BAB I
PENDAHULUAN

1.1 Latar Belakang

Rekayasa kebutuhan merupakan tahap fundamental dalam pengembangan perangkat lunak karena kualitas kebutuhan menentukan keberhasilan tahap desain, implementasi, dan pemeliharaan. Namun, kesalahan dalam mengidentifikasi dan mengklasifikasikan kebutuhan masih menjadi masalah klasik. Berbagai laporan industri menunjukkan bahwa kegagalan memahami kebutuhan, khususnya *Non-Functional Requirements* (NFR), menyumbang 60–80% masalah pada proyek perangkat lunak dan berdampak langsung pada biaya maupun kualitas sistem [1]. Bahkan dalam konteks *agile*, kelalaian terhadap NFR dapat meningkatkan risiko kegagalan proyek hingga 60% [2].

Sementara itu, proses klasifikasi kebutuhan secara manual masih bersifat lambat, mahal, dan rentan inkonsistensi antar-analis. Tantangan ini semakin besar karena NFR sering tersebar, implisit, dan ditulis dalam kalimat panjang yang ambigu [1]. Ketidakakuratan dalam mengklasifikasikan *Functional Requirements* (FR) dan NFR dapat berdampak fatal, seperti desain arsitektur yang salah arah, duplikasi fitur, pengujian yang tidak tepat, hingga kegagalan sistem.

Berbagai pendekatan *machine learning* telah diteliti untuk mengotomatiskan proses klasifikasi kebutuhan. Pendekatan tradisional berbasis TF-IDF, SVM, dan Naive Bayes menunjukkan performa yang cukup baik namun memiliki keterbatasan pada representasi fitur karena mengabaikan konteks linguistik dan struktur kalimat [1]. Kemunculan model berbasis *deep learning* seperti LSTM dan CNN meningkatkan akurasi, tetapi masih memiliki kelemahan pada pemahaman konteks yang lebih dalam [2].

Perkembangan terbaru menunjukkan bahwa model transformer seperti

BERT, RoBERTa, dan variannya memberikan kinerja unggul untuk klasifikasi kebutuhan karena kemampuannya memahami hubungan semantik dua-arah dalam bahasa alami. Studi internasional terbaru memperlihatkan bahwa BERT dan turunannya dapat mencapai akurasi hingga 95% dalam membedakan FR dan NFR pada bahasa Inggris dan Turki [3]. Model *hybrid* seperti BERT-BiCNN atau BERT-CNN juga terbukti meningkatkan performa klasifikasi [1].

Namun, hampir seluruh penelitian tersebut dilakukan pada bahasa Inggris, Turki, atau bahasa lain dengan sumber daya bahasa yang relatif kaya. Dalam konteks Bahasa Indonesia, penelitian NLP masih menghadapi keterbatasan dataset anotasi, kelangkaan *benchmark*, dan minimnya penelitian sistematis untuk model klasifikasi kebutuhan [4]. Meskipun telah tersedia IndoBERT sebagai model pra-latih Bahasa Indonesia, sebagian besar studi mengarah pada *dialog systems*, NER, atau *sentiment analysis*, bukan pada domain *Requirements Engineering* [5].

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, penelitian ini merumuskan masalah berikut:

1. Bagaimana membangun dataset klasifikasi kebutuhan perangkat lunak berbahasa Indonesia yang tervalidasi, terstruktur, dan dapat direplikasi, mengingat belum adanya dataset FR/NFR berbahasa Indonesia yang tersedia secara publik?
2. Bagaimana performa metode pembelajaran mesin klasik seperti *Support Vector Machine* (SVM) dan model *deep learning* seperti *Long Short-Term Memory* (LSTM) dalam melakukan klasifikasi *Functional Requirements* (FR) dan *Non-Functional Requirements* (NFR) pada data berbahasa Indonesia?
3. Bagaimana kinerja model transformer berbahasa Indonesia, khususnya IndoBERT dan DistilBERT, dibandingkan SVM dan LSTM dalam tugas

klasifikasi FR/NFR, serta model mana yang memberikan akurasi terbaik dalam konteks sumber daya linguistik Indonesia yang terbatas?

4. Faktor linguistik atau pola apa saja yang menyebabkan model tertentu (SVM, LSTM, IndoBERT, DistilBERT) mengalami kesalahan klasifikasi terhadap FR dan NFR pada Bahasa Indonesia?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah, penelitian ini memiliki tujuan sebagai berikut:

1. Menghasilkan dataset klasifikasi kebutuhan perangkat lunak berbahasa Indonesia yang tervalidasi, terstruktur, dan layak digunakan sebagai acuan penelitian lanjutan dalam domain klasifikasi FR dan NFR.
2. Mengevaluasi performa metode pembelajaran mesin klasik, khususnya *Support Vector Machine* (SVM), dalam mengklasifikasikan *Functional Requirements* (FR) dan *Non-Functional Requirements* (NFR) pada data berbahasa Indonesia.
3. Menganalisis kinerja model *deep learning* berbasis LSTM dalam mengolah karakteristik linguistik Bahasa Indonesia untuk tugas klasifikasi FR/NFR.
4. Mengevaluasi secara empiris efektivitas model transformer berbahasa Indonesia, terutama IndoBERT dan DistilBERT, dalam tugas klasifikasi FR/NFR, serta membandingkannya dengan pendekatan SVM dan LSTM.
5. Mengidentifikasi pola kesalahan klasifikasi yang muncul pada setiap model, meliputi aspek linguistik, struktur kalimat, serta karakteristik FR dan NFR yang menyebabkan model gagal membedakan kedua kategori.

1.4 Batasan Masalah

Penelitian ini dibatasi oleh ruang lingkup berikut:

1. Data yang digunakan hanya mencakup kebutuhan perangkat lunak berbahasa Indonesia, baik yang berasal dari dokumen publik, studi kasus akademik,

maupun sumber terbuka yang telah melalui proses seleksi dan validasi.

2. Klasifikasi yang dilakukan terbatas pada dua kategori, yaitu *Functional Requirements* (FR) dan *Non-Functional Requirements* (NFR), tanpa membedakan sub-kategori NFR secara lebih rinci.
3. Model pembelajaran yang dievaluasi dibatasi pada tiga pendekatan utama, yaitu:
   (a) Metode pembelajaran mesin (SVM).
   (b) Model *deep learning* berbasis LSTM.
   (c) Model transformer IndoBERT dan DistilBERT.
4. Eksplorasi *hyperparameter* dilakukan dalam batas wajar dan tidak mencakup *grid search* ekstensif.
5. Penelitian tidak membahas proses pengembangan sistem pendukung keputusan, melainkan berfokus pada evaluasi model klasifikasi dan analisis hasilnya.
6. Evaluasi performa terbatas pada metrik klasifikasi umum, seperti akurasi, presisi, *recall*, dan *F1-score*, tanpa mengevaluasi aspek efisiensi komputasi secara mendalam.

1.5 Kontribusi Penelitian

Penelitian ini memberikan kontribusi sebagai berikut:

1. Pembuatan dataset klasifikasi kebutuhan perangkat lunak berbahasa Indonesia yang tervalidasi, terstruktur, dan dapat direplikasi. Dataset ini menutup kekosongan sumber data FR/NFR berbahasa Indonesia yang sebelumnya tidak tersedia di literatur maupun repositori publik.
2. Penyusunan *baseline* model untuk klasifikasi FR dan NFR dalam Bahasa Indonesia, mencakup pendekatan pembelajaran mesin (SVM), *deep learning* (LSTM), serta dua model transformer berbahasa Indonesia (IndoBERT dan DistilBERT). *Baseline* ini memberikan landasan evaluasi bagi penelitian lanjutan di bidang *Requirements Engineering* berbasis

NLP.

3. Evaluasi komparatif yang sistematis terhadap tiga pendekatan klasifikasi (SVM, LSTM, transformer) pada dataset Bahasa Indonesia, sehingga menunjukkan model yang paling efektif dalam konteks sumber daya linguistik Indonesia yang terbatas.
4. Analisis mendalam terhadap pola kesalahan klasifikasi yang mencakup aspek linguistik, struktur kalimat, dan karakteristik FR/NFR. Analisis ini menghasilkan pemahaman baru tentang tantangan khusus Bahasa Indonesia dalam tugas klasifikasi kebutuhan, serta titik lemah dan kekuatan masing-masing model.
5. Penyediaan fondasi awal bagi pengembangan otomatisasi rekayasa kebutuhan berbahasa Indonesia, terutama pada tahap analisis awal dokumen kebutuhan, sehingga hasil penelitian dapat digunakan sebagai referensi akademik maupun aplikasi praktis di lingkungan industri.

BAB II
TINJAUAN PUSTAKA

2.1 State of the Art

Penelitian terkait klasifikasi kebutuhan perangkat lunak telah berkembang dalam empat fase besar: (1) pendekatan berbasis *Information Retrieval* (IR), (2) metode *Machine Learning* (ML) klasik, (3) arsitektur *Deep Learning* (DL), dan (4) model transformer.

2.1.1 Pendekatan IR dan Machine Learning Klasik

Pada tahap awal, klasifikasi FR dan NFR banyak mengandalkan pendekatan IR berbasis pencocokan kata. Metode ini efektif untuk kebutuhan sederhana, tetapi tidak mampu menangkap konteks kalimat yang lebih kompleks.

Perkembangan berikutnya adalah penggunaan metode ML seperti TF-IDF, Naïve Bayes, dan Support Vector Machine (SVM). Kaur dan Kaur menunjukkan bahwa pendekatan ini mendominasi penelitian klasifikasi kebutuhan sebelum munculnya model berbasis transformer [1]. Namun, metode ini bergantung pada fitur yang diekstraksi secara manual sehingga rentan kehilangan makna semantik [1].

2.1.2 Pendekatan Deep Learning

Kemunculan arsitektur DL seperti CNN dan LSTM meningkatkan kemampuan model dalam memahami konteks linguistik. Li dan Nong mengembangkan NFRNet (gabungan Bi-LSTM dan BERT) dan berhasil mencapai F1-Score 91% pada dataset yang diperluas [2]. Namun, performanya masih sangat dipengaruhi ukuran dataset dan struktur bahasa yang kompleks.

2.1.3 Model Transformer

Transformers mengubah lanskap NLP melalui mekanisme *self-attention* yang memungkinkan pemahaman konteks dua arah. Yucalar menunjukkan bahwa versi BERT khusus bahasa (misalnya BERTurk) jauh mengungguli model multilingual dalam tugas klasifikasi kebutuhan [3]. Kaur dan Kaur juga membuktikan bahwa arsitektur hibrida BERT-CNN mampu meningkatkan akurasi klasifikasi pada dataset PROMISE [1].

Dalam konteks Bahasa Indonesia, penelitian NLP masih berada pada kategori *low-resource*. Koto et al. menegaskan bahwa minimnya dataset anotasi menghambat kinerja model pra-latih [4], dan Di et al. menunjukkan bahwa model Indonesia perlu pelatihan tambahan karena perbedaan struktur morfologi [5].

2.1.4 Efektivitas Model Kecil dan ML Klasik

Studi komparatif terbaru oleh El-Hajjami et al. memperlihatkan bahwa model klasik seperti SVM tetap kompetitif terhadap LLM dalam klasifikasi kebutuhan, terutama ketika dataset kecil atau domainnya spesifik [6]. Temuan ini memperkuat relevansi penelitian yang membandingkan tiga generasi model sekaligus: SVM, arsitektur LSTM, dan model transformer Indonesia.

2.1.5 Ringkasan Penelitian Terdahulu

Tabel 2.1 Ringkasan Penelitian Terdahulu

No.	**Studi**	**Fokus**	**Metode**	**Hasil Utama**
1.	Yucalar et al. (2023) [3]	Dataset Turki	BERTurk vs ML/DL	Model spesifik bahasa unggul (F1 95%).

No.	Studi	Fokus	Metode	Hasil Utama
2.	Kaur & Kaur (2023) [1]	Fitur semantik	BERT-BiCNN	Akurasi meningkat dengan arsitektur hibrida.
3.	Li & Nong (2022) [2]	Ambiguitas NFR	NFRNet	F1 91% pada dataset diperluas.
4.	El-Hajjami et al. (2024) [6]	Efektivitas LLM	SVM vs GPT	Model klasik tetap kompetitif.
5.	Subahi (2023) [7]	Green IT NFR	Fine-tuned BERT	BERT efektif pada domain spesifik.

2.2 Kualitas Penulisan dan Sintesis Literatur

Literatur yang ada menunjukkan pola perkembangan yang jelas: setiap pendekatan lahir untuk mengatasi kelemahan pendekatan sebelumnya. Pendekatan ML klasik menawarkan efisiensi tetapi gagal memahami konteks. DL menawarkan pemrosesan sekuensial, namun tetap terbatas di bahasa dengan morfologi kompleks. Transformer hadir untuk menjawab kebutuhan pemahaman semantik yang lebih akurat.

Sintesis ini penting karena menggambarkan bahwa:

1. Penelitian sebelumnya tidak hanya berbeda dalam metode, tetapi dalam alasan teknis mengapa metode tersebut muncul;
2. Solusi linguistik untuk Bahasa Indonesia tidak bisa langsung mengadopsi hasil penelitian Inggris atau Turki;

3. Pemilihan model SVM, LSTM, dan DistilBERT dalam penelitian ini didasarkan pada fondasi perkembangan ilmiah, bukan arbitrer.

2.3 Research Gap

Berdasarkan tinjauan literatur, terdapat beberapa celah penelitian yang belum terisi:

1. **Belum tersedia dataset FR/NFR berbahasa Indonesia** yang terstandarisasi, sebagaimana dinyatakan dalam studi *low-resource* oleh Koto et al. [4].
2. **Tidak ada studi yang membandingkan SVM, LSTM, dan transformer khusus Indonesia** dalam klasifikasi FR/NFR.
3. **Minimnya analisis kesalahan berbasis linguistik Bahasa Indonesia**, padahal struktur afiks dan kalimat panjang sering menjadi sumber error.
4. **Belum ada baseline komparatif** yang dapat dijadikan acuan bagi penelitian lanjutan dalam Kebutuhan Perangkat Lunak berbasis NLP.

Research gap inilah yang menjadi dasar rasional penelitian ini.

2.4 Dasar Teori

2.4.1 Klasifikasi Kebutuhan

Kebutuhan perangkat lunak secara umum dibagi menjadi:

1. **Kebutuhan Fungsional (FR):** Mendefinisikan fungsi atau perilaku sistem.
2. **Kebutuhan Non-Fungsional (NFR):** Menjelaskan batasan kualitas operasi sistem seperti kinerja dan keamanan [7].

2.4.2 Algoritma Klasifikasi

2.4.2.1 Support Vector Machine (SVM)

SVM adalah algoritma *supervised learning* yang mencari *hyperplane* optimal sebagai pemisah antar kelas. Dikombinasikan dengan TF-IDF, SVM unggul dalam menangani teks berdimensi tinggi. Selain itu, SVM relatif stabil pada dataset kecil sehingga cocok sebagai *baseline* pada penelitian dengan

sumber data terbatas. Kemampuannya mengontrol margin menjadikan SVM tetap kompetitif dibandingkan model modern pada tugas klasifikasi teks yang sederhana namun padat informasi.

2.4.2.2 Bi-directional LSTM (Bi-LSTM)

Bi-LSTM memproses input dari dua arah sehingga mampu menangkap konteks panjang dalam kalimat [5]. Arsitektur dua arah ini membantu model memahami relasi kata yang saling bergantung, khususnya pada kalimat kebutuhan yang sering mengandung struktur kompleks. Meski begitu, Bi-LSTM tetap sensitif terhadap panjang sekuens dan membutuhkan pelatihan lebih lama dibandingkan metode berbasis fitur tradisional.

2.4.2.3 Transformer (DistilBERT)

DistilBERT adalah versi ringan dari BERT yang mengurangi ukuran model namun tetap mempertahankan performa tinggi melalui mekanisme *self-attention*. Mekanisme ini memungkinkan DistilBERT mempelajari hubungan semantik antar kata secara kontekstual, sebuah kemampuan yang sangat penting dalam membedakan FR dan NFR yang sering ambigu. Keunggulan efisiensinya menjadikan DistilBERT lebih mudah dilatih pada dataset kecil tanpa kehilangan kualitas representasi.

2.4.3 Metrik Evaluasi

Penelitian ini menggunakan F1-Score sebagai metrik utama:

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

(Rumus 2.1)

BAB III
METODE PENELITIAN

3.1 Alur Penelitian

Penelitian ini menggunakan pendekatan eksperimental komparatif untuk mengevaluasi kinerja tiga generasi model pembelajaran mesin dalam klasifikasi *Functional Requirements* (FR) dan *Non-Functional Requirements* (NFR). Proses penelitian meliputi konstruksi dataset, pra-pemrosesan adaptif, rekayasa fitur, pelatihan model, serta evaluasi menggunakan *Stratified 5-Fold Cross Validation*.

Diagram alir metodologi ditunjukkan pada Gambar 3.1.

Tahapan penelitian:

1. Konstruksi dataset mandiri.
2. Pra-pemrosesan data adaptif berdasarkan arsitektur model.
3. Rekayasa fitur: TF-IDF, trainable embedding, contextual embedding.
4. Pelatihan tiga model: SVM, Bi-LSTM, dan DistilBERT.
5. Evaluasi numerik dan analisis kesalahan.
6. Reproducibility dan kontrol variabel.

3.2 Konstruksi Dataset

3.2.1 Sumber Data

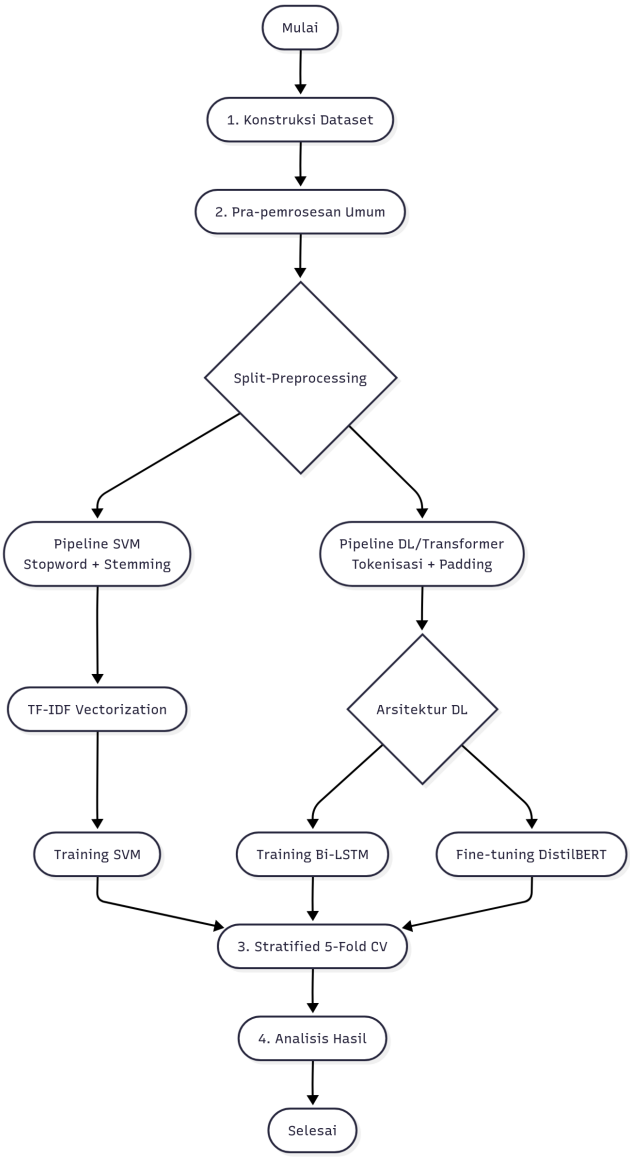
Dataset dikumpulkan dari:

1. Dokumen Spesifikasi Kebutuhan Perangkat Lunak (SKPL).
2. Laporan Tugas Akhir bidang Informatika.
3. Dokumen kebutuhan proyek perangkat lunak *open-source*.

3.2.2 Statistik Dataset

Dataset akhir berjumlah 550 kalimat, terdiri dari:

1. FR: 320 kalimat (58.18%)



Gambar 3.1 Diagram Alir Tahapan Penelitian

2. NFR: 230 kalimat (41.82%)
3. Total: 550 kalimat
4. Rata-rata panjang kalimat: 17 token
5. Panjang maksimum: 55 token

Distribusi ini mengikuti karakteristik alami dokumen kebutuhan di mana FR lebih dominan namun tidak terlalu timpang sehingga tetap cocok untuk model klasifikasi biner.

3.2.3 Protokol Anotasi

Setiap kalimat dianotasi oleh dua validator dengan pengalaman minimal 3 tahun di bidang RPL. Protokol anotasi meliputi:

1. Pelabelan manual berbasis pedoman ISO/IEC 29148.
2. Validasi silang pada 20% sampel dataset.
3. Reliabilitas inter-annotator diuji menggunakan Cohen's Kappa dengan nilai 0.72 (kategori *substantial agreement*).

3.3 Pra-pemrosesan Data

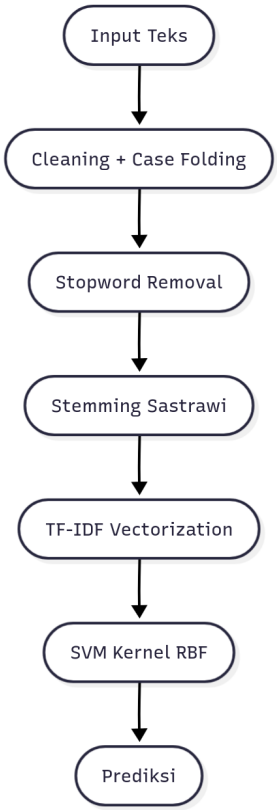
Penelitian menerapkan pendekatan **Split-Preprocessing** untuk menyesuaikan kebutuhan arsitektur model berbeda.

3.3.1 Pra-pemrosesan Umum

Langkah umum:

1. Pembersihan teks: penghapusan URL, tag HTML, karakter non-ASCII, dan simbol.
2. Case folding menjadi huruf kecil.
3. Normalisasi spasi dan tanda baca.

3.3.2 Pipeline A: SVM



Gambar 3.2 Pipeline Pra-pemrosesan dan Pelatihan SVM

Untuk SVM, digunakan pendekatan reduksi fitur agresif:

1. Stopword removal menggunakan Sastrawi.
2. Stemming Nazief–Adriani.
3. Tokenisasi berbasis whitespace.

3.3.3 Pipeline B: Bi-LSTM dan DistilBERT

1. Stopword removal dan stemming **tidak diterapkan** agar konteks semantik tidak hilang.

2. Tokenisasi:
    - **Bi-LSTM**: word-level tokenization.
    - **DistilBERT**: WordPiece tokenizer.
3. Padding dan truncation pada panjang tetap 128 token.

3.4 Representasi Fitur

3.4.1 TF-IDF (SVM)

Parameter TF-IDF:

1. `max_features` = 5000
2. `ngram_range` = (1,2)
3. Token pattern: `[a-zA-Z]+`

3.4.2 Trainable Embedding (Bi-LSTM)

Embedding dilatih dari awal:

1. Dimensi embedding: 100
2. Panjang sekuens: 128 token

3.4.3 Contextual Embedding (DistilBERT)

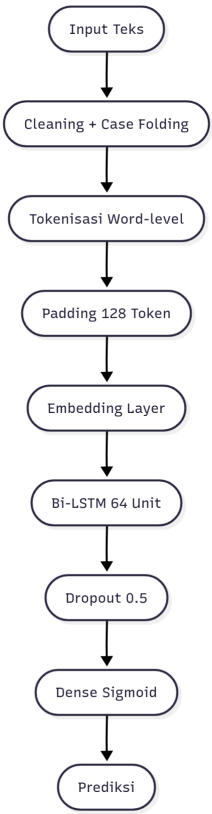
Menggunakan representasi vektor `[CLS]` dari hidden state terakhir DistilBERT.

3.5 Arsitektur dan Implementasi Model

3.5.1 Support Vector Machine (SVM)

1. Kernel: RBF
2. C = 1.0
3. Gamma = scale
4. class_weight = balanced

3.5.2 Bi-LSTM



Gambar 3.3 Arsitektur dan Proses Pelatihan Bi-LSTM

Arsitektur:

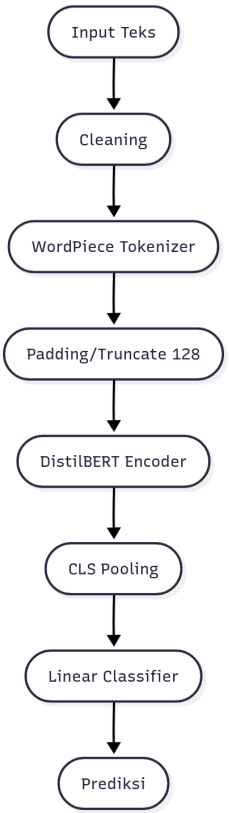
1. Embedding (100 dimensi)
2. Bi-LSTM (64 unit)
3. Dropout 0.5
4. Dense-Sigmoid

Hyperparameter:

1. Optimizer: Adam
2. Learning rate: 1e-3

3. Batch size: 32
4. Epoch: 10

3.5.3 DistilBERT



Gambar 3.4 Alur Fine-tuning DistilBERT

Konfigurasi:

1. Model: distilbert-base-multilingual-cased
2. Learning rate: 2e-5
3. Batch size: 16
4. Epoch: 3–5

5. Optimizer: AdamW
6. Gradient clipping: 1.0

3.6 Skenario Pengujian

3.6.1 Pembagian Data

1. Train/Test = 80:20
2. Stratified sampling
3. Random seed = 42

3.6.2 Stratified 5-Fold Cross Validation

Validasi performa dilakukan menggunakan stratified 5-fold.

3.7 Evaluasi dan Validasi

3.7.1 Confusion Matrix

Evaluasi dilakukan dengan TP, TN, FP, dan FN.

3.7.2 Metrik Performa

1. Metrik utama: F1-Score
2. Metrik tambahan: Precision dan Recall

3.8 Analisis Kesalahan

Analisis kesalahan dilakukan dengan mengamati sampel FN dan FP untuk:

1. Mengidentifikasi pola linguistik yang menyebabkan model gagal.
2. Menemukan jenis kalimat yang rawan salah (misal: negasi, kalimat bercabang, frasa implisit).
3. Membandingkan kelemahan tiap model berdasarkan pola error.

3.9 Reproducibility dan Kontrol Variabel

Untuk memastikan hasil dapat direplikasi:

1. Semua eksperimen dijalankan menggunakan seed = 42.
2. Preprocessing pipeline dikunci per model (A untuk SVM, B untuk

LSTM/BERT).

3. Seluruh model dilatih menggunakan dataset identik.
4. Pelatihan dilakukan pada runtime yang sama (Google Colab GPU).

3.10 Lingkungan Pengembangan

Eksperimen dijalankan pada:

1. Google Colab GPU (T4/V100)
2. Python 3.10
3. TensorFlow 2.15 dan PyTorch 2.1
4. HuggingFace Transformers, Scikit-learn, Sastrawi, NumPy, Pandas