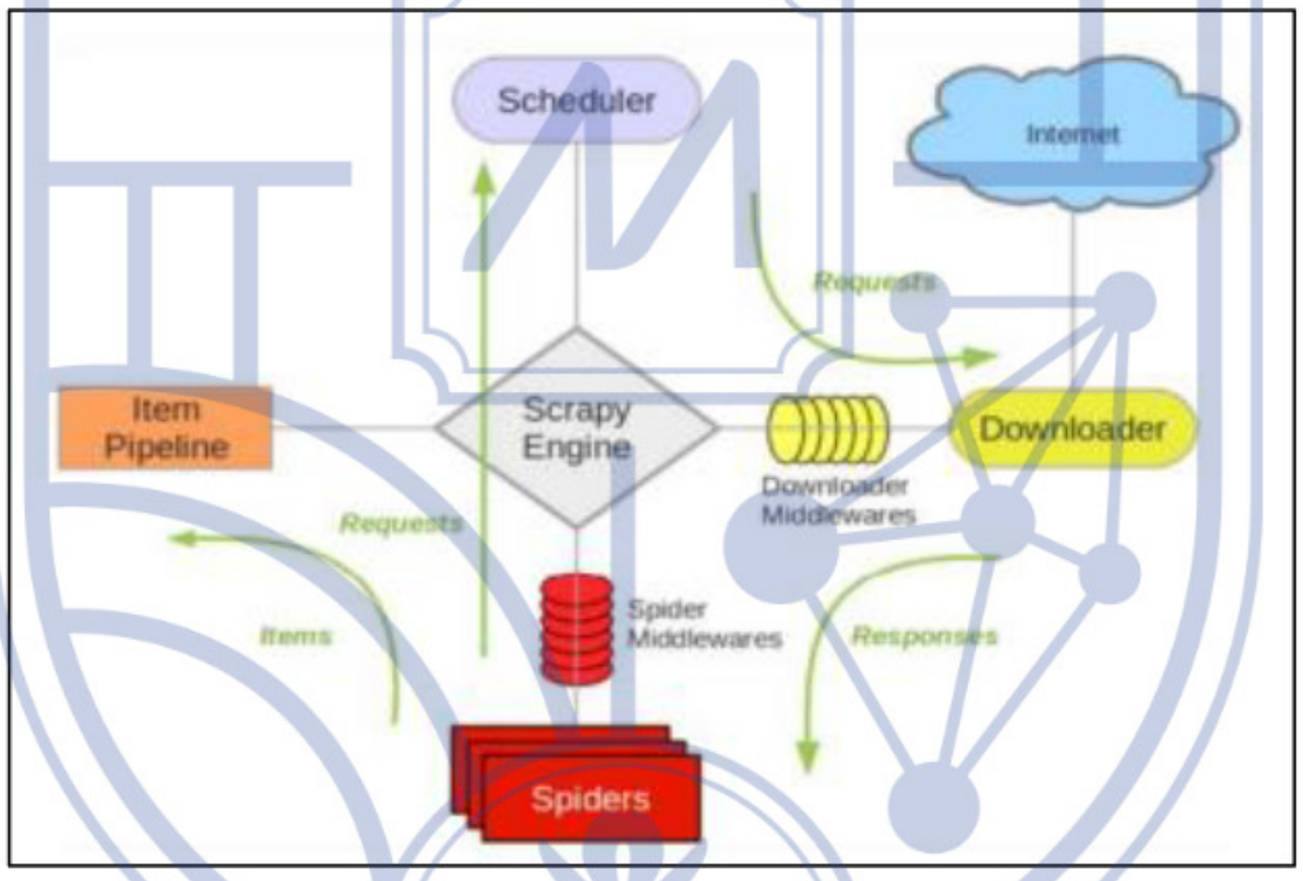
BAB II
KAJIAN LITERATUR

2.1 *Crawling* Data

Crawling Data adalah proses mengekstraksi informasi dari situs *web* melalui perangkat lunak komputer. Perangkat lunak ini dapat meniru cara manusia dalam menelusuri halaman *web* seperti *Mozilla Firefox* atau *Internet Explorer* dan memasukkan data ke dalam *database* lokal. *Crawling* merupakan teknik untuk mengubah data *web* yang tidak terstruktur menjadi data terstruktur yang dapat disimpan dan dianalisis dalam *database* atau *spreadsheet*. Hal ini memungkinkan *bot* untuk mengambil data sesuai dengan kata kunci yang diberikan dalam jumlah besar dan dalam waktu yang singkat, memungkinkan implementasi teknik *crawling* di berbagai bidang seperti pada sektor bisnis. Teknik ini dapat dimanfaatkan untuk menentukan harga dan biaya produksi dengan menilai dari data yang telah di-*crawling* [14, 15].



Gambar 2.1 *Framework Scrapy* Berbasis Python [16]

Teknik *crawling* dimaksudkan untuk menggantikan metode pengumpulan data lama seiring dengan banyaknya bisnis yang berupaya mengejar tren baru yang diikuti oleh konsumen. Sejumlah bahasa pemrograman menawarkan berbagai kemungkinan untuk melakukan *crawling* data dari internet, dan di dalam bahasa-bahasa tersebut terdapat banyak fungsionalitas tambahan yang dapat disesuaikan terkait *crawling*. Bahasa yang paling populer untuk *data crawling* yaitu Python. Python yang dikembangkan oleh lisensi *Open Source Initiative* (OSI) merupakan bahasa pemrograman serbaguna yang digunakan dalam pengembangan *web* serta analisis data dan menawarkan *library* yang berguna dalam proses *crawling* seperti *Requests*, *BeautifulSoup*, atau *Scrapy*. Namun, terdapat juga banyak alat

lain seperti *R*, *Java*, atau *Perl* yang berfungsi dengan baik untuk kebutuhan *data crawling* [16, 17].

2.2 *Data Preprocessing*

Preprocessing data menjadi tahap awal dalam klasifikasi teks untuk mempersiapkan, membersihkan dan menormalisasi data teks sebelum digunakan pada tahapan penelitian selanjutnya. Beberapa tahapan dalam *data preprocessing* meliputi [18, 19]:

1. *Data Cleaning* yaitu proses menghilangkan karakter spesial yang bukan teks seperti tanda baca, angka, simbol, *link URL*, dan *username* untuk mengurangi gangguan (*noise*) saat proses klasifikasi. Sebagai contoh, kata "@TokoHp?." akan diubah menjadi "TokoHp", sehingga memudahkan proses analisis selanjutnya.
2. *Case Folding* yaitu proses mengubah teks menjadi huruf kecil sehingga semua teks ulasan yang diproses berada dalam bentuk yang sama. Misalnya, kata "TokoHP" akan diubah menjadi "tokohp".
3. *Tokenizing* yaitu proses membagi serangkaian karakter dalam teks menjadi kata-kata individu atau *token*. Sebagai contoh, teks "hpnya lumayan mahal ya" akan ditokenisasi dalam bentuk *array* menjadi ["hpnya", "lumayan", "mahal", "ya"].
4. *Normalization* yaitu proses mengubah kata-kata tidak baku, informal, gaul dan *slang* menjadi kata yang baku dan formal. Misalnya, kata "gk" akan diubah menjadi "tidak".
5. *Stopword Removal* yaitu proses menghapus kata-kata umum yang tidak memiliki makna dalam teks. Kata-kata seperti "dan", "atau", "tetapi" akan dihapus agar model dapat fokus pada inti dari teks ulasan.
6. *Lemmatization* yaitu proses mereduksi kata menjadi bentuk dasar, tujuan dari lemmatisasi adalah mengubah kata menjadi bentuk kanonik atau kamusnya yang membantu dalam mengurangi dimensi data. Kata dasar seperti kata "membaca" dan "dibaca" keduanya akan diubah menjadi "baca" untuk mengurangi kata-kata unik pada *dataset* yang memiliki arti yang sama.

2.3 Pelabelan Data

Pelabelan data (*data labeling*) merupakan proses yang melibatkan pemberian label, kategori, atau kode tertentu pada setiap entitas atau elemen data untuk mengidentifikasi atau mengklasifikasikan data berdasarkan karakteristik atau atribut tertentu. Proses ini sangat krusial untuk berbagai tugas seperti klasifikasi data dan *entity recognition*. Pelabelan data

merupakan landasan bagi *supervised machine learning*, di mana model belajar untuk memetakan input ke *output* yang diinginkan berdasarkan contoh-contoh data yang telah diberikan label. Proses ini melibatkan penetapan label atau informasi tambahan pada data mentah (teks, gambar, audio, video) untuk membuat kumpulan data pelatihan [20]. Terdapat tiga pendekatan utama dalam pelabelan data [21, 22]:

1. Pelabelan manual mengandalkan anotator untuk menetapkan label ke poin data secara teliti. Hal ini memerlukan pemahaman mendalam tentang konteks dan tujuan pelabelan, di samping presisi dan konsistensi. Meskipun pelabelan manual menjamin kualitas tinggi dan label yang akurat yang sangat krusial untuk pelatihan model, proses ini bisa memakan waktu lama dan memakan banyak sumber daya terutama untuk *dataset* besar.
2. Pelabelan otomatis memanfaatkan algoritma atau model yang telah dilatih sebelumnya untuk mengotomatiskan proses pelabelan, sehingga secara signifikan mengurangi waktu dan sumber daya khususnya pada *dataset* yang sangat luas. Teknik yang populer meliputi model *pre-trained*, algoritma *clustering*, dan *rule-based model*. Meskipun pelabelan otomatis tidak mencapai akurasi yang sama dengan pelabelan manual, metode ini berfungsi sebagai alat yang berharga untuk membuat *dataset* awal atau meminimalkan proses manual dengan mengurangi beban kerja awal.
3. Pelabelan semi-otomatis menggabungkan pendekatan manual dan otomatis secara strategis untuk memanfaatkan kekuatan masing-masing metode. Algoritma otomatis akan memberikan label awal yang kemudian ditinjau dan disempurnakan secara teliti oleh anotator untuk memastikan akurasi pelabelan. Pendekatan ini mendorong efisiensi dan kecepatan dalam proses pelabelan dengan tetap menjaga kualitas data yang dilabeli. Metode ini sering digunakan dalam proyek besar di mana pelabelan manual menjadi tidak memungkinkan karena volume data, namun akurasi tinggi tetap menjadi hal yang utama.

Kualitas pelabelan data berdampak langsung pada performa model. Label yang tidak akurat atau tidak konsisten dapat menyebabkan model mempelajari pola yang salah dari data yang tersedia. Akibatnya, model yang dihasilkan menjadi kurang optimal, menghasilkan prediksi yang keliru, serta berpotensi menimbulkan bias. Kesalahan dalam pelabelan juga dapat menurunkan kemampuan model dalam melakukan generalisasi terhadap data baru, sehingga performa model pada tahap pengujian maupun penerapan di dunia nyata menjadi

kurang memadai. Oleh karena itu, sangat penting untuk melakukan proses validasi atau pengecekan ulang untuk mengevaluasi dan mengontrol kualitas pelabelan [21].

2.4 *Data Mining*

Data Mining merupakan proses menemukan pola dan *insight* dari kumpulan *dataset* menggunakan algoritma *machine learning*, statistik dan *artificial intelligent* (AI). *Data mining* digunakan untuk mengekstrak informasi atau pengetahuan bermanfaat yang sebelumnya tersembunyi dari berbagai sumber data tidak terstruktur dan kompleks untuk melakukan proses analisis dan prediksi [23]. Meskipun *data mining* dapat membantu mengungkap pola dan hubungan dalam *dataset*, *data mining* tidak dapat memberi tahu pengguna nilai atau signifikansi dari pola tersebut, sehingga teknik ini memerlukan spesialis teknis dan analitis yang terampil yang dapat menyusun analisis serta menginterpretasikan hasil agar lebih mudah dipahami [24].

Teknik-teknik *data mining* dapat diterapkan dalam berbagai sektor seperti *marketing*, *fraud detection*, pengembangan produk, layanan kesehatan, dan pendidikan. Selain itu, teknik *data mining* digunakan oleh organisasi untuk menyelesaikan masalah bisnis seperti meningkatkan pendapatan, mengakuisisi pelanggan baru, meningkatkan strategi *cross-selling* dan *up-selling*, serta meningkatkan ROI (*Return On Investment*) organisasi [25]. *Data mining* dapat dikategorikan menjadi dua area utama yaitu model prediktif dan deskriptif. Tugas model prediktif meliputi [26]:

1. Klasifikasi

   Klasifikasi merupakan teknik dalam *data mining* yang memetakan data ke dalam kelompok ataupun kelas yang sudah ditentukan. Teknik ini termasuk ke dalam *supervised learning* karena kelas-kelas dari data sudah ditentukan sebelumnya. *Pattern recognition* merupakan salah satu jenis klasifikasi di mana pola yang diinput akan diklasifikasikan ke salah satu dari beberapa kelas berdasarkan kemiripannya dengan kelas yang ditentukan.

2. *Time Series Analysis*

   Time series analysis merupakan proses analisis data kronologis yang diperoleh pada titik waktu dengan interval yang sama, misalnya harian, mingguan atau bulanan. Data tersebut dianggap sebagai satu kesatuan dan berhubungan satu sama lain. Analisis ini bertujuan untuk mengidentifikasi sifat dari fenomena yang direpresentasikan oleh urutan observasi dan memprediksi kemungkinan perubahan atau perkembangan seiring waktu.

Tugas model deskriptif pada *data mining* meliputi [27]:

1. *Clustering*

   Clustering adalah metode *unsupervised* dalam *machine learning* yang disebut juga dengan segmentasi data. Teknik clustering serupa dengan klasifikasi dalam hal pengelompokan data ke dalam kelas-kelas, namun pada *clustering* kelas-kelas tersebut tidak ditentukan sebelumnya.

2. *Association Rules*

   Association rules digunakan untuk mengidentifikasi hubungan antar *item* dan pola *if/then* yang sering muncul pada data. *Support* dan *confidence* merupakan kriteria pengukuran *association rules* yang menunjukkan hubungan antar *item* dianggap menarik jika nilai hubungan tersebut memenuhi batas *support* minimum dan batas *confidence* minimum yang ditentukan.

2.5 *Natural Language Processing*

Natural Language Processing (NLP) merupakan subbidang *Artificial Intelligence* (AI) yang berfokus pada interaksi antara manusia dan komputer dengan menggunakan bahasa alami. Tujuan utama dari NLP adalah untuk mengembangkan algoritma yang mampu memahami, menganalisis, dan menghasilkan bahasa manusia secara otomatis sehingga memungkinkan manusia berkomunikasi dengan komputer [28]. Terdapat dua komponen dalam NLP yaitu *Natural Language Understanding* (NLU) yang bertugas dalam memetakan input yang diberikan dalam bahasa alami ke dalam representasi tertentu, serta menganalisis berbagai aspek bahasa tersebut dan *Natural Language Generation* (NLG) yang bertugas dalam menghasilkan frasa dan kalimat bermakna dalam bentuk bahasa alami dari suatu representasi internal. NLP digunakan pada beberapa bidang seperti layanan kesehatan, keuangan dan hukum, di mana data yang dimiliki sangat kaya dan berharga. Beberapa pengaplikasiannya meliputi [28, 29]:

1. *Machine Translation*

   Machine Translation adalah tugas menerjemahkan teks dari satu bahasa ke bahasa lain menggunakan algoritma komputer. Ini adalah tugas yang menantang dalam NLP karena membutuhkan pemahaman makna semantik teks dalam kedua bahasa tersebut. Berbagai algoritma *machine learning* telah diusulkan, termasuk *Statistical Machine Translation* dan *Neural Machine Translation*. Model berbasis *Transformer*, seperti BERT dan GPT-2, menunjukkan peningkatan hasil yang signifikan dalam tugas-tugas penerjemahan mesin.

2. *Text Classification*

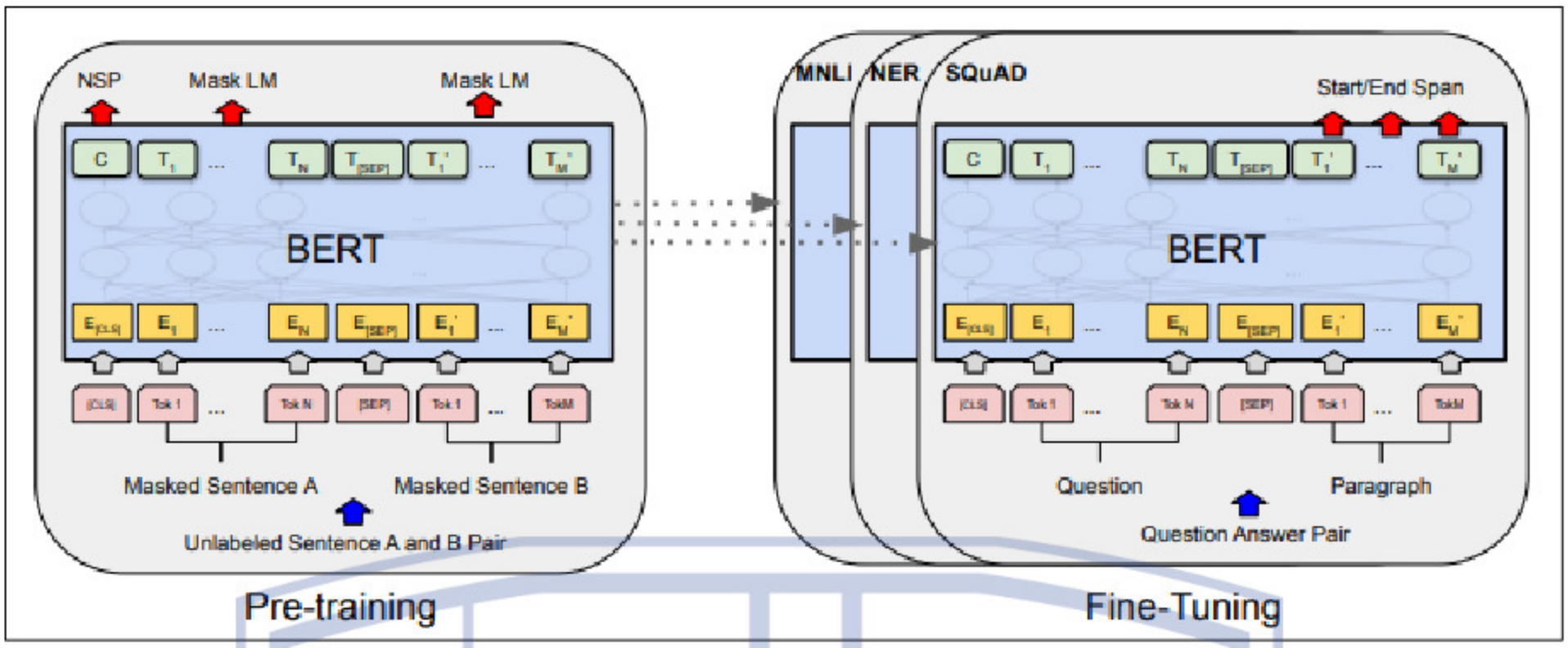
   Text Classification adalah tugas dasar dalam NLP yang melibatkan pengkategorian potongan teks ke dalam kategori yang telah ditentukan sebelumnya berdasarkan isinya. Beberapa pengaplikasian *text classification* termasuk analisis sentimen, deteksi spam, dan klasifikasi berita. Berbagai algoritma *machine learning* telah diusulkan untuk menyelesaikan tugas ini, termasuk Naïve Bayes dan SVM. Pendekatan berbasis *deep learning*, seperti *Convolutional Neural Networks* (CNN) dan *Recurrent Neural Networks* (RNN), menunjukkan peningkatan signifikan dalam tugas klasifikasi teks.

3. *Word Embedding*

   Word embedding adalah teknik populer yang digunakan dalam NLP untuk representasi teks. *Word embedding* merepresentasikan setiap kata sebagai sebuah vektor dalam ruang berdimensi tinggi, di mana posisi vektor tersebut menangkap makna semantik dari kata tersebut. Salah satu penelitian yang paling berpengaruh di bidang ini adalah algoritma Word2Vec yang merupakan algoritma berbasis *neural network* sederhana yang mempelajari *word embedding* dari korpus data teks yang besar. Sejak diperkenalkan, Word2Vec telah digunakan dalam berbagai tugas NLP, seperti analisis sentimen, *question answering*, dan penerjemahan mesin.

2.6 BERT

Bidirectional Encoder Representation from Transformer (BERT) adalah representasi *encoder* dari model *Transformer*, sebuah arsitektur NLP yang menggunakan mekanisme *self-attention* untuk menggantikan jaringan berulang dan mampu menangkap hubungan antara kata-kata yang jauh secara kontekstual [30]. BERT memanfaatkan pendekatan *bidirectional* di mana model menganalisis teks dari kedua arah. Dengan memperhitungkan kata-kata sebelum dan sesudahnya, BERT dapat digunakan untuk menganalisis sentimen dengan tingkat akurasi yang tinggi [31]. Diperkenalkan oleh grup peneliti dari *Google AI Language Laboratory* yang dipimpin oleh J. Devlin pada tahun 2018, BERT adalah model representasi berbasis *fine-tuning* pertama yang mencapai kinerja yang baik pada rangkaian besar tugas *sentence-level* dan *token-level*, dan mengungguli banyak arsitektur *task-specific.* BERT memiliki 12 *transformer encoder layers* dengan 768 *hidden units,* 12 *attention heads* dan 110 juta parameter [32].



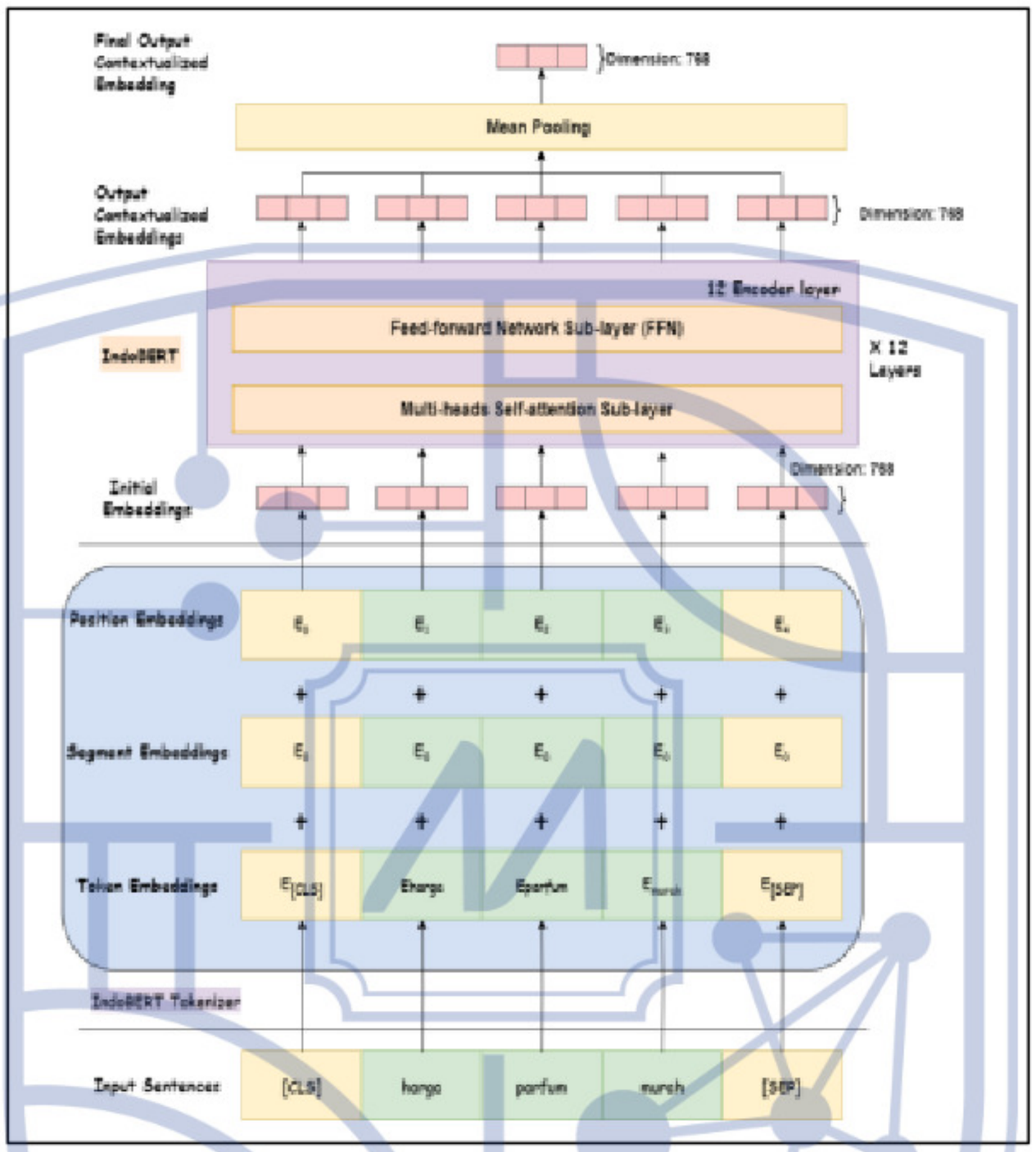
Gambar 2.2 Arsitektur BERT [32]

BERT menggunakan dua *framework* yaitu *framework pretraining* dan *framework fine-tuning*. *Framework pretraining* memiliki dua tugas yaitu *Masked Language Model* (MLM) dan *Next Sentence Prediction* (NSP) yang bertujuan untuk mempelajari representasi bahasa universal pada korpus data tak berlabel berskala besar. MLM melakukan *masking* pada sebagian *token* input secara acak kemudian melakukan prediksi pada token-token yang di *masking* sebelumnya. NSP yang melakukan tugas prediksi apakah kalimat bersifat berurutan sehingga hal ini dapat meningkatkan kemampuan model dalam memodelkan hubungan antar-kalimat [33]. Pada *framework fine-tuning* model BERT, terlebih dahulu dijalankan menggunakan *pre-trained parameters* yang sudah memiliki data berlabel dari *downstream tasks*. Setiap *downstream* memiliki model *fined-tuned* yang berbeda-beda meskipun dijalankan dengan *pre-trained parameters* yang sama [32].

2.6.1 IndoBERT

IndoBERT merupakan model bahasa alami berbasis *transformer* yang dikembangkan dari BERT, dilatih menggunakan korpus Indo4B yang terdiri atas sekitar 4 miliar kata dan 250 juta kalimat berbahasa Indonesia, yang bersumber dari berbagai media seperti Tempo, Kompas, Wikipedia, dan Twitter. Dengan demikian, IndoBERT lebih efektif dalam menangani teks berbahasa Indonesia dibandingkan dengan BERT yang dilatih menggunakan teks berbahasa Inggris. Model ini dirancang untuk memahami konteks kata dalam suatu kalimat berbahasa Indonesia secara mendalam, sehingga dapat digunakan dalam berbagai tugas NLP seperti analisis sentimen, klasifikasi teks, pemahaman bahasa, dan ekstraksi informasi [34]. Arsitektur *transformer* memungkinkan pemrosesan konteks secara dua arah (*bidirectional*) dan keberagaman sumber data, ini memungkinkan IndoBERT menangkap nuansa bahasa dan konteks yang lebih kaya serta hubungan semantik antar kata

dalam suatu kalimat berbahasa Indonesia dengan baik [35]. Beberapa varian model IndoBERT antara lain IndoBERT-LiteBase, IndoBERT-Base, dan IndoBERT-Large yang masing-masing memiliki jumlah parameter, *layer*, dan ukuran *hidden layer* yang berbeda-beda.



Gambar 2.3 Alur Kerja IndoBERT [39]

Model IndoBERT mengharuskan setiap teks input untuk ditokenisasi menjadi unit-unit yang lebih kecil menggunakan tokenizer spesifik. Cara kerja model IndoBERT pada Gambar 2.2 dapat dijabarkan sebagai berikut [36, 37, 38, 39, 40, 41, 42]:

1. Teks pada awalnya ditokenisasi menggunakan *tokenizer WordPiece* dengan melakukan proses pemecahan per kata menjadi *token* yang kemudian pada *token embedding* ditambahkan *classification token* [CLS] yang diletakkan di awal setiap urutan input dan digunakan sebagai representasi dari keseluruhan kalimat atau urutan untuk tugas klasifikasi diikuti oleh *separator token* [SEP], yang berfungsi memisahkan dua kalimat dalam tugas di mana IndoBERT memproses pasangan kalimat, sekaligus menandai batas di antara keduanya. Setelahnya, setiap *token* dipetakan menjadi id unik sesuai dengan korpus dari *WordPiece*, jika *token* tidak ada di dalam korpus maka *token* akan diberikan label [UNK].

2. Setiap id unik akan diubah menjadi representasi vektor, lalu pada *segment embedding* ditambahkan representasi vektor berupa indeks 0 dan indeks 1. Jika seluruh input hanya terdiri dari satu kalimat, maka segmen yang disematkan adalah indeks nol. Proses ini membedakan kata pada kalimat segmen 0 dan kata pada kalimat segmen 1 untuk mengetahui informasi kontekstual mengenai batas kalimat.
3. Pada *positional embedding* diterapkan pada *lookup table* berdimensi (n, 768), di mana n mewakili panjang kalimat dan 768 merupakan dimensi vektor yang sama dengan dimensi *token embedding*. *Token* pertama akan diberikan *index* 0, *token* kedua akan diberikan *index* 1 dan seterusnya.
4. Setelahnya hasil vektor *embedding* kemudian masuk ke dalam *multi-head self-attention* untuk menghitung *attention score* yang menentukan seberapa besar tingkat keterkaitan suatu kata terhadap konteks kata lainnya.
5. *Output* dari *multi-head self-attention* tersebut akan diteruskan ke dua lapisan *feed-forward network* (FFN) yang memiliki parameter yang berbeda. Setiap lapisan *feed-forward* merupakan fungsi berbasis *position-wise* yang memproses vektor secara independen. Fungsi aktivasi non-linear GELU (*Gaussian Error Linear Units*) digunakan dalam *feed-forward* untuk mempelajari representasi dan pola data yang kompleks serta menambahkan sifat non-linear dari input ke *output*. *Feed-forward* memiliki neuron yang hanya terhubung dengan neuron berikutnya, mengalir dalam satu arah dari input sampai *output* dalam bentuk *hidden states* dan tidak memiliki hubungan ke belakang.
6. Selanjutnya pada *hidden states* dilakukan teknik *mean pooling* dengan mengambil rata-rata dari *hidden states* milik seluruh n *token embeddings* dalam sebuah *layer* untuk mendapatkan *sentence embedding* tunggal dan mengecualikan *token* [PAD] menggunakan *attention masks*.

2.6.2 *Attention Mechanism*

Attention Mechanism, yang diperkenalkan pada tahun 2014 oleh [43] adalah pendekatan *deep learning* antara *encoder* dan *decoder* yang menyediakan *decoder* dengan informasi dari setiap *encoder hidden state*. Dengan mekanisme ini, model dapat secara selektif fokus pada bagian-bagian berguna dari urutan input dan mempelajari penyelarasan antara urutan input dan juga *output*. *Attention* bisa membuat satu representasi vektor dari setiap *time-step encoder* sehingga akan membantu model untuk mengatasi kalimat input yang panjang dan memberikan akurasi analisis yang lebih baik [44]. Penggunaan *attention*

semakin populer setelah diperkenalkannya *Transfromer* pada tahun 2017 oleh Vaswani et al., *attention* yang pada awalnya digunakan hanya pada RNN memperlihatkan perkembangan yang sangat signifikan saat diimplementasikan pada model *transformer* dan mencapai *state of the art* pada *machine translation*. Dengan adanya *attention* model *transformer* yang awalnya hanya dikembangkan untuk *machine translation*, diadopsi untuk melakukan tugas lain seperti *image processing*, *video processing* dan *recommender systems* [45, 46].

Pada *paper* "*Attention Is All You Need*" bersamaan dengan *transformer*, Vaswani et al. juga memperkenalkan *Scaled Dot-Product Attention* dan *Multi-Head Attention. Scaled Dot-product Attention* diperkenalkan sebagai formula untuk mencari nilai *attention score*. *Self-attention*, yang terkadang disebut sebagai *intra-attention*, merupakan mekanisme yang menghubungkan posisi-posisi berbeda dari satu urutan kalimat tunggal untuk menghitung representasi dari urutan tersebut secara bersamaan (paralel) [46]. *Multi-head attention* meningkatkan daya representasi dari lapisan *attention* dengan memungkinkan beberapa *self-attention* beroperasi pada berbagai proyeksi berdimensi rendah dari input secara berbeda. Setiap *attention head* bekerja secara paralel dan *output* dari setiap *attention head* ini kemudian digabungkan (*concatenated*) untuk menghasilkan hasil akhir [47]. *Scaled Dot-Product Attention* dan *Multi-Head Self-Attention* dapat dirumuskan sebagai berikut [48]:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad \text{.....(2.1)}$$

$$MultiHead(Q, K, V) = Concat(head_1, \dots, head_h)W^O \quad \text{.....(2.2)}$$

$$head_1 = Attention(QW_i^Q, KW_i^K, VW_i^V)\ ) \quad \text{.....(2.3)}$$

Di mana: *Dot product* dihitung untuk mengukur kesamaan skor antara Q $(Query)$ dan keseluruhan K (Key) yang di *transpose* dan membaginya dengan $\sqrt{d_k}$ yang berfungsi sebagai *scaler*. Fungsi *softmax* juga diaplikasikan untuk menormalisasi distribusi bobot *attention* dan mendapatkan bobot pada V $(Values)$. Proses tersebut dilakukan secara linier sebanyak h, yang kemudian nilai-nilai tersebut digabungkan (*concatenated)* untuk menjadi nilai akhir.

2.6.3 *Fine-Tuning* IndoBERT

Fine-tuning merujuk pada teknik *transfer learning* sebagai proses memperbarui seluruh parameter model yang menggunakan pengetahuan dari *pre-trained neural networks* untuk menyelesaikan tugas-tugas baru yang relevan. Di bidang NLP, *fine-tuning* sering

digunakan pada *Large Language Models* (LLM) seperti BERT, IndoBERT dan GPT untuk tugas-tugas spesifik seperti klasifikasi teks, analisis sentimen dan *entity recognition*. Dengan memanfaatkan pengetahuan dari model *pre-trained*, *fine-tuning* dapat meningkatkan kemampuan generalisasi model secara signifikan pada tugas-tugas baru [49]. *Fine-tuning* IndoBERT melibatkan penggantian *layer output* dari IndoBERT dengan *layer* baru yang diinisialisasi secara acak untuk menyelesaikan tugas spesifik menggunakan data berlabel. *Layer* yang telah dimodifikasi tersebut kemudian dioptimalkan dengan menggunakan *AdamW Optimizer* agar performa model sesuai dengan yang diharapkan [50]. Beberapa komponen dalam proses *fine-tuning* yang digunakan dalam upaya mengoptimalkan performa model yaitu:

1. *AdamW Optimizer*

   AdamW merupakan suatu *optimizer* varian dari *Adam* (*Adaptive Moment Estimation*), yang merupakan salah satu metode optimisasi yang sangat populer dalam *deep learning*. *Optimizer* ini berperan penting dalam menyesuaikan parameter seperti bobot dan bias yang bertujuan untuk mengurangi fungsi *loss* dengan terus menerus memperbarui nilai bobot [51]. *AdamW* memodifikasi L_2 *regularization* yang ada pada *optimizer Adam* dengan memisahkan *weight decay* dari proses *update* parameter sehingga menghasilkan generalisasi yang lebih baik [52].

2. *Dropout*

   Dropout adalah salah satu metode regularisasi yang populer untuk mencegah model *deep learning* dari *overfitting* selama fase pelatihan. *Dropout* berfungsi untuk menonaktifkan *neuron* dalam *neural network* secara acak sehingga mencegah model untuk menghafal hubungan antar *neuron* dalam data *training*. Subnet baru yang dihasilkan *dropout* kemudian digunakan dalam pelatihan untuk setiap iterasi yang berbeda yang dapat meningkatkan kemampuan generalisasi model [53].

3. *Learning Rate*

   Learning rate adalah salah satu *hyperparameter* terpenting yang memengaruhi performa pelatihan *Deep Neural Network* (DNN). Parameter ini menentukan kecepatan dalam mencapai titik minimum (*minima*) yang berdampak pada biaya komputasi optimasi misalnya, *learning rate* yang lebih tinggi mungkin mencapai area sekitar *minima* lebih cepat daripada *learning rate* yang lebih rendah. *Learning rate* juga menentukan jenis *minima* yang dicapai, *learning rate* yang rendah akan cenderung menghasilkan model dengan *minima* yang sempit dan sebaliknya.

Learning rate memiliki dampak yang signifikan pada tingkat sensitivitas, akurasi dan generalisasi pada model [54].

4. *Warmup*

   Warm-up merupakan parameter yang memiliki peran penting dalam meningkatkan generalisasi dan stabilitas model. *Warmup* berfungsi untuk meningkatkan *learning rate* dengan jumlah yang konstan secara bertahap pada setiap iterasi hingga mencapai nilai *learning rate* yang diinginkan. Proses ini bertujuan untuk menghindari lompatan *learning rate* secara cepat yang memungkinkan untuk optimasi yang lebih stabil dan mencegah model menyimpang (*diverge*) pada tahap awal pelatihan. Setelah proses *warmup* selesai ditambahkan juga parameter *learning rate scheduler* untuk menurunkan nilai *learning rate* secara perlahan hingga *epoch* terakhir untuk mengurangi lompatan *overfitting* di *epoch* akhir [55].

5. *Batch Size*

   Batch size adalah *hyperparameter* yang menentukan jumlah sampel data yang diproses dalam satu iterasi pelatihan. Data akan dibagi menjadi beberapa *batch*, dan pada setiap *batch* model akan melakukan prediksi terhadap sampel . Pada akhir setiap *batch*, hasil prediksi tersebut akan dibandingkan dengan variabel *output* yang diharapkan dan nilai *error* dihitung. Dari nilai *error* ini, parameter model akan diperbarui menggunakan algoritma optimisasi seperti *AdamW* untuk memperbaiki bobot model, misalnya dengan bergerak turun mengikuti *error gradien*. Proses ini dilakukan pada setiap *batch* hingga seluruh data dalam satu *epoch* selesai diproses. [56].

6. *Epoch*

   Epoch adalah *hyperparameter* yang menentukan berapa kali algoritma akan bekerja memproses seluruh *dataset* pelatihan secara total. Satu *epoch* berarti setiap sampel dalam *dataset* pelatihan telah mendapatkan satu kali kesempatan untuk memperbarui parameter internal model [57]. *Epoch* dapat dibayangkan sebagai sebuah perulangan untuk jumlah *epoch*, di mana setiap perulangan akan menyisir seluruh *dataset* pelatihan. Di dalam setiap perulangan terdapat perulangan bersarang (*nested for-loop*) yang melakukan iterasi pada setiap sampel *batch*, di mana satu *batch* memiliki jumlah sampel sesuai dengan ukuran *batch* yang telah ditentukan [56].

Dalam proses pelatihan, model akan menghasilkan *output* berupa *logits score* yaitu nilai mentah dari *layer output* yang belum dikonversi menjadi prediksi probabilitas kelas

positif dan negatif. Untuk menginterpretasikan *logits* sebagai probabilitas digunakan fungsi *softmax* yang dapat dirumuskan sebagai berikut [58]:

$$sm(y)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad \text{.....(2.4)}$$

Di mana: Fungsi eksponensial diterapkan pada setiap elemen z_i dari vektor input z, dan nilai-nilai yang dihasilkan kemudian dinormalisasi dengan cara membaginya dengan jumlah total dari semua nilai eksponensial tersebut. Proses normalisasi ini menjamin bahwa elemen-elemen dari vektor *output* $sm(y)$ jika dijumlahkan hasilnya akan tepat sama dengan 1.

Setelah prediksi dihasilkan dari fungsi *softmax* (sm), dilakukan pengukuran *gap* antara prediksi model dengan label sebenarnya y_i menggunakan fungsi *cross-entropy loss*. Fungsi ini membantu model dalam mengukur *error* selama proses pelatihan (*training & validation loss*). *Cross-entropy loss* dapat dirumuskan sebagai berikut [59]:

$$CED(y_i : sm) = -\sum_{j=1}^{K} y_i log(sm) \quad \text{.....(2.5)}$$

Di mana: Fungsi logaritma diterapkan pada nilai *softmax* untuk memperbesar *penalty* jika model melakukan kesalahan prediksi. Dikarenakan hasil dari logaritma $0 - 1$ akan selalu negatif maka dilakukan perkalian dengan tanda $(-)$ untuk menetapkan konsistensi hasil *loss* agar selalu positif.

Untuk mengoptimalkan performa model, digunakan metode *backpropagation*, yaitu teknik propagasi yang mengirimkan nilai *training loss* dari *output* ke *layer* awal untuk menghitung kontribusi kesalahan dari setiap parameter dan menyesuaikan bobot. Proses penyesuaian bobot model dilakukan dengan menghitung *gradien* dari fungsi *loss* terhadap *logit* yang dapat dirumuskan sebagai berikut [60]:

$$\frac{\partial(CED)}{\partial z_j} = (sm_{ij} - y_{ij}) \quad \text{.....(2.6)}$$

Di mana: Nilai *gradien loss* terhadap *logit* $(\frac{\partial(CED)}{\partial z_j})$ didapat dari pembagian turunan parsial antara *cross-entropy loss* dengan *logit score*. Nilai tersebut harus sesuai dengan nilai *gradien error* yang didapat melalui pengurangan antara nilai *sm* dan y. Jika terdapat nilai yang berbeda maka gradien harus menyesuaikan ulang bobot dan parameter model.

Hasil penghitungan *gradien* kemudian digunakan dalam proses optimasi parameter menggunakan algoritma *AdamW*. Untuk memperbarui bias momentum pertama (*moving average*) dan bias momentum kedua (*varians*) dari *gradien*, dilakukan perhitungan rata-rata dan rata-rata kuadrat dari *gradien* untuk mengukur besar *update* parameter yang dibutuhkan. Momentun dan *varians gradien* dapat dirumuskan sebagai berikut:

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \quad (2.7)$$

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \quad (2.8)$$

Selanjutnya, dikarenakan nilai awal dari momentum ($\hat{m}$) dan *varians* ($\hat{v}$) diinisiasi dari 0 hasil iterasi awal akan lebih bias ke arah 0. Untuk mendapatkan nilai yang lebih akurat dapat digunakan rumus *bias correction* yang dapat dirumuskan sebagai berikut:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \quad (2.9)$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \quad (2.10)$$

AdamW Optimizer memisahkan mekanisme *weight decay* dari pembaruan berbasis momentum yang diregularisasi untuk menekan risiko *overfitting* dan dapat dirumuskan sebagai berikut [61]:

$$\theta_t = \theta_{t-1} - \alpha\left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon} + \lambda\theta_{t-1}\right) \quad (2.11)$$

Di mana: θ merepresentasikan parameter model yaitu bobot (*weight*) dan bias. Setelah fungsi $\theta_t\,(theta)$ dihitung, nilai theta yang baru akan menggantikan nilai θ_{t-1} dan model kemudian mengambil data berikutnya dengan bobot dan bias yang sudah diperbarui.

2.7 Analisis Sentimen

Analisis sentimen adalah sebuah metode yang digunakan untuk mengekstrak data opini, memahami serta mengolah data tekstual secara otomatis untuk melihat sentimen yang terkandung dalam sebuah opini. Analisis sentimen bertujuan untuk menghubungkan ribuan sumber *online* di internet dan menentukan sikap atau perasaan penulis teks terhadap suatu subjek, yang bisa berupa produk, layanan, individu, organisasi, atau topik tertentu. Hal tersebut menjadi wawasan baru tentang bagaimana cara mengidentifikasi perubahan sentimen dari waktu ke waktu terhadap keberhasilan pemasaran dan bagaimana penyedia produk dapat meningkatkan reputasi produknya [62, 63]. Analisis sentimen digunakan dalam berbagai aplikasi, termasuk analisis *feedback* pelanggan, pemantauan media sosial, riset pasar, dan *opinion mining*. Ini memungkinkan perusahaan dan peneliti untuk memahami perasaan dan pendapat pelanggan dalam skala besar, membantu dalam pengambilan keputusan yang lebih baik berdasarkan data tekstual yang tidak terstruktur [63].

Metode analisis sentimen telah mengalami transformasi yang signifikan, dimulai dari teknik awal yang bersifat konvensional hingga penggunaan kecerdasan buatan yang kompleks. Secara garis besar, terdapat tiga kategori utama dalam pendekatan ini:

1. *Lexicon-based*

   Suatu pendekatan dalam analisis sentimen yang menggunakan kamus atau leksikon berisi daftar kata-kata beserta nilai sentimennya. Metode ini mengidentifikasi dan menilai sentimen dalam teks dengan mencocokkan kata-kata yang ditemukan dalam teks tersebut dengan kamus sentimen [64]. Namun, metode ini sangat bergantung pada kualitas kamus leksikon yang digunakan apabila suatu kata tidak ditemukan di dalam kamus tersebut, maka kata tersebut akan diabaikan. Sehingga penggunaan leksikon dalam mengidentifikasi bahasa informal seperti *slang*, singkatan dan akronim internet yang sangat banyak digunakan di media sosial tidak terakomodasi [65].

2. *Machine Learning*

   Machine learning merupakan subset dari *Artificial Intelligence*, sebagai salah satu teknologi yang telah menghadirkan berbagai inovasi baru di berbagai bidang. *Machine learning* merujuk pada konsep pelatihan mesin sedemikian rupa sehingga sistem mampu belajar melalui data yang diberikan. Implementasi konsep ini dapat diterapkan secara luas di berbagai sektor dengan memanfaatkan beragam algoritma yang tersedia. Algoritma dilatih menggunakan data untuk kemudian melakukan klasifikasi atau prediksi, yang dapat membantu dalam pengambilan keputusan [66]. Algoritma *machine learning* yang digunakan dalam analisis sentimen dapat dikategorikan menjadi [67]:

   a. Supervised Learning Algorithms

   Algoritma *supervised learning* pada analisis sentimen melibatkan penggunaan *dataset* berlabel untuk melatih model yang mampu mengklasifikasikan atau memprediksi sentimen. *Supervised learning* mempelajari pasangan *input-output* selama fase *training*, sehingga memungkinkan model tersebut untuk melakukan prediksi atau klasifikasi pada data baru. Algoritma yang sering digunakan seperti *Support Vector Machines* (SVM), *K-Nearest Neighbor* (KNN), *Random Forest*, *Decision Tree*, dan Naïve Bayes.

   b. Unsupervised Learning Algorithms

   Algoritma *unsupervised learning* diterapkan dalam analisis sentimen untuk mengungkap pola atau struktur tersembunyi di dalam data yang tidak berlabel. *Unsupervised learning* sangat berguna dalam skenario di mana data berlabel bersifat langka atau tidak tersedia. Teknik seperti *clustering*, *dimensionality reduction* dan *topic modeling* merupakan metode yang umum digunakan untuk

menganalisis sentimen dan opini tanpa pengetahuan awal mengenai label data tersebut.

3. *Deep Learning*

   Deep learning menerapkan jaringan saraf tiruan (*artificial neural networks*) pada tugas analisis sentimen dengan menggunakan jaringan yang terdiri dari beberapa *layer*. Metode ini mampu mengeksploitasi *neural networks learning* secara lebih mendalam, yang sebelumnya hanya dianggap efisien jika digunakan pada satu atau dua *layer* serta dengan jumlah data yang terbatas. Terinspirasi dari struktur otak, *neural networks* terdiri dari sejumlah besar unit pemrosesan informasi (*neuron*) yang terorganisir dalam beberapa *layer* dan bekerja secara serentak. Sistem ini dapat belajar untuk melakukan tugas-tugas tertentu seperti klasifikasi dengan menyesuaikan bobot koneksi antar neuron [68]. Beberapa model *deep learning* yang populer meliputi RNN (*Recurrent Neural Network*), CNN (*Convolutional Neural Network*), LSTM (*Long Short-Term Memory*) dan *Transformer*.

Analisis sentimen merupakan salah satu tugas dalam NLP yang paling dikenal dalam identifikasi opini, komentar, emosi, pandangan, serta sikap dari penulis teks tersebut. Proses ini mengekstrak emosi penulis dalam bentuk subjektivitas seperti objektif dan subjektif, polaritas seperti positif, negatif dan netral, serta emosi seperti senang, marah, terkejut, sedih dan cemburu. Pesan dan teks ini ditemukan di berbagai media sosial, blog, atau forum situs *web*. Informasi yang ditemukan tersebut merupakan alat yang kuat dan bermanfaat untuk memahami serta merencanakan proses bagi pengguna dalam menghasilkan informasi yang diperlukan [69].

2.8 *E-Commerce*

E-commerce didefinisikan sebagai segala bentuk transaksi komersial yang dilakukan secara elektronik, termasuk pembelian dan penjualan barang dan jasa. *E-commerce* melibatkan distribusi, penjualan, pembelian, pemasaran, dan pelayanan produk melalui sistem elektronik seperti internet dan jaringan komputer lainnya yang merupakan bagian dari *e-business* yang lebih luas mencakup kolaborasi bisnis dan layanan pelanggan [70]. Untuk memastikan kesuksesan dan efisiensi *e-commerce*, keterhubungan antara dunia fisik dan digital serta pengelolaan data menjadi krusial. Pengguna perlu terkoneksi dengan internet dan bersedia melakukan pembelian secara *online* [71].

Ada empat jenis utama *e-commerce*, yang dibedakan berdasarkan tipe partisipannya yaitu [72, 73, 74, 75]:

1. *Business-to-Business* (B2B) menggambarkan pelaksanaan transaksi antar bisnis, seperti antara produsen dan distributor, atau antara distributor dan pengecer. B2B selalu menguntungkan perusahaan untuk menghemat biaya. B2B juga digunakan dalam konteks komunikasi dan kolaborasi antar perusahaan atau pedagang.
2. *Business-to-Customer* (B2C) menggambarkan aktivitas bisnis yang melayani konsumen akhir dengan produk atau layanan. Konsumen mengunjungi situs *web* suatu organisasi untuk melakukan pembelian seolah-olah mereka telah mendatangi pedagang tersebut secara langsung. Pesanan kemudian dikirim kepada pembeli dan pemilik *website* mendapatkan komisi dari setiap penjualan yang berhasil. B2C menyediakan sistem bagi organisasi yang memungkinkan mereka untuk mengarahkan konsumen kepada pedagang dan mendapatkan keuntungan dari imbalan komisi yang diberikan oleh pedagang tersebut.
3. *Customer-to-Customer* (C2C) melibatkan individu yang memperdagangkan barang dan jasa, terkadang terjadi di *platform* media sosial seperti Facebook dan eBay. Di dalam *marketplace* C2C, transaksi dapat berbentuk lelang atau pertukaran langsung antar individu. Model ini bertujuan untuk mengurangi ketergantungan pada pihak ketiga yang memvalidasi transaksi, sehingga menurunkan biaya administrasi oleh *platform* dan menghindari potensi monopoli dari penyedia *platform*.
4. *Consumer-to-Business* (C2B) merujuk pada model bisnis yang menyediakan produk dan layanan yang dipersonalisasi kepada konsumen berdasarkan permintaan konsumen. Model C2B juga disebut sebagai lelang terbalik (*reverse auction*) atau model pengumpulan permintaan (*demand collection*) dengan memungkinkan pembeli menentukan harga mereka sendiri sesuai pilihan mereka yang kemudian akan menghasilkan permintaan untuk barang atau jasa tertentu.

2.9 Tokopedia

Tokopedia yang didirikan oleh Willian Tanuwijaya dan Leontinus Alpha Edison ditahun 2009 merupakan salah satu perusahaan jual beli *online* terbesar di Indonesia dengan berbagai kategori produk yang dijual seperti pakaian, peralatan rumah tangga, perlengkapan olahraga, dan elektronik. Tokopedia beroperasi sebagai *mall online* sekaligus *marketplace* yang membuat produk yang ditawarkan kepada konsumen lebih beragam serta memudahkan setiap perusahaan untuk mendapatkan sasaran pasar sesuai dengan komoditasnya [76]. Tokopedia memanfaatkan strategi marketing seperti *flash sale*, *free shipping* dan kolaborasi dengan artis atau aktor ternama sebagai *brand ambassador* yang diharapkan dapat

menjangkau konsumen yang lebih luas [77]. Tokopedia memberikan rasa keamanan bertransaksi kepada konsumen dengan melakukan kerja sama dengan bank seperti Bank BCA, Mandiri, BNI dan BRI untuk proses jual beli yang lebih aman dan terjamin [78].

Gambar 2.4 PT. GoTo Gojek Tokopedia Tbk. [80]

Tokopedia melakukan *merger* dengan Gojek pada tahun 2021. Gojek merupakan perusahaan teknologi yang melayani jasa transportasi, *delivery* makanan dan barang serta dompet digital dalam bentuk *gopay*. Penggabungan ini membentuk perusahaan baru yaitu GoTo *Group*, yang memiliki tujuan untuk memperluas pasar dari sektor *e-commerce* dan transportasi yang ada dalam satu ekosistem serta membangun pengalaman yang lebih kohesif dan efisien bagi pelanggan [79]. *Merger* ini tidak hanya berdampak pada internal perusahaan, tetapi juga memberikan implikasi yang luas terhadap struktur dan dinamika ekosistem bisnis digital di Indonesia. GoTo diharapkan mampu menciptakan efisiensi operasional yang signifikan, meningkatkan kinerja keuangan, dan integrasi teknologi yang mendukung pertumbuhan jangka panjang. Sinergi antara layanan transportasi, *e-commerce*, dan keuangan digital dalam satu ekosistem GoTo menjadi keunggulan kompetitif yang membedakannya dari para pesaing regional [80].

2.10 Penjualan *Smartphone*

Smartphone adalah sebuah telepon yang *internet-enabled* yang menyediakan fungsi *Personal Digital Assistant* (PDA) seperti fungsi kalender, kalkulator, buku alamat, buku agenda, dan catatan [81]. Penjualan adalah kegiatan yang terpadu untuk mengembangkan rencana-rencana strategis yang diarahkan kepada usaha pemuasan kebutuhan serta keinginan pembeli/konsumen, guna mendapatkan penjualan yang menghasilkan laba atau keuntungan. Penjualan *smartphone* merujuk kepada kegiatan penukaran *smartphone* dengan uang atas persetujuan antara pemilik produk dan pemilik uang, pemilik produk menawarkan produknya kepada pembeli dengan harapan pembeli menyerahkan sejumlah uang sebagai alat ukur produk tersebut sebesar harga jual yang telah disetujui [82].

Gambar 2.5 Total Penjualan *Smartphone* di Dunia (Q1 2023 – Q3 2025) [1, 83]

Penjualan *smartphone* terus meningkat di seluruh dunia dari tahun 2023 hingga 2025, tercatat 322 juta unit terjual pada kuartal III tahun 2025 menurut laporan dari IDC [1]. Hal ini dapat dilihat pada Gambar 2.5 jika dibandingkan dengan tahun sebelumnya yaitu kuartal III tahun 2024 sekitar 316 juta unit, penjualan *smartphone* meningkat sebesar 2,1% [83]. Faktor yang mempengaruhi minat beli *smartphone* dapat diukur menggunakan indikator harga, desain, fitur, kamera, kualitas koneksi, *audio music*, kualitas layar, daya tahan baterai, garansi dan servis pasca beli, ketersediaan produk di pasar, merek, dukungan lokasi dan tampilan antar muka [84].

2.11 *Confusion Matrix*

Confusion matrix merupakan metode evaluasi yang digunakan sebagai pengukur kinerja model *machine learning* dalam tugas klasifikasi data dengan cara membandingkan hasil klasifikasi yang dilakukan oleh model yang digunakan dengan hasil klasifikasi sebenarnya [85]. *Confusion matrix* seperti pada Tabel 2.1 menjadi dasar dalam perhitungan berbagai metrik evaluasi, seperti *accuracy*, *precision*, *recall*, dan *F1-score* yang memberikan pemahaman lebih mendalam mengenai seberapa baik model memprediksi setiap kelas. *Confusion matrix* memiliki 4 komponen utama yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) dan *False Negative* (FN) [86] [87].

Tabel 2.1 *Confusion Matrix* [87]

Prediksi	Kelas Aktual	
	Postive	*Negative*
Positive	*True Positive* (TP)	*False Positive* (FP)
Negative	*False Negative* (FN)	*True Negative* (TN)

Dari *confusion matrix* kita dapat menghasilkan perspektif lain terhadap performa model yaitu *classification report* yang terdiri dari [88]:

1. *Accuracy*

   Accuracy mengukur jumlah data yang diklasifikasikan dengan benar dibandingkan dengan total jumlah sampel. Meskipun banyak digunakan, metrik ini dapat menghasilkan interpretasi yang bias terhadap *dataset* dengan distribusi kelas yang tidak seimbang.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2.12)$$

2. *Precision*

   Precision mengukur rasio prediksi positif yang benar-benar positif terhadap keseluruhan sampel yang diprediksi sebagai positif.

$$Precision = \frac{TP}{TP+FP} \quad (2.13)$$

3. *Recall*

   Recall mengukur sejauh mana model dapat memprediksi semua sampel positif yang ada pada keseluruhan sampel positif.

$$Recall = \frac{TP}{TP+FN} \quad (2.14)$$

4. *F1-score*

   F1-score menghitung rata-rata antara metrik *precision* dan *recall* yang mengindikasikan keseimbangan kedua metrik.

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision+Recall} \quad (2.15)$$