BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian mengenai deteksi kekuatan sentimen (sentiment strength) berkembang pesat seiring kebutuhan untuk memahami opini pengguna secara lebih mendalam. Berbeda dengan analisis sentimen konvensional yang hanya mengklasifikasikan polaritas (positif, negatif, netral), deteksi kekuatan sentimen bertujuan mengukur tingkat intensitas emosi dalam teks—memberikan informasi yang lebih granular dan kaya untuk berbagai bidang seperti pemasaran, sistem rekomendasi, dan analisis perilaku konsumen. Pendekatan ini memanfaatkan fitur tambahan dan karakteristik linguistik untuk meningkatkan akurasi prediksi, sehingga menghasilkan representasi opini yang lebih baik dibandingkan klasifikasi polaritas biasa [16].

Pendekatan berbasis leksikon (lexicon-based approach) juga banyak digunakan dalam analisis sentimen dengan memanfaatkan kumpulan kata yang telah diberi bobot sentimen untuk menentukan polaritas maupun intensitas opini. Kualitas dan cakupan leksikon—seperti Opinion Lexicon, SSS-Lex, VADER, dan SentiWordNet—menjadi faktor penting yang memengaruhi hasil analisis, karena leksikon berfungsi sebagai sumber pengetahuan untuk menentukan orientasi dan intensitas sentimen suatu kata atau frasa secara lebih akurat [17]. Semakin baik kualitas leksikon yang digunakan, semakin tinggi kemampuan sistem dalam memahami makna emosional suatu teks.

Perkembangan metode deep learning mendorong penggunaan arsitektur Convolutional Neural Network (CNN) dalam analisis sentimen, yang mampu mengekstraksi pola lokal dari teks melalui proses konvolusi untuk menangkap informasi penting terkait sentimen. Performa CNN dapat ditingkatkan dengan mengintegrasikan informasi sentimen dari leksikon, sehingga model mampu memahami konteks sentimen secara lebih baik

dibandingkan hanya mengandalkan representasi kata. Model CNN yang terintegrasi dengan leksikon sentimen dan mekanisme attention telah dikembangkan dan terbukti meningkatkan kemampuan model dalam memahami konteks sentimen dibandingkan CNN standar [18].

Pengembangan *CNN* untuk tugas deteksi kekuatan sentimen kemudian dilakukan dengan mempertimbangkan aspek konteks dalam penggunaan kata. Pendekatan ini memungkinkan model memahami perubahan makna sentimen yang dipengaruhi oleh susunan kalimat. Integrasi informasi intensitas sentimen dari leksikon terbukti dapat membantu model menghasilkan prediksi yang lebih akurat. Hasil penelitian menunjukkan bahwa pendekatan tersebut mampu mengungguli beberapa metode pembanding yang digunakan sebagai *baseline*. Model *Context-Dependent Lexicon-Based CNN* dikembangkan serta menunjukkan bahwa kombinasi CNN dengan leksikon berbasis konteks menghasilkan performa yang lebih baik dalam tugas deteksi kekuatan sentimen.

Selain *CNN*, arsitektur *Long Short-Term Memory* (*LSTM*) juga banyak digunakan dalam analisis sentimen karena kemampuannya mempertahankan informasi jangka panjang pada data sekuensial. Kemampuan tersebut memungkinkan model memahami hubungan antar kata yang saling memengaruhi makna suatu kalimat. Hal ini menjadi penting karena sentimen dalam teks sering kali dipengaruhi oleh konteks yang muncul pada bagian kalimat yang berjauhan. Oleh sebab itu, *LSTM* banyak diterapkan dalam tugas prediksi sentimen dan intensitas sentimen. Model *Stacked Residual LSTM* dikembangkan untuk prediksi intensitas sentimen dan menunjukkan bahwa arsitektur tersebut mampu meningkatkan performa model dalam memahami hubungan antarkata pada teks.

Perkembangan lebih lanjut pada arsitektur *LSTM* dilakukan melalui integrasi dengan model representasi bahasa berbasis *transformer*. Pendekatan ini bertujuan memanfaatkan kemampuan representasi kontekstual yang lebih baik sebelum data diproses oleh *LSTM*. Integrasi beberapa metode terbukti mampu meningkatkan kualitas representasi fitur yang digunakan dalam proses

klasifikasi maupun prediksi sentimen. Kombinasi tersebut juga memungkinkan model memahami konteks kalimat secara lebih mendalam. Model *RoBERTa-LSTM* dikembangkan dan menunjukkan bahwa kombinasi representasi kontekstual dengan *LSTM* mampu meningkatkan akurasi analisis sentimen dibandingkan penggunaan *LSTM* secara tunggal.

Penelitian terkait *Bidirectional Long Short-Term Memory* (*Bi-LSTM*) menunjukkan bahwa pemrosesan informasi dua arah mampu memberikan pemahaman konteks yang lebih baik dibandingkan model satu arah. Arsitektur ini memanfaatkan informasi dari arah maju dan arah mundur secara bersamaan sehingga dapat menangkap hubungan kata secara lebih komprehensif. Kemampuan tersebut sangat bermanfaat pada teks yang mengandung makna implisit atau struktur kalimat yang kompleks. Oleh karena itu, *Bi-LSTM* banyak digunakan dalam berbagai tugas analisis sentimen. Kombinasi Bi-LSTM dan *CNN* dikembangkan untuk mendeteksi sarkasme pada teks serta memperoleh hasil yang lebih baik dibandingkan penggunaan model tunggal.

Keunggulan *Bi-LSTM* dalam memahami konteks kalimat juga didukung oleh berbagai penelitian lainnya. Pemrosesan informasi dari dua arah memungkinkan model memperoleh representasi teks yang lebih lengkap dibandingkan pendekatan sekuensial satu arah. Dengan demikian, model dapat mengenali hubungan antar kata yang berkontribusi terhadap pembentukan sentimen secara lebih akurat. Kemampuan tersebut menjadikan *Bi-LSTM* sebagai salah satu arsitektur yang banyak digunakan dalam analisis sentimen modern. Model Bi-LSTM menunjukkan efektivitas dalam menangkap konteks kalimat sehingga mampu meningkatkan performa analisis sentimen secara signifikan [19].

Pengembangan model berbasis *Bi-LSTM* juga dilakukan melalui integrasi dengan mekanisme *attention* dan *CNN*. Pendekatan ini bertujuan menggabungkan kemampuan ekstraksi fitur lokal dari *CNN* dengan kemampuan pemahaman konteks dari *Bi-LSTM*. Hasilnya, model dapat mempelajari hubungan kontekstual dan pola lokal secara bersamaan.

Kombinasi tersebut terbukti mampu meningkatkan performa klasifikasi sentimen dibandingkan penggunaan satu arsitektur saja. Model *CNN-BiLSTM Multi-Attention Fusion Mechanism* dikembangkan dan menunjukkan bahwa model tersebut mampu meningkatkan performa pada tugas klasifikasi sentimen karena dapat mempelajari fitur lokal dan hubungan kontekstual secara simultan.

Penelitian lain juga menunjukkan bahwa kombinasi beberapa arsitektur *deep learning* dapat menghasilkan performa yang lebih baik dibandingkan penggunaan model tunggal. Integrasi berbagai model memungkinkan pemanfaatan keunggulan masing-masing arsitektur secara bersamaan. Pendekatan tersebut menjadi salah satu tren dalam pengembangan sistem analisis sentimen modern. Kombinasi model juga dinilai mampu meningkatkan kemampuan sistem dalam memahami kompleksitas bahasa alami. Model *Attention-Based Bidirectional CNN-RNN Deep Model (ABCDM)* dikembangkan dan menunjukkan bahwa kombinasi *CNN* dan jaringan rekuren mampu menghasilkan performa yang lebih baik dibandingkan penggunaan satu arsitektur secara terpisah [20].

Pada sisi leksikon, salah satu pendekatan yang paling banyak digunakan dalam analisis sentimen adalah *Valence Aware Dictionary and Sentiment Reasoner* (*VADER*). Leksikon ini dirancang khusus untuk menangani karakteristik bahasa informal yang umum ditemukan pada media sosial. Keunggulan utama *VADER* terletak pada kemampuannya mempertimbangkan negasi, *intensifier*, kapitalisasi huruf, dan tanda baca dalam menentukan skor sentimen. Kemampuan tersebut menjadikan *VADER* sebagai salah satu leksikon yang paling banyak digunakan dalam penelitian analisis sentimen berbasis teks. Metode *VADER d*ikembangkan dan menunjukkan bahwa metode tersebut mampu memberikan performa yang kompetitif serta efisien dalam mengidentifikasi sentimen pada teks media sosial [21].

Berdasarkan berbagai penelitian terdahulu, dapat disimpulkan bahwa model *deep learning* dan leksikon sentimen memiliki peran yang sama penting dalam keberhasilan deteksi kekuatan sentimen. Namun, sebagian besar

penelitian masih berfokus pada pengembangan satu model atau satu jenis leksikon secara terpisah. Penelitian yang secara langsung membandingkan performa *CNN*, *LSTM*, dan *Bi-LSTM* menggunakan lebih dari satu jenis leksikon masih relatif terbatas. Selain itu, penggunaan metrik regresi seperti *Mean Absolute Error* (*MAE*) dan *Predictive R-Square* (*pR²*) juga belum banyak diterapkan pada domain ulasan produk makanan cepat saji. Oleh karena itu, penelitian ini dilakukan untuk mengisi kesenjangan tersebut dengan membandingkan kinerja *CNN*, *LSTM*, dan *Bi-LSTM* yang dikombinasikan dengan *SSS-Lex* dan *VADER* dalam mendeteksi kekuatan sentimen pada ulasan produk McDonald's. Berikut ini tabel 2.1 mengenai penelitian terdahulu beserta kelebihan dan kelemahan masing-masing penelitian:

Tabel 2.1. Penelitian Terdahulu

No	Referensi	Metode & Dataset	Hasil Penelitian	Kelebihan & Kelemahan
1	Huang *et al.* (2018), *Information Sciences*	*Context-Dependent Lexicon-Based CNN; Dataset: BBC, Digg, MySpace, Runners World, Twitter,* dan *YouTube.*	Model mampu memprediksi kekuatan sentimen lebih baik dibandingkan baseline serta menghasilkan leksikon sentimen berkualitas tinggi.	Kelebihan: Menggabungkan leksikon berbasis konteks dengan *CNN* sehingga meningkatkan akurasi deteksi intensitas sentimen. Kelemahan: Hanya diuji pada dataset berbahasa Inggris sehingga generalisasi ke bahasa lain masih terbatas.
2	Yu *et al.* (2018), *IEEE/ACM Transactions on Audio, Speech, and Language Processing*	*Sentiment Intensity Lexicon* dengan *CNN, DAN, Bi-LSTM,* dan *Tree-LSTM; Dataset: SemEval* dan *SST.*	Meningkatkan kualitas *word embedding* dan performa klasifikasi sentimen *biner*, *ternary*, maupun *fine-grained.*	Kelebihan: Dapat meningkatkan performa berbagai model *deep learning*. Kelemahan: Bergantung pada leksikon intensitas sentimen yang berkualitas tinggi.
3	Fang *et al.* (2019), *Artificial Intelligence Review*	*FBIOP Mining, Sentiment Intensity, SVM,* dan *CNN; Dataset: PCauto.com.cn, IMOPCs, IMOPRs.*	Model yang diusulkan mengungguli *CNN* dan *SVM* dalam mendeteksi opini implisit.	Kelebihan: Efektif mendeteksi opini implisit yang sulit dikenali metode konvensional. Kelemahan: Proses ekstraksi pola cukup kompleks dan bergantung pada domain tertentu.
4	Yang *et al.*	Pembangunan	*SentiRuc*	Kelebihan: Menghasilkan

No	Referensi	Metode & Dataset	Hasil Penelitian	Kelebihan & Kelemahan
	(2019), *Computational Intelligence and Neuroscience*	leksikon *SentiRuc; Dataset: HIT Tongyicicilin, NLPCC* 2013 dan 2014.	memberikan representasi semantik sentimen yang lebih baik dibandingkan beberapa leksikon sebelumnya.	leksikon sentimen yang komprehensif. Kelemahan: Berfokus pada pembangunan leksikon, bukan model klasifikasi.
5	Meisheri *et al.* (2017), *WASSA 2017 (EMNLP Workshop)*	*CNN, LSTM, Hybrid CNN-LSTM; Dataset: Amazon Reviews.*	Model hybrid memberikan hasil lebih baik pada data pengembangan, sedangkan *LSTM* lebih baik pada data uji.	Kelebihan: Mengombinasikan keunggulan *CNN* dan *LSTM*. Kelemahan: Performa model *hybrid* belum konsisten pada seluruh dataset.
6	Ertugrul *et al.* (2018), *Lecture Notes in Computer Science*	*SVM* dan *SVR; Dataset: Turkish Tweets.*	Pendekatan regresi sedikit meningkatkan akurasi dibandingkan klasifikasi berbasis label diskrit.	Kelebihan: Memprediksi intensitas sentimen secara kontinu menggunakan regresi. Kelemahan: Peningkatan performa relatif kecil dan hanya diuji pada bahasa Turki.
7	Syamala & Nalini (2022), *International Journal of Engineering Trends and Technology*	*CNN + Bi-RNN + Language Model; Dataset: YouTube Review* dan *LibriSpeech.*	Meningkatkan *WER* dan *CER* serta menghasilkan akurasi analisis sentimen sekitar 90%.	Kelebihan: Mengintegrasikan *ASR* dan analisis sentimen dalam satu sistem. Kelemahan: Membutuhkan proses transkripsi sehingga komputasi lebih tinggi.
8	Yang, Yang, & Zhou (2014), *Applied Mechanics and Materials*	*CSLS + SVM; Dataset: Chinese Microblog NLP&CC* 2012.	Leksikon yang dibangun terbukti efektif untuk analisis sentimen *microblog*.	Kelebihan: Efektif untuk analisis sentimen bahasa Mandarin. Kelemahan: Terbatas pada bahasa Mandarin.
9	Hajek *et al.* (2019), *IFIP Advances in Information and Communication Technology*	*DNN-WSA; Dataset: Amazon Reviews (Kaggle).*	Asosiasi kata-sentimen meningkatkan performa dibandingkan embedding kata biasa.	Kelebihan: Memperkaya representasi kata dengan asosiasi sentimen. Kelemahan: Membutuhkan sumber asosiasi kata-sentimen yang lengkap.
10	Yue *et al.* (2022), *Buildings*	*LDA + CNN-BiLSTM; Dataset: Weibo Smart City.*	Berhasil mengidentifikasi perubahan persepsi masyarakat terhadap *smart city* berdasarkan topik dan sentimen.	Kelebihan: Menggabungkan topic modeling dan analisis sentimen secara komprehensif. Kelemahan: Lebih berfokus pada analisis topik daripada peningkatan performa model sentimen.

Berdasarkan Tabel 2.1 dapat diketahui bahwa sebagian besar penelitian terdahulu berfokus pada peningkatan performa model *deep learning* atau pengembangan metode representasi teks dalam analisis sentimen. Beberapa penelitian memanfaatkan pendekatan lexicon untuk memperkaya representasi fitur, sementara penelitian lainnya lebih menitikberatkan pada pengembangan arsitektur model seperti *CNN, LSTM, BiGRU,* maupun *Transformer*. Namun demikian, masih terdapat keterbatasan karena sebagian besar penelitian hanya menggunakan satu metode pelabelan sentimen atau hanya mengevaluasi satu jenis model sehingga belum memberikan gambaran mengenai pengaruh metode pelabelan terhadap performa berbagai arsitektur *deep learning*.

Penelitian ini berupaya mengisi kesenjangan tersebut dengan membandingkan dua pendekatan pelabelan berbasis *lexicon*, yaitu *SSS-Lex* dan *VADER*, yang dikombinasikan dengan tiga model deep learning, yaitu *CNN*, *LSTM*, dan *Bi-LSTM*. Selain melakukan komparasi antar model, penelitian ini juga mengevaluasi pengaruh metode pelabelan terhadap kemampuan model dalam memprediksi kekuatan sentimen menggunakan metrik *Mean Absolute Error (MAE)* dan *predictive-R^2 (pR^2).* Dengan demikian, penelitian ini tidak hanya membandingkan performa algoritma, tetapi juga memberikan kontribusi dalam mengevaluasi efektivitas pendekatan *lexicon* sebagai dasar pembentukan label sentimen pada tugas prediksi kekuatan sentimen.

2.2 Teori yang berkaitan

2.2.1 Analisis Sentimen

Analisis sentimen merupakan teknik yang penting dalam bidang *Natural Language Processing* (*NLP*) yang digunakan untuk mengidentifikasi, mengekstraksi, dan menganalisis opini, emosi, serta sikap yang terkandung dalam suatu teks. Perkembangan teknologi informasi dan media digital telah menyebabkan meningkatnya jumlah data teks yang dihasilkan pengguna, sehingga diperlukan metode yang mampu memahami makna dan kecenderungan opini secara otomatis.

Analisis sentimen dijelaskan tidak hanya berfokus pada klasifikasi polaritas sentimen menjadi positif, negatif, dan netral, tetapi juga dapat dikembangkan untuk mengukur tingkat intensitas sentimen yang terkandung dalam teks [22]. Analisis sentimen dapat dilakukan menggunakan berbagai pendekatan, mulai dari pendekatan berbasis leksikon, *machine learning*, hingga *deep learning* [23]. Analisis sentimen memiliki peran yang sangat penting dalam memahami opini publik yang berasal dari media sosial, forum diskusi, maupun ulasan produk sehingga dapat mendukung proses pengambilan keputusan secara lebih objektif [24]. Selain itu, analisis sentimen juga memungkinkan identifikasi tingkat kekuatan sentimen sehingga mampu memberikan gambaran yang lebih rinci mengenai persepsi pengguna terhadap suatu produk atau layanan.

2.2.2 Deteksi Kekuatan Sentimen

Deteksi kekuatan sentimen merupakan pengembangan dari analisis sentimen yang tidak hanya menentukan polaritas sentimen, tetapi juga mengukur tingkat intensitas emosi yang terkandung dalam suatu teks. Pendekatan ini menjadi penting karena setiap opini memiliki tingkat ekspresi yang berbeda-beda sehingga tidak cukup hanya dikategorikan sebagai sentimen positif atau negatif. Kekuatan sentimen dijelaskan sebagai aspek yang penting dalam memahami persepsi pengguna secara lebih mendalam karena mampu menunjukkan tingkat emosi yang sebenarnya terkandung dalam suatu teks [25]. Intensitas sentimen juga dinyatakan mampu memberikan informasi yang lebih rinci dibandingkan klasifikasi sentimen biasa karena dapat membedakan opini yang lemah, sedang, maupun sangat kuat. Selain itu, Pendekatan berbasis leksikon intensitas dijelaskan dapat digunakan untuk mengukur kekuatan emosi secara lebih akurat sehingga menghasilkan interpretasi sentimen yang lebih komprehensif [26]. Oleh karena itu, deteksi kekuatan sentimen

menjadi pendekatan yang relevan dalam berbagai penelitian yang berfokus pada pemahaman opini pengguna secara mendalam.

2.2.3 Ulasan Produk McDonald's

Ulasan produk merupakan salah satu sumber informasi yang penting untuk memahami persepsi dan pengalaman konsumen terhadap suatu produk atau layanan. Informasi yang terkandung dalam ulasan dapat digunakan untuk mengevaluasi kualitas produk, mengidentifikasi kekurangan layanan, serta mengetahui tingkat kepuasan pelanggan. Ulasan pelanggan dijelaskan dapat dimanfaatkan sebagai sumber data yang bernilai untuk mendukung pengambilan keputusan bisnis dan pengembangan strategi pemasaran [27].

McDonald's merupakan salah satu jaringan restoran cepat saji terbesar di dunia yang melayani jutaan pelanggan setiap harinya. Banyaknya interaksi pelanggan menghasilkan ulasan (*review*) dalam jumlah besar pada berbagai *platform digital*, khususnya *Google Reviews*. Ulasan tersebut berisi opini pelanggan mengenai kualitas produk, pelayanan, kebersihan, maupun pengalaman berkunjung, sehingga menjadi sumber informasi yang bernilai untuk analisis sentimen. Dalam penelitian ini, data yang digunakan berasal dari dataset *McDonald's Store Reviews* yang tersedia di *platform Kaggle* dan dikumpulkan dari *Google Reviews*. Data ulasan tersebut mengandung berbagai opini yang mencerminkan pengalaman konsumen terhadap produk maupun layanan yang diberikan. Oleh karena itu, ulasan produk McDonald's menjadi objek penelitian yang relevan dalam pengembangan sistem deteksi kekuatan sentimen karena mampu merepresentasikan persepsi konsumen secara langsung.

2.2.4 *Text Mining*

Text mining merupakan salah satu cabang ilmu yang berfokus pada proses ekstraksi informasi dan pengetahuan dari data berbentuk teks yang tidak terstruktur (*unstructured data*). Berbeda dengan data numerik yang

telah tersusun dalam bentuk tabel, data teks umumnya memiliki struktur yang kompleks sehingga memerlukan tahapan pengolahan sebelum dapat dianalisis. Melalui *text mining*, informasi yang tersembunyi di dalam dokumen teks dapat diidentifikasi dan diubah menjadi pengetahuan yang bermanfaat untuk mendukung proses pengambilan keputusan. *Text mining* merupakan proses penemuan pola dan informasi yang sebelumnya tidak diketahui dari sekumpulan dokumen teks dengan memanfaatkan kombinasi teknik *information retrieval*, *natural language processing* (NLP), *machine learning*, dan *data mining* [28]. Oleh karena itu, *text mining* menjadi salah satu pendekatan yang banyak diterapkan pada berbagai bidang, seperti analisis sentimen, klasifikasi dokumen, deteksi spam, sistem rekomendasi, hingga analisis opini pengguna di media sosial.

Pada perkembangannya, *text mining* telah menjadi salah satu metode yang penting dalam mengolah data digital karena jumlah data berbentuk teks terus meningkat setiap harinya. Data tersebut berasal dari berbagai sumber, seperti media sosial, forum diskusi, ulasan produk, berita daring, maupun dokumen elektronik lainnya yang mengandung informasi bernilai. Menurut Tujuan utama *text mining* bukan hanya untuk mencari kata yang sering muncul, tetapi juga untuk menemukan hubungan, pola, maupun informasi yang dapat digunakan dalam menghasilkan pengetahuan baru dari kumpulan dokumen. Informasi tersebut kemudian dapat dimanfaatkan untuk memahami perilaku pengguna, mengidentifikasi tren, maupun mendukung pengambilan keputusan berdasarkan data. Dengan demikian, *text mining* tidak hanya berfungsi sebagai proses pengolahan teks, tetapi juga sebagai teknik eksplorasi pengetahuan yang mampu memberikan nilai tambah terhadap data yang sebelumnya tidak terstruktur.

Secara umum, proses *text mining* terdiri atas beberapa tahapan yang saling berkaitan sehingga menghasilkan informasi yang siap digunakan dalam proses analisis. Tahapan pertama adalah pengumpulan data (*data collection*), yaitu memperoleh data teks dari berbagai sumber sesuai kebutuhan penelitian. Selanjutnya dilakukan *text preprocessing* yang

bertujuan membersihkan data dari karakter, simbol, maupun kata-kata yang tidak memiliki makna sehingga kualitas data menjadi lebih baik untuk dianalisis. Setelah proses pembersihan selesai, data akan dikonversi menjadi representasi numerik melalui proses ekstraksi fitur (*feature extraction*) atau *word embedding* agar dapat diproses oleh algoritma *machine learning* maupun *deep learning*. Tahapan terakhir adalah proses analisis menggunakan algoritma klasifikasi atau prediksi serta evaluasi hasil untuk mengukur performa model yang dikembangkan [29].

Dalam penelitian analisis sentimen, *text mining* memiliki peran yang sangat penting karena sebagian besar data yang digunakan berupa ulasan pelanggan dalam bentuk teks bebas. Data tersebut umumnya mengandung berbagai karakteristik, seperti penggunaan bahasa informal, singkatan, kesalahan penulisan, tanda baca, hingga simbol yang dapat memengaruhi hasil analisis apabila tidak diproses dengan baik. Oleh karena itu, penerapan text mining memungkinkan data teks diolah secara sistematis melalui serangkaian tahapan *preprocessing* sehingga informasi penting yang berkaitan dengan opini pelanggan dapat diekstraksi secara lebih efektif. Hasil dari proses tersebut kemudian digunakan sebagai masukan bagi model klasifikasi untuk menentukan polaritas maupun intensitas sentimen dari setiap ulasan. Dengan demikian, kualitas proses *text mining* akan sangat memengaruhi performa model analisis sentimen yang dibangun karena menentukan kualitas fitur yang digunakan selama proses pelatihan dan pengujian model.

2.2.5 *Data Preprocessing*

Data preprocessing merupakan tahapan awal yang sangat penting dalam proses analisis sentimen karena bertujuan untuk membersihkan dan menyiapkan data sebelum digunakan dalam proses pelatihan model. Data teks yang berasal dari media sosial maupun ulasan daring umumnya mengandung berbagai bentuk *noise* yang dapat memengaruhi kualitas hasil analisis. Kualitas data dijelaskan sangat menentukan performa model yang

dibangun karena model akan belajar berdasarkan informasi yang tersedia pada data tersebut [30]. Data teks juga dinyatakan sering mengandung kesalahan penulisan, simbol, angka, dan karakter yang tidak relevan sehingga memerlukan proses pembersihan sebelum dilakukan analisis lebih lanjut. Selain itu, Keberhasilan analisis sentimen ditegaskan sangat dipengaruhi oleh kualitas proses *preprocessing* yang dilakukan terhadap data. Oleh karena itu, penelitian ini menerapkan beberapa tahapan *preprocessing* yang meliputi *tokenization*, *punctuation removal*, *number removal*, *stemming, lemmatization*, *stopwords removal*, dan *spell correction*.

2.2.5.1 *Tokenization*

Tokenization merupakan proses memecah teks menjadi unit-unit yang lebih kecil yang disebut *token*. Proses ini bertujuan untuk mengubah data teks yang tidak terstruktur menjadi bentuk yang lebih mudah diproses oleh sistem komputer. *Tokenization* dijelaskan sebagai salah satu tahapan awal yang sangat penting dalam *text preprocessing* karena menjadi dasar bagi proses analisis selanjutnya [31]. Proses *tokenization* dinyatakan membantu dalam ekstraksi fitur dengan memisahkan setiap kata yang terdapat dalam dokumen menjadi unit yang berdiri sendiri. Proses ini biasanya dilakukan berdasarkan spasi, tanda baca, atau aturan tertentu yang disesuaikan dengan karakteristik data. Dengan adanya *tokenization*, representasi teks menjadi lebih sistematis sehingga mempermudah proses analisis sentimen.

2.2.5.2 *Punctuation Removal*

Punctuation removal merupakan proses penghapusan tanda baca yang dianggap tidak memiliki kontribusi signifikan terhadap makna sentimen dalam suatu teks. Keberadaan tanda baca sering kali menambah kompleksitas data tanpa memberikan informasi yang relevan terhadap proses analisis. Penghapusan tanda baca dijelaskan dapat membantu menyederhanakan representasi teks sehingga proses ekstraksi fitur menjadi lebih efektif [32]. Tanda baca seperti titik,

koma, tanda tanya, tanda seru, maupun simbol lainnya umumnya dihapus untuk mengurangi *noise* pada data. Dengan menghilangkan karakter yang tidak diperlukan, model dapat lebih fokus pada kata-kata yang mengandung informasi sentimen. Oleh karena itu, *punctuation removal* menjadi salah satu tahapan penting dalam meningkatkan kualitas data teks.

2.2.5.3 *Number Removal*

Number removal merupakan proses penghapusan angka atau karakter numerik yang terdapat dalam teks sebelum dilakukan analisis lebih lanjut. Tahapan ini bertujuan untuk mengurangi jumlah fitur yang tidak relevan sehingga proses pembelajaran model menjadi lebih efisien. Angka dijelaskan tidak memiliki kontribusi yang signifikan terhadap polaritas maupun kekuatan sentimen pada sebagian besar kasus analisis sentimen sehingga dapat dihilangkan dari data [33]. Selain mengurangi kompleksitas data, penghapusan angka juga membantu menghindari munculnya fitur yang tidak memiliki makna sentimen. Meskipun demikian, pada beberapa domain tertentu angka dapat memiliki makna penting sehingga penggunaannya perlu disesuaikan dengan tujuan penelitian. Dalam penelitian ini, angka dihapus karena fokus analisis berada pada ekspresi sentimen yang terkandung dalam kata-kata pada ulasan produk.

2.2.5.4 *Stemming*

Stemming merupakan proses mengubah kata berimbuhan menjadi bentuk dasarnya dengan cara menghilangkan awalan, akhiran, maupun sisipan yang melekat pada kata tersebut. Tahapan ini bertujuan untuk mengurangi variasi bentuk kata sehingga kata-kata yang memiliki akar yang sama dapat direpresentasikan secara seragam. *Stemming* dijelaskan membantu menyederhanakan kosakata yang digunakan dalam proses analisis teks sehingga model dapat lebih

mudah mengenali hubungan antarkata. [34]. *Stemming* juga dinyatakan mampu mengurangi dimensi fitur karena beberapa kata yang memiliki makna serupa akan dipetakan ke bentuk dasar yang sama. Dengan jumlah kosakata yang lebih sedikit, proses pelatihan model menjadi lebih efisien dan cepat. Oleh karena itu, *stemming* menjadi salah satu tahapan yang banyak digunakan dalam berbagai penelitian analisis sentimen.

2.2.5.5 *Lemmatization*

Lemmatization merupakan proses mengubah kata ke bentuk dasarnya atau *lemma* dengan mempertimbangkan struktur tata bahasa dan konteks penggunaannya dalam kalimat. Berbeda dengan *stemming*, metode ini menghasilkan bentuk dasar kata yang lebih sesuai secara linguistik. *Lemmatization* dijelaskan memiliki kemampuan yang lebih baik dalam mempertahankan makna kata karena mempertimbangkan informasi gramatikal selama proses transformasi kata [35]. *Lemmatization* juga dinyatakan dapat meningkatkan kualitas representasi teks karena mampu mengelompokkan kata-kata yang memiliki makna sama ke dalam bentuk dasar yang identik. Dengan demikian, hubungan semantik antar kata dapat dipahami dengan lebih baik oleh model. Oleh karena itu, *lemmatization* banyak diterapkan pada penelitian analisis sentimen modern yang membutuhkan representasi teks yang lebih akurat.

2.2.5.6 *Stopwords Removal*

Stopwords removal merupakan proses penghapusan kata-kata umum yang sering muncul dalam teks tetapi tidak memiliki kontribusi yang signifikan terhadap makna utama suatu kalimat. Kata-kata seperti “dan”, “atau”, “yang”, “di”, dan “ke” umumnya tidak mengandung informasi sentimen sehingga dapat dihilangkan dari data. Penghapusan *stopwords* dijelaskan dapat membantu

meningkatkan kualitas fitur dengan mengurangi kata-kata yang tidak relevan dalam proses analisis teks [36]. Selain itu, proses ini juga dapat mengurangi kompleksitas data dan mempercepat proses komputasi selama pelatihan model. Dengan berkurangnya jumlah kata yang tidak penting, model dapat lebih fokus pada kata-kata yang memiliki nilai informatif tinggi. Oleh karena itu, *stopwords removal* menjadi salah satu tahapan yang umum digunakan dalam *text preprocessing*.

2.2.5.7 *Spell Correction*

Spell correction merupakan proses memperbaiki kesalahan penulisan atau ejaan yang terdapat pada data teks sebelum dilakukan analisis lebih lanjut. Kesalahan ejaan sering ditemukan pada data ulasan produk karena pengguna cenderung menulis secara informal atau terburu-buru. Kesalahan penulisan dijelaskan dapat menyebabkan kata yang sebenarnya memiliki makna sama dianggap sebagai kata yang berbeda oleh sistem sehingga menurunkan kualitas analisis yang dihasilkan [37]. Proses *spell correction* dilakukan untuk memastikan bahwa setiap kata direpresentasikan dalam bentuk yang benar sesuai dengan kamus bahasa yang digunakan. Selain meningkatkan konsistensi data, tahapan ini juga membantu mengurangi ambiguitas dan meningkatkan kualitas fitur yang diekstraksi dari teks. Dengan demikian, model dapat memahami isi dokumen dengan lebih baik dan menghasilkan performa analisis sentimen yang lebih akurat.

2.2.6 Metode-Metode Analisis Sentimen

2.2.6.1 *Convolutional Neural Networks* (CNN)

CNN merupakan salah satu algoritma *deep learning* yang awalnya dikembangkan untuk pengolahan citra, namun saat ini telah banyak diterapkan pada berbagai tugas *Natural Language Processing (NLP)*, termasuk analisis sentimen. *CNN* dijelaskan

memiliki kemampuan untuk mengekstraksi fitur secara otomatis melalui proses konvolusi sehingga mampu mengenali pola-pola penting dalam data [38]. *CNN d*ijelaskan bekerja dengan memanfaatkan filter atau kernel untuk menemukan karakteristik tertentu dari data masukan. *CNN* dinyatakan menjadi salah satu metode *deep learning* yang populer karena mampu menghasilkan representasi fitur yang efektif dengan kompleksitas komputasi yang relatif rendah.

Arsitektur *CNN* pada penelitian ini terdiri atas *embedding layer, convolution layer, max pooling layer, fully connected layer,* dan *output layer*. *Embedding layer* berfungsi mengubah kata menjadi representasi vektor numerik yang dapat diproses oleh jaringan saraf. *Convolution layer* melakukan proses ekstraksi fitur menggunakan filter yang bergerak pada urutan kata. *Max pooling layer* digunakan untuk memilih fitur-fitur yang paling dominan sehingga dapat mengurangi dimensi data sekaligus mempertahankan informasi penting. Selanjutnya, *fully connected layer* menggabungkan seluruh fitur yang telah diekstraksi untuk menghasilkan keputusan akhir pada *output layer*.

Pada *convolution layer*, dilakukan proses operasi konvolusi untuk mengekstraksi fitur-fitur penting dari representasi teks. Operasi ini memungkinkan model mengenali pola lokal berupa kata maupun frasa yang memiliki hubungan dengan sentimen suatu ulasan. Proses konvolusi dijelaskan dilakukan dengan menggeser filter (*kernel*) pada representasi vektor kata sehingga menghasilkan *feature map* yang merepresentasikan karakteristik penting dari data masukan [39]. Perhitungan operasi konvolusi pada penelitian ini mengacu pada Persamaan (2.1)

Persamaan (2.1) Operasi *Convolution* pada *CNN*

$$c_i = f(\mathrm{w} \cdot \mathrm{x}_{i:i+h-1} + b) \tag{2.1}$$

Keterangan:

a. c_i= nilai fitur hasil konvolusi.
b. $x_{i:i+h-1}$= vektor kata pada jendela (*window*) berukuran h.
c. w= filter (*kernel*) yang dipelajari selama proses pelatihan.
d. b= nilai bias.
e. $f(\cdot)$= fungsi aktivasi, umumnya menggunakan ReLU.

Sumber: *Diadaptasi Dari* [39]

Berdasarkan persamaan (2.1), setiap filter melakukan operasi perkalian terhadap representasi vektor kata yang berada pada suatu jendela tertentu. Hasil operasi tersebut kemudian dijumlahkan dengan nilai *bias* dan diproses menggunakan fungsi aktivasi untuk menghasilkan nilai fitur. Proses konvolusi dilakukan secara berulang pada seluruh bagian kalimat sehingga menghasilkan *feature map* yang berisi informasi mengenai pola-pola penting pada teks. Selanjutnya, *feature map* diproses menggunakan *pooling layer* untuk memilih fitur yang paling representatif sebelum diteruskan menuju lapisan klasifikasi (*fully connected layer*).

Tabel 2.2 *Pseudocode CNN* untuk Klasifikasi Sentimen

Langkah	Proses
1	Masukkan data ulasan (*review text*) sebagai masukan model.
2	Lakukan *text preprocessing* untuk membersihkan data teks.
3	Ubah setiap kata menjadi representasi numerik (*embedding vector*).
4	Terapkan operasi konvolusi menggunakan beberapa filter (*multiple filters*).
5	Terapkan fungsi aktivasi ReLU pada hasil konvolusi.
6	Lakukan proses *max pooling* untuk memilih fitur yang paling dominan.
7	Ubah *feature map* menjadi vektor satu dimensi (*flatten*).
8	Masukkan hasil *flatten* ke lapisan *fully connected*.
9	Hitung probabilitas keluaran menggunakan fungsi *Softmax*.
10	Tentukan kelas sentimen berdasarkan probabilitas terbesar.

Sumber: Diadaptasi dari [39].

Pseudocode pada tabel 2.2 menunjukkan bahwa proses klasifikasi menggunakan *CNN* dimulai dari penerimaan data teks yang kemudian melalui tahap *preprocessing*. Setelah data dibersihkan, setiap kata dikonversi menjadi

representasi numerik menggunakan *word embedding*. Representasi tersebut diproses oleh lapisan *convolution* untuk mengekstraksi fitur-fitur penting yang berkaitan dengan sentimen. Selanjutnya dilakukan fungsi aktivasi *ReLU* dan *max pooling* untuk memperoleh fitur yang paling dominan sebelum diteruskan menuju lapisan *fully connected*. Pada tahap akhir, fungsi *Softmax* digunakan untuk menentukan kelas sentimen berdasarkan probabilitas terbesar yang dihasilkan model.

Dalam analisis sentimen, *CNN* bekerja dengan mengidentifikasi pola lokal berupa kata atau frasa yang sering muncul pada sentimen positif maupun negatif. Kata-kata seperti *"excellent", "delicious",* atau *"terrible"* dapat dikenali sebagai fitur penting yang memengaruhi kekuatan sentimen suatu ulasan. Kemampuan ini membuat *CNN* efektif dalam menangkap hubungan lokal antar kata yang sering menjadi indikator utama sentimen pelanggan. *CNN* dipilih sebagai salah satu model dalam penelitian ini karena kemampuannya dalam menangkap pola lokal yang sering muncul pada teks ulasan pelanggan. Penelitian yang dilakukan menunjukkan bahwa CNN mampu menghasilkan performa yang sangat baik pada tugas klasifikasi kalimat dan analisis sentimen. Integrasi fitur leksikon ke dalam CNN ditunjukkan mampu meningkatkan performa analisis sentimen. Hasil penelitian juga membuktikan bahwa kombinasi CNN dengan leksikon berbasis konteks dapat meningkatkan kemampuan deteksi kekuatan sentimen.

2.2.6.2 *Long Short-Term Memory* (*LSTM)*

LSTM merupakan pengembangan dari *Recurrent Neural Network (RNN)* yang dirancang untuk mengatasi masalah vanishing gradient dalam pemrosesan data sekuensial. *LSTM* dijelaskan memiliki mekanisme memori yang memungkinkan model mempertahankan informasi penting dalam jangka waktu yang lebih panjang dibandingkan *RNN* konvensional [40]. Kemampuan tersebut menjadikan *LSTM* sangat sesuai untuk memproses data teks karena hubungan antar kata sering kali dipengaruhi oleh konteks sebelumnya. *LSTM*

juga dinyatakan banyak digunakan pada berbagai aplikasi *NLP* karena mampu memahami pola urutan data secara efektif.

Arsitektur *LSTM* terdiri atas beberapa komponen utama, yaitu *input gate, forget gate, cell state,* dan *output gate. Forget gate* berfungsi untuk menentukan informasi yang perlu dipertahankan atau dihapus dari memori. *Input gate* mengatur informasi baru yang akan disimpan ke dalam *cell state. Cell state* berperan sebagai media penyimpanan informasi jangka panjang, sedangkan *output gate* menghasilkan keluaran berdasarkan informasi yang telah diproses. Secara matematis, mekanisme pada *LSTM* dapat dijelaskan melalui beberapa persamaan berikut.

1. *Forget Gate*

Forget gate merupakan gerbang pertama pada arsitektur *LSTM* yang bertugas menentukan informasi mana dari *cell state* sebelumnya yang masih perlu dipertahankan dan informasi mana yang harus dihilangkan. Keputusan tersebut dihasilkan berdasarkan kombinasi antara data masukan pada waktu ke-*t* dengan *hidden state* dari waktu sebelumnya. Nilai keluaran *forget gate* berada pada rentang 0 hingga 1 yang diperoleh menggunakan fungsi aktivasi sigmoid. Nilai yang mendekati 1 menunjukkan bahwa informasi akan dipertahankan, sedangkan nilai yang mendekati 0 menunjukkan bahwa informasi tersebut akan dihapus dari *cell state*. Mekanisme ini memungkinkan *LSTM* menghilangkan informasi yang tidak lagi relevan sehingga kapasitas memori dapat digunakan secara lebih efektif. Perhitungan *forget gate* pada penelitian ini mengacu pada Persamaan (2.2)

Persamaan (2.2) Perhitungan *forget gate*.

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad (2.2)$$

Keterangan:

a. f_t= nilai *forget gate*
b. W_f= bobot *forget gate*
c. h_{t-1}= *hidden state* sebelumnya
d. x_t= data masukan saat ini

e. b_f= bias

f. σ= fungsi sigmoid

Sumber: Diadaptasi dari [41]

Berdasarkan Persamaan (2.2), nilai keluaran *forget gate* berada pada rentang 0 hingga 1. Nilai yang mendekati 1 menunjukkan bahwa informasi akan dipertahankan, sedangkan nilai yang mendekati 0 menunjukkan bahwa informasi akan dihapus dari *cell state*. Mekanisme ini memungkinkan *LSTM* menghilangkan informasi yang tidak lagi relevan sehingga kapasitas memori dapat dimanfaatkan secara lebih efektif.

2. *Input Gate*

Setelah menentukan informasi yang harus dipertahankan, *LSTM* akan menghitung informasi baru yang akan ditambahkan ke dalam *cell state* melalui *input gate*. Gerbang ini terdiri atas dua proses utama, yaitu menghitung nilai gerbang menggunakan fungsi sigmoid dan menghasilkan kandidat informasi baru menggunakan fungsi aktivasi *tanh*. Kedua hasil tersebut kemudian dikombinasikan untuk memperbarui isi *cell state*. Dengan mekanisme tersebut, *LSTM* hanya menyimpan informasi yang benar-benar dianggap penting untuk proses pembelajaran berikutnya. Proses ini berperan besar dalam meningkatkan kemampuan model untuk memahami konteks kalimat yang panjang. Perhitungan *input gate* pada penelitian ini mengacu pada Persamaan (2.3), perhitungan kandidat *cell state* Persamaan (2.4) dan persamaan 2.5. Perhitungan Pembaruan *cell state.*

Persamaan (2.3) Perhitungan *input gate*.

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (2.3)$$

Keterangan:

a. i_t= nilai *input gate*

b. W_i= bobot *input gate*

c. h_{t-1}= *hidden state* sebelumnya

d. x_t= data masukan saat ini

e. b_i= bias

f. σ= fungsi sigmoid

Sumber: Diadaptasi dari [41]

Berdasarkan Persamaan (2.3), *input gate* menentukan informasi baru yang layak disimpan ke dalam *cell state*. Nilai *input gate* dihasilkan menggunakan fungsi aktivasi *sigmoid* sehingga berada pada rentang 0 hingga 1. Semakin besar nilai yang dihasilkan, semakin besar pula kontribusi informasi baru terhadap proses pembaruan memori. Hasil perhitungan *input gate* selanjutnya dikombinasikan dengan kandidat *cell state* untuk memperbarui *cell state* pada tahap berikutnya.

Selanjutnya, kandidat informasi baru dihitung menggunakan Persamaan (2.4). Kandidat *cell state* merupakan informasi baru yang berpotensi disimpan ke dalam memori. Nilai ini dihasilkan menggunakan fungsi aktivasi *tanh* sehingga berada pada rentang −1 hingga 1. Informasi tersebut selanjutnya dikombinasikan dengan hasil *input gate* untuk memperbarui *cell state* pada tahap berikutnya. Kandidat informasi baru dihitung menggunakan Persamaan (2.4). Kandidat *Cell State*

Persamaan 2.4 Perhitungan kandidat *cell state*:

$$\widetilde{C_t} = \tanh(W_c[h_{t-1}, x_t] + b_c) \quad (2.4)$$

Keterangan

a. Ct~\tilde{C_t}Ct~ = kandidat *cell state*
b. WcW_cWc = bobot kandidat *cell state*
c. bcb_cbc = *bias*

Sumber: Diadaptasi dari [41]

Berdasarkan Persamaan (2.4), kandidat *cell state* dihasilkan melalui fungsi aktivasi *tanh* yang memanfaatkan data masukan saat ini dan *hidden state* sebelumnya. Fungsi *tanh* menghasilkan nilai pada rentang −1 hingga 1 sehingga mampu merepresentasikan informasi baru yang akan dipertimbangkan untuk disimpan dalam memori. Nilai kandidat tersebut belum langsung menjadi bagian dari *cell state*, melainkan akan dikombinasikan dengan keluaran *input gate* pada

proses pembaruan *cell state*. Mekanisme ini memungkinkan LSTM menyimpan informasi baru yang relevan tanpa menghilangkan informasi penting yang masih diperlukan pada langkah waktu berikutnya.

Setelah kedua komponen diperoleh, pembaruan *cell state* dilakukan menggunakan Persamaan (2.5). Proses ini menggabungkan informasi lama dengan informasi baru yang telah dipilih. Informasi yang tidak relevan akan dikurangi pengaruhnya, sedangkan informasi penting akan dipertahankan. Hasil pembaruan *cell state* kemudian digunakan sebagai dasar dalam pembentukan *hidden state* pada tahap berikutnya.

Persamaan 2.5 Perhitungan pembaruan *cell state*:

$$C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C}_t \quad (2.5)$$

Keterangan

a. CtC_tCt = *cell state* saat ini
b. Ct−1C_{t-1}Ct−1 = *cell state* sebelumnya
c. ⊙\odot⊙ = perkalian elemen (*element-wise multiplication*)

Sumber: Diadaptasi dari [41]

Berdasarkan Persamaan (2.5), informasi lama dan informasi baru digabungkan sehingga *cell state* hanya menyimpan informasi yang dianggap penting untuk proses pembelajaran berikutnya. Mekanisme ini memungkinkan *LSTM* mempertahankan informasi yang masih relevan sekaligus menghilangkan informasi yang tidak diperlukan. Proses pembaruan *cell state* dilakukan secara adaptif sehingga model mampu mempelajari hubungan antarkata dalam suatu urutan teks secara lebih efektif. Kemampuan tersebut menjadikan *LSTM* lebih baik dalam menangkap dependensi jangka panjang pada data sekuensial.

3. Output Gate

Output gate merupakan gerbang terakhir yang berfungsi menentukan informasi yang akan diteruskan sebagai *hidden state* menuju langkah waktu berikutnya maupun lapisan berikutnya pada jaringan. Nilai keluaran diperoleh menggunakan fungsi sigmoid yang dikombinasikan dengan hasil aktivasi *tanh* pada *cell state* yang telah diperbarui. Informasi yang dihasilkan oleh *output gate*

menjadi representasi konteks yang digunakan dalam proses klasifikasi maupun prediksi. Dengan demikian, *output gate* memiliki peran penting dalam memastikan bahwa hanya informasi yang paling relevan yang diteruskan ke proses selanjutnya. Mekanisme tersebut menjadikan *LSTM* lebih efektif dalam menangkap hubungan jangka panjang dibandingkan *RNN* konvensional. Perhitungan *output gate* dilakukan untuk menentukan seberapa besar informasi pada *cell state* yang akan diteruskan menjadi *hidden state*. Nilai yang dihasilkan berfungsi sebagai pengontrol keluaran memori pada setiap langkah waktu. Proses perhitungan tersebut memanfaatkan *hidden state* sebelumnya dan data masukan saat ini. Perhitungan *output gate* pada penelitian ini mengacu pada Persamaan (2.6).

Persamaan (2.6) Perhitungan *Output Gate*

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (2.6)$$

Keterangan:

1. o_t= nilai *output gate*
2. W_o= bobot *output gate*
3. h_{t-1}= *hidden state* sebelumnya
4. x_t= data masukan saat ini
5. b_o= bias
6. σ= fungsi sigmoid

Sumber: Diadaptasi dari [41]

Berdasarkan Persamaan (2.6), nilai *output gate* dihasilkan menggunakan fungsi aktivasi *sigmoid* sehingga berada pada rentang 0 hingga 1. Nilai tersebut menentukan proporsi informasi dari *cell state* yang akan diteruskan sebagai *hidden state*. Semakin besar nilai *output gate*, semakin banyak informasi penting yang diteruskan ke langkah waktu berikutnya. Hasil perhitungan *output gate* kemudian digunakan bersama *cell state* yang telah diperbarui untuk menghasilkan *hidden state* pada proses selanjutnya. Selanjutnya, *hidden state* dihitung menggunakan Persamaan (2.7).

Persamaan (2.7) Perhitungan *hidden state*

$$h_t = o_t \odot \tanh(C_t) \quad (2.7)$$

Keterangan

a. hth_tht = *hidden state*

b. CtC_tCt = *cell state*

Sumber: Diadaptasi dari [41]

Berdasarkan Persamaan (2.7), *hidden state* yang dihasilkan menjadi representasi konteks yang diteruskan menuju langkah waktu berikutnya maupun lapisan klasifikasi. Mekanisme ini memungkinkan *LSTM* mempertahankan informasi penting dalam urutan kalimat yang panjang sehingga lebih efektif dibandingkan *RNN* konvensional dalam mengatasi permasalahan *vanishing gradient*. Persamaan tersebut menggambarkan mekanisme seleksi informasi dalam *LSTM* yang memungkinkan model mengontrol informasi yang disimpan maupun dihapus dalam proses pembelajaran. Formulasi *LSTM* pertama kali diperkenalkan oleh Hochreiter dan Schmidhuber dalam pengembangan arsitektur jaringan saraf berulang yang memiliki kemampuan memori jangka panjang.

Dalam analisis sentimen, *LSTM* bekerja dengan memproses kata demi kata secara berurutan sehingga mampu mempertahankan informasi penting yang muncul pada bagian awal kalimat hingga akhir kalimat. Kemampuan ini memungkinkan model memahami hubungan antar kata yang berjauhan dan menghasilkan interpretasi konteks yang lebih baik. Kelebihan utama *LSTM* adalah kemampuannya dalam memahami konteks jangka panjang dan mengatasi permasalahan vanishing gradient. Pemilihan *LSTM* dalam penelitian ini didasarkan pada kemampuannya mempertahankan informasi penting sepanjang urutan kalimat. Model ini pertama kali diperkenalkan oleh Hochreiter dan Schmidhuber dan hingga saat ini masih menjadi salah satu arsitektur utama dalam *NLP*. Penelitian menunjukkan bahwa LSTM mampu menghasilkan performa yang baik dalam prediksi intensitas sentimen karena dapat menangkap hubungan antar kata secara lebih efektif [42].

Tabel 2.3 *Pseudocode LSTM*

Langkah	Proses
1	Inisialisasi *hidden state* (h0h_0h0) dan *cell state* (C0C_0C0).
2	Membaca urutan data masukan (*input sequence*) xtx_txt.

Langkah	Proses
3	Menghitung nilai *forget gate* menggunakan Persamaan (2.2).
4	Menghitung nilai *input gate* menggunakan Persamaan (2.3).
5	Menghasilkan kandidat *cell state* menggunakan Persamaan (2.4).
6	Memperbarui *cell state* menggunakan Persamaan (2.5).
7	Menghitung nilai *output gate* menggunakan Persamaan (2.6).
8	Menghasilkan *hidden state* menggunakan Persamaan (2.7).
9	Mengulangi proses hingga seluruh urutan data selesai diproses.
10	Menghasilkan *hidden representation* sebagai keluaran model.

Sumber: Diadaptasi dari [42]

Pseudocode pada Tabel 2.3 menggambarkan alur kerja *algoritma LSTM* dalam memproses data sekuensial. Proses dimulai dengan menginisialisasi *hidden state* dan *cell state*, kemudian setiap data masukan diproses secara berurutan melalui *forget gate*, *input gate*, dan *output gate*. Ketiga gerbang tersebut bekerja sama untuk menentukan informasi yang dipertahankan, diperbarui, maupun diteruskan ke langkah waktu berikutnya. Setelah seluruh urutan selesai diproses, *hidden state* terakhir digunakan sebagai representasi konteks yang selanjutnya diteruskan menuju lapisan klasifikasi. Kemampuan mempertahankan informasi penting dalam urutan data menjadikan *LSTM* efektif digunakan pada berbagai tugas *NLP*, termasuk analisis sentimen.

2.2.6.3 *Bidirectional Long Short-Term Memory* (*Bi-LSTM*)

Bi-LSTM merupakan pengembangan dari arsitektur *LSTM* yang dirancang untuk memproses data sekuensial dari dua arah, yaitu arah maju (*forward*) dan arah mundur (*backward*). Berbeda dengan *LSTM* konvensional yang hanya memanfaatkan informasi dari urutan sebelumnya menuju urutan berikutnya, *Bi-LSTM* memanfaatkan informasi dari kedua arah secara bersamaan sehingga mampu menghasilkan representasi konteks yang lebih lengkap. Pendekatan ini memungkinkan model memahami hubungan antarkata tidak hanya berdasarkan kata-kata yang muncul sebelumnya, tetapi juga mempertimbangkan kata-kata yang muncul setelahnya dalam suatu kalimat. Kemampuan tersebut sangat penting dalam tugas *NLP*, terutama analisis sentimen, karena makna suatu kata sering kali dipengaruhi oleh

konteks sebelum maupun sesudah kata tersebut. Oleh karena itu, *Bi-LSTM* banyak digunakan pada penelitian analisis sentimen karena terbukti mampu meningkatkan akurasi klasifikasi dibandingkan *LSTM* satu arah [43].

Pada analisis sentimen, *Bi-LSTM* bekerja dengan menjalankan dua buah jaringan *LSTM* secara paralel. Jaringan pertama membaca urutan data dari awal hingga akhir (*forward LSTM*), sedangkan jaringan kedua membaca data dari akhir menuju awal (*backward LSTM*). Informasi yang dihasilkan dari kedua arah tersebut kemudian digabungkan sehingga model memperoleh representasi konteks yang lebih kaya dibandingkan *LSTM* konvensional. Dengan memanfaatkan informasi dari dua arah, *Bi-LSTM* mampu mengenali hubungan antar kata yang berjauhan dan memahami struktur kalimat secara lebih baik. Oleh karena itu, algoritma ini banyak diterapkan pada berbagai penelitian klasifikasi teks, pengenalan entitas, penerjemahan mesin, serta analisis sentimen berbasis *deep learning*.

Dalam penelitian ini, *Bi-LSTM* digunakan setelah proses ekstraksi fitur dilakukan oleh *CNN*. *CNN* bertugas mengekstraksi fitur lokal berupa pola atau frasa penting dari teks, sedangkan *Bi-LSTM* mempelajari hubungan kontekstual antar fitur tersebut dari arah maju dan arah mundur. Kombinasi kedua algoritma tersebut memungkinkan model memperoleh representasi fitur yang lebih komprehensif sehingga dapat meningkatkan kemampuan klasifikasi sentimen. Selain itu, *Bi-LSTM* juga mampu mengurangi kehilangan informasi yang sering terjadi pada model satu arah karena mempertimbangkan seluruh konteks kalimat sebelum menghasilkan prediksi. Dengan demikian, penggunaan *CNN-BiLSTM* diharapkan mampu memberikan performa yang lebih baik dibandingkan penggunaan *CNN* atau *LSTM* secara terpisah. Pendekatan dua arah ini dijelaskan memungkinkan model memperoleh informasi yang lebih lengkap dibandingkan *LSTM* konvensional. Konteks suatu kata dijelaskan sering dipengaruhi oleh kata sebelum maupun sesudahnya sehingga

pemrosesan dua arah dapat meningkatkan pemahaman model terhadap makna kalimat.

Arsitektur *Bi-LSTM* terdiri atas dua jaringan *LSTM* yang berjalan secara paralel. *Forward LSTM* memproses data dari awal menuju akhir kalimat, sedangkan *backward LSTM* memproses data dari akhir menuju awal kalimat. Hasil kedua proses tersebut kemudian digabungkan untuk menghasilkan representasi fitur yang lebih kaya. Cara kerja *Bi-LSTM* diawali dengan menerima data masukan berupa urutan kata yang telah dikonversi menjadi representasi numerik menggunakan teknik *word embedding*. Selanjutnya, data tersebut diproses secara bersamaan oleh dua jaringan *LSTM* yang memiliki arah pemrosesan berbeda. Jaringan *forward LSTM* membaca urutan data mulai dari kata pertama hingga kata terakhir, sedangkan *backward LSTM* membaca urutan data dari kata terakhir menuju kata pertama. Masing-masing jaringan menghasilkan *hidden state* yang merepresentasikan informasi kontekstual berdasarkan arah pemrosesannya. Kedua hasil tersebut kemudian digabungkan (*concatenate*) sehingga menghasilkan representasi akhir yang mengandung informasi dari seluruh konteks kalimat.

Setelah proses penggabungan selesai, representasi hasil *Bi-LSTM* diteruskan menuju lapisan klasifikasi, seperti *fully connected layer* dan fungsi aktivasi *Softmax*. Lapisan *fully connected* bertugas menghubungkan seluruh fitur hasil ekstraksi menjadi representasi yang lebih komprehensif sebelum dilakukan proses klasifikasi. Selanjutnya, fungsi *Softmax* menghitung probabilitas masing-masing kelas sentimen sehingga model dapat menentukan kelas dengan probabilitas tertinggi sebagai hasil prediksi.

Selama proses pelatihan, seluruh parameter jaringan diperbarui menggunakan algoritma *Backpropagation Through Time* (*BPTT*) sehingga bobot jaringan terus disesuaikan untuk meminimalkan nilai *loss function*. Dengan mekanisme tersebut, *Bi-LSTM* mampu mempelajari hubungan jangka panjang dari dua arah sekaligus sehingga menghasilkan

representasi konteks yang lebih baik dibandingkan *LSTM* satu arah. Pada *Bi-LSTM*, proses pertama dilakukan oleh jaringan *LSTM* yang bekerja dari arah kiri ke kanan (*forward*). Jaringan ini menghasilkan *hidden state* pada setiap langkah waktu berdasarkan data masukan saat ini dan *hidden state* sebelumnya. Perhitungan *forward hidden state* dilakukan menggunakan Persamaan (2.8).

Persamaan (2.8) Perhitungan *Forward Hidden State*

$$\overrightarrow{h_t} = LSTM_f(x_t, \overrightarrow{h_{t-1}}) \quad (2.8)$$

Keterangan

a. ht→\overrightarrow{h_t}ht = *hidden state* arah maju (*forward hidden state*)
b. xtx_txt = data masukan pada waktu ke-ttt
c. ht−1→\overrightarrow{h_{t-1}}ht−1 = *hidden state* arah maju pada waktu sebelumnya
d. LSTMfLSTM_fLSTMf = jaringan LSTM arah maju

Sumber: Diadaptasi dari [43]

Berdasarkan Persamaan (2.8), jaringan *LSTM* arah maju mempelajari hubungan antar kata dari awal hingga akhir kalimat sehingga mampu menangkap informasi kontekstual berdasarkan urutan normal teks. Selain memproses data dari arah maju, *Bi-LSTM* juga menggunakan jaringan *LSTM* kedua yang bekerja dari arah kanan ke kiri (*backward*). Tujuannya adalah memperoleh informasi dari kata-kata yang muncul setelah posisi kata saat ini. Perhitungan *backward hidden state* dilakukan menggunakan Persamaan (2.9).

Persamaan (2.9) Perhitungan *Backward Hidden State*

$$\overleftarrow{h_t} = LSTM_b(x_t, \overleftarrow{h_{t+1}}) \quad (2.9)$$

Keterangan

a. ht←\overleftarrow{h_t}ht = *hidden state* arah mundur (*backward hidden state*)
b. xtx_txt = data masukan pada waktu ke-ttt
c. ht+1←\overleftarrow{h_{t+1}}ht+1= *hidden state* arah mundur pada waktu berikutnya
d. LSTMbLSTM_bLSTMb = jaringan *LSTM* arah mundur

Sumber: Diadaptasi dari [43]

Berdasarkan Persamaan (2.9), jaringan *LSTM* arah mundur mempelajari hubungan antar kata dari akhir menuju awal kalimat sehingga mampu menangkap informasi kontekstual dari arah yang berlawanan. Setelah diperoleh *forward hidden state* dan *backward hidden state*, kedua representasi tersebut digabungkan menggunakan operasi konkatenasi (*concatenation*) sehingga diperoleh representasi konteks akhir. Proses penggabungan tersebut dihitung menggunakan Persamaan (2.10).

Persamaan (2.10) Penggabungan *Hidden State*

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}] \quad (2.10)$$

Keterangan

a. hth_tht = representasi akhir hasil penggabungan
b. ht→\overrightarrow{h_t}ht = *forward hidden state*
c. ht←\overleftarrow{h_t}ht = *backward hidden state*

Sumber: Diadaptasi dari [43]

Berdasarkan Persamaan (2.10), representasi akhir diperoleh dengan menggabungkan informasi dari dua arah pemrosesan. Representasi tersebut mengandung konteks sebelum dan sesudah suatu kata sehingga menghasilkan informasi yang lebih kaya dibandingkan *LSTM* satu arah. Kemampuan tersebut menjadikan *Bi-LSTM* banyak digunakan pada berbagai tugas analisis sentimen karena mampu memahami hubungan antar

kata secara lebih komprehensif. Persamaan tersebut menunjukkan bahwa *Bi-LSTM* memproses data secara independen pada dua arah yang berbeda, kemudian menggabungkan hasilnya menjadi satu representasi. Representasi gabungan tersebut mengandung informasi mengenai konteks sebelum dan sesudah suatu kata sehingga mampu meningkatkan kemampuan model dalam memahami makna kalimat secara keseluruhan. Dibandingkan dengan *LSTM* satu arah, pendekatan ini menghasilkan representasi yang lebih kaya sehingga banyak digunakan pada berbagai tugas pemrosesan bahasa alami. Berikut ini adalah *pseudocode*nya:

Tabel 2.4 *Pseudocode Bi-LSTM*

Langkah	Proses
1	Inisialisasi jaringan *Forward LSTM*
2	Inisialisasi jaringan *Backward LSTM*.
3	Membaca urutan data masukan (*input sequence*).
4	Memproses urutan data dari kiri ke kanan menggunakan *Forward LSTM*.
5	Memproses urutan data dari kanan ke kiri menggunakan *Backward LSTM*.
6	Menghasilkan *forward hidden state*
7	Menghasilkan *backward hidden state*.
8	Menggabungkan (*concatenate*) kedua *hidden state*.
9	Meneruskan hasil konkatenasi ke lapisan *fully connected*.
10	Menghitung probabilitas menggunakan fungsi *Softmax*.
11	Menghasilkan prediksi kelas sentiment.

Sumber: Diadaptasi dari [43].

Pseudocode pada Tabel 2.4 menggambarkan alur kerja algoritma Bi-LSTM dalam memproses data sekuensial dari dua arah secara bersamaan. Proses diawali dengan menginisialisasi jaringan *Forward LSTM* dan *Backward LSTM* yang bekerja secara independen. Selanjutnya, urutan data diproses dari arah kiri ke kanan dan kanan ke kiri sehingga masing-masing menghasilkan *hidden state*. Kedua *hidden state* tersebut kemudian digabungkan menggunakan operasi *concatenate* untuk membentuk representasi konteks yang lebih lengkap. Representasi hasil penggabungan diteruskan menuju lapisan *fully connected* sebelum dihitung probabilitas kelas menggunakan fungsi *Softmax*. Hasil akhir berupa kelas sentimen dengan probabilitas tertinggi digunakan sebagai keluaran model klasifikasi. Pendekatan dua arah ini memungkinkan *Bi-LSTM* memahami hubungan antar kata secara lebih komprehensif sehingga

sering menghasilkan performa yang lebih baik dibandingkan *LSTM* satu arah pada tugas analisis sentimen.

Keunggulan utama *Bi-LSTM* adalah kemampuannya memanfaatkan informasi kontekstual dari dua arah secara bersamaan sehingga menghasilkan representasi teks yang lebih kaya dibandingkan *LSTM* konvensional. Namun, penggunaan dua jaringan *LSTM* menyebabkan kebutuhan memori dan waktu komputasi menjadi lebih besar. Pada penelitian ini, *Bi-LSTM* dipilih karena mampu memanfaatkan konteks sebelum dan sesudah suatu kata sehingga berpotensi meningkatkan akurasi deteksi kekuatan sentimen. Penelitian menunjukkan bahwa *Bi-LSTM* mampu meningkatkan performa deteksi sentimen dibandingkan pendekatan satu arah. Hasil serupa juga ditunjukkan oleh penelitian yang menyatakan bahwa kombinasi *CNN* dan *Bi-LSTM* menghasilkan performa yang unggul pada tugas klasifikasi sentimen. *Bi-LSTM* juga dibuktikan efektif dalam memahami pola linguistik yang kompleks pada data teks. Oleh karena itu, *Bi-LSTM* menjadi salah satu algoritma yang efektif untuk meningkatkan performa klasifikasi sentimen berbasis teks. Keunggulan utama *Bi-LSTM* adalah kemampuannya memahami konteks secara lebih lengkap sehingga sering menghasilkan performa yang lebih baik dibandingkan *LSTM* biasa. Namun demikian, penggunaan dua jaringan *LSTM* menyebabkan kebutuhan memori dan waktu komputasi menjadi lebih besar.

Bi-LSTM dipilih dalam penelitian ini karena mampu memanfaatkan konteks dari dua arah sekaligus sehingga berpotensi menghasilkan deteksi kekuatan sentimen yang lebih akurat. Penelitian menunjukkan bahwa *Bi-LSTM* mampu meningkatkan performa deteksi sentimen dibandingkan pendekatan satu arah karena dapat menangkap konteks secara lebih komprehensif. Hasil serupa juga ditemukan dalam penelitian yang menunjukkan bahwa kombinasi *CNN* dan *Bi-LSTM* menghasilkan performa yang unggul pada tugas klasifikasi sentimen. *Bi-LSTM* juga

dibuktikan efektif dalam memahami pola linguistik yang kompleks pada data teks.

2.2.6.4 *Sentiment Strength Scoring Lexicon* (SSS-Lex)

Sentiment Strength Scoring Lexicon (SSS-Lex) merupakan kumpulan kata atau frasa yang telah diberikan nilai tertentu untuk merepresentasikan polaritas maupun intensitas sentimen yang terkandung di dalamnya. Setiap kata dalam leksikon memiliki bobot yang menunjukkan kecenderungan sentimen positif, negatif, atau netral sehingga dapat digunakan sebagai dasar dalam proses analisis sentimen. Pendekatan berbasis leksikon memanfaatkan pengetahuan linguistik yang telah tersedia untuk mengidentifikasi makna emosional suatu kata tanpa memerlukan proses pelatihan model yang kompleks. Oleh karena itu, metode ini banyak digunakan sebagai salah satu teknik dasar dalam analisis sentimen karena mampu memberikan interpretasi yang mudah dipahami dan dapat dikombinasikan dengan berbagai algoritma *machine learning* maupun *deep learning*. Selain itu, penggunaan leksikon sentimen dapat meningkatkan kualitas representasi fitur karena setiap kata tidak hanya direpresentasikan sebagai token, tetapi juga memiliki informasi mengenai kecenderungan emosinya [44].

SSS-Lex merupakan pengembangan dari pendekatan leksikon sentimen yang dirancang untuk mengukur kekuatan sentimen suatu kata melalui pemberian skor intensitas. Berbeda dengan leksikon konvensional yang hanya mengelompokkan kata ke dalam kategori positif atau negatif, *SSS-Lex* memberikan nilai numerik yang menunjukkan tingkat kekuatan emosi dari setiap kata. Sebagai contoh, kata *good*, *great*, dan *excellent* sama-sama memiliki polaritas positif, tetapi masing-masing memiliki tingkat intensitas yang berbeda sehingga diberikan skor yang berbeda pula. Demikian pula pada kata *bad*, *poor*, dan *terrible*, yang memiliki tingkat kekuatan sentimen negatif yang berbeda.

Dengan adanya skor intensitas tersebut, analisis sentimen tidak hanya mampu menentukan arah sentimen, tetapi juga dapat mengukur seberapa kuat opini yang disampaikan oleh pengguna sehingga menghasilkan representasi sentimen yang lebih rinci dibandingkan pendekatan berbasis polaritas saja. Konsep pengukuran intensitas sentimen menjadi penting karena dalam praktiknya tidak semua opini memiliki tingkat emosi yang sama. Dua ulasan yang sama-sama memiliki sentimen positif belum tentu menunjukkan tingkat kepuasan yang setara. Sebagai contoh, kalimat *"The food is good"* dan *"The food is absolutely amazing"* sama-sama menunjukkan sentimen positif, tetapi kalimat kedua memiliki tingkat kepuasan yang lebih tinggi. Oleh karena itu, penggunaan *SSS-Lex* memungkinkan sistem membedakan variasi intensitas sentimen sehingga informasi yang diperoleh menjadi lebih representatif terhadap opini pengguna. Pendekatan tersebut banyak digunakan dalam penelitian *sentiment intensity detection* maupun *opinion mining* karena mampu memberikan informasi yang lebih kaya dibandingkan klasifikasi sentimen berbasis polaritas.

Dalam penelitian ini, *SSS-Lex* digunakan sebagai sumber informasi semantik yang memberikan skor intensitas pada setiap kata hasil *text preprocessing*. Skor tersebut kemudian dipadukan dengan representasi *word embedding* sebelum diproses menggunakan model *CNN-BiLSTM*. Integrasi antara informasi semantik dari leksikon dan kemampuan model *deep learning* dalam mempelajari hubungan kontekstual antar kata diharapkan mampu menghasilkan representasi fitur yang lebih baik. Dengan demikian, model tidak hanya mempelajari pola kata berdasarkan distribusi kemunculannya, tetapi juga mempertimbangkan tingkat kekuatan sentimen yang terkandung pada setiap kata. Pendekatan ini diharapkan dapat meningkatkan performa klasifikasi sentimen pada ulasan pelanggan McDonald's.

Cara kerja *SSS-Lex* dimulai dengan melakukan proses *text preprocessing* terhadap data ulasan pelanggan. Tahapan ini meliputi *case*

folding, *tokenization*, *stopword removal*, serta *lemmatization* untuk menghasilkan kumpulan kata yang bersih dan siap dianalisis. Setelah proses tersebut selesai, setiap kata akan dicocokkan dengan entri yang terdapat pada kamus SSS-Lex. Jika kata ditemukan dalam leksikon, sistem akan mengambil skor intensitas sentimen yang sesuai, sedangkan kata yang tidak terdapat dalam leksikon tidak akan memberikan kontribusi terhadap nilai sentimen. Proses pencocokan tersebut dilakukan terhadap seluruh kata sehingga diperoleh skor sentimen untuk setiap kata yang mengandung opini.

Setelah seluruh kata memperoleh skor sentimen, sistem akan menghitung nilai *sentiment strength* dengan menjumlahkan seluruh skor yang diperoleh. Nilai tersebut menggambarkan tingkat kekuatan sentimen dari suatu dokumen atau kalimat. Semakin besar nilai positif yang dihasilkan, maka semakin kuat kecenderungan sentimen positif pada dokumen tersebut. Sebaliknya, semakin kecil atau semakin negatif nilai yang diperoleh, maka semakin kuat kecenderungan sentimen negatif yang terkandung di dalamnya. Nilai *sentiment strength* tersebut selanjutnya digunakan sebagai fitur tambahan yang dikombinasikan dengan representasi *word embedding* sebelum diproses oleh model *CNN-BiLSTM* sehingga model memperoleh informasi semantik dan kontekstual secara bersamaan. Berikut ini tahapan kerja *SSS-Lex:*

Tabel 2.5 Tahapan Kerja *Sentiment Strength Scoring Lexicon (SSS-Lex)*

Langkah	Proses
1	Melakukan *text preprocessing* terhadap data ulasan.
2	Melakukan *tokenization* sehingga diperoleh daftar kata.
3	Mencocokkan setiap token dengan kamus SSS-Lex
4	Mengambil skor intensitas sentimen untuk setiap kata yang ditemukan.
5	Menjumlahkan seluruh skor sentimen.
6	Menghasilkan nilai *sentiment strength* sebagai label setiap ulasan.

Sumber: Diadaptasi dari [44].

Pada Tabel 2.5 menunjukkan bahwa proses pelabelan menggunakan *SSS-Lex* diawali dengan melakukan *text preprocessing* sehingga diperoleh token yang bersih dan konsisten. Selanjutnya setiap

token dicocokkan dengan kamus *SSS-Lex* untuk memperoleh skor sentimen. Apabila suatu kata tidak ditemukan pada leksikon, maka kata tersebut tidak memberikan kontribusi terhadap skor sentimen. Seluruh skor yang diperoleh kemudian dijumlahkan sehingga menghasilkan nilai *sentiment strength* yang digunakan sebagai label target pada proses pelatihan model *deep learning*.

Perhitungan *sentiment strength* dilakukan dengan menjumlahkan seluruh skor sentimen dari kata-kata yang terdapat pada suatu dokumen. Setiap kata memiliki skor yang menunjukkan tingkat kekuatan emosinya sehingga kata dengan intensitas yang lebih tinggi akan memberikan kontribusi yang lebih besar terhadap nilai akhir sentimen. Pendekatan ini memungkinkan sistem membedakan ulasan yang memiliki polaritas sama tetapi intensitas yang berbeda. Dengan demikian, hasil analisis sentimen menjadi lebih representatif terhadap opini pengguna. Nilai akhir yang diperoleh selanjutnya digunakan sebagai salah satu fitur dalam proses klasifikasi menggunakan CNN-BiLSTM. Nilai *sentiment strength* diperoleh dengan menjumlahkan seluruh skor sentimen dari kata-kata yang terdapat dalam suatu dokumen. Proses perhitungan tersebut dilakukan menggunakan Persamaan (2.11).

Persamaan (2.11) Perhitungan *Sentiment Strength*

$$SS = \sum_{i=1}^{n} Score(w_i) \tag{2.11}$$

Keterangan:

a. SS = nilai *sentiment strength*.
b. Score (w_i) = skor sentimen dari kata ke-i.
c. n = jumlah kata yang memiliki skor sentimen.

Sumber: Diadaptasi dari [44]

Berdasarkan Persamaan (2.11), nilai *sentiment strength* diperoleh dengan menjumlahkan seluruh skor sentimen dari kata-kata

yang terdapat pada suatu ulasan. Nilai positif menunjukkan kecenderungan sentimen positif, sedangkan nilai negatif menunjukkan kecenderungan sentimen negatif. Semakin besar nilai absolut yang diperoleh, semakin tinggi intensitas sentimen yang terkandung dalam ulasan tersebut. Pada penelitian ini, nilai *sentiment strength* digunakan sebagai label target untuk melatih model *CNN*, *LSTM*, dan *Bi-LSTM* dalam memprediksi kekuatan sentimen pelanggan McDonald's.

Dalam penelitian ini, *SSS-Lex* digunakan sebagai sumber pembobotan sentimen untuk menghasilkan representasi fitur yang lebih informatif. Skor sentimen yang dihasilkan oleh leksikon kemudian digunakan sebagai fitur tambahan pada model *CNN*, *LSTM*, dan *Bi-LSTM*. Integrasi tersebut bertujuan untuk membantu model memahami tidak hanya konteks linguistik tetapi juga tingkat intensitas sentimen yang terkandung dalam teks. Dengan demikian, proses prediksi kekuatan sentimen diharapkan dapat dilakukan secara lebih akurat. Penggunaan *SSS-Lex* juga memungkinkan dilakukan evaluasi terhadap efektivitas leksikon berbasis intensitas dibandingkan leksikon umum seperti *VADER*.

2.2.6.5 *Valence Aware Dictionary and Sentiment Reasoner* (VADER)

Valence Aware Dictionary and Sentiment Reasoner (VADER) merupakan metode analisis sentimen berbasis leksikon (*lexicon-based sentiment analysis*) untuk menganalisis sentimen pada teks media sosial maupun teks informal lainnya. *VADER* dirancang dengan menggabungkan kamus sentimen yang telah diberi skor valensi dengan serangkaian aturan linguistik sehingga mampu menghasilkan penilaian sentimen yang lebih mendekati persepsi manusia. Berbeda dengan metode leksikon konvensional, *VADER* tidak hanya memperhatikan polaritas kata, tetapi juga mempertimbangkan berbagai karakteristik bahasa, seperti penggunaan huruf kapital, tanda baca, kata negasi, kata

penguat (*intensifier*), serta emotikon yang sering digunakan dalam komunikasi digital.

Pendekatan tersebut membuat *VADER* mampu menangkap nuansa emosi yang terdapat pada suatu kalimat secara lebih baik dibandingkan metode berbasis leksikon yang hanya mengandalkan pencocokan kata. Oleh karena itu, *VADER* menjadi salah satu metode analisis sentimen berbasis leksikon yang banyak digunakan sebagai acuan maupun metode pembanding dalam berbagai penelitian analisis sentimen [45]. Keunggulan utama *VADER* terletak pada kemampuannya melakukan analisis sentimen tanpa memerlukan proses pelatihan (*training*) model. Seluruh proses analisis dilakukan berdasarkan kamus leksikon yang telah memiliki skor valensi dan aturan linguistik yang telah ditentukan sebelumnya. Dengan demikian, *VADER* dapat diterapkan pada berbagai jenis data teks tanpa membutuhkan dataset berlabel dalam jumlah besar. Selain memiliki kompleksitas komputasi yang relatif rendah, metode ini juga mudah diimplementasikan karena tersedia sebagai pustaka (*library*) pada berbagai bahasa pemrograman, seperti *Python*. Karakteristik tersebut menjadikan *VADER* sebagai salah satu metode yang banyak digunakan untuk mengevaluasi performa model *machine learning* maupun *deep learning* dalam penelitian analisis sentimen.

Lexicon VADER merupakan kamus sentimen yang berisi ribuan kata, frasa, singkatan, emotikon, dan simbol yang telah diberikan skor valensi berdasarkan hasil penilaian manusia. Setiap entri dalam kamus memiliki nilai yang menunjukkan tingkat polaritas sentimen, mulai dari sangat negatif hingga sangat positif. Nilai tersebut diperoleh melalui proses anotasi yang melibatkan sejumlah penilai (*human raters*) sehingga skor yang dihasilkan merepresentasikan persepsi manusia terhadap makna emosional suatu kata. Dengan adanya skor valensi tersebut, *VADER* mampu menentukan kecenderungan sentimen suatu kalimat secara lebih objektif. Pendekatan ini menjadikan *VADER* lebih

adaptif terhadap karakteristik bahasa yang umum digunakan pada media sosial maupun ulasan *daring*.

Selain memanfaatkan skor valensi setiap kata, *VADER* juga menerapkan beberapa aturan linguistik untuk menyesuaikan nilai sentimen berdasarkan konteks kalimat. Misalnya, keberadaan kata negasi seperti *not* atau *never* dapat mengubah arah polaritas suatu kata, sedangkan kata penguat seperti *very* atau *extremely* dapat meningkatkan nilai sentimen suatu kata. *VADER* juga mempertimbangkan penggunaan huruf kapital, pengulangan tanda seru, serta emotikon sebagai indikator tambahan dalam menentukan kekuatan sentimen. Dengan mempertimbangkan aspek-aspek tersebut, hasil analisis sentimen menjadi lebih representatif terhadap makna sebenarnya dari suatu kalimat. Oleh karena itu, *VADER* dinilai mampu memberikan hasil analisis yang lebih baik dibandingkan pendekatan leksikon yang hanya mengandalkan pencocokan kata. Tahapan kerja *VADER* pada penelitian ini dilakukan untuk menghasilkan nilai *compound score* yang merepresentasikan kekuatan sentimen setiap ulasan pelanggan.

Tabel 2.6 Tahapan Kerja VADER

Langkah	Proses
1	Melakukan *text preprocessing* terhadap ulasan pelanggan
2	Memasukkan teks ke dalam *SentimentIntensityAnalyzer*.
3	Mencocokkan setiap kata dengan kamus leksikon VADER.
4	Menerapkan aturan linguistik (negasi, *intensifier*, kapitalisasi, tanda baca, dan emotikon).
5	Menghitung skor *positive, neutral, negative*, dan *compound*.
6	Menggunakan nilai *compound score* sebagai label sentimen setiap ulasan.

Sumber: Diadaptasi dari [45]

Pada Tabel 2.6 menunjukkan bahwa proses pelabelan menggunakan *VADER* dimulai dengan melakukan *text preprocessing* terhadap data ulasan. Selanjutnya, setiap kalimat diproses menggunakan *SentimentIntensityAnalyzer* untuk memperoleh skor sentimen berdasarkan kamus leksikon dan aturan linguistik yang dimiliki *VADER*. Hasil akhir berupa nilai *compound score* digunakan sebagai representasi kekuatan sentimen (*sentiment strength*) sekaligus sebagai label target pada proses pelatihan model *CNN, LSTM,* dan *Bi-LSTM*.

Compound score merupakan skor utama yang dihasilkan oleh *VADER* untuk menunjukkan kecenderungan sentimen suatu kalimat secara keseluruhan. Nilai ini diperoleh melalui proses normalisasi terhadap total skor valensi seluruh kata yang terdapat pada kalimat sehingga menghasilkan nilai dalam rentang −1 hingga +1. Nilai yang mendekati +1 menunjukkan sentimen positif yang kuat, sedangkan nilai yang mendekati −1 menunjukkan sentimen negatif yang kuat. Apabila nilai *compound score* mendekati nol, maka kalimat tersebut cenderung memiliki sentimen netral. Skor inilah yang paling sering digunakan sebagai dasar dalam proses klasifikasi sentimen menggunakan *VADER.*

Dalam praktiknya, nilai *compound score* digunakan untuk menentukan kategori sentimen berdasarkan ambang batas (*threshold*) tertentu. Apabila nilai *compound score* ≥ 0,05 maka kalimat diklasifikasikan sebagai sentimen positif. Sebaliknya, apabila nilai *compound score* ≤ −0,05 maka kalimat dikategorikan sebagai sentimen negatif. Nilai yang berada di antara −0,05 hingga 0,05 diklasifikasikan sebagai sentimen netral. Pendekatan tersebut memungkinkan *VADER* melakukan klasifikasi sentimen secara sederhana namun tetap memiliki tingkat akurasi yang baik pada berbagai jenis data teks. Selain menghasilkan *compound score, VADER* juga menghasilkan tiga skor tambahan, yaitu *positive (pos), neutral (neu),* dan *negative (neg).* Ketiga skor tersebut menunjukkan proporsi sentimen positif, netral, dan negatif yang terkandung dalam suatu kalimat. Nilai masing-masing skor berada pada rentang 0 hingga 1, dan jumlah keseluruhannya sama dengan 1. Berbeda dengan *compound score,* ketiga nilai ini tidak digunakan secara langsung untuk menentukan kelas sentimen, melainkan untuk memberikan gambaran mengenai distribusi polaritas dalam suatu teks.

Sebagai contoh, sebuah kalimat dapat memiliki nilai *pos = 0,65, neu = 0,20, dan neg = 0,15.* Hasil tersebut menunjukkan bahwa sebagian besar isi kalimat mengandung sentimen positif meskipun masih terdapat unsur netral maupun negatif. Informasi tersebut dapat

dimanfaatkan untuk memahami karakteristik suatu ulasan secara lebih rinci. Dalam penelitian ini, *compound score* digunakan sebagai pembanding terhadap skor yang dihasilkan oleh *SSS-Lex* sehingga dapat diketahui perbedaan representasi sentimen antara kedua pendekatan berbasis leksikon tersebut. *VADER* menghasilkan empat skor sentimen, yaitu *positive (pos), neutral (neu), negative (neg),* dan *compound*. Pada penelitian ini digunakan *compound score* karena skor tersebut merepresentasikan kecenderungan sentimen suatu kalimat secara keseluruhan. Nilai *compound score* diperoleh melalui proses normalisasi terhadap total skor valensi sebagaimana dinyatakan pada Persamaan (2.12). Proses normalisasi dilakukan agar nilai sentimen berada pada rentang yang seragam. Hal ini memudahkan proses interpretasi dan perbandingan antarulasan. Nilai yang dihasilkan mencerminkan tingkat kecenderungan sentimen dari suatu teks. Oleh karena itu, *compound score* digunakan sebagai skor utama dalam proses analisis sentimen menggunakan VADER.

Persamaan (2.12) Perhitungan *Compound Score VADER* [45].

$$Compound = \frac{x}{\sqrt{x^2 + \alpha}} \tag{2.12}$$

Keterangan:

a. *Compound* = skor sentimen akhir.
b. x = jumlah seluruh skor valensi kata dalam kalimat.
c. α = konstanta normalisasi, yaitu 15.

Sumber: Diadaptasi dari [45]

Berdasarkan Persamaan (2.12), nilai *compound score* diperoleh melalui proses normalisasi terhadap total skor valensi seluruh kata pada suatu kalimat sehingga menghasilkan rentang nilai antara −1 hingga +1. Nilai yang mendekati +1 menunjukkan sentimen positif yang semakin kuat, sedangkan nilai yang mendekati −1 menunjukkan sentimen negatif yang semakin kuat. Apabila nilai berada di sekitar 0, maka

kalimat cenderung memiliki sentimen netral. Nilai *compound* ≥ 0,05 dikategorikan sebagai sentimen positif, nilai *compound* ≤ −0,05 dikategorikan sebagai sentimen negatif, sedangkan nilai yang berada di antara kedua batas tersebut dikategorikan sebagai sentimen netral.

2.2.6.6 Alasan Pemilihan *CNN*, *LSTM*, dan *Bi-LSTM*

Pemilihan *Convolutional Neural Network* (*CNN*), *Long Short-Term Memory* (*LSTM*), dan *Bidirectional Long Short-Term Memory* (*Bi-LSTM*) dalam penelitian ini didasarkan pada perbedaan karakteristik masing-masing model dalam memahami informasi tekstual. Ketiga model tersebut merupakan arsitektur *deep learning* yang telah banyak digunakan dalam berbagai penelitian analisis sentimen karena mampu mempelajari representasi fitur secara otomatis tanpa memerlukan proses *feature engineering* yang kompleks. Model *deep learning* dijelaskan memiliki kemampuan untuk mengekstraksi pola penting dari data secara langsung melalui proses pembelajaran yang berlapis. Karakteristik tersebut memungkinkan model menghasilkan representasi data yang lebih baik dibandingkan metode konvensional. Oleh karena itu, penggunaan ketiga model tersebut diharapkan dapat memberikan gambaran yang komprehensif mengenai pengaruh perbedaan arsitektur terhadap deteksi kekuatan sentimen.

CNN dipilih karena memiliki kemampuan yang sangat baik dalam mengekstraksi pola lokal berupa kata atau frasa yang sering muncul pada teks sentimen. *CNN* ditunjukkan mampu menghasilkan performa yang tinggi dalam tugas klasifikasi kalimat dan analisis sentimen karena efektif dalam mengenali fitur-fitur penting pada teks. Integrasi informasi leksikon ke dalam *CNN* dibuktikan mampu meningkatkan kemampuan model dalam memahami konteks sentimen. Kombinasi *CNN* dengan leksikon berbasis konteks terbukti mampu meningkatkan performa deteksi kekuatan sentimen dibandingkan beberapa metode *baseline*. Berdasarkan temuan tersebut, *CNN* dipilih karena memiliki

keseimbangan yang baik antara efisiensi komputasi dan kemampuan ekstraksi fitur.

LSTM dipilih karena memiliki kemampuan mempertahankan informasi jangka panjang melalui mekanisme memori yang dimilikinya. Karakteristik ini memungkinkan model memahami hubungan antar kata yang berjauhan dalam suatu kalimat. Pengembangan arsitektur *LSTM* terbukti mampu meningkatkan performa prediksi intensitas sentimen melalui kemampuan model dalam menangkap dependensi jangka panjang pada data teks. Kombinasi representasi kontekstual dan *LSTM* terbukti mampu meningkatkan akurasi analisis sentimen dibandingkan penggunaan *LSTM* secara tunggal. Oleh karena itu, *LSTM* dipilih untuk mengevaluasi kemampuan model dalam memahami konteks sekuensial yang memengaruhi kekuatan sentimen.

Sementara itu, *Bi-LSTM* dipilih karena mampu memanfaatkan konteks dari dua arah secara bersamaan sehingga menghasilkan representasi teks yang lebih lengkap. Kemampuan ini memungkinkan model memahami hubungan antar kata secara lebih komprehensif dibandingkan *LSTM* satu arah. *Bi-LSTM* terbukti mampu meningkatkan performa deteksi sentimen karena dapat menangkap konteks kalimat secara lebih baik. *Bi-LSTM* juga terbukti efektif dalam memahami pola linguistik yang kompleks sehingga mampu meningkatkan kualitas analisis sentimen. Oleh karena itu, *Bi-LSTM* dipilih sebagai representasi model yang memiliki kemampuan pemahaman konteks paling lengkap di antara ketiga arsitektur yang digunakan.

Selain membandingkan model *deep learning*, penelitian ini juga menggunakan dua jenis leksikon yang berbeda, yaitu *SSS-Lex* dan *VADER*. Penggunaan *SSS-Lex* didasarkan pada kemampuannya dalam merepresentasikan intensitas sentimen secara rinci sehingga sesuai dengan tujuan penelitian yang berfokus pada deteksi kekuatan sentimen. Sementara itu, *VADER* dipilih karena merupakan salah satu

leksikon yang paling banyak digunakan dalam penelitian analisis sentimen dan telah terbukti efektif dalam menangani teks informal. Kombinasi ketiga model *deep learning* dengan kedua jenis leksikon tersebut memungkinkan dilakukan evaluasi yang lebih menyeluruh terhadap pengaruh karakteristik model dan leksikon terhadap performa deteksi kekuatan sentimen. Dengan demikian, penelitian ini diharapkan dapat mengidentifikasi kombinasi model dan leksikon yang paling efektif untuk mendeteksi kekuatan sentimen pada ulasan produk *McDonald's*.

2.3 Framework

Framework merupakan suatu kerangka kerja yang digunakan sebagai pedoman dalam melaksanakan penelitian secara sistematis dan terstruktur. Dalam penelitian *data mining*, *framework* berfungsi untuk mengorganisasi setiap tahapan penelitian, mulai dari pemahaman permasalahan, pengumpulan data, pengolahan data, pemodelan, hingga evaluasi hasil. Penggunaan *framework* membantu memastikan bahwa setiap proses penelitian dilakukan secara terarah dan sesuai dengan tujuan yang ingin dicapai. Selain itu, *framework* juga memudahkan peneliti dalam mendokumentasikan setiap tahapan penelitian sehingga hasil yang diperoleh lebih sistematis dan dapat dipertanggungjawabkan. Berikut merupakan beberapa *framework data mining* yang dapat diterapkan pada penelitian deteksi kekuatan sentimen, serta *framework* yang digunakan dalam penelitian ini.

2.3.1 *Cross-Industry Standard Process for Data Mining (CRISP-DM)*

CRISP-DM merupakan salah satu kerangka kerja yang paling banyak digunakan dalam proses data mining dan analisis data karena menyediakan tahapan yang sistematis dan terstruktur. *CRISP-DM* terdiri atas enam tahapan utama, yaitu *business understanding*, *data understanding*, *data preparation*, *modeling*, *evaluation*, dan *deployment* [46]. Setiap tahapan saling berkaitan dan membantu peneliti dalam

mengelola proyek analisis data secara menyeluruh mulai dari identifikasi kebutuhan penelitian hingga implementasi hasil.

Dalam penelitian analisis sentimen, *CRISP-DM* sangat membantu dalam memastikan bahwa proses pengolahan data, pembangunan model, dan evaluasi dilakukan secara terarah sesuai tujuan penelitian. Selain itu, kerangka kerja ini memiliki fleksibilitas yang tinggi karena dapat diterapkan pada berbagai metode analisis dan *algoritma machine learning* maupun *deep learning*. Meskipun demikian, penerapan *CRISP-DM* sering dianggap cukup kompleks untuk penelitian berskala kecil karena melibatkan tahapan yang rinci dan membutuhkan dokumentasi yang baik pada setiap prosesnya. Namun, karena mampu memberikan alur kerja yang jelas dan sistematis, *CRISP-DM* tetap menjadi salah satu metodologi yang paling banyak digunakan dalam penelitian data mining dan analisis sentimen.

2.3.2 *Knowledge Discovery in Databases (KDD)*

KDD merupakan pendekatan yang digunakan untuk menemukan pengetahuan baru, pola tersembunyi, serta informasi yang bermanfaat dari kumpulan data dalam jumlah besar. *KDD* terdiri atas beberapa tahapan penting, yaitu pemahaman masalah, pemahaman data, seleksi data, transformasi data, pemodelan, evaluasi, dan penyajian hasil [47]. Pendekatan ini menekankan pentingnya pemahaman terhadap karakteristik data sebelum dilakukan proses analisis sehingga hasil yang diperoleh dapat lebih relevan dan akurat. Dalam penelitian berbasis teks, *KDD* dapat digunakan untuk mengidentifikasi pola-pola sentimen yang muncul dari kumpulan ulasan atau komentar pengguna.

Salah satu keunggulan utama *KDD* adalah kemampuannya menghasilkan pengetahuan baru yang sebelumnya tidak terlihat secara langsung dari data mentah. Selain itu, pendekatan ini memungkinkan peneliti untuk memahami hubungan antar variabel secara lebih mendalam melalui proses eksplorasi data yang sistematis. Namun, proses *KDD* relatif membutuhkan waktu yang lebih lama dibandingkan metode

lainnya karena setiap tahapan harus dilakukan secara berurutan dan memerlukan pemahaman domain yang baik agar hasil yang diperoleh benar-benar bermakna.

2.3.3 *Sample, Explore, Modify, Model, Assess (SEMMA)*

SEMMA merupakan metodologi data mining yang dikembangkan oleh *SAS Institute* untuk membantu proses analisis data secara sistematis dan terstruktur. SEMMA terdiri atas lima tahapan utama, yaitu *sample*, *explore*, *modify*, *model*, dan *assess* [48]. Tahap *sample* digunakan untuk memilih data yang representatif sehingga dapat menggambarkan keseluruhan populasi, sedangkan tahap *explore* bertujuan memahami karakteristik data serta mengidentifikasi pola awal yang terdapat dalam dataset. Selanjutnya, tahap *modify* digunakan untuk melakukan transformasi dan pembersihan data agar siap digunakan dalam proses pemodelan. Setelah model dibangun pada tahap *model*, dilakukan evaluasi pada tahap *assess* untuk mengukur performa model yang dihasilkan. Keunggulan *SEMMA* terletak pada fokusnya terhadap kualitas data sehingga dapat membantu meningkatkan akurasi model yang dibangun. Namun, pendekatan ini sering kali membutuhkan waktu yang cukup panjang pada tahap eksplorasi dan modifikasi data karena peneliti harus melakukan analisis data secara mendalam sebelum memasuki tahap pemodelan.

2.4 *Tools*

2.4.1 *Deep Learning Frameworks*

Deep Learning Frameworks merupakan perangkat lunak yang menyediakan berbagai fungsi dan komponen untuk membangun, melatih, dan mengimplementasikan model deep learning secara efisien. Framework *deep learning* berperan penting dalam mempercepat proses pengembangan model karena menyediakan berbagai algoritma, fungsi optimasi, serta dukungan komputasi yang telah terintegrasi [49].

Penggunaan *framework deep learning* memungkinkan peneliti untuk fokus pada pengembangan model tanpa harus membangun seluruh komponen jaringan saraf dari awal.

Dalam penelitian analisis sentimen, *framework deep learning* digunakan untuk membangun model *CNN, LSTM,* dan *Bi-LSTM* yang mampu mempelajari pola sentimen secara otomatis dari data teks. Selain itu, *framework* ini juga menyediakan dukungan terhadap pemrosesan data dalam jumlah besar serta pemanfaatan perangkat keras seperti *Graphic Processing Unit (GPU)* untuk mempercepat proses pelatihan model. Oleh karena itu, penggunaan *framework deep learning* menjadi komponen penting dalam penelitian yang melibatkan algoritma pembelajaran mendalam.

2.4.1.1 *TensorFlow*

TensorFlow merupakan salah satu *framework deep learning* yang paling populer dan banyak digunakan baik dalam penelitian akademik maupun industri. TensorFlow menyediakan lingkungan pengembangan yang fleksibel untuk membangun berbagai jenis model *deep learning* dengan performa yang tinggi [50]. *Framework* yang dikembangkan oleh *Google* ini mendukung berbagai arsitektur jaringan saraf seperti *CNN*, *RNN*, *LSTM, Bi-LSTM,* hingga transformer yang banyak digunakan dalam bidang *NLP*. Selain itu, *TensorFlow* memiliki kemampuan untuk menjalankan komputasi secara paralel sehingga proses pelatihan model dapat dilakukan dengan lebih cepat. Ketersediaan dokumentasi yang lengkap dan komunitas pengguna yang besar juga menjadi salah satu faktor yang mendukung popularitas *TensorFlow*. Oleh karena itu, *TensorFlow* menjadi salah satu *framework* utama yang banyak digunakan dalam penelitian analisis sentimen dan aplikasi kecerdasan buatan lainnya.

2.4.1.2 *Keras*

Keras merupakan *high-level API* yang dirancang untuk mempermudah pengembangan model deep learning dengan sintaks yang sederhana dan mudah dipahami. Keras memungkinkan pembangunan model jaringan saraf secara cepat tanpa harus memahami detail implementasi tingkat rendah yang kompleks [51]. *Framework* ini berjalan di atas *TensorFlow* sehingga dapat memanfaatkan seluruh kemampuan *TensorFlow* dengan antarmuka yang lebih sederhana. Dalam penelitian analisis sentimen, Keras sering digunakan karena menyediakan berbagai komponen siap pakai seperti *layer*, fungsi aktivasi, *optimizer*, dan mekanisme evaluasi model. Selain itu, Keras memungkinkan proses eksperimen model dilakukan dengan lebih cepat sehingga sangat cocok digunakan dalam penelitian akademik. Kemudahan penggunaan dan fleksibilitas yang dimiliki menjadikan Keras sebagai salah satu *framework* yang paling banyak digunakan dalam pengembangan model *deep learning* berbasis teks.

2.4.2 *Natural Language Processing (NLP) Libraries*

NLP Libraries merupakan kumpulan pustaka yang menyediakan berbagai fungsi untuk memproses, menganalisis, dan memahami bahasa manusia dalam bentuk teks. NLP Libraries berperan penting dalam proses pengolahan data teks karena menyediakan berbagai fitur, seperti tokenisasi, *stemming, lemmatization*, ekstraksi fitur, hingga analisis sentimen [52]. Penggunaan *NLP Libraries* memungkinkan proses pengolahan data dilakukan secara lebih efisien dan konsisten dibandingkan jika seluruh proses dilakukan secara manual. Dalam penelitian analisis sentimen, *NLP Libraries* digunakan untuk mendukung tahapan *preprocessing* sehingga data teks dapat diubah menjadi format

yang siap digunakan oleh model *deep learning*. Selain itu, *NLP Libraries* juga membantu dalam meningkatkan kualitas representasi data yang akan digunakan pada tahap pemodelan. Oleh karena itu, keberadaan *NLP Libraries* menjadi salah satu komponen penting dalam penelitian berbasis pemrosesan bahasa alami.

2.4.3 *Natural Language Toolkit (NLTK)*

NLTK merupakan salah satu *NLP Libraries* yang paling populer dalam bahasa pemrograman *Python* dan banyak digunakan dalam penelitian maupun pembelajaran *NLP*. *NLTK* menyediakan berbagai modul yang mendukung proses tokenisasi, *stemming*, *lemmatization*, analisis sintaksis, hingga analisis sentimen [53]. Pustaka ini memiliki koleksi korpus dan sumber daya linguistik yang lengkap sehingga sangat membantu dalam berbagai eksperimen pemrosesan bahasa alami. Selain itu, *NLTK* memiliki dokumentasi yang komprehensif sehingga mudah dipelajari oleh peneliti maupun mahasiswa yang baru mempelajari *NLP*. Dalam penelitian ini, *NLTK* digunakan untuk mendukung proses preprocessing data teks sebelum dilakukan pemodelan menggunakan algoritma deep learning. Dengan fitur yang lengkap dan fleksibilitas yang tinggi, *NLTK* menjadi salah satu *NLP libraries* yang paling banyak digunakan hingga saat ini.

2.4.4 *TextBlob*

TextBlob merupakan *NLP libraries* berbasis *Python* yang dirancang untuk memberikan kemudahan penggunaan dalam berbagai tugas pemrosesan bahasa alami. *TextBlob* menyediakan antarmuka yang sederhana untuk melakukan tokenisasi, klasifikasi teks, ekstraksi frasa, koreksi ejaan, serta analisis sentimen [54]. Keunggulan utama *TextBlob* terletak pada kemudahan implementasinya sehingga pengguna dapat melakukan berbagai proses *NLP* hanya dengan beberapa baris kode. Selain itu, *TextBlob* sangat cocok digunakan untuk penelitian atau pengembangan aplikasi yang memerlukan analisis teks secara cepat tanpa

konfigurasi yang rumit. Pustaka ini juga mendukung integrasi dengan berbagai alat *NLP* lainnya sehingga dapat digunakan sebagai pelengkap dalam proses pengolahan data teks. Oleh karena itu, *TextBlob* sering digunakan dalam penelitian yang membutuhkan solusi *NLP* yang praktis dan mudah diterapkan.

2.4.5 *Gensim*

Gensim merupakan *NLP libraries* yang secara khusus dirancang untuk pemodelan topik dan pembentukan representasi kata atau *word embedding*. Gensim mampu membangun model representasi kata yang dapat menangkap hubungan semantik antar kata berdasarkan konteks kemunculannya dalam korpus teks [55]. Kemampuan tersebut menjadikan *Gensim* sangat bermanfaat dalam berbagai tugas *NLP* seperti klasifikasi teks, analisis sentimen, pencarian informasi, dan pemodelan topik. Selain itu, *Gensim* dirancang untuk menangani data teks dalam jumlah besar dengan penggunaan memori yang relatif efisien. Dalam penelitian berbasis *deep learning,* representasi kata yang dihasilkan oleh *Gensim* dapat digunakan sebagai input model untuk meningkatkan kualitas pemahaman konteks teks. Oleh karena itu, *Gensim* menjadi salah satu pustaka yang banyak digunakan dalam penelitian yang melibatkan analisis teks skala besar.

2.5 Evaluasi Model

Evaluasi model merupakan tahapan penting dalam penelitian *machine learning* maupun *deep learning* karena digunakan untuk mengukur kemampuan model dalam menghasilkan prediksi yang mendekati nilai sebenarnya. Pada penelitian analisis sentimen yang berorientasi pada *sentiment strength prediction,* keluaran model berbentuk nilai kontinu sehingga evaluasi tidak lagi menggunakan metrik klasifikasi seperti *Accuracy, Precision, Recall*, ataupun *F1-Score*, melainkan menggunakan metrik regresi. Evaluasi pada prediksi intensitas sentimen umumnya

menggunakan ukuran kesalahan prediksi yang mampu menggambarkan kedekatan antara nilai hasil prediksi dengan nilai target sebenarnya. Penggunaan metrik evaluasi yang sesuai sangat penting agar performa model dapat dibandingkan secara objektif. Pada penelitian ini digunakan dua metrik evaluasi, yaitu *Mean Absolute Error (MAE)* dan *Predictive-R² (pR²).*

2.5.1 *Mean Absolute Error* (*MAE)*

Mean Absolute Error (MAE) merupakan metrik evaluasi yang digunakan untuk menghitung rata-rata selisih absolut antara nilai aktual dengan nilai hasil prediksi model. Semakin kecil nilai *MAE*, maka semakin baik performa model karena menunjukkan bahwa hasil prediksi semakin mendekati nilai sebenarnya. *MAE* menjadi salah satu metrik yang banyak digunakan dalam penelitian *sentiment strength prediction* karena memberikan interpretasi yang sederhana terhadap tingkat kesalahan prediksi. Perhitungan *MAE* pada penelitian ini dilakukan menggunakan Persamaan (2.13).

Persamaan (2.13) Perhitungan *Mean Absolute Error (MAE)*

$$MAE = \frac{1}{n}\sum_{i=1}^{n} | y_i - \hat{y_i} | \tag{2.13}$$

Keterangan:

a. n = jumlah data
b. y_i = nilai aktual
c. $\hat{y}_i$ = nilai prediksi model

Sumber: Diadaptasi dari [56]

Berdasarkan Persamaan (2.13), nilai *MAE* diperoleh dengan menghitung rata-rata selisih absolut antara nilai aktual dan nilai prediksi untuk seluruh data pengujian. Penggunaan nilai absolut bertujuan agar kesalahan prediksi positif maupun negatif tidak saling meniadakan.

Dengan demikian, *MAE* memberikan gambaran langsung mengenai besarnya rata-rata kesalahan prediksi yang dihasilkan model.

Interpretasi nilai MAE adalah sebagai berikut.

a. MAE mendekati 0 menunjukkan bahwa hasil prediksi sangat baik.
b. MAE kecil menunjukkan bahwa kesalahan prediksi rendah.
c. MAE besar menunjukkan bahwa prediksi model kurang akurat.

2.5.2 *Predictive-R²* (pR²)

Predictive-R² (pR²) digunakan untuk mengetahui kemampuan model dalam menjelaskan variasi data aktual melalui hasil prediksi yang dihasilkan. Nilai pR² berada pada rentang 0 sampai 1, di mana nilai yang semakin mendekati 1 menunjukkan bahwa model mampu menjelaskan variasi data dengan semakin baik. Penggunaan pR² dalam penelitian *sentiment strength prediction* juga diterapkan sebagai indikator kemampuan model dalam merepresentasikan hubungan antara data aktual dan hasil prediksi. Perhitungan *pR²* pada penelitian ini dilakukan menggunakan Persamaan (2.14).

Persamaan (2.14) Perhitungan *Predictive-R² (pR²)*

$$pR^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} \quad (2.14)$$

Keterangan:

a. yi = nilai aktual
b. ŷi = nilai prediksi
c. ȳ = rata-rata nilai aktual

Sumber: Diadaptasi dari [57]

Berdasarkan Persamaan (2.14), nilai *pR²* dihitung dengan membandingkan jumlah kuadrat kesalahan prediksi terhadap jumlah kuadrat variasi data aktual terhadap rata-ratanya. Nilai *pR²* yang tinggi menunjukkan bahwa model mampu menjelaskan sebagian besar variasi

data aktual, sedangkan nilai yang rendah menunjukkan bahwa kemampuan model dalam merepresentasikan pola data masih terbatas.

Interpretasi nilai pR^2 adalah sebagai berikut:

a. pR^2 mendekati 1 menunjukkan model memiliki kemampuan prediksi yang sangat baik.
b. pR^2 mendekati 0 menunjukkan kemampuan model rendah dalam menjelaskan variasi data.