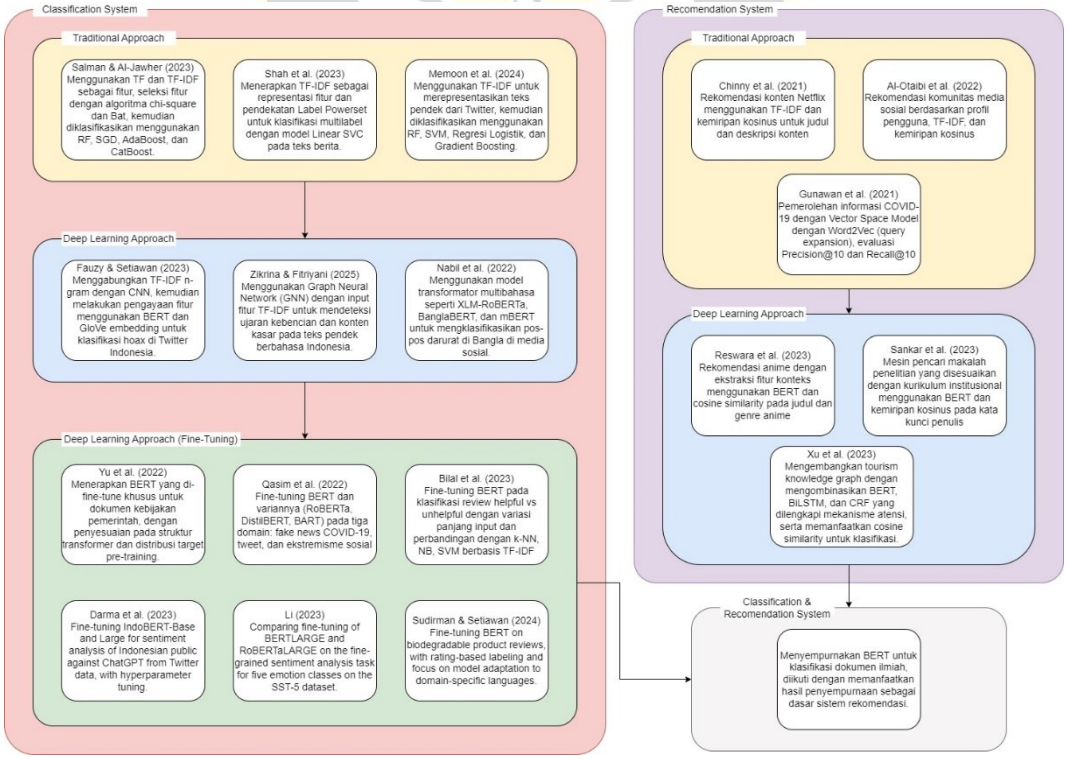
BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

Bab ini membahas tinjauan pustaka yang memuat hasil-hasil penelitian terdahulu terkait sistem klasifikasi dan rekomendasi berbasis teks, serta dasar teori yang menjadi landasan konseptual penelitian. Uraian meliputi perkembangan metode klasifikasi teks, pendekatan sistem rekomendasi, konsep-konsep utama dalam *Natural Language Processing* (NLP), hingga teori model BERT dan proses fine-tuning yang digunakan dalam penelitian ini.

2.1 Tinjauan Pustaka

Penelitian-penelitian sebelumnya mencakup pengembangan sistem klasifikasi dan sistem rekomendasi yang memanfaatkan pendekatan tradisional maupun *deep learning*. Beberapa studi juga melakukan *fine-tuning* terhadap model yang sudah ada sesuai dengan kebutuhan kasus tertentu. Diagram *state of the art* yang merangkum pendekatan-pendekatan tersebut dapat dilihat pada Gambar 2.1.



Gambar 2.1 Diagram *state of the art* dalam penelitian terkait

2.1.1 *State of the Art* Sistem Klasifikasi Teks

Perkembangan sistem klasifikasi teks meliputi tiga pendekatan utama, yaitu metode klasik berbasis *machine learning*, metode modern berbasis *deep learning*, dan metode *fine-tuning* model *transformer* yang menjadi standar terkini dalam pemrosesan bahasa alami.

1. Pendekatan Klasik Berbasis *Machine Learning*

Pada tahap awal, sistem klasifikasi teks banyak mengandalkan algoritma *machine learning* tradisional. Metode seperti *Naive Bayes*, *Support Vector Machine (SVM)*, *Logistic Regression*, dan *Random Forest* menjadi pilihan utama karena kemudahan implementasi serta performa yang cukup kompetitif pada dataset skala menengah. Representasi teks yang umum digunakan pada pendekatan ini adalah *Bag-of-Words* dan *Term Frequency–Inverse Document Frequency (TF-IDF)*, yang berfokus pada frekuensi kata tanpa memperhitungkan konteks. Rangkuman penelitian dengan pendekatan klasik berbasis *machine learning* dapat dilihat pada Tabel 2.1.

Tabel 2.1 *State of the art* klasifikasi klasik berbasis *machine learning*

Peneliti	Dataset	Metode	Hasil Utama
Salman & Al-Jawher (2023)	BBC News (2.225 dokumen, 5 kategori)	TF, TF-IDF + Chi-square + Bat Optimization + RF, SGD, AdaBoost, CatBoost	*Random Forest* terbaik: Akurasi 87,57%, F1 87,91%
Shah dkk. (2023)	60.000 artikel, 24 label (*multi-label*)	*Binary Relevance, Label Powerset, Classifier Chain* + *CountVectorizer*, TF-IDF + LR, NB, *Linear SVC*	*Label Powerset* + TF-IDF + *Linear SVC*: Akurasi 91%, F1 91%
Memoon dkk. (2024)	Twitter (687 entri, 20 kategori)	TF-IDF + SVM, NB, LR, RF, *Gradient Boosting*	*Random Forest* terbaik: Akurasi 92%, F1 93%

Metode klasifikasi dokumen *multi-class* telah dikembangkan dengan mengombinasikan teknik ekstraksi fitur TF dan TF-IDF kemudian seleksi fitur berbasis *chi-square* dengan algoritma *Bat* serta pengujian pada empat model

klasifikasi yaitu *Random Forest*, *Stochastic Gradient Descent*, *AdaBoost*, dan *CatBoost*. Pada dataset BBC *News* yang terdiri dari 2.225 dokumen dalam lima kategori, *Random Forest* menghasilkan performa terbaik dengan akurasi 87,57%, *precision* 88,58%, *recall* 87,56%, dan F1-score 87,91%. Hasil tersebut menunjukkan bahwa kombinasi metode seleksi fitur statistik dan optimasi metaheuristik mampu meningkatkan akurasi sekaligus mengurangi kompleksitas model serta risiko *overfitting* pada data berdimensi tinggi (Salman & Al-Jawher, 2023).

Studi pada klasifikasi *multi-label* kategori berita memanfaatkan pendekatan transformasi masalah seperti *Binary Relevance*, *Label Powerset*, dan *Classifier Chain* yang dikombinasikan dengan metode representasi fitur *Count Vectorizer* dan TF-IDF. Tiga model utama digunakan, yakni *Logistic Regression*, *Naive Bayes*, dan *Linear Support Vector Classifier*, dengan dataset berjumlah sekitar 60.000 artikel yang diklasifikasikan ke dalam 24 label. Hasil terbaik diperoleh dari kombinasi *Label Powerset*, TF-IDF, dan *Linear* SVC dengan akurasi 91%, *precision* 92%, dan F1-score 91%, menegaskan efektivitas *Label Powerset* dalam menangkap korelasi antar label serta keunggulan TF-IDF dalam representasi teks (Shah dkk., 2023).

Klasifikasi berita berbasis *Twitter* dilakukan dengan memanfaatkan dataset dari akun *Los Angeles Times* yang terdiri atas 687 entri pada 20 kategori. Setelah melalui proses *preprocessing* dan ekstraksi fitur menggunakan TF-IDF. Terdapat 5 model *machine learning* diuji yaitu SVM, *Naive Bayes*, *Logistic Regression*, *Random Forest*, dan *Gradient Boosting*. *Random Forest* mencatatkan performa tertinggi dengan akurasi 92% dan F1-score sebesar 93%, menunjukkan kemampuannya dalam menangani teks pendek dan tidak terstruktur. Kategori teknologi dan perjalanan bahkan mencapai F1-score sempurna yaitu 1,00 yang mencerminkan tingkat akurasi yang sangat tinggi dalam klasifikasi granular pada data media sosial (Memoon dkk., 2024).

2. Pendekatan Modern Berbasis *Deep Learning*

Seiring meningkatnya kompleksitas data teks, penelitian mulai beralih ke pendekatan berbasis *deep learning*. Model seperti *Convolutional Neural Network* (CNN) dan *Recurrent Neural Network* (RNN), termasuk variannya seperti *Long*

Short-Term Memory (LSTM) dan *Gated Recurrent Unit* (GRU), terbukti lebih unggul dalam menangkap pola sekuensial dan struktur semantik teks. Berbeda dengan metode klasik, pendekatan ini tidak hanya mengandalkan frekuensi kata, tetapi juga memanfaatkan representasi vektor kata (*word embedding*) seperti *Word2Vec* atau *GloVe* yang mampu menggambarkan hubungan semantik antar kata. Beberapa penelitian dengan pendekatan modern berbasis *deep learning* dirangkum pada Tabel 2.2.

Tabel 2.2 *State of the art* klasifikasi modern berbasis deep learning

Peneliti	Dataset	Metode	Hasil Utama
Fauzy & Setiawan (2023)	Twitter Indo (25.325 *tweet*, hoaks vs non-hoaks)	CNN + TF-IDF, *BERT embedding*, *GloVe*	CNN + TF-IDF bigram + *BERT* + *GloVe*: Akurasi 98,57%
Zikrina & Fitriyani (2025)	Twitter Indo (13.169 *tweet*)	*Graph Neural Network* (GNN) (TF-IDF) vs *RNN*	*GNN*: Akurasi 92,9% (abusif), 89,78% (*hate speech*), ungguli *RNN*
Nabil dkk. (2023)	Sosmed Bangla (5.839 postingan, 9 kategori darurat)	*Transformer* (XLM-R, BanglaBERT, mBERT)	XLM-R terbaik: F1 95,25%

Pendekatan klasifikasi untuk deteksi hoaks di Twitter berbahasa Indonesia menggabungkan *Convolutional Neural Network* (CNN) dengan berbagai representasi teks seperti TF-IDF *n-gram*, *embedding* BERT, dan *GloVe*. Dataset terdiri dari 25.325 *tweet* dalam dua kelas yaitu hoaks dan non-hoaks. Kombinasi CNN dengan TF-IDF *bigram*, BERT, dan *GloVe* menunjukkan hasil terbaik dengan akurasi 98,57%, jauh melebihi *baseline* berupa CNN yang dikombinasikan dengan TF-IDF yang hanya mencapai 94,15%. Hasil ini menunjukkan bahwa integrasi representasi semantik yang beragam dapat secara signifikan meningkatkan efektivitas deteksi hoaks di media sosial (Fauzy & Setiawan, 2023).

Klasifikasi ujaran kebencian dan konten abusif dalam bahasa Indonesia dilakukan dengan pendekatan *Graph Neural Network* menggunakan fitur dari TF-IDF. Dataset berjumlah 13.169 *tweet* yang dikategorikan secara manual ke dalam dua kelas. GNN mencatatkan akurasi 92,90% untuk konten abusif dan 89,78% untuk *hate speech*, dengan F1-score masing-masing 90,23% dan 86,94%. Hasil ini

mengungguli performa *Recurrent Neural Network* (RNN) yang hanya mencapai akurasi sekitar 86%, menunjukkan keunggulan GNN dalam memahami struktur dan konteks lokal pada teks pendek (Zikrina & Fitriyani, 2025).

Klasifikasi postingan darurat dalam bahasa Bangla pada media sosial diuji menggunakan model berbasis *transformer*. Dataset terdiri atas 5.839 postingan dari *Facebook* dan *Twitter* yang dikategorikan ke dalam sembilan jenis darurat. Model XLM-RoBERTa menghasilkan F1-score tertinggi sebesar 95,25%, melampaui BanglaBERT dengan skor 94,63% dan mBERT dengan skor 93,59%. Keunggulan model ini terletak pada kemampuannya dalam menangkap dependensi kontekstual dan mengatasi bahasa campuran sehingga relevan untuk sistem deteksi dini pada bahasa dengan sumber daya terbatas (Nabil dkk., 2023).

3. Pendekatan *Fine-Tuning* Model *Transformer*

Perkembangan terkini dalam klasifikasi teks ditandai dengan hadirnya arsitektur *transformer*, khususnya *Bidirectional Encoder Representations from Transformers* (BERT) dan turunannya. Model *transformer* memanfaatkan mekanisme *self-attention* yang memungkinkan pemahaman konteks kata secara dua arah sehingga menghasilkan representasi teks yang lebih akurat. Melalui proses *fine-tuning*, model yang telah dilatih sebelumnya pada data berskala besar dapat disesuaikan dengan dataset spesifik untuk tugas klasifikasi. Pendekatan ini terbukti melampaui performa metode klasik maupun *deep learning* konvensional dalam berbagai *benchmark*. Rangkuman penelitian yang menerapkan *fine-tuning* pada model *transformer* ditunjukkan pada Tabel 2.3.

Tabel 2.3 *State of the art* klasifikasi dengan *fine-tuning transformer*

Peneliti	Dataset	Metode	Hasil Utama
Yu dkk. (2022)	Dokumen kebijakan (10.000 dokumen)	*Fine-tuning* BERT vs *CNN,* BiLSTM	BERT: F1 93,25%
Qasim dkk. (2022)	Berita palsu COVID, *tweet* COVID, ekstremisme	RoBERTa, BART, BERT *fine-tuning*	Akurasi 98–99% (RoBERTa 99,71%, *BART* 98,83%)
Bilal & Almazroi (2022)	*Yelp review* (10.000 ulasan)	*Fine-tuning* BERT (*sequence length* 320)	Akurasi 70,7%, F1 71,7%

Peneliti	Dataset	Metode	Hasil Utama
Darma dkk. (2023)	*Tweet* Indonesia (6.408 *tweet*)	IndoBERT-Base & IndoBERT-Large	IndoBERT-Large: Akurasi 79%
J. Li (2023)	SST-5 (*fine-grained sentiment*)	RoBERTa-Large vs BERT-Large	RoBERTa: Akurasi 58,2% (unggul 3,7%)
Sudirman & Setiawan (2024)	Amazon review (949 ulasan)	*Fine-tuning* BERT	Akurasi 85,37%

Klasifikasi dokumen kebijakan dari sembilan instansi pemerintahan dilakukan pada 10.000 dokumen menggunakan pendekatan *fine-tuning* terhadap arsitektur BERT. Model BERT disesuaikan dengan karakteristik teks kebijakan yang cenderung pendek dan bersifat formal. Hasil menunjukkan bahwa BERT berhasil mencapai F1-score tertinggi sebesar 93,25% melampaui CNN dengan skor 88,32%. BiLSTM dengan skor 89,37% serta kombinasi CNN dan BiLSTM dengan skor 91,34%. Optimalisasi model dilakukan melalui pengurangan kedalaman *transformer* dan penyesuaian pada tugas *next sentence prediction* menjadi efektif untuk menangani teks kebijakan yang minim konteks namun mengandung informasi bermakna (Yu dkk., 2022).

Transfer learning berbasis *transformer* telah diuji dalam berbagai domain teks, termasuk berita palsu COVID-19, *tweet* COVID-19, dan ekstremisme. Model RoBERTa-base mencatat akurasi 99,71% pada dataset berita palsu, BART-large mencapai 98,83% pada *tweet* COVID-19 dan BERT-base mencatatkan akurasi 99,71% pada dataset ekstremisme. Ketiga model tersebut menunjukkan peningkatan akurasi signifikan hingga 20% dibandingkan metode klasik seperti CNN dan XGBoost, menandakan dominasi arsitektur *transformer* dalam pemahaman konteks yang kompleks (Qasim dkk., 2022).

Klasifikasi ulasan pengguna *Yelp* terhadap label *helpful* dan *unhelpful* dilakukan dengan *fine-tuning* BERT pada 10.000 data ulasan. Model terbaik menggunakan *sequence length* 320 menghasilkan akurasi 70,7% dan F1-score 71,7% mengungguli SVM yang hanya mencapai akurasi 67,9%. Ulasan yang dianggap *helpful* memiliki panjang rata-rata 186 kata sedangkan *unhelpful* hanya 99 kata menunjukkan bahwa panjang teks dan kedalaman semantik memengaruhi persepsi kegunaan sebuah ulasan (Bilal & Almazroi, 2022).

Analisis sentimen publik Indonesia terhadap ChatGPT dilakukan menggunakan IndoBERT-Base dan IndoBERT-Large pada 6.408 tweet dengan 5.000 *tweet* berlabel digunakan untuk pelatihan model. IndoBERT-Large mencatat akurasi tertinggi sebesar 79% diperoleh pada *training* selama 50 *epoch* dengan *learning rate* sebesar 2e-5. Hasil tersebut sedikit lebih tinggi dibandingkan IndoBERT-Base yang mencapai akurasi 78%. Pengujian pada 1.327 *tweet* tanpa label menunjukkan konsistensi prediksi yang memperkuat kemampuan generalisasi model IndoBERT terhadap sentimen dalam bahasa Indonesia (Darma dkk., 2023).

Klasifikasi emosi dalam tugas *fine-grained sentiment analysis* menggunakan model RoBERTa-Large pada dataset SST-5 menunjukkan hasil yang lebih tinggi dibandingkan BERT-Large. Model RoBERTa-Large mencatat akurasi sebesar 58,2% unggul 3,7% dibandingkan BERT-Large yang memperoleh akurasi 55,4%. Selisih performa tersebut berasal dari strategi *pre-training* yang telah dioptimalkan seperti penggunaan *dynamic masking, training* tanpa tugas *next sentence prediction*, dan penerapan *batch* berukuran besar yang terbukti meningkatkan kemampuan model dalam memahami emosi yang bersifat nuansial (J. Li, 2023).

Analisis sentimen terhadap produk kantong sampah *biodegradable* dilakukan melalui *fine-tuning* model BERT pada 949 ulasan dari platform *Amazon*. Model mencatat akurasi validasi sebesar 85,37% dengan distribusi sentimen terdiri atas 71% ulasan positif dan 29% ulasan negatif. Ditemukan sebanyak 125 ulasan dengan *rating* sempurna yaitu 5,0 namun tetap mengandung sentimen negatif yang menunjukkan pentingnya analisis sentimen yang mampu menangkap emosi campuran dalam satu entitas teks (Sudirman & Setiawan, 2024).

2.1.2 *State of the Art* Sistem Rekomendasi Teks

State of the art sistem rekomendasi teks meliputi metode tradisional yang berbasis kesamaan kata atau vektor, serta pendekatan representasi kontekstual seperti BERT yang mampu menghasilkan rekomendasi lebih relevan.

1. Rekomendasi Berbasis Metode Tradisional

Pendekatan tradisional pada sistem rekomendasi teks umumnya memanfaatkan teknik *information retrieval* klasik seperti *vector space model*, *term frequency–inverse document frequency* (TF-IDF), atau metode berbasis *content-based*

filtering. Beberapa penelitian juga menggabungkan metode statistik dengan algoritma *machine learning* untuk memperbaiki hasil pencarian. Namun, pendekatan ini masih terbatas karena hanya melihat kesamaan kata secara permukaan tanpa memperhatikan makna kontekstual dalam kalimat. Rangkuman penelitian yang menggunakan metode tradisional pada sistem rekomendasi teks ditunjukkan pada Tabel 2.4.

Tabel 2.4 *State of the art* sistem rekomendasi berbasis tradisional

Peneliti	Dataset	Metode	Hasil Utama
Chiny dkk. (2022)	Streaming (7.787 entri)	TF-IDF + *Cosine Similarity* (judul, deskripsi)	Skor *similarity* maksimum 0,2195, efisien meski fitur terbatas
Al-Otaibi dkk. (2022)	80 profil, 10 komunitas	TF-IDF + *Cosine Similarity* (profil & komunitas)	Akurasi 90%
Gunawan dkk. (2021)	18.596 artikel COVID-19	*Vector Space* Model (VSM) + *Word2Vec* (*query expansion*)	*Precision*@10 = 0,825, *Recall*@10 = 0,1952

Sistem rekomendasi konten pada *platform streaming* dikembangkan menggunakan algoritma TF-IDF dan *cosine similarity* dengan memanfaatkan 7.787 entri unik. Sistem rekomendasi tersebut menganalisis kemiripan antara judul dan deskripsi untuk menghasilkan saran konten dengan skor kemiripan tertinggi sebesar 0,2195 untuk konten seperti seri "NCIS". Meskipun pendekatan yang digunakan hanya memanfaatkan dua fitur teks yaitu judul dan deskripsi, metode ini terbukti efisien dan menunjukkan potensi untuk ditingkatkan melalui penambahan atribut seperti durasi, *rating*, aktor, dan demografi pengguna (Chiny dkk., 2022).

Rekomendasi komunitas di media sosial berbasis aplikasi *mobile* memanfaatkan algoritma TF-IDF dan *cosine similarity* untuk membandingkan profil pengguna dengan karakteristik komunitas. Pengujian terhadap 80 profil pengguna pada 10 komunitas menghasilkan akurasi hingga 90%, dengan tingkat kesalahan sebesar 10%. Sistem rekomendasi yang dikembangkan terbukti efektif dalam mengurangi kelebihan informasi serta mendukung pencarian komunitas yang relevan dengan mempertimbangkan interaksi pengguna, penggunaan bahasa *multi-language*, dan penerapan arsitektur modular 3 lapis (Al-Otaibi dkk., 2022).

Salah satu penelitian mengembangkan sistem pemerolehan informasi artikel terkait COVID-19 dengan mengombinasikan metode *Vector Space Model* dan *Word2Vec* untuk *query expansion*. Data yang digunakan sebanyak 18.596 artikel jurnal COVID-19. Evaluasi sistem dilakukan menggunakan metrik *Precision*@10 dan *Recall*@10 sebagai ukuran relevansi hasil pencarian. Hasil pengujian menunjukkan bahwa penggunaan *Word2Vec* pada tahap *query expansion* mampu meningkatkan performa sistem, dengan nilai *Precision*@10 mencapai 0,825 dan *Recall*@10 sebesar 0,1952, lebih tinggi dibandingkan sistem tanpa *query expansion*. Temuan ini memperkuat bahwa metrik relevansi seperti *Precision*@K dan *Recall*@K lazim digunakan dalam penelitian sistem temu kembali informasi maupun rekomendasi dokumen (Gunawan dkk., 2021).

2. Rekomendasi Berbasis Representasi Kontekstual (BERT dan Variannya)

Perkembangan terbaru menunjukkan penggunaan model *transformer* seperti *BERT* dalam sistem rekomendasi teks. Model ini mampu menghasilkan representasi semantik dua arah yang lebih akurat melalui mekanisme *self-attention*. Dengan melakukan *fine-tuning* pada dataset spesifik, *BERT* dan variannya dapat menangkap makna kontekstual dokumen secara lebih mendalam, sehingga sistem rekomendasi tidak hanya berbasis pada kesamaan kata, tetapi juga kesamaan makna. Pendekatan ini terbukti meningkatkan kualitas rekomendasi dibanding metode tradisional. Beberapa penelitian yang mengimplementasikan sistem rekomendasi berbasis *BERT* dirangkum pada Tabel 2.5.

Tabel 2.5 *State of the art* sistem rekomendasi berbasis BERT

Peneliti	Dataset	Metode	Hasil Utama
Reswara dkk. (2023)	17.562 anime	BERT (judul & genre) + *Cosine Similarity*	*Similarity* Naruto 0,994; unggul namun terbatas pada *franchise* besar
Xu dkk. (2023)	22.600 ulasan pariwisata	BERT + *BiLSTM* + *CRF* + *Attention* + *Cosine Similarity*	*Precision*@5 = 0,130; *Recall*@5 = 0,100
Sankar dkk. (2023)	4.112 dokumen (Math, CS, Eng)	BERT + *Cosine Similarity*	*Similarity* keyword 0,43 vs judul 0,37

Sistem rekomendasi *anime* dikembangkan menggunakan model BERT untuk mengekstraksi fitur kontekstual dari judul dan genre yang kemudian dibandingkan menggunakan *cosine similarity*. Pengujian pada 17.562 entri menunjukkan nilai *similarity* tertinggi sebesar 0,994 untuk *Naruto*, 0,991 untuk *Boruto*, dan 0,993 untuk *Dragon Ball Super*. Sistem rekomendasi berbasis BERT ini terbukti unggul dalam menyarankan *anime* yang serupa secara kontekstual. Namun, sistem tersebut masih memiliki keterbatasan dalam menangani waralaba besar (*franchise*) yang memiliki banyak seri karena judul dari waralaba yang sama cenderung mendominasi hasil rekomendasi (Reswara dkk., 2023).

Penelitian lain mengembangkan *tourism knowledge graph* menggunakan kombinasi BERT, BiLSTM, dan CRF dengan mekanisme atensi serta melakukan klasifikasi berbasis *cosine similarity*. Data yang digunakan sebanyak 22.600 ulasan pariwisata. Pada evaluasi ekstraksi entitas, model yang diusulkan berhasil mencapai *Precision*@5 sebesar 0,130 dan *Recall*@5 sebesar 0,100, melampaui *baseline* PCNN yang hanya memperoleh *Precision*@5 sebesar 0,118 dan *Recall*@5 sebesar 0,072. Hasil ini menunjukkan bahwa integrasi metode *deep learning* dan *similarity-based classification* dapat meningkatkan kualitas klasifikasi dan ekstraksi entitas dalam domain pariwisata (Xu dkk., 2023).

Pencarian makalah riset berdasarkan kurikulum dilakukan menggunakan sistem pencari berbasis BERT dan *cosine similarity* pada 4.112 dokumen di kategori Matematika, Teknik, dan Ilmu Komputer. *Input* berupa topik mata kuliah digunakan untuk melakukan *encoding* vektor kata kunci dan menghitung tingkat kesamaan dengan dokumen yang tersedia. Sistem pencarian berbasis BERT mampu menjawab seluruh topik tanpa mengalami kegagalan pencarian sementara *platform Scopus* tercatat mengalami hingga delapan kegagalan setiap mata kuliah. Skor *similarity* berdasarkan kata kunci mencapai 0,43 dibandingkan dengan skor 0,37 untuk pencocokan berdasarkan judul yang menegaskan efektivitas penggunaan kata kunci dalam sistem pencarian akademik (Sankar dkk., 2023).

Melihat berbagai pendekatan yang telah dikaji sebelumnya, terlihat bahwa model *transformer* seperti BERT dan variannya menunjukkan performa yang unggul dalam berbagai tugas klasifikasi teks, baik dalam konteks umum, domain

spesifik, maupun bahasa dengan sumber daya terbatas. Keunggulan tersebut tidak hanya terletak pada kemampuannya dalam memahami konteks semantik secara mendalam, tetapi juga pada fleksibilitasnya untuk disesuaikan melalui proses *fine-tuning* terhadap karakteristik data tertentu.

Berdasarkan temuan-temuan tersebut, penelitian ini akan mengembangkan sistem klasifikasi berbasis *fine-tuning* menggunakan model BERT dengan arsitektur *bert-base-uncased* yang diambil dari repositori *Hugging Face*. Penyesuaian dilakukan terhadap sejumlah *hyperparameter*, seperti jumlah *epoch*, *learning rate*, *batch size*, dan jenis *optimizer* yang digunakan. Dataset yang digunakan merupakan kumpulan dokumen karya ilmiah di bidang Ilmu Komputer dan Informatika yang kemudian dibagi menjadi tiga bagian yaitu data *training* untuk pelatihan model, data *validation* untuk memantau dan mengevaluasi performa selama pelatihan serta data *testing* untuk mengukur kemampuan generalisasi model terhadap data yang belum pernah dilihat sebelumnya. Model terbaik dipilih berdasarkan kombinasi parameter yang menghasilkan performa tertinggi pada data *validation*, dengan mempertimbangkan kestabilan nilai *loss*, akurasi dan F1-score antar *epoch* untuk memastikan bahwa model tidak mengalami *overfitting*. Model klasifikasi hasil *fine-tuning* tersebut selanjutnya akan digunakan dalam sistem rekomendasi berbasis konten yang memanfaatkan keluaran prediksi klasifikasi sebagai dasar pencocokan antar entitas. Proses rekomendasi dilakukan dengan menghitung kemiripan semantik antar representasi vektor dokumen menggunakan metode *cosine similarity* sehingga sistem mampu memberikan saran yang lebih relevan. Untuk mengukur kualitas saran yang diberikan, sistem rekomendasi dievaluasi berdasarkan tingkat relevansi, menggunakan metrik seperti *Precision*@K dan *Recall*@K untuk menilai kesesuaian antara hasil rekomendasi dan kategori dokumen yang diharapkan.

2.2 Dasar Teori

2.2.1 Dokumen Karya Ilmiah

Dokumen Karya ilmiah merupakan tulisan yang disusun secara sistematis berdasarkan kaidah ilmu pengetahuan dan prinsip-prinsip penulisan ilmiah. Tulisan ini berisi analisis, temuan, atau hasil penelitian yang diperoleh melalui studi

literatur, observasi langsung, atau pengalaman penulis. Selain itu, karya tulis ilmiah berkontribusi terhadap perkembangan serta kemajuan ilmu pengetahuan dan teknologi (Yanti dkk., 2024).

Karya ilmiah di bidang ilmu komputer umumnya diklasifikasikan ke dalam tiga kategori utama yaitu *Intelligent System*, *Embedded System*, dan *Enterprise System*. Proses kategorisasi dilakukan melalui analisis sistematis terhadap judul, abstrak, dan kata kunci dari setiap dokumen berdasarkan area riset yang umum digunakan. Karya yang membahas kecerdasan buatan seperti *machine learning* atau sistem otonom dikategorikan sebagai *Intelligent System* (Hristova dkk., 2022). Topik yang berkaitan dengan perangkat keras seperti *mikrokontroler*, *Internet of Things* (IoT), arsitektur jaringan, dan keamanan siber termasuk dalam *Embedded System* (Alhababi & Mulhim, 2024). Sementara itu, penelitian yang berfokus pada sistem informasi organisasi seperti *enterprise resource planning* (ERP) atau pemodelan proses bisnis dimasukkan ke dalam kategori *Enterprise System* (Mayasari dkk., 2023). Contoh dokumen karya ilmiah berdasarkan dari ketiga kategori utama tersebut dapat dilihat pada Tabel 2.6.

Tabel 2.6 Contoh karya ilmiah bidang ilmu komputer dan informatika

No	Judul Karya Ilmiah	Abstrak	Lab
1	Classification of Lombok Songket Based on GLCM and Invariant Moment Features and Linear Discriminant Analysis (LDA) Classifier	Songket is one of Indonesia's cultural heritage that is still present today. One of the most famous songket woven fabrics is the Lombok songket...	Intelligent System
2	Application of Rule Base Algorithm with Hexadecimal Approach on TransliteRation of Bima Script into Latin Letters	Aksara Bima is a text-Based inFormation exchange media that has never been lost. At this time the Bima script began to be studied again indicated, with researchers...	Intelligent System
3	Smart Plant Monitoring System Using Wireless Sensor Network and Evolutionary Fuzzy Association Rule Mining	Plant conditions monitoring requires specific knowledge in agriculture. The knowledge includes decision support to describe between good (ideal)...	Embedded System
4	Design of IoT and Android-Based Public	Public Street Lighting (PSL) is an element or component that	Embedded System

No	Judul Karya Ilmiah	Abstrak	Lab
	Street Lighting Monitoring and Controlling System	must be present on every road, functioning as street lighting at night for the safety of motorists...	
5	Analysis of Information Technology Governance to define the IT process in STIE 45 Mataram using COBIT 4,1 Framework	STIE 45 Mataram in the stage of implementing information technology in achieving the vision and mission that exists...	Enterprise System
6	Design of Electronic Service Information System for Engineering Study Program Informatics Faculty of Engineering, University of Mataram	Design of the PSTI Electronic Services Information System Faculty of Engineering, Mataram University, is a sophisticated System that supports the rheumatic process requested by students to...	Enterprise System

Contoh karya ilmiah pada Tabel 2.6 yang sering digunakan sebagai referensi meliputi artikel jurnal, makalah konferensi, tesis, disertasi, dan laporan penelitian akademik. Publikasi ilmiah tersebut tersimpan dalam repositori akademik yang dikelola oleh institusi pendidikan maupun platform publik. Keberadaannya menjadi sumber utama bagi mahasiswa dan peneliti dalam mengembangkan penelitian lebih lanjut.

2.2.2 *Natural Language Processing* (NLP)

Natural Language Processing (NLP) merupakan cabang dari kecerdasan buatan yang berfokus pada interaksi antara manusia dan komputer melalui bahasa alami. Teknologi NLP memungkinkan sistem untuk memahami, menganalisis, memanipulasi serta menghasilkan bahasa manusia baik dalam bentuk teks maupun suara. NLP terdiri atas beberapa komponen utama, yaitu sebagai berikut.

1. Pemrosesan Teks

Tahapan ini mencakup proses seperti *tokenizer*, *stemming*, *lemmatization* serta analisis sintaksis dan semantik. Melalui tahapan ini, sistem dapat memecah teks ke dalam satuan-satuan bahasa dan memahami struktur gramatikalnya secara menyeluruh.

2. Pemahaman Bahasa

Proses pemahaman bahasa memungkinkan komputer untuk menafsirkan makna suatu kata, frasa atau kalimat dalam konteks tertentu. Makna yang ditangkap

dapat bersifat eksplisit maupun implisit, tergantung pada struktur kalimat dan relasi antar kata.

3. Generasi Bahasa

Komputer dapat menghasilkan kalimat yang tersusun secara gramatikal dan bermakna sebagaimana ditunjukkan dalam aplikasi seperti *chatbot*, sistem penjawab otomatis, atau alat bantu penulisan.

4. Penerjemahan Bahasa

NLP telah mendukung penerjemahan otomatis dari satu bahasa ke bahasa lain dengan tetap mempertahankan konteks, makna, dan struktur kalimat agar hasil terjemahan lebih relevan dan mudah dipahami.

Dalam perkembangannya, NLP menggabungkan pendekatan statistik, pembelajaran mesin, dan pembelajaran mendalam. NLP telah diterapkan secara luas dalam berbagai aspek yaitu sebagai berikut.

1. Asisten Virtual dan *Chatbot*

NLP digunakan dalam perangkat seperti *Siri*, *Google Assistant*, dan *Alexa* untuk memahami perintah suara dan memberikan respons secara alami. *Chatbot* pada layanan pelanggan pun memanfaatkan NLP untuk merespons pertanyaan pengguna secara otomatis.

2. Mesin Pencari Informasi

Sistem seperti *Google* dan *Bing* menggunakan NLP untuk memahami maksud dari pencarian pengguna termasuk konteks dan niat di balik kata kunci kemudian menyajikan hasil yang relevan dan sesuai dengan kebutuhan informasi secara lebih akurat.

3. Analisis Sentimen

Pada media sosial dan aplikasi bisnis, NLP digunakan untuk menganalisis opini publik terhadap produk, isu, dan layanan tertentu dengan mengidentifikasi polaritas emosi seperti positif, negatif, dan netral dalam suatu pernyataan.

4. Otomatisasi Proses Bisnis

NLP membantu dalam klasifikasi *email*, pengambilan informasi dari dokumen serta pengisian formulir secara otomatis yang mempercepat proses kerja dan mengurangi beban administratif.

5. Sistem Pembelajaran Adaptif

Dalam bidang pendidikan, NLP memungkinkan sistem pembelajaran untuk menyesuaikan materi dengan gaya belajar, tingkat pemahaman, dan kebutuhan masing-masing siswa secara dinamis sehingga proses belajar menjadi lebih personal dan efektif.

6. Pelayanan Kesehatan

NLP digunakan untuk mengekstraksi informasi dari rekam medis seperti gejala, diagnosis, dan riwayat pasien serta membantu dalam analisis klinis dan pengambilan keputusan medis secara lebih cepat dan akurat.

7. Keamanan dan Penegakan Hukum

NLP telah diterapkan untuk mendeteksi ujaran kebencian, ancaman, dan aktivitas mencurigakan dalam komunikasi digital guna mendukung upaya pemantauan, pencegahan kejahatan siber, dan penegakan hukum secara proaktif.

Dengan cakupan aplikasi yang begitu luas, NLP berperan penting dalam mendukung efisiensi, otomatisasi serta pemahaman informasi dalam berbagai sektor berbasis teks (Widiantoro dkk., 2024).

2.2.3 Klasifikasi Teks

Klasifikasi adalah proses mengelompokkan atau memberikan label pada data atau objek baru berdasarkan sifat tertentu. Teknik klasifikasi melibatkan analisis variabel yang diperoleh dari data yang tersedia. Tujuan utama klasifikasi untuk melakukan prediksi berdasarkan kategori yang sesuai untuk item yang belum dikenali (Pebdika dkk., 2023). Proses klasifikasi dapat dilihat pada Gambar 2.2.

Gambar 2.2 Blok diagram model klasifikasi (Sari dkk., 2024)

Blok diagram model klasifikasi pada Gambar 2.2 menggambarkan tahapan dalam melakukan klasifikasi, dimulai dari pemetaan vektor fitur x ke salah satu label kelas y melalui pelatihan atau pembelajaran terhadap fungsi target f. Tahapan

klasifikasi melibatkan penggunaan *training set* yang terdiri dari sejumlah atribut baik berupa data kontinu maupun kategori. Salah satu atribut pada data *training* menunjukkan kelas atau label yang sesuai berdasarkan setiap rekaman data (Sari dkk., 2024).

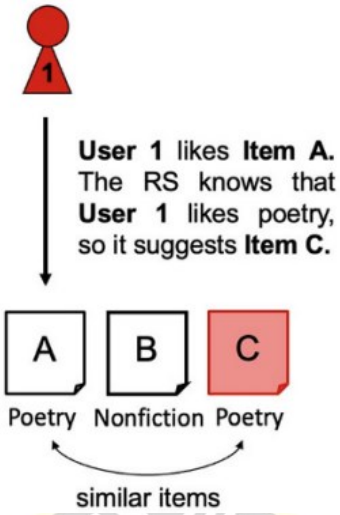
Dalam pengolahan teks, klasifikasi digunakan sebagai metode dalam pengolahan *Natural Language Processing*. Klasifikasi teks dirancang untuk menetapkan label atau mengelompokkan teks berdasarkan karakteristik tertentu. Proses klasifikasi teks mencakup pengelompokan data ke dalam kategori yang telah ditentukan serta memprediksi kategori untuk data baru yang belum teridentifikasi (Khoirunnisaa dkk., 2024).

2.2.4 Sistem Rekomendasi

Sistem rekomendasi adalah sebuah sistem yang dirancang untuk memberikan rekomendasi terhadap item tertentu guna membantu pengguna dalam mengambil keputusan (Oktavian dkk., 2023). Sistem rekomendasi memerlukan pendekatan yang tepat agar yang direkomendasikan sesuai dengan keinginan pengguna.

1. *Content-Based Filtering*

Pendekatan yang umum digunakan dalam pengembangan sistem rekomendasi adalah *content-based filtering* yang bekerja dengan menganalisis atribut atau fitur dari suatu item. Proses analisis dilakukan untuk mencocokkan karakteristik item dengan kebutuhan pengguna sehingga menghasilkan rekomendasi berdasarkan kesamaan fitur antar item. Pendekatan *content-based filtering* sering diterapkan karena relatif sederhana dan efektif dalam menghasilkan rekomendasi yang relevan berdasarkan karakteristik item (Zakharia dkk., 2024). Ilustrasi mengenai konsep *content-based filtering* dapat dilihat pada Gambar 2.3.



Gambar 2.3 Ilustrasi *content-based filtering* (Knees dkk., 2024)

Ilustrasi konsep dari *content-based filtering* pada Gambar 2.3 merupakan cara kerja dari sistem rekomendasi dengan memanfaatkan preferensi pengguna terhadap suatu item untuk memberikan rekomendasi. Sebagai contoh, Pengguna 1 menyukai Item A yang termasuk dalam kategori *Poetry*. Sistem merekomendasikan Item C yang memiliki kesamaan kategori dengan Item A untuk memastikan relevansi dengan preferensi pengguna.

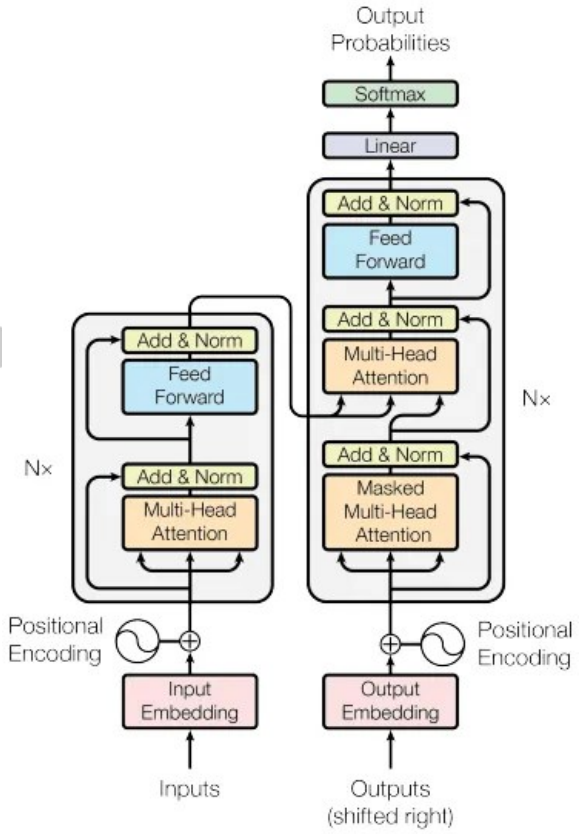
Salah satu contoh lain dalam penerapan *content-based filtering* adalah pada sistem rekomendasi film. Sistem menganalisis data dari film yang telah ditonton pengguna seperti genre, pemeran utama, sutradara, atau sinopsis. Apabila seorang pengguna menyukai film bergenre *action* yang dibintangi oleh aktor tertentu, maka sistem akan merekomendasikan film lain yang memiliki genre serupa atau dibintangi oleh aktor yang sama (Suryantara, 2024).

2.2.5 *Bidirectional Encoder Representations from Transformers* (BERT)

Bidirectional Encoder Representations from Transformers (BERT) merupakan model bahasa yang dikembangkan oleh *Google* pada tahun 2018 dan dirancang untuk memahami konteks kata dalam sebuah kalimat secara dua arah (*Bidirectional*). BERT dibangun berdasarkan arsitektur *transformer* yaitu model pembelajaran mesin yang dirancang untuk menangani data berurutan seperti teks. Model BERT memiliki sejumlah besar parameter yang dapat dilatih untuk berbagai tugas pemrosesan bahasa alami seperti klasifikasi teks, analisis sentimen, dan pemahaman kalimat dalam dokumen (Choudhary & Arora, 2024).

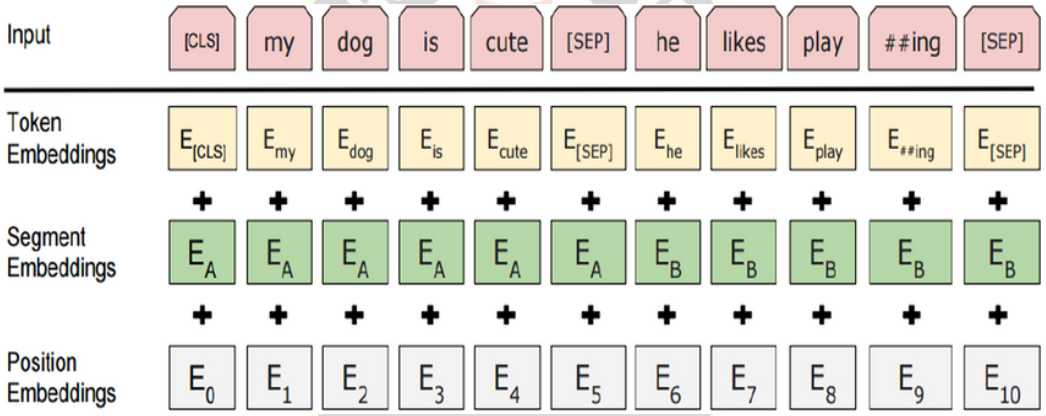
Salah satu keunggulan utama dari BERT adalah kemampuannya dalam menghasilkan representasi kata yang mempertimbangkan konteks kalimat secara menyeluruh. Untuk mencapai hal tersebut, BERT dilatih terlebih dahulu melalui proses *pre-training* menggunakan data teks dalam jumlah besar tanpa label. Pada tahap ini, BERT mempelajari bagaimana memprediksi kata yang hilang berdasarkan konteks sekitarnya sehingga model mampu membangun pemahaman yang lebih mendalam terhadap hubungan antar kata dalam satu kalimat.

Berbeda dengan pendekatan sebelumnya yang hanya memperhitungkan konteks dari satu arah, BERT menggabungkan informasi dari dua arah sekaligus. Pendekatan BERT memungkinkan memahami makna kata dengan mempertimbangkan keseluruhan kalimat. Kemampuan tersebut menjadikan BERT sangat efektif dalam menyelesaikan berbagai tugas NLP seperti sistem *question answering*, terjemahan bahasa, dan klasifikasi teks. Model BERT memanfaatkan struktur arsitektur *transformer* yang memiliki dua komponen utama yaitu *encoder* dan *decoder*. Arsitektur *transformer* dapat dilihat pada Gambar 2.4.



Gambar 2.4 *High-level transformer architecture* (Vaswani dkk., 2017)

Arsitektur *transformer* pada Gambar 2.4 bahwa model tersebut terdiri atas blok *encoder* dan *decoder* yang bekerja secara berurutan. Setiap blok memiliki mekanisme *multi-head attention* yang memungkinkan model memperhatikan beberapa bagian dari *input* secara simultan. Pendekatan tersebut membantu model memperoleh pemahaman yang lebih utuh terhadap konteks kalimat. Selain itu, *transformer* menggunakan *positional encoding* untuk menjaga informasi urutan kata dalam kalimat. *Positional encoding* mengkodekan posisi setiap *token* dalam bentuk vektor numerik, sehingga model dapat membedakan urutan kata dengan lebih akurat. Pada tahap awal pemrosesan, BERT mengonversi kalimat menjadi representasi vektor melalui beberapa tahap *embedding*. Representasi *input* BERT dapat dilihat pada Gambar 2.5.



Gambar 2.5 Representasi dari *input* BERT (Devlin dkk., 2019)

Representasi *input* BERT pada Gambar 2.5 dapat dilihat bahwa setiap *input* kalimat dimulai dengan *token* khusus [CLS] yang digunakan untuk tugas klasifikasi. Kalimat diakhiri dengan *token* [SEP] yang berfungsi sebagai pemisah antar kalimat atau antar segmen. BERT menambahkan *segment embedding* untuk membedakan kata yang berasal dari kalimat pertama dan kedua serta *positional embedding* untuk menandai posisi *token*. *Input* akhir ke dalam *encoder* BERT merupakan hasil penjumlahan dari ketiga komponen *embedding* yang terdiri dari *token embedding*, *segment embedding*, dan *positional embedding*.

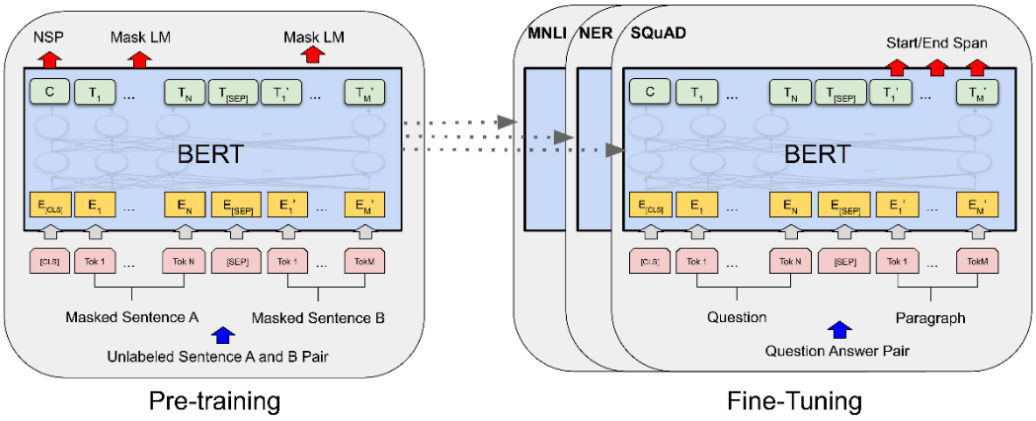
Setiap *token* dalam kalimat diproses melalui sejumlah *layer* dalam arsitektur BERT yaitu 12 *layer* pada versi BERT *base* dan 24 *layer* pada versi BERT *large*. Setiap *layer* menghasilkan vektor representasi yang mencerminkan makna

semantik dari *token* berdasarkan konteks global kalimat. BERT menggunakan *attention mechanism* dalam setiap *layer* untuk memprioritaskan kata-kata yang paling relevan dalam kalimat. Representasi akhir dari *token* tersebut dapat dimanfaatkan untuk berbagai tugas NLP seperti klasifikasi teks, analisis sentimen, dan pemahaman dokumen.

Proses pelatihan BERT dilakukan secara tidak terawasi (*unsupervised learning*) menggunakan korpus teks besar seperti Wikipedia dan koleksi buku digital. Pada tahap *pre-training*, BERT dilatih untuk menyelesaikan dua tugas utama yaitu *Masked Language Modeling* (MLM) dan *Next Sentence* Prediction (NSP).

Pada tugas MLM, sebagian kata dalam sebuah kalimat secara acak diganti dengan *token* [MASK] lalu model diminta untuk memprediksi kata asli berdasarkan konteks kiri dan kanan. Tugas ini memungkinkan BERT mempelajari hubungan semantik dua arah dalam kalimat. Sedangkan pada tugas NSP, model diberikan dua kalimat dan diminta untuk memprediksi apakah kalimat kedua merupakan lanjutan logis dari kalimat pertama. Tugas ini membantu model memahami keterkaitan antar kalimat dalam satu dokumen.

Setelah proses *pre-training* selesai, model BERT dapat digunakan untuk *fine-tuning* terhadap berbagai tugas NLP tertentu. Proses *fine-tuning* dilakukan dengan menggunakan data berlabel, di mana parameter BERT disesuaikan untuk mempelajari tugas-tugas spesifik seperti klasifikasi dokumen, analisis sentimen, atau ekstraksi informasi. Proses ini dikenal sebagai *transfer learning* karena memanfaatkan parameter hasil pelatihan umum untuk disesuaikan dengan tugas yang lebih spesifik. Skema *pre-training* dan *fine-tuning* BERT dapat dilihat pada Gambar 2.6.



Gambar 2.6 *Pre-training* dan *fine-tuning* BERT (Devlin dkk., 2019)

Gambar 2.6 menjelaskan proses pelatihan dua tahap pada BERT yang dimulai dengan *pre-training* menggunakan data tanpa label. Pada tahap ini, BERT dilatih untuk memahami struktur dan konteks bahasa melalui dua tugas utama, yaitu *Masked Language Modeling* (MLM) dan *Next Sentence Prediction* (NSP). Dalam MLM, beberapa *token* dalam kalimat A dan B disamarkan, dan model ditugaskan untuk memprediksi *token* yang hilang berdasarkan konteks sekitarnya. Sementara itu, NSP bertujuan melatih model dalam memahami hubungan antar kalimat dengan menilai apakah kalimat B merupakan kelanjutan logis dari kalimat A. *Input* yang digunakan terdiri dari *token* [CLS], kalimat A dan B yang dipisahkan oleh *token* [SEP], serta *embedding* posisi dan segmen yang semuanya diproses melalui arsitektur *transformer* BERT secara dua arah untuk menghasilkan representasi kontekstual yang mendalam.

Setelah *pre-training*, BERT dilanjutkan ke tahap *fine-tuning* dengan menggunakan data berlabel untuk menyelesaikan berbagai tugas pemrosesan bahasa alami. Beberapa contoh tugas tersebut meliputi klasifikasi hubungan antar kalimat seperti pada MNLI, penandaan entitas penting dalam teks atau NER, dan pencarian jawaban dari paragraf bacaan berdasarkan pertanyaan seperti pada SQuAD. Pada tahap ini, struktur utama BERT tetap dipertahankan dan hanya bagian *output* yang disesuaikan dengan masing-masing tugas. Meskipun tugas yang diselesaikan berbeda-beda, seluruh model *fine-tuned* memulai dari parameter yang sama hasil *pre-training*. Pendekatan dua tahap ini menjadikan BERT fleksibel, efisien, dan sangat efektif dalam berbagai aplikasi NLP (Asri dkk., 2025).

2.2.6 *bert-base-uncased*

Model BERT yang disediakan melalui *platform Hugging Face* hadir dalam dua varian utama yaitu *bert-base-cased* dan *bert-base-uncased*. Perbedaan mendasarnya terletak pada perlakuan terhadap huruf kapital dalam teks masukan. Varian *bert-base-cased* membedakan antara huruf besar dan kecil sehingga kata yang sama tetapi ditulis dengan kapitalisasi berbeda akan dianggap sebagai *token* yang berbeda. Sebaliknya, *bert-base-uncased* mengabaikan kapitalisasi dengan mengubah semua teks menjadi huruf kecil sebelum diproses. Hal ini membuat model *uncased* memiliki ukuran kosakata yang lebih kecil dan cenderung lebih sederhana namun tidak mempertahankan informasi yang mungkin terkandung dalam huruf kapital. Perbedaan dari kedua varian tersebut memengaruhi bagaimana model memahami struktur, makna, dan konteks dalam teks terutama ketika kapitalisasi berfungsi sebagai penanda penting seperti pada nama diri atau awal kalimat (Geetha & Renuka, 2021).

Sebagai contoh, dalam kalimat "*This research applies Convolutional Neural Network to image data*", model *bert-base-cased* akan mempertahankan huruf kapital pada "*Convolutional Neural Network*" sehingga lebih mudah mengenali istilah tersebut sebagai entitas atau istilah teknis. Sebaliknya, *bert-base-uncased* akan mengubahnya menjadi "*convolutional neural network*" dan mengandalkan konteks kalimat untuk memahami bahwa itu adalah nama metode dalam bidang pembelajaran mesin. Dengan demikian, perbedaan antara model *cased* dan *uncased* terutama terletak pada apakah informasi kapitalisasi dipertahankan atau diabaikan. Oleh karena itu, pemilihan antara model *cased* atau *uncased* perlu mempertimbangkan karakteristik data dan sejauh mana kapitalisasi berkontribusi terhadap pemaknaan istilah dalam teks.

2.2.7 *Fine-Tuning* BERT

Fine-tuning merupakan proses pelatihan ulang model BERT yang telah melalui tahap *pre-training* menggunakan data berskala besar, agar model tersebut dapat diadaptasi secara optimal terhadap tugas-tugas spesifik seperti klasifikasi teks, NER, dan analisis sentimen. Pada proses ini, model tidak dilatih dari awal, melainkan disesuaikan kembali bobot parameter internalnya menggunakan dataset

yang lebih kecil namun berlabel. Secara umum, proses *fine-tuning* BERT mencakup beberapa tahapan sebagai berikut.

1. *Pre-training* BERT

Model BERT awalnya dilatih pada korpus teks besar dan tidak berlabel seperti Wikipedia, dengan dua tugas utama yaitu *Masked Language Modeling* (MLM) dan *Next Sentence Prediction* (NSP). Tujuan dari *pre-training* adalah agar model memahami struktur dan makna bahasa secara umum sebelum diterapkan pada tugas tertentu.

2. Pemahaman Representasi Kata

Setelah tahap *pre-training*, model BERT memiliki kemampuan untuk membentuk representasi vektor dari setiap kata dalam sebuah kalimat berdasarkan konteks dua arah. Representasi ini disebut *embedding* dan digunakan sebagai masukan pada tugas-tugas lanjutan.

3. *Fine-Tuning* untuk Tugas Spesifik

Model BERT yang telah melalui tahap *pre-training* kemudian diberikan dataset berlabel sesuai dengan kebutuhan tugas seperti klasifikasi dokumen. Dataset tersebut berisi pasangan teks dan label kelas yang ingin diprediksi.

4. Pelabelan Data dan Pengaturan Parameter

Data *training* yang digunakan harus sudah memiliki label yang sesuai. Dalam tahap ini, parameter penting seperti *batch size*, *learning rate*, dan jumlah *epoch* disesuaikan agar model dapat beradaptasi secara efektif terhadap dataset.

5. Proses *Fine-Tuning*

Model BERT dilatih ulang dengan data berlabel, dan bobot-bobot model diperbarui menggunakan algoritma optimasi. *Fine-tuning* memungkinkan model untuk menyempurnakan kemampuannya dalam memahami konteks spesifik dari tugas tertentu.

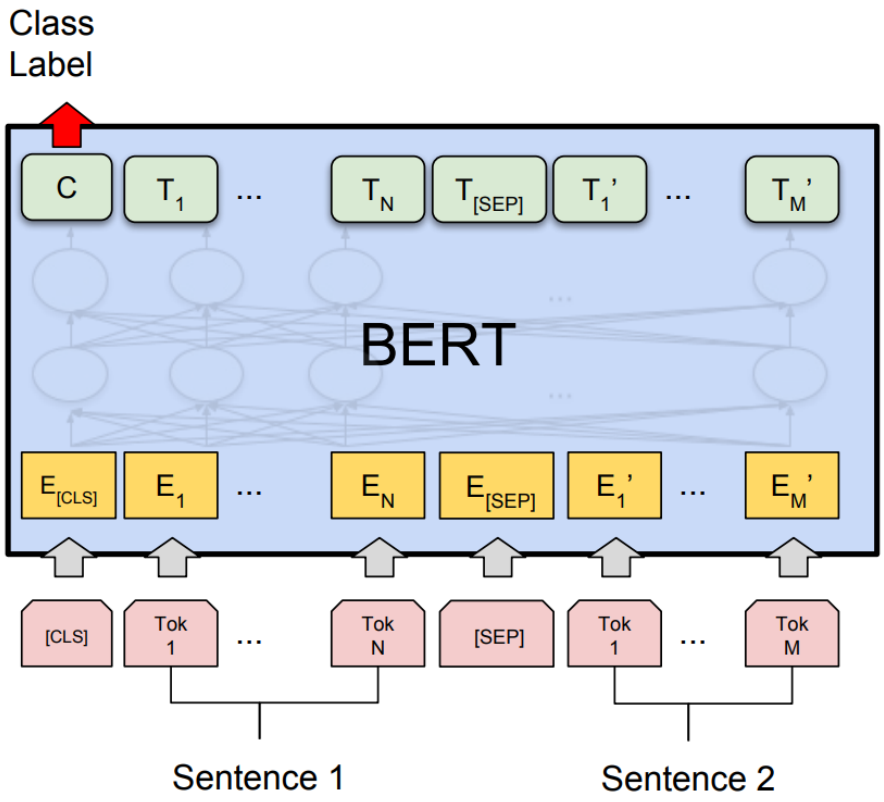
6. Evaluasi dan Penyetelan

Setelah proses pelatihan selesai, model diuji menggunakan data uji untuk mengetahui performanya. Jika hasilnya belum optimal, dilakukan penyesuaian parameter tambahan.

7. Penggunaan Model yang Disesuaikan

Model yang telah dilakukan *fine-tuning* dapat digunakan untuk menyelesaikan berbagai tugas NLP seperti klasifikasi teks, ekstraksi entitas, dan sistem rekomendasi berbasis teks.

Fine-tuning BERT dilakukan dengan menambahkan *classifier* pada bagian atas model BERT. Bagian tersebut bertugas mengkategorikan *output* dari *token* [CLS] menjadi kelas tertentu. Biasanya, hanya beberapa *layer* atas dari BERT yang dilatih ulang agar model tetap efisien dan tidak kehilangan pengetahuan umum yang telah dipelajari pada tahap *pre-training*. Selain itu, model dapat digunakan baik untuk *input* berupa kalimat tunggal maupun pasangan kalimat tergantung pada jenis tugas yang dihadapi. Skema proses *fine-tuning* BERT dapat dilihat pada Gambar 2.7.



Gambar 2.7 *Fine-tuning* BERT untuk klasifikasi (Devlin dkk., 2019)

Fine-tuning BERT untuk klasifikasi pada Gambar 2.7 merupakan proses penyesuaian model BERT agar sesuai dengan tugas klasifikasi, seperti analisis sentimen atau pengenalan hubungan antar kalimat. *Input* terdiri atas dua kalimat yang sudah melalui proses tokenisasi, dengan *token* khusus [CLS] di awal dan *token*

[SEP] sebagai pemisah antar kalimat. Setiap *token* dikonversi menjadi vektor *embedding*, lalu masuk ke dalam lapisan BERT. *Token* [CLS] menghasilkan vektor C yang berperan sebagai representasi gabungan dari seluruh *input*. Vektor ini diteruskan ke bagian akhir model untuk menghasilkan label kelas. Selama pelatihan, parameter BERT ikut disesuaikan berdasarkan data tugas klasifikasi yang digunakan, sehingga model dapat beradaptasi dan menghasilkan prediksi yang akurat (Asri dkk., 2024).

2.2.8 *Hyperparameter Tuning*

Hyperparameter tuning merupakan proses penyesuaian nilai-nilai parameter pelatihan untuk mengoptimalkan performa model selama *fine-tuning*. Fokus utama *tuning* adalah mencapai keseimbangan antara kecepatan konvergensi, generalisasi, dan stabilitas pelatihan. Beberapa *hyperparameter* utama yang umum disesuaikan dalam *fine-tuning* BERT yaitu sebagai berikut.

1. *Batch Size*

Ukuran *batch* yang umum digunakan adalah 16 dan 32. Nilai ini dianggap cukup ideal untuk menjaga efisiensi penggunaan memori GPU serta memberikan kestabilan dalam proses pelatihan. Pemilihan *batch size* dapat disesuaikan dengan kapasitas perangkat keras yang digunakan (Fatwanto dkk., 2024).

2. *Learning Rate*

Nilai *learning rate* yang kecil seperti 2e-5, 3e-5, dan 5e-5 sering digunakan untuk memastikan proses pembaruan bobot yang stabil dan akurat. *Learning rate* yang terlalu besar dapat menyebabkan pelatihan tidak konvergen, sedangkan nilai yang terlalu kecil memperlambat proses pelatihan secara signifikan (X. Li dkk., 2021).

3. *Epoch*

Jumlah *epoch* yang digunakan pada proses *fine-tuning* BERT biasanya berada dalam kisaran 3 hingga 5, karena dalam rentang ini model cenderung mencapai konvergensi yang stabil pada tugas-tugas klasifikasi (Yu dkk., 2022). Penggunaan *epoch* yang terlalu banyak dapat meningkatkan risiko *overfitting* atau memberikan peningkatan performa yang sangat kecil dibandingkan dengan beban komputasi tambahan (Prasanthi dkk., 2023).

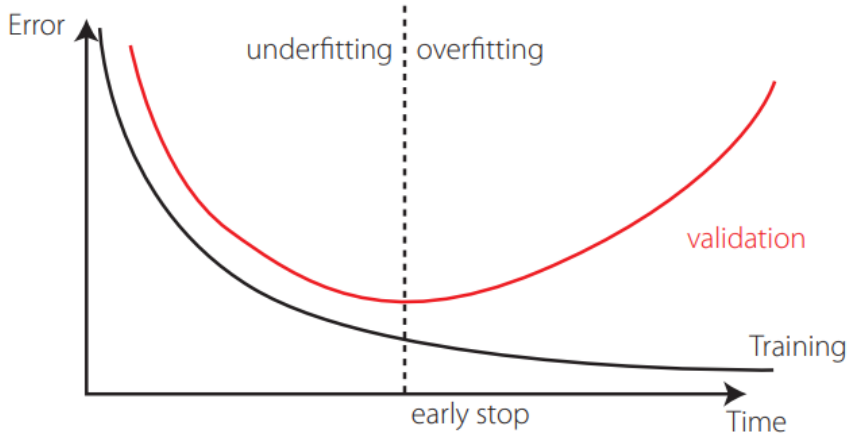
4. *Optimizer*

Dua jenis *optimizer* yang sering diterapkan dalam *fine-tuning* BERT adalah *Adam* dan *AdamW*. *Adam* merupakan pilihan standar dalam banyak arsitektur *deep learning*, sedangkan *AdamW* dikenal lebih efektif dalam mengatur regularisasi melalui mekanisme *weight decay* yang menjadikannya lebih sesuai untuk model *transformer* (Nabila & Setiawan, 2024).

Selain parameter tersebut, terdapat beberapa parameter tambahan yang dapat disesuaikan seperti *Max sequence length*, ukuran *embedding*, dan jumlah *layer* aktif yang dilatih. Pemilihan parameter yang tepat sangat bergantung pada karakteristik dataset dan kompleksitas tugas yang sedang dikerjakan. Oleh karena itu, *fine-tuning* BERT bukan hanya tentang melatih ulang model, tetapi telah melibatkan strategi evaluasi dan penyetelan parameter agar hasil akhir yang diperoleh relevan, akurat, dan dapat diterapkan (Asri dkk., 2024).

2.2.9 *Loss Function*

Fungsi *loss* digunakan untuk mengukur perbedaan antara hasil prediksi model dengan nilai sebenarnya. Nilai *loss* menjadi indikator utama dalam proses pelatihan karena digunakan oleh algoritma optimisasi untuk memperbarui bobot model. Dalam klasifikasi teks berbasis deep learning, fungsi *loss* yang umum dipakai adalah *cross-entropy loss* karena mampu mengukur perbedaan distribusi probabilitas kelas prediksi terhadap distribusi target (Terven dkk., 2025). *Underfitting* dan *overfitting* terhadap *error* selama proses pelatihan dapat dilihat pada Gambar 2.8.



Gambar 2.8 Kurva *underfitting* dan *overfitting* (Abambres dkk., 2019)

Kurva *error* pada Gambar 2.8 merupakan dinamika *error* pada data pelatihan dan data validasi selama proses pelatihan model. Pada awal pelatihan, *error* relatif tinggi karena model belum mampu mengenali pola, kondisi ini disebut *underfitting*. Seiring waktu, *error* pada data pelatihan terus menurun, sedangkan *error* pada data validasi menurun hingga titik tertentu lalu kembali meningkat. Peningkatan *error* pada data validasi menandakan terjadinya *overfitting*, yaitu kondisi ketika model terlalu menyesuaikan diri dengan data pelatihan dan kehilangan kemampuan generalisasi. Garis vertikal bertanda *early stop* menunjukkan titik optimal ketika proses pelatihan sebaiknya dihentikan, yaitu saat *error* validasi berada pada titik terendah sebelum meningkat kembali (Abambres dkk., 2019). Dengan demikian, pemantauan nilai *loss* pada data validasi berperan penting dalam menjaga keseimbangan antara ketepatan pada data latih dan kemampuan generalisasi pada data uji.

2.2.10 Evaluasi Kinerja

Evaluasi kinerja merupakan proses penting untuk mengukur seberapa baik model dalam melakukan prediksi atau memberikan rekomendasi yang sesuai dengan tujuan sistem. Dalam klasifikasi, evaluasi difokuskan pada ketepatan prediksi terhadap kelas data, sementara dalam sistem rekomendasi, penilaian dilakukan berdasarkan relevansi hasil terhadap kebutuhan pengguna. Berbagai metrik digunakan untuk menilai kinerja ini secara kuantitatif, seperti akurasi, *precision*, *recall*, F1-score serta metrik khusus seperti *cosine similarity*, *Precision*@K, dan *Recall*@K.

1. *Confusion Matrix*

Confusion Matrix merupakan tabel yang digunakan untuk mengevaluasi kinerja model klasifikasi dengan membandingkan hasil prediksi dengan nilai sebenarnya. Tabel ini terdiri dari empat kategori utama yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN) yang menunjukkan sejauh mana model mengklasifikasikan data dengan benar atau salah. Ilustrasi *Confusion Matrix* dapat dilihat pada Gambar 2.9.

		Kenyataan	
		Positif	Negatif
Prediksi	Positif	TP	FP
	Negatif	FN	TN

Gambar 2.9 *Confusion matrix* (Kurniawan, 2022)

Confusion Matrix pada Gambar 2.9 merupakan distribusi hasil prediksi model dalam klasifikasi. Prediksi yang sesuai dengan kenyataan terjadi pada TP saat model mengidentifikasi kelas positif dengan benar dan TN saat model mengenali kelas negatif dengan benar. Sementara itu, kesalahan klasifikasi terjadi pada FP ketika model salah mengklasifikasikan kelas negatif sebagai positif serta FN ketika model salah mengenali kelas positif sebagai negatif (Kurniawan, 2022). Adapun *Confusion Matrix* untuk *multi-class* dapat dilihat pada Gambar 2.10.

		Nilai prediksi			
		1	**2**	**3**	**4**
Nilai aktual	**1**	*A*	*B*	*C*	*D*
	2	*E*	*F*	*G*	*H*
	3	*I*	*J*	*K*	*L*
	4	*M*	*N*	*O*	*P*

Gambar 2.10 *Confusion matrix* untuk multi-class (Widodo, 2022)

Confusion matrix pada kasus *multi-class* dalam Gambar 2.10 memiliki pola perhitungan yang berbeda dibandingkan klasifikasi dua kelas. Pada klasifikasi *multi-class*, proses evaluasi dilakukan satu per satu untuk setiap kelas dengan cara menganggap satu kelas sebagai kelas yang sedang diuji dan semua kelas lainnya sebagai kelas pembanding.

Ketika evaluasi difokuskan pada kelas 1, nilai *True Positive* (TP) berada pada sel A karena mewakili jumlah prediksi yang benar untuk kelas tersebut. Pada saat fokus berada pada kelas 2, nilai TP berada di sel F. Hal yang sama berlaku untuk kelas 3 dan kelas 4 yang masing-masing memiliki nilai TP pada sel K dan P. Untuk menghitung nilai *True Negative* (TN), seluruh elemen matriks dihitung kecuali baris dan kolom yang berkaitan dengan kelas yang sedang dianalisis. Pada fokus kelas 1,

nilai TN terdiri atas sel F, G, H, J, K, L, N, O, dan P. *False Positive* (FP) mencakup jumlah prediksi dari kelas lain yang salah diklasifikasikan sebagai kelas 1, yaitu sel E, I, dan M. Sementara itu, *False Negative* (FN) mencerminkan kasus kelas 1 yang tidak berhasil dikenali dan diklasifikasikan ke dalam kelas lain, yaitu sel B, C, dan D. Dengan pendekatan ini, evaluasi terhadap model dapat dilakukan secara adil dan akurat untuk setiap kelas dalam sistem klasifikasi *multi-class* (Widodo, 2022).

2. Akurasi

Akurasi menggambarkan tingkat kedekatan antara nilai prediksi dengan nilai aktual. Semakin tinggi nilai akurasi, maka semakin baik performa sistem dalam melakukan prediksi (Susandri & Ratri, 2022). Perhitungan akurasi ditunjukkan pada persamaan 2.1.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

3. *Precision*

Precision digunakan untuk mengukur tingkat ketepatan antara informasi yang diminta pengguna dengan jawaban sistem (Susandri & Ratri, 2022). Perhitungan *Precision* ditunjukkan pada persamaan 2.2.

$$precision = \frac{TP}{TP + FP} \tag{2.2}$$

4. *Recall*

Recall atau *sensitivity* adalah metrik lain yang digunakan untuk mengevaluasi tingkat keberhasilan sistem dalam menemukan kembali informasi relevan (Susandri & Ratri, 2022). Perhitungan *recall* ditunjukkan pada persamaan 2.3.

$$recall = \frac{TP}{TP + FN} \tag{2.3}$$

5. F1-Score

F1-score adalah pengaruh relatif hasil kombinasi nilai *precision* dengan nilai *recall*. F1-score digunakan untuk mengukur kinerja dari sistem klasifikasi yang merupakan rata-rata harmonis dari *precision* dan *recall* (Susandri & Ratri, 2022). Perhitungan F1-score ditunjukkan pada persamaan 2.4.

$$F1\ Score = \frac{2\ (recall.precision)}{(recall + precision)} \tag{2.4}$$

6. *Cosine Similarity*

Cosine similarity adalah metode yang digunakan untuk mengukur tingkat kesamaan antara dua dokumen atau teks dalam konteks NLP. *Cosine similarity* mengukur sudut antara dua vektor dalam ruang vektor berdimensi tinggi untuk menentukan kemiripan dokumen berdasarkan vektor yang mewakilinya (Putra dkk., 2024). Perhitungan *cosine similarity* ditunjukkan pada persamaan.2.5.

$$Cosine\ Similarity = \frac{A.B}{||A||||B||} \tag{2.5}$$

dengan:

A dan B adalah dua vektor yang merepresentasikan dokumen atau teks

$||A||$ dan $||B||$ adalah magnitudo (panjang) dari masing-masing vektor

Hasil perhitungan *cosine similarity* berada dalam rentang 0 hingga 1 yang di mana nilai 1 menunjukkan kemiripan sempurna, sedangkan nilai 0 menunjukkan tidak adanya kesamaan (Lumbansiantar dkk., 2023). *Cosine similarity* digunakan baik dalam sistem pencarian maupun sistem rekomendasi. Dalam pencarian, metode ini mengukur kesamaan antara kueri pengguna dan dokumen untuk menampilkan hasil paling relevan.

Dalam penerapannya, *cosine similarity* umumnya disertai dengan penetapan ambang batas (*similarity threshold*) untuk menentukan apakah dua dokumen dianggap relevan atau tidak. Nilai *threshold* tidak bersifat universal, melainkan dapat bervariasi tergantung pada karakteristik data, konteks penelitian, serta tujuan analisis. Secara umum, *threshold* optimal berada pada rentang nilai menengah (sekitar 0,4 hingga 0,6) yang mampu menjaga *recall* tetap tinggi sekaligus mengurangi beban kerja secara signifikan. Oleh karena itu, pemilihan *threshold* terbaik dilakukan secara adaptif pada setiap studi agar tercapai keseimbangan antara ketepatan hasil dan efisiensi kerja (Wiwatthanasetthakarn dkk., 2024).

2.2.11 Konsep Relevansi dalam Sistem Rekomendasi

Dalam sistem rekomendasi, istilah relevansi merujuk pada tingkat kesesuaian antara item hasil pencarian atau rekomendasi dengan kebutuhan atau konteks pengguna. Relevansi tidak hanya ditentukan oleh kemiripan teks secara permukaan, tetapi juga oleh kesesuaian semantik maupun kategori terhadap konteks acuan.

Untuk mengukur relevansi, penelitian sebelumnya secara luas menggunakan metrik *Precision*@K dan *Recall*@K pada tugas *top*-N *recommendation* (Jayarana dkk., 2025).

1. *Precision*@K

Evaluasi terhadap akurasi sistem pencarian atau rekomendasi memerlukan metrik yang mampu mengukur relevansi hasil. Salah satu metrik yang umum digunakan adalah *Precision*@K yang menghitung proporsi item relevan di antara k hasil teratas yang ditampilkan kepada pengguna (KS & Shajan, 2024). Perhitungan *Precision*@K ditunjukkan pada persamaan 2.6.

$$Precision@k = \frac{Jumlah\ item\ relevan\ dalam\ top-k}{k} \tag{2.6}$$

dengan:

Jumlah item relevan dalam *top*-k adalah item relevan yang muncul di hasil teratas k adalah jumlah total hasil yang dievaluasi

Persamaan tersebut menggambarkan proporsi hasil yang benar-benar relevan dibandingkan dengan keseluruhan hasil hingga posisi ke-k. Semakin tinggi nilai *precision*, semakin besar konsentrasi item relevan pada posisi awal yang penting bagi efisiensi pengguna.

Untuk memberikan gambaran yang lebih menyeluruh, *precision* dapat dirata-ratakan pada beberapa nilai *cutoff* (Armacanqui dkk., 2024). Perhitungan ini dikenal sebagai *Mean Average Precision* (MAP) dan ditunjukkan pada persamaan 2.7.

$$MAP = \frac{\sum_{k=1}^{K} Precision_k}{K} \tag{2.7}$$

dengan:

K = jumlah *cutoff* yang dievaluasi (misalnya K = 5, 10, 20)

Precision k = nilai *precision* pada *cutoff* ke-k

2. *Recall*@K

Menilai performa sistem juga memerlukan pengukuran terhadap seberapa luas sistem dapat menjangkau item yang relevan. Untuk itu, *Recall*@K digunakan untuk mengetahui seberapa besar proporsi item relevan yang berhasil ditemukan dalam k hasil teratas (KS & Shajan, 2024). Perhitungan *Recall*@K ditunjukkan pada persamaan 2.8.

$$Recall@k = \frac{Jumlah\ item\ relevan\ dalam\ top-k}{Total\ item\ relevan\ yang\ tersedia} \quad (2.8)$$

dengan:

Jumlah item relevan dalam *top*-k adalah item relevan yang ditemukan

Total item relevan yang tersedia adalah keseluruhan item relevan yang ada

Melalui rumus tersebut, *recall* menyoroti kemampuan sistem dalam menemukan sebanyak mungkin item relevan dari seluruh yang tersedia. Semakin tinggi nilai *recall*, semakin baik sistem dalam menjangkau item relevan, terutama pada konteks di mana kelengkapan hasil lebih diutamakan daripada ketepatan.

Untuk memperoleh evaluasi yang lebih menyeluruh, *recall* dapat dirata-ratakan pada beberapa nilai *cutoff* (Armacanqui dkk., 2024). Perhitungan ini dikenal sebagai *Mean Average Recall* (MAR) dan ditunjukkan pada persamaan 2.9.

$$MAR = \frac{\sum_{k=1}^{K} Recall_k}{K} \quad (2.9)$$

dengan:

K = jumlah cutoff yang dievaluasi (misalnya K = 5, 10, 20)

Recall k = nilai *recall* pada *cutoff* ke-k

Perhitungan *Precision*@K dan *Recall*@K selalu didasarkan pada *top*-k hasil rekomendasi, bukan pada seluruh koleksi dokumen. Dengan demikian, *Precision*@K lebih menekankan pada ketepatan rekomendasi awal yang benar-benar diperhatikan pengguna, sedangkan *Recall*@K menggambarkan sejauh mana sistem mampu menjangkau dokumen relevan yang tersedia secara keseluruhan.