

BAB II

TINJAUAN PUSTAKA

2.1 Penelitian yang Relevan

Penelitian ini berfokus pada implementasi *spelling checker* Bahasa Indonesia menggunakan teknik *text preprocessing* dan model *Convolutional Neural Network* pada data ulasan PLN *Mobile*. Oleh karena itu, penelitian terdahulu yang digunakan sebagai rujukan adalah penelitian yang berkaitan dengan *spelling checker*, *spelling correction*, korpus kesalahan ejaan Bahasa Indonesia, normalisasi teks, model *deep learning*, serta penelitian yang menggunakan data ulasan aplikasi PLN *Mobile*. Penelitian-penelitian tersebut menjadi dasar dalam menentukan metode, objek data, dan celah penelitian yang dikembangkan dalam penelitian ini.

Penelitian pertama dilakukan oleh (Setiabudi et al., 2021) dengan judul ***Enhancing Text Classification Performance by Preprocessing Misspelled Words in Indonesian Language***. Penelitian ini membahas pengaruh perbaikan kata salah tulis pada tahap *preprocessing* terhadap performa klasifikasi teks berbahasa Indonesia. Data yang digunakan berupa teks *user feedback* berbahasa Indonesia. Metode yang digunakan adalah algoritma Levenshtein Distance untuk menangani kata salah tulis sebelum data digunakan pada model klasifikasi. Hasil penelitian menunjukkan bahwa *preprocessing* terhadap kata salah tulis mampu meningkatkan akurasi model sebesar 8,2%. Penelitian ini relevan karena menunjukkan bahwa penanganan *Typo* atau kata salah tulis pada tahap *preprocessing* dapat meningkatkan kualitas pengolahan teks Bahasa Indonesia. Namun, penelitian tersebut belum menggunakan model CNN dan belum menggunakan data ulasan PLN *Mobile*.

Penelitian kedua dilakukan oleh (Asri et al., 2022) dengan judul ***Pelabelan Otomatis Lexicon Vader dan Klasifikasi Naive Bayes dalam Menganalisis Sentimen Data Ulasan PLN Mobile***. Penelitian tersebut menggunakan 1.000 data ulasan PLN *Mobile* yang diperoleh dari Google Play Store. Metode yang digunakan adalah pelabelan otomatis menggunakan *Lexicon Vader* dan klasifikasi menggunakan Naive Bayes. Hasil penelitian menunjukkan bahwa model memperoleh akurasi sebesar 70%.

Penelitian ini relevan karena menggunakan objek data yang sama, yaitu ulasan PLN *Mobile*. Namun, penelitian tersebut berfokus pada analisis sentimen, bukan pada pengembangan *spelling checker* Bahasa Indonesia. Penelitian ini relevan karena menggunakan objek data yang sama, yaitu ulasan PLN *Mobile*. Namun, penelitian tersebut berfokus pada analisis sentimen, bukan pada pengembangan *spelling checker* Bahasa Indonesia.

Penelitian ketiga dilakukan oleh (Musyafa et al., 2022) dengan judul ***Automatic Correction of Indonesian Grammatical Errors Based on Transformer***. Penelitian tersebut membahas koreksi kesalahan tata bahasa Bahasa Indonesia menggunakan arsitektur Transformer. Penelitian ini menunjukkan bahwa Bahasa Indonesia masih memiliki keterbatasan sumber daya dalam pengembangan sistem koreksi kesalahan otomatis. Meskipun fokusnya adalah *grammatical error correction*, penelitian ini relevan karena sama-sama berkaitan dengan pengembangan sistem koreksi kesalahan pada teks Bahasa Indonesia. Namun, penelitian tersebut berfokus pada kesalahan tata bahasa, bukan pada *spelling checker* tingkat kata menggunakan CNN.

Penelitian keempat dilakukan oleh (Toleu et al., 2022) dengan judul ***Data-Driven Approach for Spellchecking and Autocorrection***. Penelitian tersebut mengembangkan sistem *spell checking* dan *autocorrection* untuk Bahasa Kazakh menggunakan pendekatan berbasis data. Data penelitian diperoleh dari korpus web yang mengandung teks dengan variasi kesalahan penulisan. *Dataset* yang digunakan terdiri dari 244.277 dokumen, 2.043.209 token, dan 27.228 pasangan kata salah dan kata benar. Metode yang digunakan membangun *error model* dan *context model* yang kemudian digabungkan untuk menghasilkan kandidat koreksi kata. Hasil penelitian menunjukkan bahwa sistem QazSpell memperoleh akurasi koreksi sebesar 96% ketika menggunakan model bahasa (*language model*) dengan konfigurasi terbaik, sedangkan tanpa model bahasa akurasi mencapai 89%. Penelitian ini relevan karena menunjukkan efektivitas pendekatan berbasis data dalam membangun sistem *spell checking* dan *autocorrection*. Namun, penelitian tersebut belum menggunakan Bahasa Indonesia, belum menggunakan data ulasan PLN *Mobile*, dan tidak menerapkan model CNN untuk deteksi kesalahan ejaan.

Penelitian kelima dilakukan oleh (Lubis et al., 2023) dengan judul ***Improving Spell Checker Performance for Bahasa Indonesia Using Text preprocessing Techniques with Deep Learning Models***. Penelitian tersebut menggunakan 10.000 data yang berasal dari komunitas media sosial. Data dibagi menjadi 80% data latih atau sebanyak 8.000 data dan 20% data uji atau sebanyak 2.000 data. Metode yang digunakan adalah teknik *text preprocessing* yang meliputi *case folding*, tokenisasi, *stemming*, dan *stopword removal*, kemudian diproses menggunakan model *Convolutional Neural Network* atau CNN. Hasil penelitian menunjukkan bahwa akurasi model berbeda pada setiap skenario *preprocessing*. Akurasi yang diperoleh sebesar 0,86 pada *dataset* tanpa *stemming*, 0,74 tanpa *stopword removal*, 0,70 tanpa *case folding*, dan 0,89 ketika seluruh tahapan *preprocessing* diterapkan. Penelitian ini sangat relevan karena memiliki kesamaan fokus dengan penelitian yang dilakukan, yaitu penerapan teknik *text preprocessing* dan model CNN pada sistem *spelling checker* Bahasa Indonesia. Namun, penelitian tersebut belum menggunakan data ulasan PLN *Mobile* sebagai objek penelitian.

Penelitian keenam dilakukan oleh (Yanfi et al., 2023) dengan judul ***SPECIL: Spell Error Corpus for the Indonesian Language***. Penelitian ini membangun korpus kesalahan ejaan khusus Bahasa Indonesia yang diberi nama SPECIL. Data yang digunakan terdiri dari lebih dari 180.000 token dalam 21.500 kalimat. Korpus tersebut mencakup beberapa jenis kesalahan ejaan, yaitu *non-word error*, *real-word error*, dan *punctuation error*. Penelitian ini menunjukkan bahwa pengembangan sistem *spelling checker* Bahasa Indonesia membutuhkan *dataset* atau korpus yang merepresentasikan kesalahan ejaan secara nyata. Relevansi penelitian ini terletak pada pentingnya pembentukan *dataset* kesalahan ejaan sebagai dasar dalam pengembangan dan evaluasi model *spelling checker*. Namun, penelitian tersebut masih berfokus pada pembentukan korpus dan belum menerapkan model CNN pada data ulasan PLN *Mobile*.

Penelitian ketujuh dilakukan oleh (Yanfi et al., 2024) dengan judul ***Multi-head Attention Based Bidirectional LSTM for Spelling Error Detection in the Indonesian Language***. Penelitian tersebut membahas deteksi kesalahan ejaan Bahasa Indonesia menggunakan pendekatan *deep learning*. Metode yang digunakan adalah Bidirectional Long Short-Term Memory atau Bi-LSTM yang

dikombinasikan dengan *Multi-Head Attention*. Penelitian ini mendeteksi beberapa jenis kesalahan ejaan, seperti penyisipan huruf, penghapusan huruf, penggantian huruf, pertukaran posisi huruf, *real-word error*, dan kesalahan tanda baca. Hasil penelitian menunjukkan bahwa model mencapai akurasi tertinggi sebesar 92,26%. Penelitian ini relevan karena memperkuat bahwa model *deep learning* dapat digunakan untuk mendeteksi kesalahan ejaan Bahasa Indonesia. Namun, metode yang digunakan bukan CNN dan objek data yang digunakan bukan ulasan PLN *Mobile*.

Penelitian kedelapan dilakukan oleh (Mansur et al., 2024) dengan judul ***A Normalization Model for Repeated Letters in Social Media Hate Speech Text Based on Rules and Spelling correction***. Penelitian tersebut membahas normalisasi huruf berulang pada teks media sosial. Metode yang digunakan adalah *rule-based pattern* dan SymSpell. Hasil penelitian menunjukkan bahwa model yang dikembangkan mampu menurunkan jumlah *out-of-Vocabulary* atau OOV menjadi 8%. Penelitian ini relevan karena teks yang dihasilkan pengguna sering mengandung pengulangan huruf, seperti “lemottt”, “baguuus”, atau “erorrr”. Bentuk penulisan seperti ini juga berpotensi muncul pada ulasan aplikasi PLN *Mobile*. Namun, penelitian tersebut belum menggunakan data PLN *Mobile* dan belum berfokus pada *spelling checker* Bahasa Indonesia berbasis CNN.

Penelitian kesembilan dilakukan oleh (Dashti et al., 2024) dengan **judul *PERCORE: A Deep Learning-Based Framework for Persian Spelling Correction with Phonetic Analysis***. Penelitian tersebut mengembangkan sistem *spelling correction* Bahasa Persia menggunakan pendekatan *deep learning* yang dikombinasikan dengan analisis fonetik. Data yang digunakan berasal dari korpus Bahasa Persia dengan jumlah sekitar 1,4 juta dokumen yang digunakan untuk membangun representasi bahasa dan proses *fine-tuning* model. Evaluasi dilakukan menggunakan beberapa *dataset* pengujian dari korpus Hamshahri dengan metrik *precision*, *recall*, dan *F1-score*. Hasil penelitian menunjukkan bahwa model PERCORE mampu mencapai F1-score sebesar 0,891 untuk koreksi kesalahan non-word, 0,890 untuk deteksi kesalahan real-word, dan 0,905 untuk koreksi kesalahan real-word menggunakan kombinasi model *deep learning* dan analisis fonetik. Penelitian ini relevan karena menunjukkan bahwa pendekatan *deep learning*

mampu menangani variasi kesalahan ejaan secara efektif. Namun, penelitian tersebut menggunakan Bahasa Persia dan tidak menerapkan model CNN pada data ulasan pengguna Bahasa Indonesia.

Penelitian kesepuluh dilakukan oleh (Asri, Kuswardani, Ferdinanda, et al., 2025) dengan judul ***Meningkatkan Performa Ulasan Berbahasa Indonesia dengan Spelling corrector Peter Norvig dan Pelabelan SentiStrength_id***. Penelitian tersebut menggunakan 11.004 data ulasan aplikasi PLN *Mobile* dari Google Play Store. Metode yang digunakan meliputi *spelling corrector* Peter Norvig, *preprocessing*, pelabelan *SentiStrength_id*, TF-IDF, dan Support Vector Machine atau SVM. Hasil penelitian menunjukkan bahwa model SVM dengan pembagian data latih dan data uji 90:10 memperoleh akurasi tertinggi sebesar 82%. Penelitian ini sangat relevan karena menggunakan objek yang sama, yaitu ulasan aplikasi PLN *Mobile*. Namun, fokus penelitian tersebut masih pada analisis sentimen, sedangkan *spelling corrector* hanya digunakan sebagai bagian dari tahap *preprocessing*. Berbeda dengan penelitian tersebut, penelitian ini menjadikan *spelling checker* sebagai fokus utama dengan menggunakan model CNN.

Berdasarkan penelitian-penelitian terdahulu tersebut, dapat disimpulkan bahwa pengembangan *spelling checker* dan *spelling correction* telah dilakukan menggunakan berbagai pendekatan, mulai dari metode berbasis kamus, algoritma kemiripan kata, normalisasi teks, hingga model *deep learning*. Penelitian (Setiabudi et al., 2021) menunjukkan bahwa penanganan kata salah tulis pada tahap *preprocessing* dapat meningkatkan performa pengolahan teks Bahasa Indonesia. Penelitian (Lubis et al., 2023) membuktikan bahwa penerapan teknik *text preprocessing* dan model CNN mampu meningkatkan performa *spelling checker* Bahasa Indonesia, tetapi data yang digunakan masih berasal dari media sosial. Penelitian (Yanfi et al., 2023) dan (Yanfi et al., 2024) menunjukkan pentingnya pengembangan korpus serta penerapan model *deep learning* untuk mendeteksi kesalahan ejaan Bahasa Indonesia. Selain itu, penelitian (Mansur et al., 2024), (Dashti et al., 2024), dan (Toleu et al., 2022) memperlihatkan bahwa pendekatan normalisasi, *deep learning*, dan pendekatan berbasis data mampu meningkatkan kemampuan sistem dalam menangani variasi kesalahan penulisan. Di sisi lain, penelitian (Asri et al., 2022), (Asri, Kuswardani, Suliyanti, et al., 2025), dan (Asri,

Kuswardani, Ferdinanda, et al., 2025) telah menggunakan data ulasan aplikasi PLN *Mobile*, tetapi penelitian tersebut masih berfokus pada analisis sentimen dan pemanfaatan *spelling corrector* sebagai tahap pendukung *preprocessing*. Belum terdapat penelitian yang secara khusus mengimplementasi *spelling checker* Bahasa Indonesia pada data ulasan PLN *Mobile* dengan menerapkan teknik *text preprocessing* dalam proses implementasi model CNN. Oleh karena itu, penelitian ini mengisi celah penelitian dengan menerapkan teknik *text preprocessing* dan model *Convolutional Neural Network* untuk membangun sistem *spelling checker* Bahasa Indonesia pada data ulasan pengguna aplikasi PLN *Mobile*.

2.1.1 State Of The Art

Untuk mengetahui posisi penelitian ini dibandingkan penelitian terdahulu, dilakukan analisis *state of the art* yang dirangkum pada Tabel 2.1.

Tabel 2. 1 State of the Art

No	Judul Penelitian	Metode / Pendekatan	Data / Objek	Hasil Utama	Keterkaitan dan Gap Penelitian
1	<i>Improving Spell Checker Performance for Bahasa Indonesia Using Text preprocessing Techniques with Deep Learning Models</i>	<i>Text preprocessing</i> dan model <i>deep learning</i> , khususnya CNN	Data komunitas media sosial Bahasa Indonesia	Akurasi terbaik sebesar 0,89 ketika seluruh tahapan <i>preprocessing</i> diterapkan	Belum menggunakan data ulasan PLN <i>Mobile</i>
2	<i>SPECIL: Spell Error Corpus for the Indonesian Language</i>	Pembentukan korpus kesalahan ejaan Bahasa Indonesia	SPECIL / korpus kesalahan ejaan Bahasa Indonesia	Membentuk korpus lebih dari 180.000 token dalam 21.500 kalimat	Belum menerapkan CNN pada data PLN <i>Mobile</i>
3	<i>Enhancing Text</i>	Peter Norvig	Teks Bahasa	Perbaikan kata salah	Menunjukkan bahwa

	Classification Performance by <i>Preprocessing</i> Misspelled Words in Indonesian Language	dan N-Gram	Indonesia	tulis meningkatkan akurasi sebesar 8,2%	penanganan <i>Typo</i> penting dalam <i>preprocessing</i> . Gap: belum menggunakan CNN dan belum memakai data PLN <i>Mobile</i>
4	<i>Multi-head Attention Based Bidirectional LSTM for Spelling Error Detection in the Indonesian Language</i>	<i>Bi-LSTM dan Multi-Head Attention</i>	Teks Bahasa Indonesia	Model mencapai akurasi tertinggi sebesar 92,26%	<i>elevan</i> karena membahas deteksi kesalahan ejaan Bahasa Indonesia dengan <i>deep learning</i> . Gap: metode bukan CNN dan tidak menggunakan data PLN <i>Mobile</i>
5	Meningkatkan Performa Ulasan Berbahasa Indonesia dengan <i>Spelling corrector</i> Peter Norvig dan Pelabelan SentiStrength_id	<i>Peter Norvig, preprocessing , SentiStrength_id, dan SVM</i>	11.004 ulasan aplikasi PLN <i>Mobile</i>	Model SVM memperoleh akurasi tertinggi sebesar 82%	Sangat relevan karena objeknya sama, yaitu PLN <i>Mobile</i> . Gap: fokus penelitian masih analisis sentimen, belum

					membangun model <i>spelling checker</i> CNN.
6	Pelabelan Otomatis <i>Lexicon Vader</i> dan <i>Klasifikasi Naive Bayes</i> dalam Menganalisis Sentimen Data Ulasan PLN <i>Mobile</i>	<i>Lexicon Vader</i> dan <i>Naive Bayes</i>	1.000 ulasan PLN <i>Mobile</i>	Model memperoleh akurasi sebesar 70%	Relevan karena memakai data PLN <i>Mobile</i> . Gap: fokus penelitian pada analisis sentimen, bukan <i>spelling checker</i>
7	<i>A Normalization Model for Repeated Letters in Social Media Hate Speech Text Based on Rules and Spelling correction</i>	<i>Rule-based pattern</i> dan <i>SymSpell</i>	Teks media sosial	Model mampu menurunkan OOV menjadi 8%	Relevan untuk menangani huruf berulang seperti “lemottt” atau “erorrr”. Gap: bukan data PLN <i>Mobile</i> dan bukan fokus <i>spelling checker</i> Bahasa Indonesia
8	<i>Automatic Correction of Indonesian Grammatical Errors Based on Transformer</i>	<i>Transformer</i>	Korpus Bahasa Indonesia	Mengembangkan model koreksi kesalahan tata bahasa Indonesia	Mendukung pengembangan sistem koreksi kesalahan Bahasa Indonesia. Gap: fokus pada

					<i>grammatical error correction</i> , bukan <i>spelling checker</i> tingkat kata menggunakan an CNN
--	--	--	--	--	--

Berdasarkan Tabel 2.1, dapat disimpulkan bahwa penelitian sebelumnya telah membahas berbagai pendekatan terkait *spelling checker*, *spelling correction*, normalisasi teks, serta pengolahan ulasan PLN Mobile. Namun, penelitian yang secara khusus membangun sistem *spelling checker* Bahasa Indonesia menggunakan model *Convolutional Neural Network* (CNN) pada data ulasan PLN Mobile masih terbatas. Penelitian berbasis CNN sebelumnya masih menggunakan data media sosial secara umum, sedangkan penelitian dengan objek PLN Mobile lebih banyak berfokus pada analisis sentimen sehingga belum membahas deteksi kesalahan ejaan pada tingkat kata.

Penelitian ini memilih CNN karena model tersebut mampu mengekstraksi pola lokal pada data berbasis karakter sehingga sesuai untuk mengenali perubahan kecil pada struktur kata akibat kesalahan ejaan, seperti penghapusan, penambahan, penggantian, maupun perpindahan karakter. Pendekatan *Character-Level* dipilih karena kesalahan ejaan pada pengguna umumnya terjadi pada tingkat karakter, sehingga informasi detail mengenai susunan karakter tetap dapat dipertahankan dibandingkan pendekatan berbasis kata. *Dataset* PLN Mobile dipilih karena ulasan pengguna aplikasi tersebut mengandung variasi penulisan nyata seperti *Typo*, slang, singkatan, dan kata tidak baku yang sesuai dengan permasalahan *spelling checker*. Oleh karena itu, penelitian ini menerapkan *text preprocessing* dan *Character-Level* CNN untuk melakukan klasifikasi kata benar dan kata salah pada ulasan pengguna aplikasi PLN Mobile.

2.1.2 Research GAP

Berdasarkan Tabel 2.1, penelitian terdahulu menunjukkan bahwa berbagai metode telah diterapkan dalam pengembangan *spelling checker* Bahasa Indonesia, mulai dari pendekatan berbasis aturan hingga *deep learning*. Namun, masih terdapat keterbatasan terkait penggunaan *dataset*, karakteristik bahasa pengguna, dan penerapan model pada domain ulasan aplikasi. Oleh karena itu, analisis *research gap* dilakukan untuk mengetahui posisi penelitian ini dibandingkan penelitian sebelumnya.

Research GAP penelitian ini disajikan pada Tabel 2.2 berikut.

Tabel 2. 2 Research GAP

Penelitian	<i>Dataset</i>	Metode	Kekurangan
Lubis et al. (2023)	Data teks komunitas media sosial Bahasa Indonesia	<i>Text preprocessing</i> dan Convolutional Neural Network (CNN) untuk <i>spelling checker</i>	Penelitian telah membuktikan penggunaan CNN untuk deteksi kesalahan ejaan Bahasa Indonesia, tetapi belum menguji model pada data ulasan aplikasi dengan karakteristik bahasa pengguna seperti slang, singkatan, dan istilah domain tertentu.
Yanfi et al. (2024)	Korpus kesalahan ejaan Bahasa Indonesia	Bi-LSTM dengan <i>Multi-Head Attention</i> untuk <i>spelling error detection</i>	Model memiliki kemampuan menangkap hubungan <i>sekuensial</i> dan konteks yang lebih panjang, tetapi belum mengevaluasi pendekatan <i>Character-Level CNN</i> yang lebih sederhana pada domain ulasan aplikasi.
Penelitian ini	Ulasan pengguna aplikasi	<i>Text preprocessing</i> dan <i>Character-Level Convolutional Neural</i>	Penelitian ini masih memiliki keterbatasan pada

	PLN Mobile	Network (CNN) untuk <i>spelling checker</i> Bahasa Indonesia serta <i>RapidFuzz</i> sebagai <i>spelling corrector</i>	jumlah data berlabel yang relatif terbatas serta penggunaan pelabelan semi- otomatis yang masih berpotensi menghasilkan <i>label noise</i> . Selain itu, model CNN belum dibandingkan dengan arsitektur deep learning lain seperti Bi-LSTM atau Transformer.
--	---------------	---	--

Berdasarkan Tabel 2.2, dapat diketahui bahwa masih terdapat celah penelitian dalam pengembangan sistem *spelling checker* Bahasa Indonesia, khususnya pada penerapan model *deep learning* terhadap data ulasan aplikasi yang memiliki karakteristik bahasa pengguna secara nyata. Penelitian (Lubis et al., 2023) telah menunjukkan bahwa kombinasi *text preprocessing* dan CNN mampu digunakan untuk mendeteksi kesalahan ejaan, tetapi *dataset* yang digunakan masih berasal dari teks komunitas media sosial. Di sisi lain, (Yanfi et al., 2024) menggunakan pendekatan Bi-LSTM dengan *Multi-Head Attention* yang memiliki kemampuan memahami hubungan konteks lebih panjang, namun belum mengevaluasi pendekatan CNN berbasis karakter pada domain ulasan aplikasi.

Berdasarkan celah tersebut, penelitian ini berfokus pada implementasi *spelling checker* Bahasa Indonesia berbasis *Character-Level CNN* pada data ulasan pengguna aplikasi PLN Mobile. Pendekatan berbasis karakter dipilih karena kesalahan ejaan pada tingkat kata umumnya terjadi akibat perubahan kecil pada susunan karakter, seperti penambahan, penghapusan, atau penggantian huruf. Selain itu, penelitian ini mengembangkan pipeline yang memisahkan fungsi CNN sebagai *spelling checker* dan *RapidFuzz* sebagai *spelling corrector* sehingga sistem tidak hanya mampu mendeteksi kata salah, tetapi juga memberikan rekomendasi koreksi kata.

Dengan demikian, kontribusi utama penelitian ini terletak pada penerapan *Character-Level CNN* untuk *spelling error detection* berbasis ulasan aplikasi PLN

Mobile yang memiliki karakteristik kesalahan penulisan nyata, serta integrasi *RapidFuzz* untuk memberikan rekomendasi koreksi kata setelah proses deteksi dilakukan.

2.2 Landasan Teori

2.2.1 Natural Language Processing

Natural *Language Processing* (NLP) merupakan cabang kecerdasan buatan yang berfokus pada kemampuan komputer dalam memahami, mengolah, dan menganalisis bahasa manusia dalam bentuk teks maupun suara. Dalam pengolahan teks, NLP digunakan untuk berbagai tugas seperti klasifikasi dokumen, analisis sentimen, ekstraksi informasi, normalisasi teks, hingga koreksi kesalahan bahasa. Data ulasan pengguna aplikasi digital, seperti PLN *Mobile*, memiliki karakteristik bahasa yang tidak formal dengan berbagai variasi penulisan, seperti *Typo*, singkatan, slang, dan kata tidak baku, sehingga diperlukan tahapan *text preprocessing* agar dapat diproses secara optimal oleh model komputasi (Asri et al., 2022). Pada penelitian ini, NLP diterapkan sebagai dasar pengembangan sistem *spelling checker* pada ulasan pengguna PLN *Mobile* melalui tahapan *preprocessing*, pembentukan *dataset* kesalahan ejaan, representasi teks berbasis karakter, serta pemodelan menggunakan *Convolutional Neural Network (CNN)*. Pendekatan kombinasi NLP dan *deep learning* tersebut terbukti mampu meningkatkan performa sistem koreksi ejaan Bahasa Indonesia (Lubis et al., 2023).

2.2.2 Spelling checker

Spelling checker merupakan sistem yang digunakan untuk mendeteksi kesalahan ejaan pada kata atau teks. Perkembangan metode berbasis *deep learning* memungkinkan sistem pemeriksa ejaan tidak hanya mendeteksi kesalahan berbasis kamus, tetapi juga mempelajari pola kesalahan berdasarkan karakter dan konteks bahasa. Pendekatan *neural spelling correction* mampu menangani variasi kesalahan ejaan yang lebih kompleks dibandingkan metode berbasis aturan tradisional (Dashti et al., 2024). Sistem ini bertujuan untuk menentukan apakah suatu kata termasuk kata benar atau kata salah. Dalam konteks Bahasa Indonesia, *spelling checker* penting karena teks digital yang ditulis pengguna sering mengandung kesalahan pengetikan, kata tidak baku, slang, singkatan, dan variasi penulisan informal.

Spelling checker berbeda dengan *spelling corrector*. *Spelling checker* berfokus pada proses pendeteksian kata salah, sedangkan *spelling corrector* berfokus pada pemberian saran kata benar atau perbaikan terhadap kata yang salah. (Toleu et al., 2022) menjelaskan bahwa *spellchecking* merupakan tugas menemukan kata yang salah dalam dokumen, sedangkan *autocorrection* berkaitan dengan pemberian atau penggantian kata salah menjadi bentuk yang benar. Dalam proses koreksi ejaan, kandidat kata dapat dibangkitkan setelah kesalahan teridentifikasi, kemudian kandidat tersebut dinilai untuk menentukan bentuk kata yang paling sesuai (Lee et al., 2022).

Kesalahan ejaan dapat terjadi dalam beberapa bentuk. (Yanfi et al., 2023) menjelaskan bahwa kesalahan ejaan dapat mencakup *non-word error*, *real-word error*, dan *punctuation error*. *Non-word error* merupakan kesalahan yang menghasilkan kata yang tidak terdapat dalam kamus, sedangkan *real-word error* merupakan kesalahan yang menghasilkan kata valid tetapi tidak sesuai konteks. Sementara itu, *punctuation error* berkaitan dengan kesalahan penggunaan tanda baca.

Pada tingkat kata, kesalahan ejaan dapat berupa *insertion*, *deletion*, *substitution*, dan *transposition* (Lubis et al., 2023). *Insertion* adalah penambahan huruf, *deletion* adalah penghilangan huruf, *substitution* adalah penggantian huruf, sedangkan *transposition* adalah pertukaran posisi huruf. Jenis kesalahan tersebut termasuk bentuk kesalahan ejaan yang dapat ditemukan dalam teks berbahasa Indonesia (Yanfi et al., 2023). Contoh bentuk kesalahan pada ulasan pengguna adalah “eror”, “gk”, “gabisa”, “lemottt”, atau “aplikasii”.

Dalam penelitian ini, *spelling checker* difokuskan pada klasifikasi kata benar dan kata salah. Kata benar diberi label 0, sedangkan kata salah diberi label 1. Proses identifikasi kata yang salah merupakan fungsi utama *spelling checker* sebelum sistem melakukan proses koreksi (Toleu et al., 2022). Kata salah dapat berupa *Typo*, slang, singkatan, kata tidak baku, atau variasi penulisan informal yang ditemukan pada data ulasan pengguna aplikasi PLN *Mobile*. Variasi kesalahan ejaan dalam bahasa Indonesia juga dapat mencakup *non-word error*, *real-word error*, dan kesalahan tanda baca (Yanfi et al., 2023).

Dalam sistem pemeriksa ejaan (*spelling checker*), proses deteksi kesalahan dan proses koreksi kata merupakan dua tahapan yang berbeda (Toleu et al., 2022). Model

Character-Level CNN pada penelitian ini berperan sebagai *spelling checker* yang bertugas melakukan identifikasi awal terhadap suatu kata apakah termasuk kata benar atau kata yang mengalami kesalahan ejaan berdasarkan pola karakter yang dipelajari dari data pelatihan. CNN mampu menangkap pola lokal pada susunan karakter yang relevan untuk proses klasifikasi teks (Soni et al., 2023). Model CNN tidak menghasilkan kata koreksi secara langsung, melainkan memberikan keputusan klasifikasi dan probabilitas kesalahan terhadap kata masukan.

Setelah suatu kata teridentifikasi sebagai kata salah, proses selanjutnya dilakukan oleh komponen *spelling corrector* menggunakan algoritma *RapidFuzz*. Dalam proses koreksi ejaan, kandidat kata dapat dibangkitkan setelah kesalahan teridentifikasi untuk menentukan bentuk kata yang paling sesuai (Lee et al., 2022). *RapidFuzz* berfungsi mencari kandidat kata yang paling mendekati kata masukan berdasarkan tingkat kemiripan karakter (*string similarity*). Dengan demikian, CNN berperan dalam tahap deteksi kesalahan ejaan, sedangkan *RapidFuzz* berperan dalam tahap pemberian rekomendasi koreksi. Pemisahan fungsi deteksi dan koreksi tersebut sesuai dengan konsep *spellchecking* dan *autocorrection* sebagai dua proses yang berbeda (Toleu et al., 2022). Kombinasi kedua pendekatan tersebut memungkinkan sistem tidak hanya menemukan kata yang salah, tetapi juga memberikan usulan perbaikan kata yang sesuai.

2.2.3 Text Preprocessing

Text preprocessing merupakan tahap awal dalam pengolahan teks yang bertujuan untuk membersihkan dan menyeragamkan data sebelum digunakan dalam proses pemodelan (Lubis et al., 2023). Tahapan ini diperlukan karena data teks pengguna umumnya mengandung *noise*, variasi bahasa informal, dan kesalahan penulisan yang dapat memengaruhi performa model (Palomino & Aider, 2022).

Lubis et al. (2023) menjelaskan bahwa data bahasa alami sering mengandung *noise* sehingga perlu dilakukan *preprocessing* sebelum digunakan dalam model *deep learning*. (Palomino & Aider, 2022) juga menunjukkan bahwa komponen *preprocessing* dan urutan penerapannya dapat memengaruhi performa model pada pengolahan teks.

2.2.3.1 Cleaning

Cleaning merupakan tahap *preprocessing* yang bertujuan mengurangi *noise* pada data teks dengan menghapus elemen yang tidak diperlukan, seperti URL, simbol, karakter khusus, tanda baca tidak relevan, dan spasi berlebih. Tahapan ini penting pada data ulasan pengguna karena teks yang dihasilkan cenderung tidak terstruktur dan memiliki banyak variasi penulisan (Lubis et al., 2023). Namun, pada sistem *spelling checker*, proses *cleaning* perlu dilakukan secara selektif karena kesalahan ejaan seperti *Typo*, penghilangan huruf, penambahan huruf, dan penggantian karakter merupakan informasi penting bagi model dalam mempelajari pola kesalahan bahasa (Yanfi et al., 2023). Oleh karena itu, penelitian ini hanya menghapus elemen yang tidak berkaitan dengan kesalahan ejaan, sementara variasi penulisan tetap dipertahankan sebagai bagian dari *dataset*.

2.2.3.2 Case Folding

Case folding merupakan tahap *preprocessing* yang bertujuan menyeragamkan bentuk teks dengan mengubah seluruh karakter menjadi huruf kecil (*lowercase*). Proses ini dilakukan untuk mengurangi variasi kapitalisasi pada kata yang sama, sehingga model tidak menganggapnya sebagai token berbeda, misalnya “PLN”, “Pln”, dan “pln” menjadi bentuk yang sama yaitu “pln”. Tahapan ini membantu menghasilkan representasi data yang lebih konsisten sebelum proses pemodelan (Lubis et al., 2023).

2.2.3.3 Tokenization

Tokenization merupakan proses memecah teks menjadi unit yang lebih kecil (*token*) berupa kata, subkata, atau karakter sesuai kebutuhan model. Pada penelitian ini, tokenisasi dilakukan pada tingkat kata untuk mengubah data ulasan menjadi kumpulan kata yang dapat diberi label sebagai kata benar atau salah dalam *dataset spelling checker*. Tahapan ini menjadi dasar sebelum proses representasi karakter dan pelatihan model CNN (Lubis et al., 2023).

2.2.4 Pembentukan Dataset Spelling checker

Dataset spelling checker merupakan *dataset* yang berisi kata-kata yang telah diberi label sebagai kata benar atau kata salah (Toleu et al., 2022). *Dataset* ini diperlukan agar model klasifikasi dapat mempelajari pola yang membedakan kedua kelas tersebut (Lubis et al., 2023).

Dalam penelitian ini, *dataset spelling checker* dibentuk dari data ulasan pengguna aplikasi PLN Mobile. Kata yang dianggap benar diberi label 0, sedangkan kata yang dianggap salah diberi label 1. Kata salah dapat berasal dari *Typo* alami dalam data, slang, singkatan, kata tidak baku, atau variasi penulisan informal (Yanfi et al., 2023). Selain itu, *Typo* sistematis dapat dibentuk dari kata benar untuk memperkaya variasi kesalahan ejaan pada data pelatihan (Liu et al., 2024).

Pembentukan *Typo* sintetis dilakukan untuk memperkaya variasi kesalahan ejaan. Penggunaan data sintetis berupa kesalahan ejaan buatan dapat membantu meningkatkan jumlah variasi kesalahan pada proses pelatihan model *spelling checker*, terutama ketika jumlah data kesalahan alami masih terbatas (Liu et al., 2024). Bentuk *Typo* sintetis dapat berupa penghapusan huruf, penambahan huruf, penggantian huruf, pertukaran posisi huruf, atau pengulangan huruf. Proses ini sejalan dengan jenis kesalahan ejaan yang dijelaskan dalam penelitian (Lubis et al., 2023), yaitu *deletion*, *insertion*, *substitution*, dan *transposition*.

Dataset dengan label kata benar dan kata salah menjadi dasar bagi model CNN untuk melakukan klasifikasi biner (Lubis et al., 2023). Dengan adanya *dataset* tersebut, model dapat mempelajari pola karakter yang menunjukkan apakah suatu kata termasuk kata benar atau kata salah (Soni et al., 2023).

2.2.5 Representasi Teks dan Word Embedding

Model CNN tidak dapat memproses teks mentah secara langsung. Oleh karena itu, data teks harus diubah terlebih dahulu ke dalam bentuk numerik. Representasi teks menjadi tahap penting dalam sistem NLP karena menentukan bagaimana informasi linguistik diterjemahkan menjadi fitur numerik yang dapat dipelajari oleh model (Minaee et al., 2021). Representasi berbasis karakter memiliki kemampuan dalam menangkap variasi morfologi dan kesalahan kecil pada susunan huruf yang sering muncul pada data teks informal (Soni et al., 2023). Proses ini disebut representasi teks. (Palanivinayagam et al., 2023) menjelaskan bahwa klasifikasi teks umumnya melalui tahapan *preprocessing*, representasi teks, pemilihan fitur, dan klasifikasi.

Pada penelitian ini, representasi teks dilakukan pada tingkat karakter. Pendekatan ini digunakan karena *spelling checker* bekerja pada tingkat kata, sedangkan kesalahan ejaan umumnya terjadi pada susunan huruf dalam kata (Yanfi et

al., 2023). Representasi berbasis karakter juga sesuai digunakan pada model CNN karena mampu menangkap pola lokal pada susunan karakter yang berdekatan (Soni et al., 2023).

Pendekatan *Character-Level* dipilih karena kesalahan ejaan umumnya terjadi akibat perubahan kecil pada susunan karakter, seperti penambahan, penghilangan, penggantian, atau pertukaran huruf (Lubis et al., 2023). Sebagai contoh, kata “aplikasi” dan “aplikasii” hanya memiliki perbedaan satu karakter tambahan. Representasi berbasis karakter memungkinkan model mempelajari pola perubahan tersebut secara lebih rinci dibandingkan representasi berbasis kata (Soni et al., 2023).

2.2.5.1 Token

Token merupakan unit teks yang digunakan dalam pemodelan dan dapat berupa kata, subkata, atau karakter (Palanivinayagam et al., 2023). Pada penelitian ini, token yang digunakan adalah karakter karena kesalahan ejaan biasanya muncul pada tingkat huruf, seperti huruf hilang, huruf berulang, huruf tertukar, atau huruf yang diganti (Yanfi et al., 2023).

2.2.5.2 Encoding

Integer encoding merupakan proses mengubah token menjadi angka agar dapat diproses oleh model komputasi (Palanivinayagam et al., 2023). Setiap karakter akan diberi indeks tertentu. Misalnya, karakter “a”, “b”, dan “c” akan direpresentasikan dengan angka yang berbeda. Proses ini diperlukan karena model CNN bekerja dengan *Input* numerik sebelum melakukan ekstraksi pola melalui lapisan jaringan saraf (Soni et al., 2023).

2.2.5.3 Sequence

Sequence merupakan urutan angka yang dihasilkan dari proses *integer encoding* dan merepresentasikan susunan token dalam bentuk numerik (Palanivinayagam et al., 2023). Setiap kata akan diubah menjadi deretan angka berdasarkan karakter penyusunnya. Misalnya, kata “token” akan diubah menjadi urutan indeks yang mewakili karakter “t”, “o”, “k”, “e”, dan “n”. Representasi berbasis urutan karakter memungkinkan model mempelajari pola lokal pada susunan karakter yang berdekatan (Soni et al., 2023).

2.2.5.4 Padding

Padding merupakan proses menyamakan panjang *sequence* agar data memiliki ukuran masukan yang seragam sebelum diproses oleh model (Palanivinayagam et al., 2023). Setiap kata memiliki jumlah karakter yang berbeda, sedangkan model CNN membutuhkan *Input* dengan ukuran yang konsisten. Oleh karena itu, kata yang lebih pendek dapat ditambahkan nilai tertentu, sedangkan kata yang terlalu panjang dapat dipotong sesuai panjang maksimum agar dapat diproses secara seragam oleh jaringan CNN (Soni et al., 2023).

2.2.5.5 Embedding Layer

Embedding layer merupakan lapisan yang mengubah indeks numerik menjadi vektor berdimensi tertentu sebagai representasi awal data teks (Palanivinayagam et al., 2023). Dengan *embedding*, model dapat mempelajari representasi karakter secara lebih informatif dibandingkan hanya menggunakan angka indeks secara langsung. Pada penelitian ini, *embedding* menjadi tahap awal sebelum data diproses oleh lapisan CNN untuk mengekstraksi pola lokal pada susunan karakter (Soni et al., 2023).

2.2.6 Deep Learning

Deep learning merupakan bagian dari *machine learning* yang menggunakan jaringan saraf tiruan dengan banyak lapisan untuk mempelajari pola dari data. Dalam bidang NLP, pendekatan deep learning banyak digunakan karena mampu melakukan ekstraksi fitur secara otomatis dari data teks tanpa membutuhkan rekayasa fitur secara manual yang kompleks (Minaee et al., 2021). Model berbasis neural network telah menunjukkan performa yang baik pada berbagai tugas klasifikasi teks (Prabhakar et al., 2022). Dalam pengolahan teks, *deep learning* dapat digunakan untuk berbagai tugas seperti klasifikasi teks, analisis sentimen, deteksi kesalahan ejaan, dan koreksi teks.

CNN telah digunakan sebagai salah satu model *deep learning* untuk meningkatkan performa *spelling checker* bahasa Indonesia (Lubis et al., 2023). Kombinasi *text preprocessing* dan model *deep learning* pada penelitian tersebut menunjukkan peningkatan performa dalam proses pemeriksaan ejaan bahasa Indonesia (Lubis et al., 2023).

Pada penelitian ini, model *deep learning* yang digunakan adalah CNN. Pemilihan CNN didasarkan pada kemampuannya dalam menangkap pola lokal pada

susunan karakter yang berdekatan (Soni et al., 2023). Kemampuan tersebut membuat CNN sesuai untuk membedakan kata benar dan kata salah melalui proses klasifikasi berbasis pola karakter (Emanuel et al., 2024).

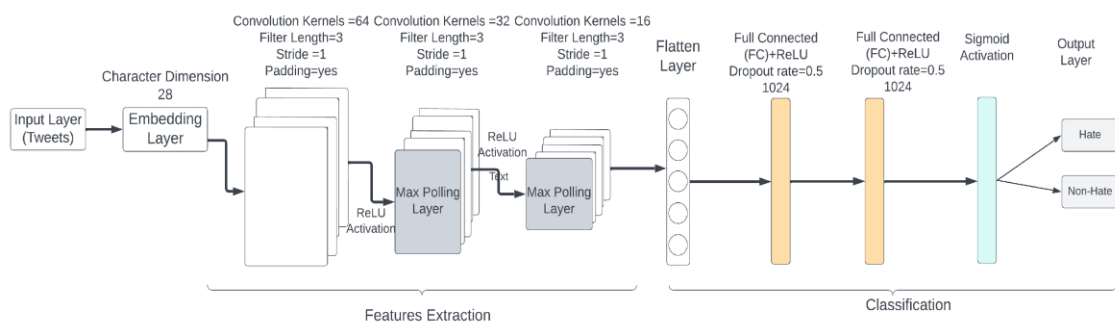
2.2.7 Convolutional Neural Network

2.2.7.1 Pengertian Convolutional Neural Network

Convolutional Neural Network atau CNN merupakan salah satu arsitektur *deep learning* yang digunakan untuk mengenali pola pada data (Soni et al., 2023). CNN banyak digunakan dalam pengolahan citra, tetapi juga dapat diterapkan pada teks. Dalam pengolahan teks, CNN bekerja dengan menangkap pola lokal dari urutan token atau karakter. Pada klasifikasi teks, CNN mampu mengekstraksi pola lokal (*local features*) secara efektif melalui operasi konvolusi sehingga dapat mengenali hubungan antar token maupun karakter yang berdekatan (Emanuel et al., 2024). Kemampuan tersebut menjadikan CNN sebagai salah satu arsitektur yang banyak digunakan dalam berbagai tugas klasifikasi teks berbasis representasi karakter maupun kata (Soni et al., 2023).

(Emanuel et al., 2024) menjelaskan bahwa CNN dapat digunakan untuk tugas klasifikasi teks karena mampu mendeteksi pola dan fitur penting. Dalam penelitian ini, CNN digunakan untuk mengklasifikasikan kata menjadi dua kelas, yaitu kata benar dan kata salah.

Dalam pengolahan teks, CNN juga dapat diterapkan menggunakan representasi berbasis karakter atau *Character-Level CNN*. (Al-Hussaeni et al., 2023) menjelaskan bahwa CNN satu dimensi dapat digunakan pada data tekstual dengan memperlakukan teks sebagai sinyal pada tingkat karakter. Pendekatan berbasis karakter memungkinkan model memproses teks berdasarkan susunan karakter serta membantu menangani variasi kesalahan ejaan dan *out-of-Vocabulary* yang dapat muncul pada representasi berbasis kata. Representasi pada tingkat karakter juga meningkatkan kemungkinan ditemukannya karakter yang sama antara data pelatihan dan data pengujian, sehingga model tidak sepenuhnya bergantung pada keberadaan kata yang identik dalam *dataset*. Selain itu, pendekatan ini memungkinkan CNN mempelajari pola lokal antar karakter yang dapat menjadi ciri penting dalam membedakan variasi bentuk kata pada data teks (Al-Hussaeni et al., 2023). Salah satu bentuk arsitektur *Character-Level CNN* ditunjukkan pada Gambar 2.1 berikut



Gambar 2. 1 Arsitektur CNN

Sumber: (Al-Hussaeni, 2023)

Berdasarkan Gambar 2.1 arsitektur *Character-Level CNN* terdiri atas *input layer*, *embedding layer*, beberapa *convolution layer*, *max-pooling layer*, *flatten layer*, *fully connected layer*, *dropout layer*, dan *output layer*. Pada arsitektur yang dikembangkan oleh (Al-Hussaeni et al., 2023), representasi karakter terlebih dahulu diproses melalui *embedding layer*, kemudian diteruskan ke tiga *convolution layer* untuk melakukan ekstraksi fitur. Dua lapisan konvolusi diikuti oleh *max-pooling layer*, sedangkan hasil ekstraksi fitur selanjutnya diteruskan ke *flatten layer* dan dua *fully connected layer*. Tahap akhir menggunakan *output layer* dengan fungsi aktivasi *sigmoid* untuk menghasilkan keluaran klasifikasi.

2.2.7.2 Embedding pada CNN

Embedding pada CNN berfungsi sebagai representasi awal dari *Input* teks yang telah dikonversi ke dalam bentuk numerik (Palanivinayagam et al., 2023). Kata yang telah diubah menjadi urutan angka melalui tokenisasi, *integer encoding*, dan *padding* akan diproses oleh *embedding layer*. Hasil *embedding* berupa vektor numerik yang kemudian menjadi *Input* bagi lapisan konvolusi untuk mengekstraksi pola lokal pada susunan karakter (Soni et al., 2023).

2.2.7.3 Convolution Layer

Convolution layer merupakan lapisan utama dalam CNN yang berfungsi untuk mengekstraksi fitur dari *Input*. Pada data teks berbasis karakter, lapisan ini mempelajari pola lokal dari susunan karakter (Soni et al., 2023). Misalnya, pola huruf berulang, huruf yang hilang, huruf tertukar, atau kombinasi karakter tertentu yang sering muncul pada kata salah (Emanuel et al., 2024).

Kernel merupakan filter yang bergerak pada *Input* untuk menangkap pola tertentu. Pada teks, kernel dapat menangkap pola beberapa karakter yang berdekatan (Soni et al., 2023). Dalam konteks *spelling checker*, kernel dapat membantu model mengenali bentuk kesalahan seperti pengulangan huruf, penggantian huruf, atau urutan karakter yang tidak umum. Pada klasifikasi teks berbasis karakter, operasi konvolusi memungkinkan model mendeteksi pola lokal yang relevan pada berbagai posisi dalam urutan karakter sehingga efektif digunakan untuk mengenali pola kesalahan ejaan dan variasi penulisan kata (Minaee et al., 2021).

2.2.7.4 Feature Map

Feature map merupakan hasil dari proses konvolusi. *Feature map* berisi informasi pola yang berhasil ditangkap oleh kernel (Soni et al., 2023). Pola-pola tersebut kemudian digunakan oleh model untuk membantu menentukan apakah suatu kata termasuk kata benar atau kata salah (Emanuel et al., 2024).

2.2.7.5 Max Pooling

Max pooling merupakan proses mengambil nilai fitur paling penting dari *Feature map* untuk merangkum informasi hasil konvolusi (Soni et al., 2023). Tahapan ini membantu mengurangi dimensi data dan mempertahankan fitur yang paling dominan (Prabhakar et al., 2022). Dalam klasifikasi kata, *max pooling* membantu model memilih pola karakter yang paling kuat untuk membedakan kata benar dan kata salah. Selain itu, proses pooling membantu mengurangi kompleksitas komputasi dan risiko overfitting dengan tetap mempertahankan informasi fitur yang paling relevan untuk proses klasifikasi (Minaee et al., 2021).

2.2.7.6 Flatten

Flatten merupakan proses mengubah hasil ekstraksi fitur menjadi bentuk satu dimensi agar dapat diproses oleh *dense layer* (Emanuel et al., 2024). Jika model menggunakan *GlobalMaxPooling1D*, proses *flatten* dapat digantikan karena fitur telah diringkas menjadi vektor satu dimensi yang siap digunakan pada tahap klasifikasi (Minaee et al., 2021).

2.2.7.7 Dense Layer

Dense layer merupakan lapisan *fully connected* yang digunakan untuk mengolah fitur hasil ekstraksi CNN (Emanuel et al., 2024). Lapisan ini

menggabungkan fitur-fitur penting yang diperoleh dari proses konvolusi dan *pooling* sebelum menghasilkan prediksi akhir (Soni et al., 2023).

2.2.7.8 Softmax dan Sigmoid

Softmax dan *sigmoid* merupakan fungsi aktivasi yang sering digunakan pada *output layer*. *Softmax* umumnya digunakan untuk klasifikasi multikelas, sedangkan *sigmoid* digunakan untuk klasifikasi biner (Prabhakar et al., 2022). Pemilihan fungsi aktivasi pada lapisan keluaran perlu disesuaikan dengan karakteristik tugas klasifikasi yang dilakukan (Emanuel et al., 2024).

Dalam penelitian ini, *output layer* menggunakan fungsi aktivasi *sigmoid* karena tugas klasifikasi hanya terdiri dari dua kelas, yaitu kata benar dan kata salah (Lubis et al., 2023). Nilai *sigmoid* berada pada rentang 0 sampai 1 dan dapat digunakan sebagai probabilitas hasil klasifikasi. Nilai yang mendekati 0 menunjukkan kecenderungan kata benar, sedangkan nilai yang mendekati 1 menunjukkan kecenderungan kata salah (Prabhakar et al., 2022).

2.2.7.9 Binary Classification

Binary classification merupakan proses klasifikasi yang membedakan data ke dalam dua kelas (Prabhakar et al., 2022). Pada penelitian ini, kelas yang digunakan adalah kata benar dan kata salah. Kata benar diberi label 0, sedangkan kata salah diberi label 1. Model CNN kemudian menghasilkan probabilitas yang menunjukkan kemungkinan suatu kata termasuk ke dalam kelas salah (Lubis et al., 2023).

2.2.7.10 Loss Function

Loss function digunakan untuk menghitung tingkat kesalahan model selama proses pelatihan dan menjadi dasar dalam proses pembaruan parameter model (Prabhakar et al., 2022). Pada klasifikasi biner, fungsi *loss* yang umum digunakan adalah *binary crossentropy*. Fungsi ini menghitung perbedaan antara label sebenarnya dan probabilitas prediksi yang dihasilkan model (Emanuel et al., 2024).

2.2.7.11 Optimizer

Optimizer digunakan untuk memperbarui bobot model agar nilai *loss* semakin kecil selama proses pelatihan (Prabhakar et al., 2022). Salah satu *optimizer* yang banyak digunakan adalah Adam karena mampu melakukan pembaruan bobot secara adaptif selama proses pembelajaran. Dalam penelitian ini, *optimizer* digunakan untuk

membantu model CNN mempelajari pola kata benar dan kata salah secara bertahap (Lubis et al., 2023).

2.2.7.12 Epoch

Epoch merupakan satu siklus penuh ketika seluruh data latih telah diproses oleh model selama proses pelatihan (Stankevičius et al., 2022). Jumlah *epoch* perlu ditentukan secara tepat karena jumlah yang terlalu sedikit dapat menyebabkan model belum belajar secara optimal, sedangkan jumlah yang terlalu banyak dapat meningkatkan risiko *overfitting* (Prabhakar et al., 2022).

2.2.7.13 Batch Size

Batch size merupakan jumlah data yang diproses dalam satu kali pembaruan bobot model (Stankevičius et al., 2022). Pemilihan *batch size* dapat memengaruhi waktu pelatihan, penggunaan sumber daya komputasi, serta kestabilan proses pembelajaran model (Prabhakar et al., 2022). Stankevičius et al. (2022) juga menjelaskan bahwa *batch size* merupakan salah satu hiperparameter penting dalam proses pelatihan karena berkaitan langsung dengan proses pembaruan bobot model.

2.2.8 Evaluasi Model

2.2.8.1 Confusion matrix

Confusion matrix merupakan metode evaluasi yang menggambarkan hubungan antara hasil prediksi model dan label sebenarnya. Pada klasifikasi biner, *confusion matrix* terdiri dari empat komponen, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN) (Prabhakar et al., 2022).

Dalam konteks *spelling checker*, TP menunjukkan kata salah yang berhasil diprediksi sebagai kata salah. TN menunjukkan kata benar yang berhasil diprediksi sebagai kata benar. FP menunjukkan kata benar yang salah diprediksi sebagai kata salah, sedangkan FN menunjukkan kata salah yang diprediksi sebagai kata benar (Lubis et al., 2023). Keempat komponen tersebut menjadi dasar dalam perhitungan berbagai metrik evaluasi untuk memperoleh gambaran performa model secara lebih menyeluruh (Minaee et al., 2021).

Komponen *confusion matrix* yang digunakan disajikan pada Tabel 3.19.

Tabel 3. 1 *Confusion matrix*

Komponen	Keterangan
True Positive (TP)	Kata salah yang diprediksi sebagai kata salah
True Negative (TN)	Kata benar yang diprediksi sebagai kata benar
False Positive (FP)	Kata benar yang diprediksi sebagai kata salah
False Negative (FN)	Kata salah yang diprediksi sebagai kata benar

Berdasarkan Tabel 3.19, *confusion matrix* digunakan untuk menggambarkan hubungan antara hasil prediksi model dengan label sebenarnya. Empat komponen utama dalam *confusion matrix*, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN), memberikan informasi detail mengenai jenis kesalahan yang dilakukan oleh model.

Dalam konteks *spelling checker*, nilai TP menunjukkan kemampuan model dalam mendeteksi kata salah secara tepat, sedangkan TN menunjukkan kemampuan model mempertahankan kata benar. Sementara itu, FP dan FN menunjukkan kesalahan klasifikasi yang perlu diperhatikan karena dapat menyebabkan kata benar dianggap salah atau kata salah tidak terdeteksi.

Keempat komponen tersebut kemudian digunakan sebagai dasar perhitungan metrik evaluasi seperti *accuracy*, *precision*, *recall*, *F1-score*, *specificity*, dan ROC-AUC untuk memperoleh gambaran performa model secara lebih komprehensif.

2.2.8.2 *Accuracy*

Accuracy merupakan metrik yang digunakan untuk mengukur tingkat ketepatan model berdasarkan jumlah prediksi yang benar dibandingkan dengan seluruh data yang dievaluasi (Prabhakar et al., 2022).

Persamaan *accuracy*:

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

Nilai *accuracy* yang semakin tinggi menunjukkan bahwa semakin banyak data yang berhasil diklasifikasikan secara tepat oleh model (Minaee et al., 2021).

2.2.8.3 Precision

Precision digunakan untuk mengukur tingkat ketepatan model ketika memprediksi suatu data sebagai kelas positif (Prabhakar et al., 2022). Dalam sistem *spelling checker*, kelas positif merupakan kata yang mengalami kesalahan ejaan.

Persamaan *precision*:

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

Nilai *precision* menunjukkan proporsi kata yang diprediksi sebagai kata salah dan benar-benar termasuk kata salah berdasarkan label sebenarnya (Lubis et al., 2023).

2.2.8.4 Recall

Recall digunakan untuk mengukur kemampuan model dalam menemukan seluruh data yang sebenarnya termasuk ke dalam kelas positif. Nilai *recall* dihitung berdasarkan perbandingan antara jumlah *true positive* terhadap seluruh data positif aktual, yaitu *true positive* dan *false negative* (Palanivinayagam et al., 2023).

Persamaan *recall*:

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

Dalam konteks *spelling checker*, *recall* menunjukkan kemampuan model dalam mendeteksi kata-kata yang benar-benar mengalami kesalahan ejaan (Lubis et al., 2023).

2.2.8.5 F1-Score

F1-score merupakan metrik yang menggabungkan *precision* dan *recall* melalui perhitungan *harmonic mean* sehingga dapat memberikan ukuran keseimbangan antara kedua metrik tersebut (Prabhakar et al., 2022).

Persamaan F1-score:

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

Nilai *F1-score* dapat digunakan ketika penelitian perlu mempertimbangkan keseimbangan antara kemampuan model menemukan kata salah dan ketepatan model ketika memberikan prediksi kata salah (Minaee et al., 2021).

2.2.8.6 Specificity

Specificity merupakan metrik yang digunakan untuk mengukur kemampuan model dalam mengenali kelas negatif secara tepat (Minaee et al., 2021). Dalam penelitian *spelling checker*, kelas negatif merupakan kata yang benar atau valid.

Persamaan *specificity*:

$$Specificity = \frac{TN}{TN + FP}$$

Nilai *specificity* menunjukkan kemampuan model dalam mempertahankan kata yang benar agar tidak salah diklasifikasikan sebagai kata yang mengalami kesalahan ejaan (Lubis et al., 2023).