BAB II
KAJIAN LITERATUR

2.1 Kebijakan Pajak Pertambahan Nilai (PPN) 12%

Pajak Pertambahan Nilai (PPN) merupakan pajak tidak langsung yang dikenakan pada setiap tahap produksi dan distribusi barang dan jasa. Pajak ini dikenakan atas pertambahan nilai dari suatu barang atau jasa di setiap rantai transaksi, mulai dari produsen hingga konsumen akhir, namun beban pajaknya hanya efektif ditanggung oleh konsumen akhir melalui mekanisme pengkreditan pajak masukan terhadap pajak keluaran bagi pelaku usaha [31]. PPN dipungut atas seluruh transaksi penjualan dan pembelian barang atau jasa yang dilakukan oleh Wajib Pajak orang pribadi maupun badan yang telah terdaftar sebagai Pengusaha Kena Pajak (PKP) [1].

Dari sisi regulasi, PPN di Indonesia pertama kali diatur melalui Undang-Undang Nomor 8 Tahun 1983 tentang Pajak Pertambahan Nilai Barang dan Jasa serta Pajak Penjualan atas Barang Mewah, yang kemudian mengalami perubahan melalui Undang-Undang Nomor 42 Tahun 2009. Perubahan paling signifikan terjadi melalui Undang-Undang Nomor 7 Tahun 2021 tentang Harmonisasi Peraturan Perpajakan (UU HPP), yang mengatur kenaikan tarif PPN secara bertahap dari 10% menjadi 11% mulai 1 April 2022, dan selanjutnya menjadi 12% yang berlaku efektif paling lambat pada 1 Januari 2025 [31][32].

Kenaikan tarif PPN ini merupakan bagian dari upaya pemerintah untuk meningkatkan penerimaan negara guna mengurangi defisit Anggaran Pendapatan dan Belanja Negara (APBN) yang semakin besar akibat dampak pandemi COVID-19, sekaligus menunjang pembiayaan pembangunan dan mendorong pemulihan ekonomi nasional dalam jangka panjang [1][31]. Mengingat pajak secara keseluruhan berkontribusi sekitar 80% dari total penerimaan negara, kenaikan tarif PPN diharapkan mampu memperbaiki struktur fiskal dan menciptakan ruang fiskal yang lebih besar untuk pembiayaan program sosial seperti pendidikan, kesehatan, dan infrastruktur [31].

Meskipun demikian, kebijakan ini tidak lepas dari berbagai kekhawatiran di masyarakat. Kenaikan tarif PPN berpotensi menurunkan daya beli masyarakat, terutama bagi kelompok konsumen yang pengeluarannya sensitif terhadap perubahan harga. Kenaikan harga barang dan jasa akibat meningkatnya biaya produksi mendorong konsumen untuk lebih selektif dalam pengeluarannya, dengan kemungkinan beralih ke produk yang lebih

terjangkau atau menunda pembelian barang tertentu [31]. Tekanan ini pada akhirnya juga berdampak pada pelaku Usaha Mikro, Kecil, dan Menengah (UMKM) yang merupakan sektor penting dalam perekonomian Indonesia dengan kontribusi besar terhadap Produk Domestik Bruto (PDB) dan lapangan kerja. Pelaku UMKM menghadapi tantangan yang signifikan akibat kebijakan ini, terutama karena mereka cenderung mengandalkan bahan baku dan barang dengan harga yang terpengaruh langsung oleh tarif PPN, sehingga kenaikan biaya produksi berpotensi memengaruhi harga jual produk dan daya saing mereka di pasar [33].

Dari hasil kajian yang ada, kenaikan PPN berkontribusi pada peningkatan Indeks Harga Konsumen (IHK) sebesar sekitar 0,8 hingga 1%, dengan dampak yang lebih signifikan dirasakan pada barang sekunder dan tersier. Sementara itu, barang kebutuhan pokok yang telah dikecualikan dari PPN relatif tidak terpengaruh secara langsung [31]. Sebagai upaya mitigasi, pemerintah menyiapkan sejumlah kebijakan kompensasi yang meliputi subsidi langsung untuk kebutuhan pokok, program Bantuan Langsung Tunai (BLT) bagi kelompok berpenghasilan rendah, serta fasilitas pengecualian PPN terhadap barang-barang esensial seperti obat-obatan dan layanan pendidikan [31].

Di sisi lain, persepsi publik yang Negatif terhadap suatu kebijakan perpajakan yang tidak direspons secara tepat berpotensi mengurangi kesediaan wajib pajak untuk memenuhi kewajibannya, sehingga berdampak pada tingkat kepatuhan pajak (*tax compliance*) secara keseluruhan [2]. Kondisi ini semakin terlihat melalui tingginya volume respons publik yang muncul di platform media sosial, khususnya platform X, di mana masyarakat mengekspresikan berbagai bentuk pendapat terhadap kebijakan PPN 12% mulai dari penolakan, kekhawatiran, hingga berbagai bentuk ekspresi lainnya. Tingginya intensitas dan keberagaman ekspresi publik terhadap kebijakan ini menjadikan isu PPN 12% sebagai objek yang relevan untuk dikaji lebih lanjut melalui pendekatan analisis sentimen berbasis kecerdasan buatan.

2.2 Media Sosial

Media sosial merupakan platform berbasis internet yang memungkinkan individu membuat, berbagi, dan berinteraksi melalui konten yang dihasilkan oleh pengguna itu sendiri (*user-generated content*). Dalam konteks penelitian ilmu sosial dan teknologi informasi, media sosial telah berkembang menjadi sumber data primer yang berharga untuk memahami opini publik, tren sosial, dan perilaku masyarakat secara *real-time*. Sifat data yang terbuka, volume yang besar, dan kecepatan produksinya menjadikan media sosial sebagai

representasi digital dari dinamika opini masyarakat yang tidak dapat dihasilkan melalui survei tradisional dengan biaya dan waktu sebanding [34].

Gambar 2.1 Ragam Platform Media Sosial [35]

Oltra-Badenes et al. (2023) dalam tinjauan sistematisnya terhadap penggunaan Twitter dalam penelitian ilmu sosial mencatat bahwa pengumpulan data dari platform X secara programatik telah menjadi metodologi standar dalam penelitian ilmu sosial komputasional, dengan berbagai pendekatan teknis yang terus berkembang seiring perubahan kebijakan akses platform [34]. Temuan ini memperkuat relevansi platform media sosial khususnya X sebagai sumber data yang sah dan representatif dalam penelitian analisis sentimen [5][6].

2.3 Platform X (Twitter)

Gambar 2.2 Platform X [36]

X, sebelumnya dikenal sebagai Twitter, adalah platform media sosial berbasis mikroblog yang memungkinkan pengguna mempublikasikan pesan singkat (*tweet*) berisi teks, gambar, video, dan tautan. Sejak peluncurannya, X berkembang menjadi salah satu platform diskusi publik terbesar di dunia dengan ratusan juta pengguna aktif bulanan yang menghasilkan miliaran *tweet* setiap harinya. Kekhasan platform ini terletak pada karakteristik berbasis teks yang terbuka, penggunaan tagar (*hashtag*) untuk kategorisasi

tematik percakapan, dan sistem percakapan berantai (*thread*) yang memudahkan pelacakan isu secara longitudinal [4].

Freelon et al. (2024) dalam analisis historis mereka terhadap akses data media sosial selama hampir dua dekade mencatat bahwa penelitian komputasional berbasis data Twitter/X telah mengalami transformasi signifikan seiring dengan pembatasan dan komersialisasi akses API oleh platform, namun platform ini tetap menjadi sumber data yang relevan dan luas digunakan dalam kajian opini publik terhadap isu-isu kebijakan [35]. Dari perspektif analisis sentimen, X memiliki keunggulan struktural berupa volume data yang besar, sifat publik *tweet*, serta kemampuan filtrasi berbasis kata kunci dan tagar yang memudahkan pengumpulan data tematik, termasuk untuk isu kebijakan PPN 12% [7][8].

Meskipun demikian, teks dari platform X memiliki tantangan tersendiri dalam pemrosesan komputasional. Teks *tweet* umumnya pendek, informal, dan mengandung berbagai elemen nonstandar seperti singkatan, bahasa gaul (*slang*), percampuran bahasa Indonesia-Inggris (*code-mixing*), emoji, dan referensi kontekstual yang membutuhkan pemahaman mendalam [8]. Keragaman linguistik ini menjadi faktor utama yang mendorong kebutuhan akan model NLP yang secara khusus dikembangkan untuk bahasa Indonesia, sebagaimana disoroti oleh Aji et al. (2022) dalam kajiannya terhadap tantangan NLP untuk bahasa-bahasa daerah Indonesia [11].

2.4 Kecerdasan Buatan

Kecerdasan buatan (*Artificial Intelligence*/AI) merupakan cabang ilmu komputer yang berfokus pada pengembangan sistem dan program komputer yang mampu melakukan tugas-tugas yang umumnya memerlukan kecerdasan manusia, seperti penalaran, pembelajaran, pengambilan keputusan, pemahaman bahasa, dan persepsi visual [36] Konsep dasar kecerdasan buatan pertama kali diperkenalkan oleh John McCarthy pada tahun 1956, yang mendefinisikannya sebagai ilmu dan rekayasa pembuatan mesin cerdas [36].

Secara umum, kecerdasan buatan dapat dikategorikan ke dalam dua pendekatan utama. Pertama, *symbolic AI* yang bekerja berdasarkan aturan logika dan representasi pengetahuan eksplisit. Kedua, *machine learning* yang memungkinkan sistem belajar secara otomatis dari data tanpa diprogram secara eksplisit untuk setiap tugas [36]. Perkembangan teknologi komputasi dan ketersediaan data dalam jumlah besar mendorong kemajuan signifikan pada pendekatan kedua, khususnya melalui *deep learning* yang menggunakan jaringan saraf tiruan berlapis untuk mengekstraksi representasi fitur secara hierarkis dari data mentah.

Kecerdasan buatan telah diterapkan secara luas di berbagai bidang, termasuk pengolahan citra, sistem rekomendasi, diagnosis medis, kendaraan otonom, hingga pemrosesan bahasa alami (*Natural Language Processing*/NLP) [36]. Penerapan kecerdasan buatan pada NLP memungkinkan komputer untuk memahami, menginterpretasikan, dan menghasilkan teks dalam bahasa manusia, yang menjadi dasar bagi berbagai aplikasi seperti analisis sentimen, penerjemahan mesin, dan ekstraksi informasi.

2.5 Pemrosesan Bahasa Alami (*Natural Language Processing*)

Pemrosesan Bahasa Alami (*Natural Language Processing*/NLP) merupakan cabang kecerdasan buatan yang berfokus pada pengembangan metode komputasional untuk memungkinkan komputer memahami, menganalisis, dan menghasilkan bahasa manusia. Bidang ini mencakup berbagai tugas pemrosesan bahasa, seperti klasifikasi teks, terjemahan mesin, pengenalan entitas bernama (*named entity recognition*), serta pembuatan teks secara otomatis [37].

Perkembangan NLP modern telah mengalami akselerasi luar biasa, terutama sejak diperkenalkannya arsitektur *Transformer* oleh Vaswani et al. dan model bahasa pralatih (*pre-trained language models*) berbasis *self-supervised learning*. Kalyan, Rajasekharan, dan Sangeetha (2021) dalam survei komprehensifnya terhadap model pralatih berbasis *Transformer* menunjukkan bahwa pendekatan *pre-training* dan *fine-tuning* telah mendominasi hampir semua *benchmark* NLP utama dan secara konsisten mengalahkan pendekatan-pendekatan sebelumnya yang mengandalkan rekayasa fitur manual [38]. Transformasi paradigma ini berdampak langsung pada cara sistem analisis sentimen dibangun dan dievaluasi dalam penelitian kontemporer.

2.6 Analisis Sentimen

Analisis sentimen, yang juga dikenal sebagai *opinion mining*, adalah tugas NLP yang bertujuan mengidentifikasi dan mengekstraksi opini, sikap, dan emosi yang terkandung dalam teks terhadap suatu entitas, topik, produk, atau kejadian tertentu. Wankhade, Rao, dan Kulkarni (2022) mendefinisikan analisis sentimen sebagai proses menentukan orientasi emosional dari suatu teks apakah Positif, Negatif, atau Netral dengan menganalisis karakteristik linguistik pada tingkat kata, frasa, kalimat, maupun dokumen secara keseluruhan [39].

Dalam konteks penelitian ini, analisis sentimen diterapkan pada tingkat *tweet* (*tweet-level sentiment analysis*), dimana setiap *tweet* diklasifikasikan secara independen ke dalam

salah satu dari tiga kelas: Positif, Negatif, atau Netral. Nandwani dan Verma (2021) dalam tinjauan mereka terhadap analisis sentimen dan deteksi emosi mengidentifikasi bahwa tantangan utama dalam analisis sentimen pada teks media sosial meliputi penggunaan bahasa informal dan *slang*, ambiguitas semantik, ironi dan sarkasme, serta keberagaman linguistik yang sangat tinggi kondisi yang semuanya sangat relevan dengan karakteristik data berbahasa Indonesia di platform X [40].

2.7 Pembelajaran Mesin Tradisional

Sebelum era *deep learning*, pendekatan pembelajaran mesin tradisional mendominasi penelitian analisis sentimen. Pendekatan ini bekerja dengan dua tahap utama: pertama, mengekstraksi fitur teks secara manual menggunakan representasi seperti *Bag-of-Words* (BoW) atau *Term Frequency-Inverse Document Frequency* (TF-IDF); kemudian melatih algoritma klasifikasi menggunakan fitur-fitur tersebut [39].

Naïve Bayes adalah algoritma probabilistik berbasis Teorema Bayes yang mengklasifikasikan teks berdasarkan probabilitas kemunculan kata pada setiap kelas sentimen. Meskipun efisien dalam pelatihan dan cukup andal pada *dataset* kecil, asumsi independensi fitur yang menjadi pondasinya jarang terpenuhi dalam konteks linguistik nyata [12]. Siagian dan Painem (2024) menerapkan Naïve Bayes untuk menganalisis sentimen publik terhadap rencana kenaikan PPN 12% di media sosial X dan memperoleh akurasi yang cukup memadai pada skala penelitian tersebut [12].

Support Vector Machine (SVM) bekerja dengan mencari *hyperplane* optimal yang memisahkan kelas-kelas data dengan margin maksimum di ruang fitur berdimensi tinggi. Algoritma ini sering dikombinasikan dengan representasi TF-IDF dan terbukti kompetitif pada berbagai tugas klasifikasi teks [8].

Decision Tree adalah algoritma klasifikasi berbasis struktur pohon yang mempartisi data secara rekursif berdasarkan nilai fitur yang paling informatif pada setiap simpul, menghasilkan aturan keputusan yang dapat diinterpretasikan secara langsung. Hardyatman dan Hasan (2025) menerapkan metode *Decision Tree* untuk menganalisis sentimen masyarakat terhadap rencana kenaikan PPN 12% di media sosial X, dan menemukan bahwa *Decision Tree* mampu mengklasifikasikan sentimen publik menjadi kategori Positif, Negatif, dan Netral dengan kemampuan interpretabilitas yang baik, meskipun algoritma ini dikenal rentan terhadap *overfitting* pada data yang kompleks dan tidak seimbang [7]. Hasan dan Bimby (2025) dalam perbandingan beberapa algoritma *machine learning* untuk analisis sentimen kebijakan PPN menunjukkan bahwa meskipun SVM, *Logistic Regression*, dan

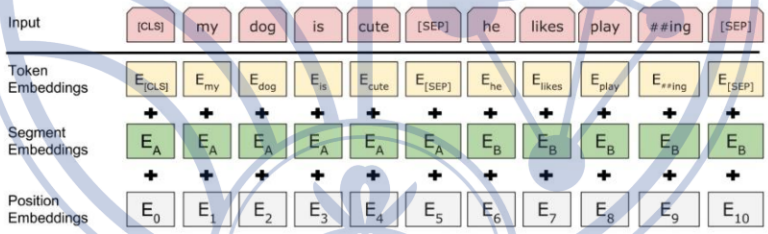
Naive Bayes mampu mengklasifikasikan sentimen publik, pendekatan tradisional ini memiliki ruang peningkatan yang signifikan dibandingkan model berbasis *deep learning* [8].

Keterbatasan mendasar dari seluruh pendekatan tradisional ini adalah ketidakmampuannya menangkap konteks semantik yang mendalam. Representasi berbasis kata tidak mampu memahami makna kata yang bergantung pada konteks, hubungan semantik antarkata, maupun fenomena linguistik kompleks seperti ironi, sarkasme, negasi, dan ambiguitas makna kondisi yang sangat umum dalam teks informal media sosial berbahasa Indonesia [14][40].

2.8 Pembelajaran Mendalam (*Deep Learning*)

Minaee et al. (2021) dalam tinjauan komprehensifnya terhadap klasifikasi teks berbasis *deep learning* mendefinisikan pendekatan ini sebagai metode yang menggunakan jaringan saraf tiruan berlapis banyak untuk mempelajari representasi fitur secara otomatis dan hierarkis dari data teks mentah, menghilangkan kebutuhan akan rekayasa fitur manual yang membutuhkan keahlian domain khusus [41]. Dalam konteks NLP, *deep learning* telah merevolusi cara komputer memahami bahasa alami, memungkinkan pemodelan semantik yang jauh lebih kaya dibandingkan metode tradisional.

2.8.1 BERT dan IndoBERT

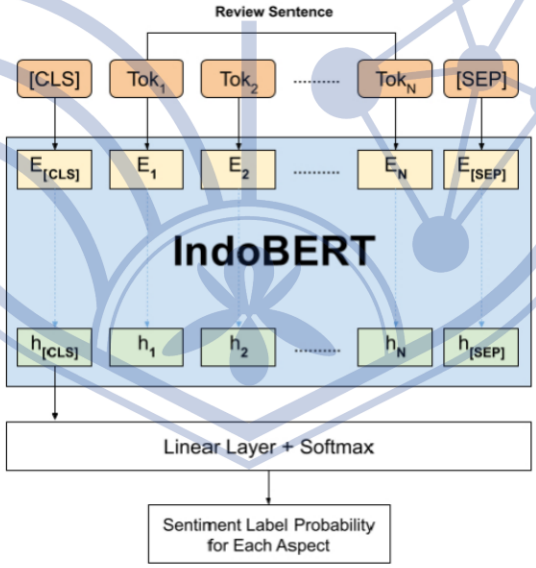


Gambar 2.3 Arsitektur Model BERT[43]

BERT (*Bidirectional Encoder Representations from Transformers*) adalah model bahasa pralatih yang memanfaatkan arsitektur *Transformer Encoder* untuk menghasilkan representasi kata yang bersifat kontekstual dan *bidirectional*. Arsitektur *Transformer* sendiri, sebagaimana diulas secara mendalam oleh Lin et al. (2022), dibangun atas mekanisme *Multi-Head Self-Attention* yang memungkinkan setiap token secara dinamis membobot seberapa besar kontribusi token lain dalam menentukan representasinya sehingga makna kata ditangkap berdasarkan konteks kalimat secara menyeluruh, bukan secara terisolasi [42].

Kalyan, Rajasekharan, dan Sangeetha (2021) dalam survei model pralatih berbasis *Transformer* menerangkan bahwa BERT dilatih menggunakan dua tugas *self-supervised* pada korpus skala besar: *Masked Language Modeling* (MLM), dimana sebagian token *input* disembunyikan dan model dilatih untuk memprediksinya dari konteks dua arah; serta *Next Sentence Prediction* (NSP), dimana model dilatih menentukan apakah dua kalimat yang diberikan merupakan pasangan yang berurutan secara alami. Melalui pralatih ini, BERT membangun representasi linguistik yang mendalam dan dapat di-*fine-tune* untuk berbagai tugas NLP hilir (*downstream tasks*) dengan sedikit data berlabel [38].

Meskipun BERT versi asli dan *bert-base-multilingual-cased* (mBERT) tersedia dalam versi multibahasa, representasinya untuk bahasa Indonesia cenderung terbatas karena dominasi data pelatihan berbahasa Inggris [18]. Untuk mengatasi keterbatasan ini, Koto et al. (2020) mengembangkan IndoBERT sebagai model BERT yang dilatih secara eksklusif menggunakan lebih dari 4 miliar token bahasa Indonesia dari sumber-sumber beragam, meliputi korpus berita lokal, Wikipedia bahasa Indonesia, dan *web crawl* [15]. Dengan basis data pelatihan yang kaya akan kosakata dan ragam bahasa informal bahasa Indonesia, IndoBERT secara inheren lebih mampu merepresentasikan nuansa semantik bahasa Indonesia, termasuk variasi informal yang lazim di media sosial dibandingkan mBERT [18][19].



Gambar 2.4 Arsitektur Model IndoBERT [45]

Gambar 2.4 mengilustrasikan alur kerja penerapan IndoBERT untuk klasifikasi sentimen berbasis aspek. Kalimat ulasan (*review sentence*) diproses melalui tokenizer berbasis *WordPiece* yang menambahkan token-token khusus [CLS] dan [SEP] sehingga menghasilkan deretan token: [CLS], Tok_1, Tok_2, ..., Tok_n, [SEP]. Setiap token kemudian

dipetakan ke dalam ruang vektor melalui proses *input embedding* yang merupakan penjumlahan dari tiga komponen, yaitu *token embedding* (t_i), *segment embedding* (s_i), dan *positional embedding* (w_i) sebagaimana didefinisikan oleh Devlin et al. (2019) [43]

$$E_i = t_i + s_i + w_i \quad (1)$$

Setiap model pralatih memiliki dimensi tetap dalam representasi vektor sebesar 768 dimensi. Vektor embedding input selanjutnya diproses oleh encoder IndoBERT yang memiliki 12 *hidden layer* masing-masing berdimensi 768, 12 *attention head*, serta *feed-forward hidden layer* berdimensi 3.072 [19]. Di dalam setiap layer, mekanisme *Multi-Head Self-Attention* menghitung bobot perhatian antar token menggunakan rumus yang dirumuskan oleh Devlin et al. (2019) [43]

$$Attention(Q, K, V) = softmax\left(QK^T/sqrt(d_k)\right)V \quad (2)$$

di mana Q, K, V adalah matriks *Query*, *Key*, dan *Value* yang diturunkan dari representasi token, dan d_k adalah dimensi *Key* sebagai faktor skala. Hasil output encoder berupa representasi kontekstual setiap token: h[CLS], h_1, h_2, ..., h_n, h[SEP]. Token [CLS] merangkum seluruh representasi kalimat, sehingga hanya vektor h[CLS] yang diteruskan ke *linear layer* untuk memetakannya ke ruang jumlah kelas sentiment [43]:

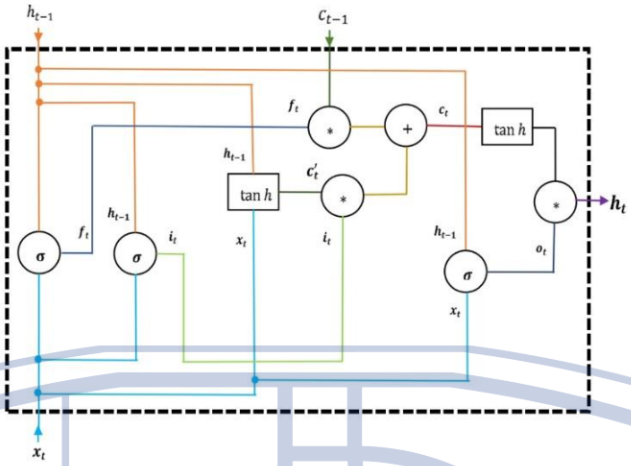
$$\hat{y} = W \cdot h_{[CLS]} + b \quad (3)$$

Output dari *linear layer* kemudian diproses menggunakan fungsi Softmax untuk menghasilkan distribusi probabilitas atas setiap kelas [43]:

$$P(y = c \mid x) = \text{softmax}(\hat{y})^c = \frac{e^{\widehat{y^c}}}{\sum_j e^{\widehat{y_j}}} \quad (4)$$

Nilai probabilitas tertinggi dari output Softmax menentukan label sentimen yang diprediksi untuk setiap aspek. Untuk masalah klasifikasi tingkat kalimat seperti analisis sentimen, IndoBERT mengambil representasi output dari token pertama [CLS] kemudian memberikannya ke sebuah *softmax classifier* untuk mendapatkan prediksi klasifikasi yang digunakan sebagai output model untuk proses *fine-tuning* [44].

2.8.2 LSTM dan BiLSTM



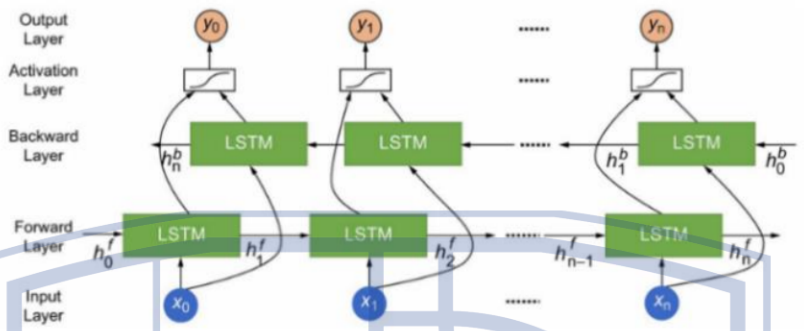
Gambar 2.5 Arsitektur Model LSTM [43]

Long Short-Term Memory (LSTM) adalah varian jaringan saraf rekuren (*Recurrent Neural Network*/RNN) yang dirancang untuk mengatasi masalah *vanishing gradient* pada RNN konvensional, yaitu kondisi dimana gradien menjadi sangat kecil saat melewati urutan yang panjang sehingga koneksi jangka jauh sulit dipelajari. Minaee et al. (2021) menjelaskan bahwa LSTM mengatasi masalah ini melalui penambahan sel memori (*memory cell*) dan tiga jenis gerbang (*gate*): *forget gate* yang mengontrol informasi mana dari memori sebelumnya yang perlu dilupakan, *input gate* yang mengontrol informasi baru yang ditambahkan ke memori, serta *output gate* yang mengontrol informasi mana yang diteruskan sebagai *output*. Kombinasi ini memungkinkan LSTM menangkap dependensi urutan kata dalam jangka panjang secara adaptif [31].

LSTM konvensional memproses urutan teks secara searah (dari kiri ke kanan), sehingga representasi setiap token hanya mempertimbangkan token-token sebelumnya. Untuk menangkap konteks yang lebih lengkap dari kedua arah, dikembangkanlah *Bidirectional* LSTM (BiLSTM). Basiri et al. (2021) dalam penelitiannya tentang model *bidirectional deep learning* untuk analisis sentimen menjelaskan bahwa BiLSTM terdiri dari dua lapisan LSTM yang bekerja secara paralel: lapisan *forward* memproses urutan dari kiri ke kanan, sementara lapisan *backward* memproses dari kanan ke kiri representasi dari kedua lapisan, kemudian digabungkan (*concatenation*) untuk menghasilkan representasi kontekstual yang mencakup informasi dari kedua arah secara simultan [45].

Keunggulan BiLSTM dibandingkan LSTM konvensional dalam analisis sentimen telah dikonfirmasi secara empiris dalam berbagai penelitian. Asnawiyah dan Putra (2025) dalam studi komparatifnya menyimpulkan bahwa BiLSTM secara konsisten menghasilkan performa klasifikasi sentimen yang lebih baik dibandingkan LSTM *unidirectional* pada data

teks media sosial, karena kemampuannya memanfaatkan konteks dua arah untuk memahami makna kata yang bergantung pada kata-kata sesudahnya, seperti dalam konstruksi negasi atau frasa ambigu [22].



Gambar 2.6 Arsitektur Model BiLSTM [45]

Gambar 2.6 mengilustrasikan arsitektur BiLSTM secara lengkap, mulai dari lapisan input hingga lapisan output. Masukan BiLSTM berupa urutan vektor token x_0, x_1, x_2, ..., x_n yang kemudian diproses oleh dua lapisan LSTM secara paralel. Di dalam setiap sel LSTM, terdapat tiga mekanisme *gate* yang bekerja secara matematis untuk mengatur aliran informasi pada setiap *time step* t. Pertama, *forget gate* memutuskan informasi mana dari memori sebelumnya yang harus dilupakan:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{5}$$

Kedua, *input gate* memutuskan informasi baru apa yang akan ditambahkan ke memori sel, disertai kandidat nilai sel baru ($\check{C}_t$):

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{6}$$

$$\widetilde{C_t} = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{7}$$

Pembaruan *cell state* (C_t) dilakukan dengan melupakan sebagian informasi lama dan menambahkan informasi baru:

$$C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C_t} \tag{8}$$

Ketiga, *output gate* menentukan informasi apa yang diteruskan sebagai *hidden state* (h_t) ke *time step* berikutnya:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{9}$$

$$h_t = o_t \odot \tanh(C_t) \tag{10}$$

di mana σ adalah fungsi sigmoid, tanh adalah fungsi hiperbolik tangen, W adalah matriks bobot, b adalah bias, dan ⊙ menyatakan perkalian elemen per elemen (*element-wise multiplication*). Lapisan *forward* LSTM memproses urutan token dari awal hingga akhir,

menghasilkan representasi $\overrightarrow{h_t}$, sementara lapisan *backward* memproses dari arah berlawanan menghasilkan representasi $\overleftarrow{h_t}$:

$$\overrightarrow{h_t} = \text{LSTM}_{forward}\left(x_t, \overrightarrow{h_{t-1}}, \overrightarrow{C_{t-1}}\right) \quad (11)$$

$$\vec{h}_t = LSTM_backward\left(x_t, \vec{h}_{t+1}, \vec{C}_{t+1}\right) \quad (12)$$

Representasi dari kedua lapisan kemudian digabungkan melalui operasi konkatenasi pada setiap *time step*, sehingga jika dimensi setiap LSTM adalah d, vektor gabungan h_t berdimensi 2d:

$$h_t = \left[\overrightarrow{h_t} \oplus \backslash overleftarrow h_t\right] \quad (13)$$

2.8.3 Tokenisasi dan *Padding*

1. Tokenisasi *WordPiece*

Tokenisasi adalah proses memecah teks menjadi unit-unit terkecil yang disebut token, yang merupakan satuan terkecil yang diproses oleh model bahasa. Dalam konteks IndoBERT, tokenisasi dilakukan menggunakan metode *WordPiece tokenization*. Mielke et al. (2021) dalam sejarah komprehensif metode tokenisasi untuk NLP menjelaskan bahwa *WordPiece* adalah algoritma *sub-word tokenization* yang membangun kosakata secara iteratif dengan menggabungkan pasangan karakter atau subkata yang memaksimalkan kemungkinan data pelatihan [46]. Song et al. (2021) dalam penelitian mereka tentang algoritma *WordPiece* secara eksplisit menjelaskan bahwa subkata yang merupakan kelanjutan dari subkata sebelumnya yakni yang berada di tengah atau akhir kata, ditandai dengan awalan "##" sebagai *suffix indicator*, sehingga memungkinkan model membedakan token awal kata dengan token lanjutan [47].

Pendekatan *sub-word* ini memungkinkan model menangani kata-kata yang tidak terdapat dalam kosakata (*out-of-vocabulary*/OOV) tanpa kehilangan informasi semantik yang signifikan, sekaligus membatasi ukuran kosakata pada batas yang dapat dikelola. Sebagai contoh, kata "pertambahan" mungkin direpresentasikan sebagai subtoken ["per", "##tam", "##bahan"]. Lin dan Nuha (2023) dalam penelitian analisis sentimen berbasis arsitektur hibrida *deep learning* menjelaskan bahwa *token embedding* dalam arsitektur BERT yang menjadi fondasi IndoBERT selalu dimulai dengan token khusus [CLS] dan diakhiri dengan token [SEP], dilengkapi dengan *segment embeddings* dan *position embeddings* yang dijumlahkan secara *element-wise* sebagai representasi *input* akhir [38] [46][48].

2. *Padding* dan *Truncation*

Dalam proses pelatihan model *deep learning* berbasis *batch*, semua sekuens *input* dalam satu *batch* harus memiliki panjang yang seragam. Lin dan Nuha (2023) menjelaskan bahwa *tokenizer* memproses kalimat dengan dua prosedur sekaligus: *truncation* yang memotong sekuens yang melebihi panjang maksimum (*max_length*), serta *padding* yang menambahkan token khusus (PAD) pada sekuens yang lebih pendek hingga mencapai panjang yang seragam menghasilkan tiga komponen *output* yaitu input_ids, attention_mask, dan token_type_ids yang digunakan sebagai *input* model [47]. Proses normalisasi panjang sekuens ini merupakan tahap penting dalam persiapan data untuk model klasifikasi teks berbasis arsitektur hibrida IndoBERT–BiLSTM.

2.8.4 *Fine Tunning*

Fine-tuning adalah proses adaptasi model bahasa pralatih (*pre-trained language model*) terhadap tugas spesifik dengan cara melatih ulang seluruh atau sebagian parameter model menggunakan data berlabel dalam jumlah terbatas. Devlin et al. (2019) mendefinisikan *fine-tuning* sebagai tahap dimana seluruh parameter model diinisialisasi dengan bobot hasil pralatih, kemudian disesuaikan secara bersama-sama (*end-to-end*) menggunakan data berlabel dari tugas hilir (*downstream task*) yang ditargetkan [43].

Secara mekanisme, *fine-tuning* dilakukan dengan menambahkan satu atau lebih lapisan *fully connected* di atas lapisan encoder terakhir model pralatih. Rogers, Kovaleva, dan Rumshisky (2020) menjelaskan bahwa selama proses *fine-tuning*, model dilatih menggunakan fungsi *loss cross-entropy* yang mengukur selisih antara prediksi model dan label sebenarnya [49]:

$$\mathcal{L} = -\sum (c = 1)^{C} y_c \log(\hat{y_c}) \qquad (14)$$

di mana y_c adalah label kelas sebenarnya, $\hat{y}_c$ adalah probabilitas prediksi dari output Softmax, dan C adalah jumlah kelas. Nilai loss ini yang kemudian menjadi sinyal untuk memperbarui bobot seluruh model melalui proses *backpropagation.

Keunggulan utama pendekatan *fine-tuning* dibandingkan melatih model dari awal (*training from scratch*) adalah efisiensi sumber daya dan data. Karena model pralatih telah memiliki pemahaman linguistik umum yang mendalam, *fine-tuning* hanya membutuhkan data berlabel yang jauh lebih sedikit dan waktu pelatihan yang jauh lebih singkat untuk mencapai performa tinggi pada tugas spesifik [43][49].

2.8.5 Hyperparameter

Hyperparameter adalah parameter yang nilainya ditetapkan sebelum proses pelatihan dimulai dan tidak dipelajari langsung dari data melalui proses pembelajaran, berbeda dengan bobot (*weight*) dan bias yang diperbarui secara otomatis melalui *backpropagation*. Hyperparameter berfungsi mengendalikan proses pembelajaran model, sehingga nilai optimalnya umumnya ditentukan melalui eksperimen seperti *grid search*, *random search*, maupun pengujian manual terhadap beberapa kombinasi nilai [50]. Dalam pengembangan model *deep learning*, beberapa hyperparameter yang paling memengaruhi performa dan kecepatan konvergensi model meliputi *learning rate*, *batch size*, dropout, serta ukuran lapisan tersembunyi (*hidden size*), yang masing-masing dijelaskan pada subbab berikut.

a. Learning Rate

Learning rate merupakan hyperparameter yang mengatur besarnya langkah pembaruan bobot model pada setiap iterasi pelatihan berdasarkan nilai gradien dari fungsi *loss* [51]. Pemilihan nilai *learning rate* menjadi krusial karena nilai yang terlalu kecil dapat menyebabkan proses pelatihan berjalan sangat lambat dan berisiko terhenti pada solusi yang kurang optimal, sedangkan nilai yang terlalu besar berisiko menyebabkan model mempelajari bobot yang sub-optimal secara terlalu cepat atau bahkan menyebabkan proses pelatihan menjadi tidak stabil [50]. Beberapa penelitian menunjukkan bahwa *learning rate*, *batch size*, dan parameter regularisasi lainnya saling terkait secara erat sehingga nilai optimalnya perlu ditentukan secara bersamaan, bukan secara terpisah satu per satu [51].

b. Batch Size

Batch size adalah hyperparameter yang menentukan jumlah sampel data yang diproses secara bersamaan dalam satu kali iterasi sebelum bobot model diperbarui melalui *backpropagation* [52]. Penentuan ukuran *batch size* yang tepat memerlukan pertimbangan antara kecepatan komputasi, kecepatan konvergensi, serta keterbatasan memori yang tersedia [52]. *Batch size* yang lebih kecil umumnya menghasilkan pembaruan bobot yang lebih sering namun dengan estimasi gradien yang lebih bervariasi, sedangkan *batch size* yang lebih besar menghasilkan estimasi gradien yang lebih stabil namun membutuhkan kapasitas komputasi yang lebih besar [50].

c. Dropout

Dropout merupakan teknik regularisasi yang digunakan untuk mengurangi risiko *overfitting* pada jaringan saraf tiruan dengan cara menonaktifkan sebagian neuron secara acak pada setiap iterasi pelatihan [53]. Probabilitas suatu neuron dinonaktifkan ditentukan oleh nilai *dropout rate* yang ditetapkan sebagai hyperparameter. Srivastava et al.

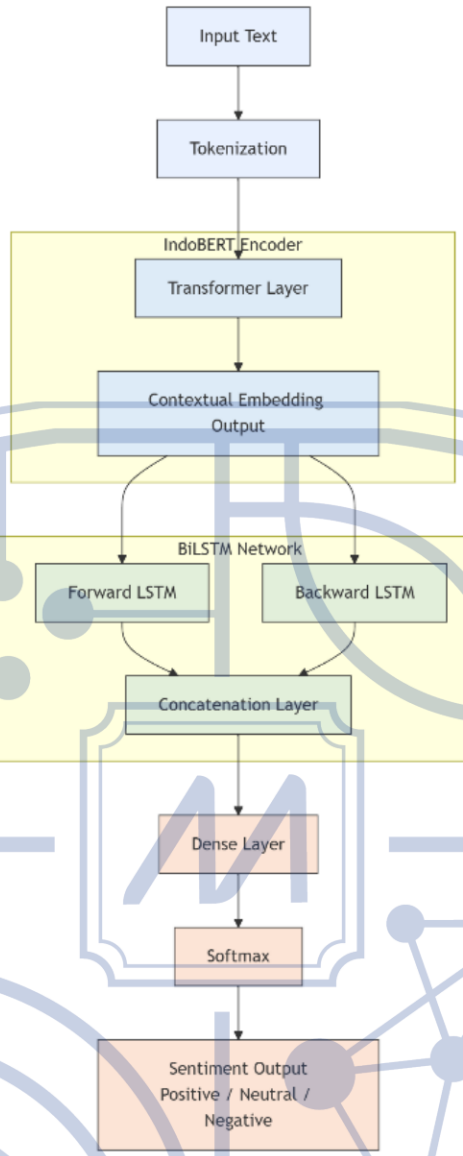
menjelaskan bahwa dropout secara konseptual mensimulasikan pelatihan terhadap sejumlah besar sub-jaringan (*thinned network*) yang berbeda secara bersamaan, sehingga model tidak menjadi terlalu bergantung pada neuron tertentu dan menghasilkan representasi yang lebih general [53]. Dalam konteks arsitektur hibrida BERT-BiLSTM untuk analisis sentimen, nilai *dropout* yang umum digunakan berkisar antara 0,1 hingga 0,5 yang diterapkan setelah lapisan BiLSTM maupun sebelum lapisan klasifikasi akhir [54].

d. Hidden Size

Hidden size merupakan hyperparameter yang menentukan dimensi atau jumlah unit pada lapisan tersembunyi suatu jaringan saraf, termasuk pada lapisan BiLSTM. Ukuran *hidden size* secara langsung memengaruhi kapasitas representasi model *hidden size* yang lebih besar memungkinkan model mempelajari pola yang lebih kompleks, namun juga meningkatkan jumlah parameter yang harus dilatih serta risiko *overfitting* apabila tidak diimbangi dengan regularisasi yang memadai [54]. Pada arsitektur hibrida BERT-BiLSTM, ukuran *hidden size* BiLSTM yang digunakan pada berbagai penelitian analisis sentimen bervariasi antara 150 hingga 300 unit per arah, dengan kombinasi dropout untuk mencegah *overfitting* pada representasi yang dihasilkan [54][44]. Pada arsitektur BiLSTM, *hidden size* menentukan dimensi vektor *hidden state* yang dihasilkan oleh masing-masing arah (*forward* dan *backward*) sebelum digabungkan melalui operasi konkatenasi.

2.8.6 Model Hibrida IndoBERT–BiLSTM

Model hibrida IndoBERT–BiLSTM menggabungkan kekuatan representasi kontekstual dari IndoBERT dengan kemampuan pemodelan sekuensial dua arah dari BiLSTM. Secara arsitektur, IndoBERT berfungsi sebagai *feature extractor* yang menghasilkan representasi vektor kontekstual berkualitas tinggi dan berdimensi untuk setiap token, yang kemudian diproses lebih lanjut oleh lapisan BiLSTM untuk menangkap pola urutan yang relevan bagi klasifikasi sentimen. *Output* BiLSTM selanjutnya diteruskan ke lapisan *fully connected* (*dense*) dengan fungsi aktivasi *softmax* untuk menghasilkan distribusi probabilitas atas tiga kelas sentimen [24][26].



Gambar 2.7 Tahapan Model Hibrida IndoBERT-BiLSTM [18][26]

Dasar pemilihan arsitektur hibrida ini bersifat substantif. Dhendra dan Utomo (2025) menunjukkan bahwa IndoBERT tanpa lapisan sekuensial berpotensi kurang optimal dalam menangkap pola urutan jangka panjang pada kalimat yang kompleks, karena mekanisme *self-attention* BERT menghitung atensi global secara paralel tanpa mekanisme pemrosesan sekuensial yang eksplisit [21]. Di sisi lain, Ahmadian et al. (2024) menegaskan bahwa BiLSTM tanpa representasi bahasa pralatih harus membangun pemahaman semantik dari nol, yang berisiko mengalami *underfitting* pada *dataset* berukuran terbatas [25]. Integrasi keduanya menghasilkan model yang saling melengkapi: IndoBERT menyediakan representasi semantik yang kaya dari hasil pralatih pada miliaran token bahasa Indonesia, sementara BiLSTM menambahkan kemampuan pemodelan pola sekuensial yang eksplisit [26][27].

Subarkah et al. (2025) memvalidasi efektivitas arsitektur IndoBERT–BiLSTM dalam penelitiannya tentang sentimen kebijakan pemerintah di media sosial X dan menyimpulkan bahwa integrasi kedua pendekatan ini menghasilkan performa klasifikasi yang baik dalam mengidentifikasi sentimen Positif, Negatif, dan Netral dari teks media sosial berbahasa Indonesia [24].

Secara matematis, alur pemrosesan model hibrida IndoBERT–BiLSTM pada setiap tahapannya dijelaskan sebagai berikut.

a. IndoBERT Embedding

Setiap token yang direpresentasikan dalam bentuk indeks *vocabulary* dipetakan ke dalam matriks embedding IndoBERT untuk menghasilkan representasi vektor kontekstual awal [25]. Proses pemetaan tersebut dinyatakan sebagai:

$$x_i = E \cdot t_i \tag{15}$$

di mana x_i adalah vektor embedding token ke-i, E adalah matriks embedding IndoBERT berukuran:

$$E \in \mathbb{R}^{V \times 768} \tag{16}$$

dengan V menyatakan ukuran *vocabulary* IndoBERT dan 768 adalah dimensi embedding yang digunakan oleh model, serta t_i adalah indeks token ke-i. Output yang dihasilkan berupa *last_hidden_state* berdimensi (N, 128, 768) yang merepresentasikan seluruh token dalam sekuens [25].

b. Pemrosesan BiLSTM

Output last_hidden_state IndoBERT selanjutnya diproses oleh lapisan BiLSTM. Lapisan *forward* memproses urutan token dari awal hingga akhir, sedangkan lapisan *backward* memproses dari arah sebaliknya, sebagaimana telah dijelaskan pada Persamaan (11) dan (12). Representasi dari kedua lapisan kemudian digabungkan melalui operasi konkatenasi sebagaimana Persamaan (13), sehingga setiap token menghasilkan representasi berdimensi 512 dan output BiLSTM secara keseluruhan berdimensi (N, 128, 512) [55].

c. Attention Pooling

Output BiLSTM diproses menggunakan mekanisme *attention pooling* untuk memberikan bobot kepentingan yang berbeda pada setiap token, sehingga model dapat lebih fokus pada token yang relevan dalam menentukan sentimen suatu kalimat [56]. Skor *attention* untuk setiap token dihitung menggunakan lapisan *linear* sebagai berikut:

$$e_t = W_a h_t + b_a \tag{17}$$

di mana e_t adalah skor *attention* token ke-t, h_t adalah output BiLSTM token ke-t, W_a adalah bobot *attention*, dan b_a adalah bias *attention*. Skor tersebut kemudian dinormalisasi menggunakan fungsi *softmax* agar totalnya bernilai 1 [56]:

$$\alpha_t = \frac{\exp(e_t)}{\sum_j \exp(e_j)} \tag{18}$$

di mana α_t adalah bobot *attention* token ke-t dan T adalah jumlah token dalam sekuens. Representasi akhir kalimat kemudian diperoleh melalui operasi *weighted sum* sebagai berikut [57]:

$$s = \sum_t \alpha_t \cdot h_t \tag{19}$$

di mana s adalah *pooled vector* berdimensi 512 yang merepresentasikan keseluruhan kalimat.

d. Dropout

Pooled vector s selanjutnya diproses menggunakan lapisan *dropout* untuk mengurangi risiko *overfitting* selama proses pelatihan [53]. Pada tahap pelatihan, *dropout* menonaktifkan sebagian elemen vektor secara acak, sedangkan pada tahap pengujian seluruh nilai vektor diteruskan tanpa perubahan. Output lapisan ini tetap berdimensi 512.

e. Klasifikasi Sentimen

Vektor hasil *dropout* selanjutnya diteruskan ke lapisan *linear classifier* untuk menghasilkan skor mentah (*logits*) pada setiap kelas sentiment [55]:

$$z = Ws + b \tag{20}$$

di mana z adalah vektor *logits*, W adalah bobot *classifier*, s adalah vektor masukan, dan b adalah bias. Nilai *logits* kemudian dikonversi menjadi distribusi probabilitas menggunakan fungsi *softmax* sebagaimana Persamaan (4), dan kelas dengan probabilitas tertinggi ditetapkan sebagai label sentimen prediksi [55].

2.9 Data Crawling

Data crawling adalah proses otomatis pengumpulan data dari internet menggunakan program komputer yang disebut *crawler* atau *scraper*. Dalam konteks penelitian berbasis media sosial, *data crawling* memungkinkan pengumpulan teks, metadata, dan konten digital dalam volume besar yang tidak dapat dicapai secara manual. Oltra-Badenes et al. (2023) dalam tinjauan sistematisnya menegaskan bahwa Twitter merupakan platform media sosial yang paling populer digunakan oleh ilmuwan sosial untuk penelitian karena kemudahan

akses datanya secara programatik melalui API, menjadikan pengumpulan data Twitter secara terprogram sebagai metodologi yang lazim dalam penelitian ilmu sosial komputasional [34].

Freelon et al. (2024) menegaskan bahwa pembatasan dan komersialisasi akses API oleh platform media sosial seperti Twitter/X telah mendorong peneliti untuk mengadopsi strategi pengumpulan data yang lebih beragam, termasuk pendekatan berbasis *scraping* dan alat-alat kolaboratif yang dikembangkan komunitas riset sebagai alternatif akses data media sosial dalam konteks penelitian akademis [35].

2.9.1 *Tweets Harvest*

Tweets Harvest adalah alat (*tool*) pengumpulan data *tweet* berbasis Python yang dirancang untuk mengotomatiskan pengambilan *tweet* dari platform X berdasarkan kata kunci, tagar, atau pengguna tertentu dalam rentang waktu yang ditentukan. Alat ini memanfaatkan mekanisme pencarian platform X secara efisien dan telah digunakan secara luas dalam penelitian analisis sentimen berbahasa Indonesia yang berbasis data media sosial [10].

Tweets Harvest digunakan untuk mengumpulkan *tweet* berbahasa Indonesia yang mengandung kata kunci berkaitan dengan kebijakan PPN 12% dalam periode yang ditetapkan. Rentang waktu tersebut dipilih untuk mencakup fase sebelum dan sesudah pemberlakuan tarif PPN 12% pada 1 Januari 2025, sehingga data yang terkumpul dapat merepresentasikan dinamika sentimen publik dalam kedua fase tersebut secara komparatif.

2.10 Pelabelan Data Manual

Pelabelan data manual (*manual data labeling*) adalah proses pemberian label atau anotasi pada data mentah oleh manusia (*human annotator*) yang berfungsi sebagai *ground truth* untuk pelatihan model *supervised learning*. Kualitas data yang dilabeli merupakan faktor determinan utama dalam performa model *machine learning*, mengingat model hanya dapat belajar sebaik label yang diberikan kepadanya. Klie, Castilho, dan Gurevych (2023) dalam analisis komprehensifnya terhadap kesalahan anotasi menegaskan bahwa kualitas anotasi bukan sekadar kebutuhan teknis, melainkan fondasi epistemis dari validitas seluruh proses penelitian berbasis data berlabel [58].

Dalam penelitian analisis sentimen, pelabelan data melibatkan pemberian label Positif, Negatif, atau Netral pada setiap *tweet* berdasarkan pemahaman kontekstual anotator. Proses ini membutuhkan anotator yang memahami konteks linguistik bahasa Indonesia

informal, pengetahuan tentang isu kebijakan PPN 12%, serta kemampuan menginterpretasikan makna tersurat maupun tersirat dalam teks media sosial [58].

Annotation guidelines (pedoman anotasi) adalah dokumen yang mendefinisikan kriteria, aturan, dan contoh yang digunakan oleh anotator dalam memberikan label pada data. Uma et al. (2021) dalam survei mereka tentang pembelajaran dari ketidaksepakatan anotator menekankan bahwa pedoman anotasi yang komprehensif dan konsisten merupakan komponen krusial dalam memastikan kualitas dan reliabilitas data berlabel, terutama ketika pelabelan melibatkan lebih dari satu anotator [59]. Ketidakjelasan pedoman anotasi merupakan salah satu sumber utama ketidakkonsistenan label yang dapat secara langsung mengurangi kualitas model yang dibangun.

Pedoman anotasi mendefinisikan tiga kelas sentimen berdasarkan karakteristik konten *tweet* sebagai berikut:

1. Positif: *Tweet* yang mengekspresikan dukungan, persetujuan, optimisme, atau pandangan yang menguntungkan terhadap kebijakan PPN 12%. Contoh: persetujuan terhadap argumen fiskal pemerintah, apresiasi terhadap pengecualian barang kebutuhan pokok, atau harapan Positif terhadap dampak kebijakan.
2. Negatif: *Tweet* yang mengekspresikan penolakan, ketidaksetujuan, kekhawatiran, kritik, atau sentimen tidak puas terhadap kebijakan PPN 12%. Contoh: keluhan atas peningkatan beban biaya hidup, kritik terhadap mekanisme kebijakan, atau kekhawatiran dampak ekonomi.
3. Netral: *Tweet* yang bersifat informatif atau faktual tanpa mengekspresikan pandangan yang jelas mendukung maupun menolak kebijakan. Contoh: penyebaran informasi faktual tentang kebijakan, pertanyaan Netral tentang implementasi.

2.10.1 Inter-Annotator Agreement dan Fleiss Kappa

Dalam proses pelabelan data yang melibatkan lebih dari satu anotator, diperlukan mekanisme untuk mengukur tingkat kesepakatan antar-anotator (*inter-annotator agreement*/IAA) guna memvalidasi konsistensi dan reliabilitas label yang dihasilkan. Tanpa pengukuran IAA, label yang diperoleh tidak dapat dinyatakan objektif dan reliabel, karena perbedaan interpretasi antar-anotator dapat mengintroduksi bias sistematis ke dalam dataset [51]. Pengukuran IAA menjadi komponen kritis dalam penelitian berbasis anotasi manusia, khususnya untuk tugas-tugas subjektif seperti analisis sentimen, di mana interpretasi teks dapat bervariasi antar-individu [60].

Fleiss Kappa (κ) merupakan statistik yang dikembangkan sebagai generalisasi dari Cohen's Kappa untuk mengakomodasi skenario pelabelan yang melibatkan lebih dari dua anotator dengan skala kategorikal nominal [61]. Berbeda dengan Cohen's Kappa yang terbatas pada dua penilai, Fleiss Kappa secara eksplisit dirancang untuk mengukur tingkat kesepakatan di antara *k* anotator yang menilai *n* subjek ke dalam *c* kategori yang saling eksklusif [60]. Keunggulan ini menjadikan Fleiss Kappa sebagai pilihan yang tepat dalam penelitian anotasi berlabel multi-kelas yang melibatkan lebih dari dua anotator, termasuk pada penelitian analisis sentimen dengan tiga kelas sentimen (Positif, Negatif, dan Netral). Secara matematis, Fleiss Kappa diformulasikan sebagai berikut:

$$\kappa = \frac{\bar{P} - \bar{P}_e}{1 - \bar{P}_e} \tag{21}$$

Di mana $\bar{P}$ merepresentasikan proporsi kesepakatan yang teramati (*observed agreement*) yang dihitung berdasarkan rata-rata proporsi pasangan anotator yang memberikan label yang sama untuk setiap subjek, sedangkan $\bar{P}_e$ merepresentasikan proporsi kesepakatan yang diharapkan secara acak (*expected agreement by chance*) berdasarkan distribusi marginal label masing-masing kategori [61]. Pembagi (1 - $\bar{P}_e$) berfungsi menormalisasi nilai kappa terhadap besarnya kesepakatan acak, sehingga nilai κ = 1 menunjukkan kesepakatan sempurna dan κ = 0 menunjukkan kesepakatan yang setara dengan kebetulan [51].

Nilai $\bar{P}$ dihitung melalui persamaan:

$$\bar{P} = \frac{1}{n \cdot k(k-1)} \sum_{i=1}^{N} \sum_{j=1}^{c} n_{ij} \left(n_{ij} - 1\right) \tag{22}$$

Sedangkan nilai $\bar{P}_e$ dihitung sebagai:

$$\bar{P}_e = \sum_{j=1}^{c} p_j^2 \tag{23}$$

Di mana *n* adalah jumlah total item yang dilabeli, *k* adalah jumlah anotator, *c* adalah jumlah kategori, n_{ij} adalah jumlah anotator yang mengategorikan item ke-*i* ke dalam kategori *j*, dan p_j adalah proporsi seluruh penilaian yang diberikan pada kategori *j* [61]. Interpretasi nilai Fleiss Kappa mengacu pada skala yang dirangkum oleh McHugh (2012), yang membagi tingkat kesepakatan ke dalam enam kategori sebagai berikut [61]:

Tabel 2.1 Nilai Interpretasi Tingkat Kesepakatan [61]

Nilai κ	**Interpretasi Tingkat Kesepakatan**
< 0,00	Poor (Buruk)
0,01 – 0,20	Slight (Sangat Lemah)
0,21 – 0,40	Fair (Cukup)
0,41 – 0,60	Moderate (Sedang)
0,61 – 0,80	Substantial (Baik)
0,81 – 1,00	Almost Perfect (Hampir Sempurna)

Dalam konteks penelitian anotasi data untuk *machine learning*, nilai kappa minimum yang dapat diterima sebagai bukti reliabilitas anotasi adalah $\kappa \geq 0{,}60$, yang merepresentasikan tingkat kesepakatan *substantial* [61]. James (2026) memperkuat hal ini dengan menyatakan bahwa nilai IAA yang rendah dapat mengindikasikan pedoman anotasi yang kurang spesifik, pelatihan anotator yang tidak memadai, atau tingginya subjektivitas pada tugas yang dievaluasi, sehingga pengukuran dan pelaporan IAA secara transparan merupakan praktik esensial dalam penelitian NLP [61]. Pada penelitian analisis sentimen khususnya, Uma et al. (2021) menekankan bahwa ketidaksepakatan antar-anotator merupakan fenomena yang inheren dalam tugas-tugas subjektif, dan pengukuran IAA menjadi alat utama untuk memverifikasi sejauh mana label yang dihasilkan dapat dipercaya sebagai *ground truth* [59].

Fleiss' Kappa digunakan untuk mengukur tingkat kesepakatan di antara tiga anotator yang melabeli *tweet* terkait kebijakan PPN 12% ke dalam tiga kategori sentimen, yaitu Positif, Negatif, dan Netral. Nilai κ yang diperoleh berfungsi sebagai bukti kuantitatif bahwa proses pelabelan manual telah memenuhi standar reliabilitas yang dapat diterima secara ilmiah.

2.11 *Exploratory Data Analysis* (EDA)

Exploratory Data Analysis (EDA) merupakan tahap awal dalam proses analisis data yang bertujuan untuk memahami karakteristik dasar dari suatu dataset sebelum dilakukan proses pemodelan atau analisis lanjutan. EDA dilakukan dengan menggunakan statistik deskriptif dan teknik visualisasi data untuk mengidentifikasi pola, distribusi data, serta

kemungkinan adanya anomali atau ketidakteraturan dalam data yang dapat memengaruhi hasil analisis selanjutnya [62].

Konsep Exploratory Data Analysis diperkenalkan oleh Tukey sebagai pendekatan analisis data yang menekankan proses eksplorasi untuk menemukan pola dan hubungan yang tersembunyi dalam data tanpa bergantung pada asumsi model statistik tertentu. Melalui proses eksplorasi ini, peneliti dapat memperoleh pemahaman awal mengenai struktur data sehingga dapat menentukan metode analisis yang lebih tepat pada tahap berikutnya [63].

Dalam konteks data science dan machine learning, EDA memiliki peran penting dalam tahap data understanding karena membantu peneliti mengidentifikasi berbagai permasalahan pada dataset seperti data duplikat, nilai yang hilang (missing values), serta ketidakseimbangan distribusi data. Selain itu, EDA juga membantu memahami pola distribusi variabel serta hubungan antar variabel sehingga proses pembangunan model prediktif dapat dilakukan secara lebih efektif dan akurat [64].

Pada penelitian berbasis teks seperti analisis sentimen, EDA biasanya dilakukan dengan menganalisis karakteristik data teks, seperti distribusi panjang teks, frekuensi kata yang muncul, serta distribusi label sentimen dalam dataset. Analisis ini memberikan gambaran awal mengenai struktur data teks yang bersifat tidak terstruktur sehingga tahap prapemrosesan dan pemodelan Natural Language Processing (NLP) dapat dilakukan dengan lebih terarah.

2.12 Prapemrosesan Data (*Text Preprocessing*)

Prapemrosesan data merupakan tahap penting dalam *pipeline Natural Language Processing* yang bertujuan membersihkan, menstandarkan, dan menyiapkan data teks mentah sebelum digunakan dalam proses pemodelan. Data yang berasal dari media sosial umumnya bersifat tidak terstruktur dan mengandung berbagai elemen seperti URL, emoji, singkatan, serta variasi bahasa informal yang dapat memengaruhi kualitas representasi teks. Oleh karena itu, proses prapemrosesan diperlukan untuk mengurangi *noise* dan meningkatkan konsistensi data sehingga model dapat mempelajari pola linguistik yang lebih relevan untuk tugas *Sentiment Analysis* [65].

Penelitian sebelumnya menunjukkan bahwa tahapan pembersihan dan normalisasi teks dapat memberikan dampak signifikan terhadap performa model klasifikasi sentimen karena membantu model fokus pada informasi semantik yang penting dibandingkan variasi tekstual yang tidak relevan [65]. Selain itu, dalam konteks bahasa Indonesia, keragaman dialek, singkatan, dan bahasa informal yang banyak digunakan di media sosial juga menjadi

tantangan tersendiri dalam pemrosesan teks, sehingga diperlukan proses normalisasi yang sistematis sebelum dilakukan pemodelan berbasis *machine learning* atau *deep learning* [11]. Beberapa tahapan prapemrosesan diterapkan secara berurutan untuk menyiapkan data teks sebelum digunakan dalam proses pemodelan menggunakan IndoBERT. Tahapan tersebut meliputi *case folding*, *noise removal*, *demojize*, *handling repetitive characters*, serta *slang transformation*.

2.12.1 *Case Folding*

Case folding merupakan proses mengubah seluruh huruf dalam teks menjadi huruf kecil (*lowercase*) untuk menghindari perbedaan representasi token akibat variasi kapitalisasi. Variasi penulisan huruf besar dan kecil pada teks media sosial dapat menyebabkan kata yang sama dianggap sebagai token yang berbeda oleh sistem pemrosesan teks. Oleh karena itu, proses *case folding* digunakan untuk menstandarkan representasi teks sehingga dapat mengurangi redundansi kosakata dalam *dataset* [65].

Tabel 2.2 Aturan Proses Case Folding [54]

Komponen	Deskripsi
Tujuan	Mengubah seluruh huruf dalam teks menjadi huruf kecil agar konsisten.
Input	Teks dengan huruf besar dan kecil.
Output	Teks yang seluruhnya telah dikonversi menjadi huruf kecil.

2.12.2 *Noise Removal*

Noise removal merupakan proses penghapusan elemen teks yang tidak memberikan kontribusi langsung terhadap analisis sentimen, seperti URL, *mention* pengguna, *hashtag*, atau simbol tertentu. Elemen-elemen tersebut sering muncul pada teks media sosial dan dapat mengganggu proses ekstraksi fitur jika tidak dihapus terlebih dahulu. Dengan menghilangkan unsur-unsur tersebut, teks yang dihasilkan menjadi lebih bersih dan relevan untuk proses analisis lebih lanjut [65].

Tabel 2.3 Komponen Noise Removal [54]

Elemen	Deskripsi
URL	Menghapus tautan *web* yang tidak berkontribusi terhadap analisis sentimen.
Mention	Menghapus referensi pengguna pada media sosial yang diawali dengan simbol “@”.

Hashtag	Menghapus simbol *hashtag* (#) yang digunakan untuk kategorisasi topik.
Punctuation	Menghapus tanda baca atau simbol yang tidak memengaruhi makna sentimen.
Numbers	Menghapus karakter numerik yang tidak relevan dengan analisis sentimen.

2.12.3 *Demojize*

Emoji merupakan elemen komunikasi digital yang sering digunakan untuk mengekspresikan emosi dalam teks media sosial. Menghapus emoji secara langsung dapat menyebabkan hilangnya informasi sentimen yang penting. Oleh karena itu, emoji dikonversi menjadi representasi teks menggunakan proses *demojize* sehingga informasi emosional tetap dapat dipertahankan dan diproses oleh model analisis sentimen [66].

Tabel 2.4 Contoh Transformasi Emoji dengan Proses Demojize [55]

Emoji	Representasi Teks	Keterangan
🙂	slightly_smiling_face	Emoji dikonversi menjadi token teks melalui proses demojize.
😂	face_with_tears_of_joy	Emoji dikonversi menjadi token teks melalui proses demojize.
😾	pouting_face	Emoji dikonversi menjadi token teks melalui proses demojize.
😢	crying_face	Emoji dikonversi menjadi token teks melalui proses demojize.

2.12.4 *Handling Repetitive Characters*

Dalam teks media sosial, pengguna sering mengekspresikan emosi dengan mengulang karakter dalam suatu kata, misalnya “bagussss” atau “tidakkkkk”. Fenomena ini dikenal sebagai *character elongation* dan dapat menghasilkan kata yang berada di luar kosakata standar. Oleh karena itu dilakukan proses normalisasi untuk membatasi pengulangan karakter sehingga kata kembali mendekati bentuk standar dan dapat dikenali oleh model NLP [65].

Tabel 2.5 Penaganan Karakter Berulang [54]

Fenomena	Deskripsi
Character Elongation	Pengulangan karakter dalam suatu kata untuk mengekspresikan emosi.
Normalisasi Karakter	Mengurangi jumlah karakter berulang agar mendekati bentuk kata standar.
Tujuan	Menghindari kata yang tidak terdapat dalam kosakata model NLP.

2.12.5 *Slang Transformation*

Bahasa yang digunakan dalam media sosial sering mengandung kata tidak baku, singkatan, atau istilah gaul yang tidak terdapat dalam kamus bahasa formal. Dalam konteks bahasa Indonesia, keragaman dialek dan penggunaan bahasa informal menjadi tantangan tersendiri dalam pemrosesan teks. Oleh karena itu dilakukan proses normalisasi teks untuk mengubah kata tidak baku menjadi bentuk baku menggunakan kamus slang sehingga konsistensi representasi kata dalam *dataset* dapat ditingkatkan [11].

Tabel 2.6 Contoh Normalisasi Kata Tidak Baku [11]

Kata Tidak Baku	Kata Baku	Keterangan
gk	gak	bentuk singkatan
bgt	banget	singkatan kata
dr	dari	penghilangan huruf vokal
udh	sudah	bentuk informal

2.13 Penanganan Ketidakseimbangan Kelas (*Class Imbalance*)

Ketidakseimbangan kelas (*class imbalance*) terjadi ketika distribusi sampel di antara kelas-kelas dalam *dataset* tidak merata secara signifikan. Dalam analisis sentimen terhadap kebijakan publik yang kontroversial seperti PPN 12%, kondisi ini sangat umum terjadi sentimen Negatif cenderung mendominasi secara tidak proporsional karena masyarakat yang tidak puas lebih terdorong untuk mengekspresikan pendapatnya di media sosial [67].

Tarawneh et al. (2022) dalam tinjauan kritis mereka terhadap berbagai teknik penanganan ketidakseimbangan kelas menegaskan bahwa model yang dilatih pada data tidak seimbang secara inheren bias terhadap kelas mayoritas, menghasilkan akurasi keseluruhan yang tinggi namun performa yang buruk pada kelas minoritas. Pendekatan berbasis

algoritma seperti *class weight* dan *focal loss* menawarkan solusi tanpa perlu memodifikasi komposisi *dataset* [67]. Penanganan *class imbalance* mengacu pada pendekatan yang diterapkan dalam analisis sentimen ulasan pelanggan Tokopedia menggunakan BiLSTM dan IndoBERT, yang membandingkan efektivitas berbagai teknik *preprocessing* dan pelabelan dalam menangani ketidakseimbangan kelas [44].

2.13.1 *Class Weight*

Class weight merupakan salah satu teknik penanganan ketidakseimbangan kelas (*class imbalance*) yang diterapkan pada tingkat algoritma dengan memberikan bobot yang berbeda pada setiap kelas selama proses pelatihan model. Pada pendekatan ini, kelas minoritas biasanya diberikan bobot yang lebih tinggi sehingga kesalahan prediksi pada kelas tersebut menghasilkan penalti yang lebih besar dalam perhitungan fungsi *loss*. Dengan demikian, model didorong untuk memberikan perhatian yang lebih seimbang terhadap seluruh kelas tanpa harus melakukan perubahan pada distribusi data dalam *dataset* pelatihan. Pendekatan ini termasuk dalam kategori *algorithm-level methods* yang umum digunakan untuk mengatasi permasalahan ketidakseimbangan kelas pada tugas klasifikasi [67].

Secara umum, bobot kelas dihitung berdasarkan frekuensi kemunculan setiap kelas dalam *dataset* pelatihan, sehingga kelas dengan jumlah sampel yang lebih sedikit memperoleh bobot yang lebih besar. Perhitungan bobot kelas dapat dinyatakan sebagai berikut:

$$w_c = \frac{n_{total}}{n_{classes} \times n_c} \tag{24}$$

dimana n_{total} merupakan jumlah total sampel dalam dataset, $n_{classes}$ adalah jumlah kelas, dan n_c adalah jumlah sampel pada kelas ke-c. Dengan pendekatan ini, kelas dengan jumlah data yang lebih sedikit akan mendapatkan penalti kesalahan yang lebih besar ketika terjadi kesalahan klasifikasi, sehingga model tidak cenderung bias terhadap kelas mayoritas selama proses pelatihan.

2.13.2 *Focal Loss*

Focal loss merupakan modifikasi dari fungsi *Cross-Entropy Loss* yang dirancang untuk mengatasi permasalahan ketidakseimbangan kelas (*class imbalance*) dengan menyesuaikan kontribusi kesalahan prediksi selama proses pelatihan model. Metode ini bekerja dengan mengurangi kontribusi relatif dari sampel yang mudah diklasifikasikan (*easy examples*), sehingga proses pelatihan lebih berfokus pada sampel yang sulit diklasifikasikan

(*hard examples*). Pendekatan ini banyak digunakan pada berbagai tugas klasifikasi teks untuk meningkatkan performa model pada *dataset* dengan distribusi kelas yang tidak seimbang [68].

Pada tugas klasifikasi sentimen teks dengan distribusi kelas yang tidak seimbang, penggunaan *focal loss* dapat membantu model memberikan perhatian lebih besar terhadap sampel yang sulit diprediksi, yang umumnya berasal dari kelas minoritas atau memiliki ambiguitas semantic [68].

Secara matematis, *focal loss* untuk klasifikasi multikelas didefinisikan sebagai berikut [69]:

$$FL(p_{t,c}) = -\sum_{c=1}^{C} \alpha_c \left(1 - p_{t,c}\right)^{\gamma} \log(p_{t,c}) \tag{25}$$

dimana Cadalah jumlah kelas (pada penelitian ini $C = 3$, yaitu Positif, Negatif, dan Netral), $p_{t,c}$merupakan probabilitas prediksi untuk kelas $c$yang diperoleh dari fungsi *softmax*, $\alpha_c$adalah faktor penyeimbang per kelas yang digunakan untuk mengatasi ketidakseimbangan distribusi kelas, dan γ(*gamma*) merupakan parameter pemfokus (*focusing parameter*) yang mengontrol tingkat penekanan terhadap sampel yang sulit diklasifikasikan. Faktor modulasi $\left(1 - p_{t,c}\right)^{\gamma}$akan mendekati nol ketika sampel mudah diklasifikasikan ($p_{t,c} \approx 1$), sehingga kontribusinya terhadap *total loss* menjadi kecil. Sebaliknya, ketika nilai $p_{t,c}$kecil, faktor ini akan mendekati satu sehingga kesalahan prediksi mendapatkan bobot yang lebih besar. Dengan mekanisme tersebut, *focal loss* dapat membantu model untuk lebih fokus pada sampel yang sulit dan mengurangi dominasi kontribusi dari sampel yang mudah selama proses pelatihan [68].

2.14 Pembagian Data (*Data Splitting*)

Pembagian data (*data splitting*) merupakan proses membagi keseluruhan *dataset* berlabel menjadi beberapa subset yang digunakan secara terpisah untuk proses pelatihan (*training*) dan evaluasi model. Praktik ini merupakan langkah penting dalam pengembangan model *machine learning* karena memungkinkan evaluasi performa model dilakukan menggunakan data yang tidak digunakan selama proses pelatihan, sehingga memberikan estimasi kemampuan generalisasi model yang lebih objektif [70].

Training set merupakan bagian data yang digunakan untuk melatih parameter model, termasuk bobot jaringan dan parameter *fine-tuning* pada model IndoBERT. Dalam beberapa pendekatan pengembangan model, sebagian data pelatihan juga dapat dialokasikan sebagai

validation set yang digunakan untuk memantau performa model selama proses pelatihan serta melakukan penyesuaian *hyperparameter*. Sementara itu, *test set* disisihkan sepenuhnya dari proses pelatihan dan hanya digunakan pada tahap evaluasi akhir untuk mengukur performa model secara independen [70][71].

Proporsi pembagian data yang umum digunakan dalam eksperimen *machine learning* adalah rasio 80:20 atau 70:30 antara data pelatihan dan data pengujian. Rasio ini sering digunakan untuk menjaga keseimbangan antara jumlah data yang cukup untuk melatih model dan data yang memadai untuk mengevaluasi performa model secara objektif [61]. Teknik *stratified splitting* diterapkan untuk memastikan distribusi kelas sentimen pada setiap subset tetap proporsional dengan distribusi kelas pada keseluruhan *dataset*. Pendekatan ini penting terutama pada *dataset* klasifikasi untuk menghindari ketidakseimbangan distribusi kelas yang dapat memengaruhi proses pelatihan maupun evaluasi model [71].

2.15 Evaluasi Performa Model

Evaluasi performa model adalah tahapan kritis dalam pengembangan sistem klasifikasi yang bertujuan mengukur secara kuantitatif kemampuan model membuat prediksi akurat pada data yang belum pernah dilihat sebelumnya. Penggunaan *multiple metrics* dalam evaluasi sangat penting karena setiap metrik mengukur aspek performa yang berbeda, dan tidak ada metrik tunggal yang memberikan gambaran performa yang lengkap terutama pada data yang tidak seimbang [47].

2.15.1 *Confusion matrix*

Confusion matrix adalah representasi tabular yang menggambarkan performa model klasifikasi secara rinci dengan membandingkan label yang diprediksi oleh model terhadap label aktual (*ground truth*) pada data uji. Untuk klasifikasi tiga kelas (Positif, Negatif, dan Netral), *confusion matrix* berupa matriks 3×3 yang memungkinkan analisis kesalahan secara granular, termasuk identifikasi kelas mana yang paling sering terkonfusi satu sama lain [72].

Tabel 2.7 Representasi *Confusion matrix* 3x3 [63]

Aktual	**Prediksi**			**Total Aktual**
	Negatif	**Netral**	**Positif**	
Negatif	TP_{Neg}	$FN_{Neg \rightarrow Net}$	$FN_{Neg \rightarrow Pos}$	ΣNeg

Netral	$FP_{Net \rightarrow Neg}$	TP_{Net}	$FN_{Net \rightarrow Pos}$	*ΣNet*
Positif	$FP_{Pos \rightarrow Neg}$	$FP_{Pos \rightarrow Net}$	TP_{Pos}	*ΣPos*
Total Prediksi	*ΣPred Neg*	*ΣPred Net*	*ΣPred Pos*	**N = Data Uji**

Diagonal utama matriks, yaitu TP_{pos}, TP_{neg}, dan TP_{net}, merepresentasikan jumlah sampel yang diprediksi secara benar untuk masing-masing kelas. Nilai di luar diagonal merepresentasikan kesalahan klasifikasi, dimana FN(*False Negative*) menunjukkan sampel dari suatu kelas yang salah diprediksi ke kelas lain, sedangkan FP(*False Positive*) menunjukkan sampel dari kelas lain yang salah diprediksi masuk ke suatu kelas [Grandini et al., 2020]. Sebagai contoh, $FN_{pos \rightarrow neg}$berarti sampel berlabel Positif yang salah diprediksi sebagai Negatif, sedangkan $FP_{neg \rightarrow pos}$berarti sampel berlabel Negatif yang salah diprediksi sebagai Positif.

Diagonal utama matriks, yaitu TP_{pos}, TP_{neg}, dan TP_{net}, merepresentasikan jumlah sampel yang diprediksi secara benar untuk masing-masing kelas. Nilai di luar diagonal merepresentasikan kesalahan klasifikasi, dimana FN(*False Negative*) menunjukkan sampel dari suatu kelas yang salah diprediksi ke kelas lain, sedangkan FP(*False Positive*) menunjukkan sampel dari kelas lain yang salah diprediksi masuk ke suatu kelas [Grandini et al., 2020]. Sebagai contoh, $FN_{pos \rightarrow neg}$berarti sampel berlabel Positif yang salah diprediksi sebagai Negatif, sedangkan $FP_{neg \rightarrow pos}$berarti sampel berlabel Negatif yang salah diprediksi sebagai Positif [72].

Dari *confusion matrix* diturunkan nilai *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN) untuk setiap kelas, yang menjadi dasar perhitungan metrik evaluasi standar.

2.15.2 Metrik Evaluasi

Takahashi et al. (2022) dalam kajian mereka tentang metrik evaluasi untuk klasifikasi multikelas menegaskan bahwa empat metrik berikut merupakan standar evaluasi yang paling komprehensif dan luas digunakan dalam tugas klasifikasi sentimen [72]:

F1-Macro merupakan rata-rata dari nilai F1-Score pada setiap kelas tanpa memperhatikan jumlah sampel pada masing-masing kelas, sehingga setiap kelas memiliki kontribusi yang sama terhadap hasil akhir evaluasi:

$$\text{F1-Macro} = \frac{1}{K}\sum_{i=1}^{K} \text{F1}_\text{i} \tag{26}$$

Accuracy mengukur proporsi total prediksi yang benar dari keseluruhan prediksi:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{27}$$

Accuracy memberikan gambaran umum performa model namun dapat menyesatkan pada data tidak seimbang, karena model yang selalu memprediksi kelas mayoritas pun dapat memiliki akurasi tinggi.

Precision mengukur proporsi prediksi Positif yang benar-benar merupakan kelas Positif:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{28}$$

Precision tinggi menunjukkan bahwa model jarang menghasilkan prediksi Positif yang salah (*False Positive* rendah).

Recall (*Sensitivity*) mengukur proporsi sampel Positif aktual yang berhasil diidentifikasi model:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{29}$$

Recall tinggi menunjukkan bahwa model mampu mendeteksi sebagian besar sampel dari kelas Positif (*False Negative* rendah).

F1-*Score* adalah rata-rata harmonik antara *Precision* dan *Recall*:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{30}$$

F1-*Score* sangat berguna pada data tidak seimbang karena mempertimbangkan baik *False Positive* maupun *False Negative* dalam satu metrik. Dalam klasifikasi multikelas, metrik dihitung per kelas dan dirata-ratakan menggunakan *macro average* (bobot sama untuk setiap kelas) atau *weighted average* (bobot proporsional terhadap jumlah sampel per kelas) [73].

2.16 Python dan *Library* Pendukung

Python merupakan bahasa pemrograman tingkat tinggi yang bersifat *interpreted*, dinamis, dan multiparadigma yang banyak digunakan dalam pengembangan perangkat lunak, analisis data, serta penelitian di bidang kecerdasan buatan. Bahasa pemrograman ini dirancang dengan sintaks yang sederhana dan mudah dibaca sehingga memudahkan pengembang dalam menulis dan memahami kode program secara efisien. Selain itu, Python

memiliki ekosistem pustaka yang sangat luas serta dukungan komunitas yang aktif, sehingga menjadikannya salah satu bahasa pemrograman yang paling populer dalam penelitian *machine learning*, *data science*, dan *natural language processing* (NLP). Berbagai penelitian menunjukkan bahwa Python telah menjadi bahasa yang dominan digunakan dalam pengembangan sistem pemrosesan teks dan analisis data karena kemampuannya dalam menangani komputasi ilmiah serta kemudahan integrasi dengan berbagai *framework* kecerdasan buatan [74][11].

Python digunakan sebagai bahasa pemrograman utama untuk mengimplementasikan proses analisis sentimen terhadap kebijakan PPN 12% pada media sosial X. Python memungkinkan pengolahan data teks dalam jumlah besar secara efisien serta mendukung pengembangan model pembelajaran mesin dan pembelajaran mendalam yang digunakan dalam analisis sentimen. Melalui kemampuan pemrosesan data dan komputasi yang fleksibel, Python digunakan untuk mengelola seluruh tahapan penelitian mulai dari pengolahan data teks, pembentukan representasi data, pelatihan model klasifikasi sentimen, hingga evaluasi performa model yang dihasilkan. Berbagai penelitian di bidang *natural language processing* juga menunjukkan bahwa Python telah menjadi bahasa pemrograman yang dominan digunakan dalam pengembangan sistem analisis teks dan analisis sentimen karena kemudahan implementasi algoritma serta dukungan ekosistem ilmiah yang luas [75][76]. Oleh karena itu, penggunaan Python memungkinkan penerapan metode analisis sentimen secara efektif untuk mengidentifikasi dan mengklasifikasikan opini publik terhadap kebijakan PPN 12% yang berkembang di media sosial.

2.17 Penelitian Terdahulu

Tinjauan terhadap penelitian terdahulu dalam bagian ini difokuskan pada studi-studi yang secara langsung relevan dengan topik analisis sentimen kebijakan PPN 12% di Indonesia, disusun secara kronologis perkembangan metode dari pendekatan tradisional menuju pendekatan berbasis model pralatih bahasa Indonesia untuk memetakan secara eksplisit kesenjangan yang menjadi motivasi penelitian ini.

Hardyatman dan Hasan (2025) melakukan analisis sentimen terhadap opini publik mengenai rencana kenaikan PPN 12% di media sosial X menggunakan metode Decision Tree [7]. Dataset yang digunakan terdiri dari 1.000 *tweet* berbahasa Indonesia dengan pembagian 80% data latih dan 20% data uji, yang diperoleh melalui proses crawling dan preprocessing teks. Dari keseluruhan dataset, terdapat 715 data bersentimen Positif dan 285 data bersentimen Negatif, menunjukkan bahwa dalam periode pengumpulan data tersebut

opini yang mendukung rencana kenaikan PPN masih lebih banyak dengan dominasi sentimen Positif sebesar 71,5%. Hasil analisis menunjukkan bahwa metode Decision Tree berhasil mencapai akurasi sebesar 81,34%, mengungguli metode Naïve Bayes yang hanya mencapai 63,1% pada topik yang sama. Meskipun akurasi yang diperoleh tergolong memadai, metode Decision Tree memiliki keterbatasan mendasar dalam menangkap konteks semantik antar kata model ini bekerja berdasarkan aturan pemisahan fitur statis sehingga tidak mampu memahami nuansa linguistik, ironi, atau ambiguitas yang lazim muncul dalam teks media sosial informal berbahasa Indonesia.

Siagian dan Painem (2024) melaksanakan penelitian serupa dengan menerapkan metode Naïve Bayes Classifier pada data Twitter/X berbahasa Indonesia terkait rencana kenaikan PPN menjadi 12% [12]. Data dikumpulkan melalui proses crawling Twitter mulai tanggal 1 Maret hingga 15 Mei 2024, menghasilkan 468 *tweet* sebagai dataset. Proses analisis melibatkan pelabelan data, preprocessing, ekstraksi fitur dengan bag of words, klasifikasi dengan Naïve Bayes Classifier, dan pengujian dengan *confusion matrix*. Hasil pengujian menunjukkan akurasi sebesar 83%, presisi sebesar 68,8%, dan recall sebesar 78,6%. Meskipun akurasi yang dihasilkan cukup kompetitif, rendahnya nilai presisi sebesar 68,8% menunjukkan bahwa model ini rentan menghasilkan prediksi Positif yang salah (*false positive*), yang mengindikasikan keterbatasan Naïve Bayes dalam membedakan nuansa sentimen pada teks informal berbahasa Indonesia terutama kalimat yang mengandung negasi atau ekspresi idiomatis.

Hasan dan Bimby (2025) melakukan perbandingan komprehensif tiga algoritma machine learning Logistic Regression, Naïve Bayes, dan Support Vector Machine (SVM) dalam mengklasifikasikan sentimen publik terhadap kenaikan PPN di Indonesia [8]. Proses analisis melibatkan tahapan preprocessing, transformasi teks menggunakan TF-IDF, serta evaluasi menggunakan metrik akurasi dan F1-score. Hasil eksperimen menunjukkan bahwa SVM memberikan performa terbaik dengan akurasi sebesar 82,24%, diikuti oleh Logistic Regression sebesar 81,15%, dan Naïve Bayes sebesar 74,63%. Temuan ini menunjukkan bahwa SVM lebih efektif dalam membedakan sentimen Positif, Negatif, dan Netral pada teks media sosial. Meskipun demikian, ketiga algoritma tersebut secara fundamental bergantung pada representasi fitur statis TF-IDF yang tidak mampu menangkap makna kontekstual, sehingga kinerjanya terbatas pada teks informal bahasa Indonesia yang kaya akan singkatan, slang, dan percampuran bahasa.

Imawan, Shiddieq, dan Roji (2025) merupakan salah satu penelitian pertama yang menerapkan pendekatan deep learning berbasis transformer pada topik PPN 12%,

menggunakan model BERT (*pre-trained BERT Classification*) untuk analisis sentimen di platform X [13]. Hasil analisis menunjukkan dominasi sentimen Negatif sebesar 48,58%, yang mencerminkan kekhawatiran masyarakat terhadap potensi dampak kebijakan, diikuti oleh sentimen Netral sebesar 42,39%, dan sentimen Positif sebesar 9,03%. Model BERT yang digunakan berhasil mencapai akurasi sebesar 83%, dengan nilai precision, recall, dan F1-score rata-rata masing-masing sebesar 83%, 82%, dan 82%. Meskipun pendekatan ini melampaui seluruh metode tradisional, model BERT yang digunakan adalah varian standar berbahasa Inggris atau multilingual bukan model yang dilatih secara khusus pada korpus bahasa Indonesia sehingga representasi semantik untuk kosakata lokal, singkatan, dan ragam informal bahasa Indonesia cenderung kurang optimal.

Manoppo et al. (2025) menjadi penelitian yang secara spesifik menggunakan IndoBERT model pralatih berbasis BERT yang dilatih pada korpus monolingual bahasa Indonesia untuk analisis sentimen terhadap kebijakan kenaikan PPN 12% di Indonesia [[20]. Penelitian ini mengumpulkan 2.581 sampel data dari platform media sosial X, Instagram, dan TikTok, yang kemudian melalui tahapan pra-pemrosesan, tokenisasi, dan label mapping sebelum dibagi dengan rasio 80/10/10 menjadi set pelatihan, validasi, dan pengujian. Model IndoBERT dasar yang di-fine-tuned selama tiga epoch menunjukkan kinerja yang signifikan pada set pengujian, dengan akurasi sebesar 84,94%, precision sebesar 85,60%, recall sebesar 84,94%, dan F1-score (weighted) sebesar 84,37%. Analisis distribusi sentimen menunjukkan bahwa sentimen publik yang dominan adalah Negatif. Meskipun demikian, arsitektur yang digunakan hanya memanfaatkan lapisan encoder IndoBERT secara langsung untuk klasifikasi, tanpa mengintegrasikan komponen pemodelan sekuensial tambahan. Keterbatasan ini berpotensi mengurangi kemampuan model dalam menangkap pola dependensi urutan jangka panjang pada kalimat kompleks yang lazim muncul dalam diskusi kebijakan publik di media sosial.

Berdasarkan tinjauan terhadap kelima penelitian terdahulu di atas, teridentifikasi tiga lapis kesenjangan yang saling berkaitan. Pertama, penelitian yang menggunakan metode tradisional seperti Decision Tree [6], Naïve Bayes [12], dan kombinasi Logistic Regression, Naïve Bayes, serta SVM [8] mencapai akurasi di kisaran 74 - 83%, terbatas oleh ketidakmampuan menangkap konteks semantik teks informal. Kedua, penelitian yang menggunakan BERT standar [13] menghasilkan akurasi 83% namun menggunakan model yang tidak dioptimalkan untuk bahasa Indonesia sehingga representasi kosakata informal terbatas. Ketiga, penelitian yang menggunakan IndoBERT [20] menunjukkan peningkatan performa lebih lanjut dengan F1-score 84,37%, namun tidak mengintegrasikan arsitektur

sekuensial yang mampu menangkap pola urutan eksplisit. Ketiga hal inilah yang secara langsung memotivasi penelitian ini untuk mengombinasikan IndoBERT dengan BiLSTM secara spesifik untuk analisis sentimen kebijakan PPN 12% di platform X.

Tabel 2.8 Penelitian Terdahulu

No.	Penelitian	Metode	Akurasi / Hasil
1	Hardyatman & Hasan (2025)	Decision Tree	Akurasi: 81,34% | Sentimen Positif dominan: 71,5%
2	Siagian & Painem (2024)	Naïve Bayes	Akurasi: 83% | Presisi: 68,8% | Recall: 78,6%
3	Hasan & Bimby (2025)	Logistic Regression, Naïve Bayes, SVM	SVM: 82,24% | LR: 81,15% | NB: 74,63%
4	Imawan et al. (2025)	BERT (pre-trained)	Akurasi: 83% | Precision: 83% | Recall: 82% | F1: 82%
5	Manoppo et al. (2025)	IndoBERT fine-tuning	Accuracy: 84,94% | Precision: 85,60% | Recall: 84,94% | F1: 84,37%