BAB I
PENDAHULUAN

1.1 Latar Belakang

Persyaratan perangkat lunak merupakan fondasi yang menentukan kualitas dan kesuksesan proyek pengembangan perangkat lunak (Delima dkk., 2024). Rekayasa persyaratan perangkat lunak melibatkan proses identifikasi, analisis, dan pendokumentasian persyaratan fungsional dan non-fungsional secara sistematis (Fatima dkk., 2023). Proses klasifikasi manual persyaratan perangkat lunak memiliki kelemahan mendasar, yaitu memakan waktu yang lama dan rentan terhadap kesalahan, terutama pada proyek berskala besar dengan jumlah persyaratan yang sangat banyak (Canedo & Mendes, 2020). Dampak dari kesalahan persyaratan, yang salah satunya dipicu oleh klasifikasi yang tidak akurat ialah dapat menurunkan kualitas perangkat lunak (Naumcheva, 2021). Untuk mengatasi tantangan tersebut, klasifikasi otomatis dapat dimanfaatkan guna meningkatkan efisiensi serta akurasi proses pengolahan persyaratan perangkat lunak. Penerapannya dapat dilakukan dalam dua skenario, yaitu klasifikasi biner (fungsional dan non-fungsional), serta klasifikasi multikelas (11 sub non-fungsional). Pemilihan dua skenario ini didasarkan pada temuan Hughes dkk. (2025) yang menunjukkan bahwa model biner meskipun efisien karena kesederhanaan kategorinya, tetapi cenderung kehilangan informasi penting dan kurang sensitif terhadap kerumitan data dibandingkan model multikelas.

Dalam konteks klasifikasi persyaratan perangkat lunak, metode *machine learning* tradisional seperti Support Vector Machine (SVM), Naïve Bayes, dan Random Forest telah menunjukkan upaya yang lebih besar dalam menangani kompleksitas bahasa alami dan keragaman struktur teks persyaratan karena pendekatan ini bergantung pada rekayasa fitur secara manual (Saleem dkk., 2023). Keterbatasan tersebut mendorong eksplorasi pendekatan *deep learning,* termasuk arsitektur model berbasis transformer seperti DistilBERT yang memungkinkan ekstraksi fitur secara otomatis (Kici dkk., 2021) DistilBERT, menawarkan

keseimbangan optimal antara performa dan efisiensi komputasi, dengan mempertahankan 97% kemampuan pemahaman bahasa BERT namun dengan parameter 40% lebih sedikit dan kecepatan inferensi 60% lebih cepat (Sanh dkk., 2019). Namun, performa DistilBERT juga bergantung pada konfigurasi hyperparameter yang digunakan dan sering kali pengaturan default masih bersifat umum sehingga belum optimal pada data domain spesifik (Abadeer, 2020). Penelitian dari Yinkfu (2025) menunjukkan bahwa *fine-tuning* dengan optimasi konfigurasi hyperparameter yang tepat dapat meningkatkan performa DistilBERT. Sementara, penyesuaian yang keliru justru dapat menurunkan performa dan bahkan mengubah model yang semula unggul menjadi berkinerja buruk (Liu & Wang, 2021). Hal ini menegaskan perlunya strategi *fine-tuning* yang tepat melalui optimasi hyperparameter yang sistematis, dengan tetap membangun model *baseline* berbasis konfigurasi default sebagai titik acuan, sehingga dampak penyesuaian dapat dievaluasi secara adil, terukur, dan sesuai dengan karakteristik data persyaratan perangkat lunak. Dengan demikian, istilah optimasi dalam penelitian ini merujuk pada upaya sistematis untuk menemukan konfigurasi hyperparameter yang lebih sesuai dengan karakteristik data dibandingkan *baseline* default.

Salah satu pendekatan yang telah banyak digunakan untuk mengatasi tantangan pengaturan hyperparameter adalah menggunakan *Tree-structured Parzen Estimator* (TPE), sebuah metode Bayesian Optimization yang memanfaatkan informasi dari evaluasi sebelumnya untuk memprediksi konfigurasi berikutnya (Watanabe, 2023). TPE dinilai efektif dalam menghadapi ruang pencarian berdimensi tinggi dan interaksi kompleks antar hyperparameter (Ozaki dkk., 2022). Secara empiris, TPE terbukti lebih efisien dibandingkan Grid Search, Random Search, dan Simulated Annealing, baik dari segi jumlah iterasi maupun waktu komputasi (Arafa dkk., 2022). TPE juga menunjukkan kinerja unggul khususnya dalam tugas klasifikasi, mengungguli metode Random Search dan Genetic Algorithm (Dasgupta & Sen, 2024). Selain itu, TPE terbukti lebih kompetitif dibandingkan pendekatan Bayesian Optimization lainnya seperti Gaussian Process dan Spearmint, terutama dalam menangani parameter kondisional dan ruang pencarian berdimensi tinggi (Eggensperger dkk., 2013).

Penelitian ini menerapkan algoritma TPE untuk mengoptimalkan hyperparameter model DistilBERT secara sistematis dalam klasifikasi persyaratan perangkat lunak, guna mengatasi keterbatasan tuning manual dan mendorong pengembangan model klasifikasi yang lebih efisien dan andal. Secara khusus, penelitian ini bertujuan untuk mengevaluasi pengaruh penggunaan algoritma TPE terhadap model DistilBERT, ditinjau dari dua aspek utama, yaitu efektivitas peningkatan metrik performa secara keseluruhan dan pemerataan kualitas klasifikasi pada masing-masing kelas. Penilaian dilakukan tidak hanya berdasarkan akurasi dan nilai macro F1-score sebagai indikator agregat, tetapi juga melalui analisis precision, recall, dan F1-score per kelas untuk mengamati seberapa baik model mengenali tiap kategori secara proporsional. Seluruh proses evaluasi dilakukan pada dua skenario klasifikasi, yakni biner dan multikelas, untuk memastikan bahwa pengaruh optimasi hyperparameter TPE terhadap DistilBERT dapat diamati secara menyeluruh baik pada tingkat kompleksitas yang sederhana maupun yang lebih tinggi.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, maka masalah dirumuskan sebagai berikut:

1. Bagaimana pengaruh Tree-structured Parzen Estimator dalam meningkatkan efektivitas model DistilBERT pada level global untuk klasifikasi persyaratan perangkat lunak dibandingkan dengan model *baseline*, pada skenario klasifikasi biner dan multikelas, ditinjau dari metrik macro F1-score dan akurasi?
2. Bagaimana pengaruh Tree-structured Parzen Estimator terhadap distribusi performa klasifikasi model DistilBERT pada level kelas untuk klasifikasi persyaratan perangkat lunak dibandingkan dengan model *baseline*, pada skenario klasifikasi biner dan multikelas, ditinjau dari metrik precision, recall, dan F1-score?

1.3 Tujuan Penelitian

Adapun tujuan penelitian ini yakni sebagai berikut:

1. Menganalisis pengaruh Tree-structured Parzen Estimator dalam meningkatkan efektivitas model DistilBERT pada level global untuk klasifikasi persyaratan perangkat lunak dibandingkan dengan model *baseline*, pada skenario klasifikasi biner dan multikelas ditinjau dari metrik macro F1-score dan akurasi.
2. Menganalisis pengaruh Tree-structure Parzen Estimator terhadap distribusi performa klasifikasi per kelas pada model DistilBERT pada level kelas untuk klasifikasi persyaratan perangkat lunak dibandingkan dengan model *baseline*, pada skenario klasifikasi biner dan multikelas, dirinjau dari metrik precision, recall, dan F1-score.

1.4 Manfaat Penelitian

Manfaat yang diharapkan dari adanya penelitian ialah sebagai berikut.

1. Manfaat Teoritis:
   a. Memberikan kontribusi ilmiah terhadap optimasi model klasifikasi melalui teknik optimasi hyperparameter, khususnya dalam memahami bagaimana Tree-structured Parzen Estimator memengaruhi model DistilBERT.
   b. Menambah wawasan tentang evaluasi performa model secara menyeluruh, tidak hanya dari segi metrik agregat seperti akurasi dan macro F1-score, tetapi juga dari sisi distribusi performa antar kelas.
2. Manfaat Praktis:
   a. Memberikan referensi empiris bagi peneliti selanjutnya dalam mengevaluasi efektivitas Tree-structured Parzen Estimator sebagai strategi optimasi hyperparameter pada tugas klasifikasi teks, khususnya yang berkaitan dengan tantangan distribusi kelas tidak seimbang di domain rekayasa perangkat lunak.
   b. Mendukung pengembangan model klasifikasi persyaratan perangkat lunak yang lebih akurat dan adil, sehingga dapat meningkatkan efisiensi dan keandalan proses otomatisasi analisis persyaratan dalam pengembangan perangkat lunak.

1.5 Ruang Lingkup Penelitian

1. Dataset digunakan dalam penelitian hanya menggunakan PROMISE_exp.
2. Klasifikasi persyaratan perangkat lunak hanya tersedia untuk pernyataan berbahasa Inggris.
3. Teknik optimasi hyperparameter yang digunakan dalam penelitian ini hanya Tree-structured Parzen Estimator (TPE).
4. Metrik evaluasi yang digunakan hanya nilai dari *precision, recall, F1-score, accuracy,* dan *macro F1-score.*
5. Penelitian dilakukan pada platform Google Colab dengan runtime dan resource yang disediakan oleh sistem. Karena sifat Google Colab yang dinamis, hasil penelitian dapat memiliki sedikit perbedaan ketika dijalankan ulang pada waktu atau lingkungan berbeda, meskipun dengan konfigurasi yang sama.