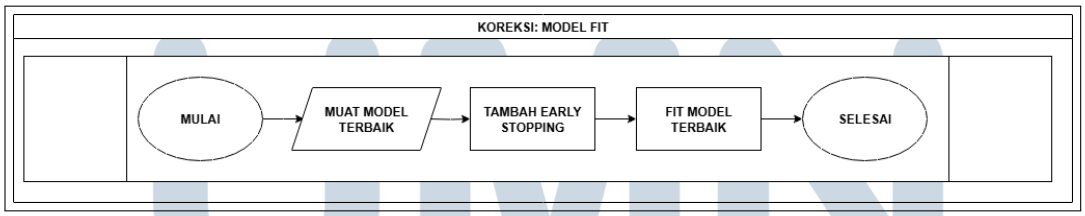
Tahapan ini menjelaskan proses perancangan dan pelatihan model untuk melakukan koreksi terhadap kesalahan penggunaan tanda baca, sebagaimana digambarkan pada Gambar 3.15. Proses diawali dengan pemuatan dataset yang telah melalui tahap pembersihan dan praproses sebelumnya. Data tersebut kemudian dipersiapkan dalam struktur model berarsitektur *sequential* seperti pada Gambar 3.16.

Langkah selanjutnya adalah mengubah data teks menjadi representasi numerik melalui layer *embedding*, yang bertujuan untuk memetakan setiap kata ke dalam ruang vektor berdimensi tetap. Model ini selanjutnya menggunakan dua lapisan *Long Short-Term Memory (LSTM)* secara bertingkat. Masing-masing lapisan LSTM dilengkapi dengan parameter *dropout* dan *recurrent dropout* guna mengurangi risiko *overfitting* dan meningkatkan kemampuan generalisasi model.

Setelah melewati lapisan LSTM, jaringan dilanjutkan dengan beberapa lapisan *Dense* beraktivasi *ReLU*, diselingi oleh lapisan *Dropout*, dan ditutup dengan layer *Dense* beraktivasi *sigmoid* yang berfungsi sebagai output untuk klasifikasi bineryaitu memberikan koreksi atau tidak terhadap suatu kesalahan tanda baca.



Gambar 3.17. Urutan *Parameter Tuning*



Gambar 3.18. *Model Fitting untuk Koreksi Tanda Baca*

Merujuk pada Gambar 3.17, model dikompilasi dengan fungsi aktivasi dan konfigurasi pelatihan terbaik yang diperoleh melalui proses *Hyperparameter Tuning*. Dalam pelatihannya, digunakan strategi *early stopping* untuk menghentikan pelatihan secara otomatis ketika tidak ada lagi peningkatan pada data validasi. Proses pemilihan konfigurasi optimal dilakukan menggunakan metode *Random Search*. Setelah konfigurasi terbaik ditentukan, model dilatih secara penuh untuk menghasilkan sistem koreksi tanda baca berbasis jaringan saraf LSTM seperti pada Gambar 3.18.

3.3.4 Koreksi: Evaluasi Model

Evaluasi terhadap model pada tahap Koreksi dilakukan untuk mengukur seberapa baik model dapat mengklasifikasikan kalimat yang membutuhkan koreksi tanda baca. Proses evaluasi ini melibatkan beberapa langkah utama, yaitu:

1. **Penentuan Threshold Optimal dengan Kurva ROC:**
   - Prediksi probabilitas terhadap data uji (`X_test`) dilakukan terlebih dahulu menggunakan fungsi `model.predict`.
   - Hasil prediksi tersebut digunakan untuk membentuk kurva *Receiver Operating Characteristic (ROC)* menggunakan fungsi `roc_curve` dari `sklearn.metrics`, yang menghasilkan tiga komponen utama:
     - `fpr` (*False Positive Rate*)
     - `tpr` (*True Positive Rate*)
     - `thresholds` (nilai ambang klasifikasi)
   - Threshold terbaik dipilih dengan mencari selisih maksimum antara nilai TPR dan FPR, untuk memperoleh pemisahan kelas yang optimal, yaitu:

     ```
     best_threshold = thresholds[np.argmax(tpr - fpr)]
     ```

2. **Evaluasi Model pada Data Uji:** Setelah proses pelatihan selesai dan threshold optimal ditentukan, langkah selanjutnya adalah mengevaluasi performa model pada data uji yang tidak pernah digunakan sebelumnya. Evaluasi ini bertujuan untuk mengukur kemampuan generalisasi model dalam mengklasifikasikan data baru secara akurat. Dengan menggunakan probabilitas hasil prediksi dan ambang batas yang telah ditentukan sebelumnya, sistem dapat mengubah nilai probabilitas menjadi label klasifikasi biner. Selanjutnya, hasil klasifikasi ini dibandingkan dengan label sebenarnya untuk menghitung berbagai metrik evaluasi model.
   - Model melakukan klasifikasi biner pada data uji berdasarkan threshold yang telah diperoleh:

     ```
     y_pred = (y_probs ≥ best_threshold).astype(int)
     ```

- Nilai prediksi tersebut kemudian dibandingkan dengan label asli (`y_test`) untuk memperoleh metrik performa model menggunakan fungsi `classification_report`.
- Laporan evaluasi menampilkan empat metrik penting:
  - **Precision:** Proporsi hasil prediksi positif yang benar.
  - **Recall:** Kemampuan model dalam mengidentifikasi kelas positif secara akurat.
  - **F1-Score:** Rata-rata harmonis antara nilai *precision* dan *recall*.
  - **Support:** Jumlah sampel aktual dalam masing-masing kelas.