BAB I
PENDAHULUAN

1.1 Latar Belakang

Revolusi *Natural Language Processing* (NLP) telah mengubah cara *Artificial Intelligence* (AI) dalam memproses bahasa manusia secara digital. Di antara berbagai inovasi, *Neural Machine Translation* (NMT) telah menjadi fondasi penting dalam teknologi penerjemahan otomatis. Performa model NMT meningkat sejak diperkenalkannya mekanisme *attention* yang dikembangkan pada arsitektur *Transformer*. Arsitektur *Transformer* memungkinkan NMT untuk menangani teks panjang dan kompleks berkat penggunaan mekanisme *Self-Attention* yang lebih efisien dibanding *Recurrent Neural Network* (RNN) dan *Long Short-Term Memory* (LSTM) (Vaswani et al., 2017).

NMT telah mencapai performa tinggi pada *high-resource language pairs* seperti Inggris–Prancis atau Inggris–Jerman, tetapi penerapannya pada *low-resource languages* seperti Bahasa Indonesia masih menghadapi berbagai tantangan. Keterbatasan data *parallel corpus* yang berkualitas tinggi menyebabkan model sulit mempelajari pola penerjemahan yang akurat, terutama untuk struktur sintaksis dan semantik yang kompleks (Lakew et al., 2018). Tantangan ini semakin nyata ketika model NMT diterapkan pada *domain-specific* seperti ornitologi, di mana terdapat terminologi teknis dan deskripsi morfologi yang jarang muncul dalam korpus umum (Noll et al., 2023). Teks ornitologi mengandung istilah spesifik seperti *secondaries* (bulu sekunder) dan *superciliary line* (garis alis) yang memerlukan pemahaman konteks biologis untuk diterjemahkan dengan tepat (Roussis et al., 2024).

Berbagai arsitektur model NMT telah dikembangkan dengan pendekatan yang berbeda dalam menangani proses penerjemahan. Model bilingual seperti OPUS-MT dirancang khusus untuk pasangan bahasa tertentu dengan fokus pada efisiensi dan performa yang optimal untuk bahasa target tunggal (Tiedemann & Thottingal, 2020). Sementara itu, model *FLAN-T5* menerapkan pendekatan *text-to-text* yang fleksibel dan dikembangkan khusus melalui proses *instruction tuning*, yaitu pelatihan lanjutan di atas model T5 dengan ratusan *dataset* instruksi dan *task* ekspresif yang bervariasi. FLAN-T5 terbukti meningkatkan generalisasi dan kemampuan adaptasi pada perintah atau format instruksi baru berbeda dengan T5 standar yang hanya *pre-training* C4, FLAN-T5 mempelajari cara mengikuti instruksi spesifik, membuatnya sangat powerful untuk evaluasi *zero-shot, few-shot*, dan adaptasi domain khusus secara lebih presisi (Chung et al., 2022; Longpre et al., 2023). Perbedaan arsitektur dan strategi *pre-training* ini menghasilkan pendekatan serta performa yang berbeda dalam menghadapi penerjemahan *domain-specific*, namun belum ada studi yang secara sistematis membandingkan kedua pendekatan ini pada domain ornitologi untuk Bahasa Indonesia.

Pemilihan *hyperparameter* yang optimal menjadi faktor krusial dalam *fine-tuning* model NMT, terutama untuk *domain-specific translation* dengan data terbatas. *Learning rate* yang terlalu tinggi dapat menyebabkan model kehilangan pengetahuan dari *pre-training* (*catastrophic forgetting*), sementara *learning rate* yang terlalu rendah membuat proses adaptasi menjadi lambat (Mosbach et al., 2020). *Batch size* mempengaruhi stabilitas gradient dan kecepatan konvergensi, sedangkan

jumlah *epoch* menentukan seberapa banyak model dapat beradaptasi dengan domain target tanpa mengalami *overfitting* (Smith, 2018). Optimasi *hyperparameter* melalui *grid search* terbukti efektif dalam menemukan konfigurasi terbaik untuk meningkatkan performa model pada domain spesifik (Dodge et al., 2020).

Penelitian ini menggunakan dataset *CUB-200-2011* yang berisi deskripsi tekstual burung dalam bahasa Inggris, mencakup lebih dari 200 spesies dengan variasi deskripsi morfologi yang kaya (Wah et al., 2011). Dataset ini dipilih karena representatif untuk domain ornitologi dan telah banyak digunakan dalam penelitian NLP (Mahadi & Utama, 2023). Dataset diterjemahkan ke Bahasa Indonesia untuk menciptakan *parallel corpus* yang digunakan dalam *fine-tuning* dan evaluasi model.

Beberapa penelitian sebelumnya telah mengeksplorasi efektivitas model NMT berbasis *attention* dalam menerjemahkan *low-resource languages*. Penelitian yang dilakukan oleh (Smirnov et al., 2022) menunjukkan bahwa OPUS-MT (Helsinki-NLP) setelah dilakukan *fine-tuning*, terjadi peningkatan skor BLEU hingga +0.2 pada OPUS-MT, menunjukkan bahwa model ini dapat disesuaikan untuk domain spesifik. Selain itu, penelitian terbaru oleh (Chung et al., 2022) dan (Longpre et al., 2023) membuktikan bahwa FLAN-T5 setelah proses *instruction tuning* mampu meningkatkan hasil pada banyak *task zero-shot/few-shot* termasuk peningkatan signifikan pada penerjemahan *low-resource language* dan aplikasi di berbagai domain khusus. Namun, sampai saat ini belum ada penelitian yang menguji langsung efisiensi dan efektivitas FLAN-T5 pada dataset domain ornitologi untuk pasangan bahasa Inggris–Indonesia.

Berdasarkan hasil penelitian sebelumnya dan tantangan yang telah diuraikan, penelitian ini akan membandingkan performa dua model NMT berbasis *attention*, yaitu OPUS-MT dan FLAN-T5 dalam menerjemahkan teks ornitologi dari Bahasa Inggris ke Bahasa Indonesia. Selain mengimplementasikan model-model tersebut, penelitian ini juga akan menyempurnakan korpus data untuk meningkatkan pemahaman model terhadap istilah spesifik dalam domain ornitologi.

Beberapa penelitian sebelumnya telah mengeksplorasi efektivitas model NMT berbasis *Transformer* untuk *low-resource languages*. *Fine-tuning* model OPUS-MT terbukti meningkatkan performa pada domain spesifik (Smirnov et al., 2022). Sementara itu, studi (Chung et al., 2022) menunjukkan FLAN-T5 mampu mencapai peningkatan signifikan pada tugas penerjemahan Bahasa Indonesia dan domain khusus lewat *instruction tuning* serta kemampuan transfer antar *task* yang tinggi. Namun, penelitian-penelitian tersebut fokus pada korpus umum dan belum ada yang secara sistematis membandingkan kedua model ini pada domain yang sama, khususnya domain ornitologi untuk pasangan bahasa Inggris–Indonesia.

Berdasarkan gap penelitian tersebut, penelitian ini bertujuan untuk membandingkan performa dua model NMT berbasis *attention* OPUS-MT dan FLAN-T5 dalam menerjemahkan teks ornitologi dari Bahasa Inggris ke Bahasa Indonesia. Kedua model dipilih karena merepresentasikan dua pendekatan berbeda: OPUS-MT sebagai model bilingual yang efisien, FLAN-T5 sebagai model *text-to-text* yang fleksibel dan *instruction-tuned*. Penelitian ini akan melakukan *fine-tuning* pada kedua model menggunakan dataset *CUB-200-2011* yang telah diterjemahkan ke Bahasa Indonesia. Optimasi *hyperparameter* dilakukan menggunakan metode *grid search* untuk menemukan konfigurasi terbaik dari setiap model. Evaluasi performa dilakukan

menggunakan metrik BLEU dan METEOR untuk mengukur kualitas terjemahan secara kuantitatif. Hasil penelitian ini diharapkan dapat memberikan rekomendasi model dan konfigurasi *hyperparameter* yang optimal untuk penerjemahan teks *domain-specific* pada *low-resource language pairs*, khususnya dalam konteks teks ilmiah ornitologi.

1.2 Landasan Teori

1.2.1 Ornitologi

Ornitologi (*Ornithology*) adalah cabang ilmu zoologi yang secara khusus mengkaji tentang burung (*Aves*). Sebagai disiplin ilmu, Bidang ini mencakup berbagai aspek kehidupan burung, mulai dari evolusi, perilaku, ekologi, genetika, hingga struktur fisik dan adaptasi mereka (Gill, 2007). Studi ornitologi memberikan landasan penting bagi penelitian keanekaragaman hayati dan konservasi, karena burung sering berfungsi sebagai indikator kesehatan ekosistem.

Fokus utama dalam domain ini, khususnya yang relevan dengan data penelitian, adalah morfologi burung. Morfologi adalah studi mengenai bentuk, struktur, dan dimensi fisik burung, seperti bentuk paruh (*beak*), struktur kaki (*tarsus*), dan warna serta pola bulu (*plumage*), yang merupakan kunci dalam taksonomi dan identifikasi spesies (Pyle et al., 2008). Setiap deskripsi morfologi bersifat presisi dan memiliki implikasi biologis yang jelas. Sebagai contoh, istilah "... *superciliary* putih dan bulu coklat" atau "... memiliki kloaka berwarna putih" bukan sekadar kata sifat deskriptif, melainkan istilah standar yang harus diterjemahkan dengan padanan yang benar dan konsisten dalam Bahasa Indonesia agar makna ilmiahnya tidak hilang atau menjadi ambigu.

Kebutuhan akan konsistensi terminologi yang tinggi dalam literatur Ornitologi menghadirkan tantangan besar bagi penerjemahan otomatis. Mayoritas publikasi ilmiah tersedia dalam Bahasa Inggris, dan volume data deskriptif yang besar menuntut solusi penerjemahan yang tidak hanya cepat, tetapi juga mampu mempertahankan akurasi teknis. Tantangan inilah yang mendorong evaluasi mendalam terhadap kapabilitas model *Neural Machine Translation* (NMT) berbasis *Attention* untuk memproses domain spesifik ini.

1.2.2 Neural Machine Translation/NMT

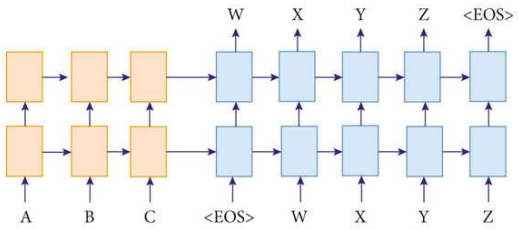
Neural Machine Translation (NMT) adalah pendekatan penerjemahan otomatis yang memanfaatkan jaringan saraf tiruan (*Artificial Neural Networks*) untuk mempelajari pemetaan langsung dari teks dalam bahasa sumber ke bahasa target. Berbeda dengan pendekatan sebelumnya yang memerlukan banyak komponen terpisah, NMT menggunakan arsitektur *end-to-end* yang dapat dilatih secara langsung pada data *parallel corpus* (Stahlberg, 2020). Pendekatan ini memungkinkan model untuk mempelajari representasi bahasa secara implisit dan menghasilkan terjemahan yang lebih natural dibandingkan dengan metode statistik tradisional.

Sebelum era NMT, bidang penerjemahan mesin didominasi oleh *Statistical Machine Translation* (*SMT*) yang menggunakan model *probabilistik berbasis frasa* dan *aturan linguistik eksplisit*. SMT memerlukan berbagai komponen terpisah seperti *language model*, *translation model*, dan *reordering model* yang masing-masing harus

dioptimasi secara independen (Koehn, 2020). Meskipun SMT berhasil mencapai performa yang baik pada beberapa pasangan bahasa, pendekatan ini memiliki keterbatasan dalam menangani *dependensi jarak jauh* dan *konteks kalimat* secara keseluruhan. Transisi dari SMT ke NMT terjadi secara signifikan pada pertengahan 2010-an ketika arsitektur *sequence-to-sequence* dengan *mekanisme attention* pertama kali diperkenalkan, menunjukkan peningkatan kualitas terjemahan yang substansial terutama pada bahasa dengan struktur sintaksis yang kompleks (Bahdanau et al., 2014).

Arsitektur dasar NMT terdiri dari dua komponen utama: *encoder* dan *decoder*, seperti yang ditunjukkan pada Gambar 1 *Encoder* bertugas membaca dan mengkodekan kalimat sumber menjadi *representasi vektor kontinu* yang menangkap *makna semantik* dari teks input, sementara *decoder* menghasilkan terjemahan dalam bahasa target secara bertahap berdasarkan representasi yang dihasilkan oleh *encoder*.

Pada awalnya, arsitektur NMT menggunakan *Recurrent Neural Network* (RNN) dan variannya seperti *Long Short-Term Memory* (LSTM) atau *Gated Recurrent Unit* (GRU) untuk memproses *sekuens* kata secara berurutan (Sutskever et al., 2014). Namun, arsitektur berbasis RNN memiliki keterbatasan dalam menangani *sekuens* yang panjang karena masalah *vanishing gradient* dan ketidakmampuannya untuk memproses data secara *paralel*, yang membuat proses *training* menjadi lambat dan sulit untuk di-*scale* (Vaswani et al., 2017).



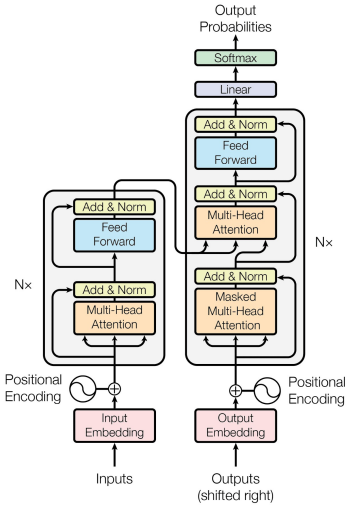
Gambar 1. Arsitektur Encoder-Decoder pada Neural Machine Translation
(Sumber: Diadaptasi dari Bahdanau et al., 2015)

1.2.3 *Attention Mechanism*

Terobosan signifikan dalam NMT terjadi dengan diperkenalkannya arsitektur Transformer pada tahun 2017 oleh Vaswani et al. Transformer menggantikan mekanisme rekursif pada RNN dengan mekanisme *Self-Attention* yang memungkinkan model untuk menangkap hubungan antar kata dalam kalimat tanpa memproses kata secara berurutan. Mekanisme *Self-Attention* menghitung bobot relevansi antara setiap pasangan kata dalam sekuens, memungkinkan model untuk fokus pada bagian-bagian penting dari input saat menghasilkan setiap kata dalam output (Vaswani et al., 2017).

Mekanisme Attention secara fundamental bekerja dengan memproyeksikan representasi input menjadi tiga vektor: *Query* (Q), *Key* (K), dan *Value* (V). Bobot *attention* dihitung dengan membandingkan *Query* token target dengan semua *Key* token sumber, biasanya menggunakan fungsi *dot product* dan *softmax*. Hasil bobot ini kemudian digunakan untuk melakukan rata-rata tertimbang (*weighted average*) pada vektor *Value* untuk menghasilkan vektor konteks yang berfokus pada informasi paling relevan, yang mengatasi masalah *bottleneck* konteks tunggal pada RNN (Vaswani et al., 2017).

Arsitektur lengkap Transformer dapat dilihat pada Gambar 2, yang menunjukkan struktur *encoder-decoder* dengan komponen-komponen utamanya. Pada sisi encoder, *input embedding* dikombinasikan dengan *positional encoding* untuk memberikan informasi posisi kata dalam sekuens. Setiap *encoder layer* terdiri dari *multi-head self-attention* yang memungkinkan setiap posisi untuk "melihat" semua posisi lain dalam input, diikuti oleh *feed-forward network* (Vaswani et al., 2017). Pada sisi decoder, selain *self-attention layer*, terdapat juga *encoder-decoder attention layer* yang memungkinkan decoder untuk fokus pada bagian relevan dari input saat menghasilkan setiap kata output.



Gambar 2. Arsitektur Transformer dengan komponen Encoder dan Decoder (Sumber: Vaswani et al., 2017)

Keunggulan utama Transformer dibandingkan arsitektur berbasis RNN terletak pada kemampuannya untuk memproses seluruh sekuens secara paralel, yang secara dramatis mempercepat proses *training* dan *inference*. *Parallelization* ini memungkinkan model Transformer untuk dilatih pada korpus data yang jauh lebih besar, menghasilkan representasi bahasa yang lebih kaya dan komprehensif (Popel & Bojar, 2018). Selain itu, mekanisme *attention* dalam Transformer mampu menangkap dependensi jarak jauh dengan lebih efektif karena setiap posisi dalam sekuens dapat langsung "melihat" semua posisi lainnya, tanpa melalui *multiple time steps* seperti pada RNN. Hal ini sangat penting dalam penerjemahan, di mana struktur gramatikal bahasa sumber dan target sering kali berbeda secara signifikan, memerlukan *reordering* kata dan pemahaman konteks global (Ott et al., 2019).

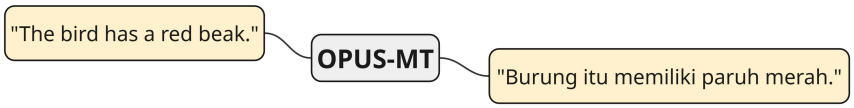
1.2.4 Model *Neural Machine Translation*

Perkembangan arsitektur Transformer telah melahirkan berbagai model *Neural Machine Translation* dengan pendekatan dan strategi pre-training yang berbeda. Dua model yang menonjol dalam kategori ini adalah OPUS-MT dan FLAN-T5, yang masing-masing merepresentasikan paradigma berbeda dalam menangani tugas penerjemahan mesin.

OPUS-MT dikembangkan sebagai model bilingual yang dioptimasi untuk pasangan bahasa spesifik (Tiedemann & Thottingal, 2020). Sementara itu, FLAN-T5 menerapkan *unified text-to-text framework* yang memperlakukan semua tugas NLP, termasuk penerjemahan, sebagai masalah generasi teks dengan format *input-output* yang konsisten (Raffel et al., 2019).

OPUS-MT (*Open Parallel Universal System for Machine Translation*) merupakan koleksi model *Neural Machine Translation* yang dikembangkan oleh *University of Helsinki* dan dilatih menggunakan korpus OPUS, sebuah *parallel corpus open-source* yang mencakup lebih dari 1000 pasangan bahasa. Model ini dibangun di atas arsitektur Transformer standar dengan *encoder-decoder* yang telah disesuaikan untuk efisiensi dan performa optimal pada pasangan bahasa spesifik (Tiedemann & Thottingal, 2020).

Secara internal, OPUS-MT dirancang untuk efisiensi komputasi dengan tetap mempertahankan struktur Transformer standar. Model ini mengadopsi mekanisme *Absolute Positional Embeddings* dengan pendekatan sinusoidal, sesuai arsitektur Transformer orisinal (Vaswani et al., 2017). Dalam pendekatan ini, informasi posisi ditambahkan ke *token embedding* menggunakan fungsi trigonometri (sinus dan cosinus) yang bersifat deterministik dan tetap (*fixed*). Vektor posisi ini kemudian dijumlahkan dengan *token embedding* sebelum masuk ke *encoder*. Mekanisme ini memungkinkan model untuk melakukan ekstrapolasi pada panjang sekuens yang belum pernah dilihat sebelumnya tanpa perlu menambah parameter yang harus dipelajari. Untuk menjaga stabilitas bobot, OPUS-MT menerapkan *Standard Layer Normalization* yang diposisikan setelah operasi residual (*Post-Norm*), serta menggunakan fungsi aktivasi yang ringan seperti *Swish* atau *ReLU*. Kombinasi arsitektur yang relatif ringan (72 juta parameter) dengan spesialisasi bilingual membuat OPUS-MT sangat efisien dalam hal *computational cost* dan *memory footprint*.

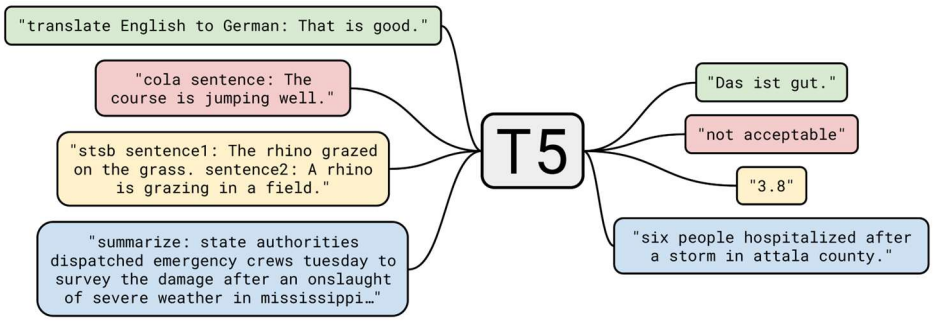


Gambar 3. Skema Penerjemahan Langsung pada Model OPUS-MT

Keunggulan utama OPUS-MT terletak pada pendekatan bilingualnya, di mana setiap model dilatih secara khusus untuk satu arah penerjemahan tertentu, misalnya Inggris ke Indonesia atau sebaliknya. Pendekatan ini memungkinkan model untuk fokus mempelajari karakteristik linguistik spesifik dari kedua bahasa tanpa *interference* dari bahasa lain, menghasilkan performa yang lebih tinggi dibandingkan model *multilingual generic* pada pasangan bahasa yang sama (Junczys-Dowmunt et al., 2016). Model OPUS-MT untuk pasangan Inggris-Indonesia dilatih menggunakan kombinasi korpus parallel dari berbagai domain termasuk, dokumen pemerintah, dan teks web, dengan total jutaan pasangan kalimat. Arsitektur yang relatif membuat OPUS-MT efisien dalam

hal *computational cost dan memory footprint*, yang penting untuk *deployment* praktis (Aulamo et al., 2020).

Berbeda dengan OPUS-MT, FLAN-T5 tidak menggunakan vektor posisi terpisah yang dijumlahkan dengan *token embedding*. Sebagai gantinya, FLAN-T5 menerapkan *Relative Positional Bias* yang diinjeksikan langsung ke dalam perhitungan *attention scores*. Mekanisme ini menambahkan nilai bias yang dipelajari (*learned scalar bias*) berdasarkan jarak relatif antar token. Sebagai contoh, jika token A berada pada jarak +1 dari token B (tepat setelahnya), atau pada jarak -2 (dua posisi sebelumnya), maka bias yang berbeda akan ditambahkan ke *attention score* sesuai dengan jarak relatif tersebut. Dengan demikian, model dapat menangkap hubungan posisional tanpa terikat pada panjang sekuens maksimum yang tetap, sehingga lebih fleksibel dalam menangani variasi panjang kalimat. Selain itu, FLAN-T5 menggunakan skema *Pre-Norm* dengan *Simplified Layer Normalization* (tanpa bias aditif) dan fungsi aktivasi *Gated GELU*. Modifikasi ini meningkatkan stabilitas pelatihan pada model yang lebih dalam (12 *layer*) dan kompleks.



Gambar 4. Mekanisme *Input* Berbasis Instruksi pada FLAN-T5

FLAN-T5 (*Fine-tuned Language Net T5*) mengadopsi pendekatan yang secara prinsip sama dengan T5 pada *text-to-text framework*, namun dioptimasi dengan dua tahap utama, yaitu *pre-training* pada *Colossal Clean Crawled Corpus (C4)* dan lanjut *instruction tuning* pada ribuan *dataset* dengan berbagai instruksi eksplisit. Model ini mampu memahami dan mengikuti banyak format perintah atau *task*, misalnya *"translate English to Indonesian: [source text]"*, dengan kualitas hasil yang lebih baik pada skenario *zero-shot* dan *few-shot*. Varian FLAN-T5 tersedia dalam beberapa ukuran, dari Flan-T5-Small (80 juta parameter) hingga Flan-T5-XXL (11 miliar parameter), dengan upaya *instruction tuning* yang jauh lebih intensif dan kemampuan generalisasi instruksi yang lebih kuat dibandingkan T5 original (Chung et al., 2022).

Untuk memahami mekanisme penerjemahan secara konkret, berikut ilustrasi aliran data pada arsitektur Transformer *encoder-decoder*. Misalkan teks sumber adalah "*the bird has yellow crown*": (1) Tokenisasi, teks dipecah menjadi token ["the", "bird", "has", "yellow", "crown"]; (2) Token Embedding, setiap token dikonversi menjadi vektor berdimensi tinggi melalui *embedding layer*; (3) Positional Encoding, informasi posisi ditambahkan ke *token embedding* melalui penjumlahan vektor posisi (OPUS-MT) atau diinjeksikan pada *attention* melalui *relative positional bias* (FLAN-T5); (4) Encoder Processing, sekuens vektor diproses melalui *multi-head self-attention* dan *feed-forward*

layers untuk menghasilkan representasi kontekstual dari teks sumber; (5) Decoder Processing, *decoder* menerima representasi dari *encoder* dan secara *autoregressive* menghasilkan token target satu per satu menggunakan *cross-attention* untuk mengakses informasi dari *encoder*; (6) Output Projection, vektor keluaran *decoder* diproyeksikan ke dimensi kosakata target melalui *linear layer*, menghasilkan *logits* untuk setiap kata dalam kosakata; (7) Softmax & Prediction, fungsi *softmax* mengkonversi *logits* menjadi distribusi probabilitas, dan token dengan probabilitas tertinggi dipilih sebagai prediksi. Proses *decoding* berlanjut hingga model menghasilkan token akhir (*End of Sequence*).

Perbandingan karakteristik utama dari kedua model dapat dilihat pada Tabel 1.

Tabel 1. Perbandingan Karakteristik Model OPUS-MT dan T5

Karakteristik	OPUS-MT-EN-ID	FLAN-T5-BASE-OPUS-EN-ID-ID-EN
Paradigma	Bilingual	Bilingual (*text-to-text, instruction tuning*)
Arsitektur Dasar	Transformer *encoder-decoder, dedicated bilingual NMT*	Transformer *encoder-decoder, Unified Text-to-Text Transfer Transformer*
Mekanisme Input	Menerima teks mentah (*raw text*) tanpa prefix/instruksi	Menerima instruksi + teks
Embedding Posisi	*Absolute Positional Embeddings*	*Relative Positional Embeddings*
Normalisasi Layer	*Standard Layer Normalization (Post-Norm)*	Pre-Norm (Simplified Layer Norm)
Fungsi Aktivasi	*Swish* atau *ReLU*	*Gated GELU*
Kedalaman Model	*6 layer encoder*, *6 layer decoder*	12 *layer encoder*, 12 *layer decoder*
Jumlah Parameter	72M	247M
Pre-training Objective	*Supervised translation*	*Span corruption* + *instruction tuning*
Jumlah Bahasa	2 (en-id)	2 (en-id dan id-en)
Pre-training Data	*Parallel corpus* (OPUS)	*Parallel corpus* (OPUS, en-id + id-en) + instruksi
Computational Cost	Rendah	Sedang
Zero-shot Translation	Tidak	Baik (berbasis instruksi)
Domain Adaptation	Baik (via *fine-tuning*)	Baik (via *fine-tuning* & *instruction tuning*)

(Sumber: Diadaptasi dari Tiedemann & Thottingal, 2020; Chung et al., 2022; Aulamo et al., 2020)

Meskipun OPUS-MT dan FLAN-T5 sama-sama dibangun di atas arsitektur dasar Transformer *encoder-decoder* (Vaswani et al., 2017), keduanya menerapkan skema arsitektur dan mekanisme input yang berbeda secara fundamental. OPUS-MT mengadopsi arsitektur bilingual yang didedikasikan untuk satu pasangan bahasa tertentu, dengan kedalaman model yang lebih efisien (6 *layer encoder* dan 6 *layer*

decoder) serta alur masukan yang langsung menerima teks sumber tanpa kebutuhan instruksi eksplisit (Tiedemann & Thottingal, 2020). Sebaliknya, FLAN-T5 menerapkan arsitektur *Unified Text-to-Text Transfer* Transformer dengan struktur yang lebih dalam (12 *layer* pada varian *base*) dan dirancang untuk menangani berbagai tugas pemrosesan teks. Dalam kerangka ini, penerjemahan dipandang sebagai salah satu dari banyak *task* generatif, sehingga model mengharuskan skema input berbasis instruksi (*prompting*) pada lapisan awal *encoder* untuk mengarahkan bobot model secara spesifik ke tugas penerjemahan.(Chung et al., 2022; Longpre et al., 2023).

Pemilihan model yang tepat untuk tugas penerjemahan *domain-specific* bergantung pada berbagai faktor termasuk ketersediaan data, *computational resources*, dan karakteristik pasangan bahasa. *Fine-tuning* pada data spesifik seperti teks ornitologi memungkinkan kedua model untuk beradaptasi dengan terminologi dan struktur bahasa khusus, meskipun masing-masing model menunjukkan karakteristik pembelajaran yang berbeda tergantung pada strategi *pre-training* dan arsitektur dasarnya (Moslem et al., 2023). Penelitian komparatif yang sistematis terhadap kedua model pada domain yang sama dengan metodologi evaluasi yang konsisten menjadi penting untuk memberikan *insight* tentang *trade-off* antara efisiensi, performa, dan kemampuan generalisasi masing-masing pendekatan.

1.2.5 *Transfer Learning* dan *Fine-Tuning*

Transfer learning merupakan pendekatan esensial dalam pengembangan *Neural Machine Translation (NMT)* masa kini, di mana model yang telah di-*pre-train* menggunakan korpus besar dan beragam dapat digunakan sebagai dasar untuk adaptasi ke domain baru atau ke tugas spesifik dengan jumlah data terbatas. Pendekatan ini sangat krusial untuk pasangan bahasa *low-resource* seperti Inggris-Indonesia, yang tidak memiliki *parallel corpus* besar dan berkualitas tinggi (Ruder et al., 2019).

Dalam praktiknya, *transfer learning* terbagi menjadi dua fase utama *pre-training* dan *fine-tuning*. Pada fase *pre-training*, model seperti OPUS-MT dilatih menggunakan *parallel corpus* (OPUS), sementara FLAN-T5 diberdayakan dengan *pre-training* pada C4 dan *instruction datasets* (Chung et al., 2022; Liu et al., 2020). Proses *pre-training* menghasilkan parameter model yang mampu memahami representasi linguistik dan struktur sintaksis secara komprehensif lintas domain (Raffel et al., 2019).

Selanjutnya, *fine-tuning* dilakukan dengan *continued training* pada *dataset target* yang berukuran lebih kecil dan *domain-spesifik*, misalnya pada data ornitologi. Model mengalami penyesuaian bertahap agar dapat mengoptimalisasikan hasil terjemahan untuk terminologi dan pola unik domain tersebut. Proses ini umumnya menggunakan *learning rate* yang lebih kecil dibandingkan *pre-training* agar parameter yang telah teroptimasi tidak langsung digantikan, tetapi diadaptasi secara gradual (Mosbach et al., 2020). Model yang diajarkan dengan data lebih sedikit pada tahap *fine-tuning* terbukti kompetitif dan lebih hemat sumber daya dibandingkan model yang dilatih dari nol (Smith, 2018).

Keberhasilan *transfer learning* dan *fine-tuning* sangat ditentukan oleh pengaturan *hyperparameter*. *Learning rate* harus dioptimalkan agar cukup kecil untuk

menjaga pengetahuan umum (mencegah *catastrophic forgetting*), namun cukup besar untuk mempercepat adaptasi (Bergstra & Bengio, 2012; Dodge et al., 2020). *Batch size* dan jumlah *epoch* juga mempengaruhi stabilitas dan generalisasi model, sedangkan teknik regularisasi seperti *weight decay* membantu menghindari *overfitting* pada data domain target (Mosbach et al., 2020).

1.2.6 Metrik Evaluasi

Evaluasi kualitas *Neural Machine Translation* merupakan aspek fundamental dalam penelitian penerjemahan mesin yang menentukan seberapa baik sistem dapat menghasilkan terjemahan yang akurat dan natural. Berbeda dengan tugas klasifikasi atau regresi yang memiliki *ground truth* objektif, evaluasi kualitas terjemahan bersifat kompleks karena terdapat *multiple valid translations* untuk satu kalimat sumber, tergantung pada pilihan kata, struktur sintaksis, dan gaya bahasa yang digunakan (Mathur et al., 2020). Evaluasi manual oleh penerjemah profesional atau *native speakers*, meskipun dianggap sebagai *gold standard*, memerlukan waktu, biaya, dan *effort* yang substansial, sehingga tidak praktis untuk evaluasi iteratif selama *development model* atau untuk membandingkan banyak sistem secara sistematis (Freitag et al., 2021). Oleh karena itu, *automatic evaluation metrics* telah dikembangkan untuk memberikan estimasi kuantitatif tentang kualitas terjemahan dengan membandingkan output sistem terhadap satu atau lebih *reference translations* yang dibuat oleh manusia. Dua metrik yang paling *established* dan *widely adopted* dalam penelitian NMT adalah BLEU (*Bilingual Evaluation Understudy*) dan METEOR (*Metric for Evaluation of Translation with Explicit ORdering*), yang masing-masing menggunakan pendekatan berbeda dalam mengukur *similarity* antara *machine translation* dan *reference translation*.

BLEU, yang diperkenalkan oleh Papineni et al. pada tahun 2002, merupakan metrik evaluasi berbasis *precision* yang mengukur *overlap n-gram* antara *candidate translation* (output sistem) dan *reference translation*. *N-gram* adalah sekuens kontinu dari n kata; misalnya, untuk kalimat *"the bird is red"*, *unigrams* adalah {the, bird, is, red}, *bigrams* adalah {the bird, bird is, is red}, dan seterusnya (Papineni et al., 2002). BLEU menghitung *modified n-gram precision* untuk berbagai nilai n (biasanya n=1 hingga n=4), di mana setiap n-gram dalam candidate diperiksa apakah muncul dalam reference, dan precision dihitung sebagai proporsi *n-gram* yang *match*. *"Modified" precision* mengatasi masalah *double-counting* dengan membatasi jumlah *maximum count* untuk setiap n-gram berdasarkan kemunculannya dalam reference (Post, 2018). Skor BLEU final dihitung sebagai *geometric mean* dari modified precision untuk semua *n-gram*, dikalikan dengan *brevity penalty* yang memberikan penalti pada terjemahan yang terlalu pendek dibandingkan reference. Formula BLEU secara matematis dapat dinyatakan pada persamaan (1) di bawah ini:

$$BLEU = BP \times exp\left(\sum_{n=1}^{N} w_n \log P_n\right) \quad (1)$$

Pada persamaan (1), *BP* adalah *brevity penalty*, w_n adalah *weight* untuk *n-gram* (biasanya uniform 1/4 untuk $n = 1,2,3,4$), dan P_n adalah *modified n-gram precision*. *BP* dihitung sebagai $\exp(1 - r/c)$ jika panjang *candidate* lebih pendek

dari *reference length* r, dan 1 jika *candidate* lebih panjang atau sama panjang (Papineni et al., 2002). Skor BLEU berkisar dari 0 hingga 100 (atau 0 hingga 1 dalam skala *normalized*), di mana skor yang lebih tinggi mengindikasikan *similarity* yang lebih besar dengan *reference*.

BLEU memiliki beberapa keunggulan signifikan: implementasinya sederhana dan *computationally efficient*, tidak memerlukan *linguistic resources* seperti *parser* atau *lexicon*, *language-agnostic* sehingga dapat diaplikasikan pada pasangan bahasa apapun, dan memiliki *strong correlation* dengan *human judgment* pada *corpus-level evaluation* (Post, 2018). Namun, BLEU juga memiliki keterbatasan: metrik ini bersifat *precision-focused* dan tidak secara eksplisit mengukur *recall* (seberapa banyak informasi dari *reference* yang tercakup dalam *candidate*), tidak mempertimbangkan *synonyms* atau *paraphrase* sehingga penalti diberikan pada variasi kata yang *semantically equivalent*, dan *precision-based approach* dapat memberikan skor tinggi pada terjemahan yang *fluent* namun tidak akurat secara semantik (Mathur et al., 2020).

METEOR dikembangkan oleh Banerjee dan Lavie pada tahun 2005 sebagai respons terhadap keterbatasan BLEU, dengan tujuan untuk mencapai *better correlation* dengan *human judgment* melalui *incorporation of linguistic knowledge*. Berbeda dengan BLEU yang murni berbasis *exact n-gram matching*, METEOR menggunakan *unigram matching* dengan beberapa *enhancement*: *exact matching* untuk kata yang identik, *stem matching* menggunakan *Porter stemmer* untuk mengenali variasi morfologi dari kata yang sama (misalnya, *"running"* dan *"runs"* dianggap *match*), dan *synonym matching* menggunakan *WordNet* untuk mengenali kata-kata yang *semantically equivalent* (Banerjee & Lavie, 2005). METEOR menghitung *harmonic mean* dari *unigram precision* dan *recall*, memberikan bobot yang lebih besar pada *recall* (biasanya dengan rasio 9:1) untuk mengatasi *bias precision* pada BLEU. Setelah *unigram matching*, METEOR menghitung *penalty* berdasarkan jumlah *chunks* sekuens kata kontinu yang *matched* dengan rasionalisasi bahwa terjemahan dengan *fewer chunks* (kata-kata yang *matched* lebih terkelompok) lebih natural dan *preserving word order* dengan lebih baik (Freitag et al., 2021). Formula METEOR dapat dinyatakan sebagai:

$$METEOR = F_{mean} \times (1 - Penalty) \tag{2}$$

Pada persamaan (2), perhitungan METEOR memerlukan nilai *Precision* (*P*) dan *Recall* (*R*) yang diperoleh dari proses *unigram matching* antara *hypothesis* (*c*) dan *reference* (*r*). *Precision* mengukur proporsi *unigram* pada *hypothesis* yang cocok dengan *reference*, sedangkan *Recall* mengukur proporsi *unigram* pada *reference* yang berhasil tercakup dalam *hypothesis*. Kedua nilai tersebut dihitung sebagai berikut:

$$P = \frac{m}{|c|} \tag{3}$$

$$\mathrm{R} = \frac{m}{|r|} \tag{4}$$

Dengan m adalah jumlah unigram yang cocok (*matches unigrams*), $|c|$ adalah jumlah unigrams pada hypotesis, dan $|r|$ adalah jumlah *unigram* pada *reference*.

Selanjutnya, nilai F_{mean} dihitung sebagai *harmonic mean* dari *precision* P dan *recall* R dengan *weight parameter* α (biasanya 0.9):

$$F_{mean} = \frac{P \times R}{\alpha \times P + (1 - \alpha) \times R} \quad (5)$$

dan *Penalty* dihitung berdasarkan *fragmentation*:

$$Penalty = \gamma \times (\frac{chunks}{matches})^{\beta} \quad (6)$$

dengan γ dan β sebagai *tuning parameters* (Denkowski & Lavie, 2014). Meskipun ketiga parameter (α, β, γ) bersifat *tunable*, Banerjee dan Lavie (2005) merekomendasikan nilai *default* α = 0.9, β = 3, dan γ = 0.5 yang telah dioptimalkan melalui eksperimen untuk mencapai korelasi tertinggi dengan penilaian manusia. Nilai-nilai ini menjadi standar yang umum digunakan dalam evaluasi NMT.

METEOR menghasilkan skor antara 0 dan 1, dengan nilai yang lebih tinggi mengindikasikan kualitas terjemahan yang lebih baik. Keunggulan METEOR terletak pada kemampuannya untuk mengenali *semantic similarity* melalui *synonym matching* dan *morphological variation* melalui *stemming*, memberikan *importance* pada *recall* selain *precision*, dan mempertimbangkan *word order* melalui *chunk-based penalty* (Freitag et al., 2021). Namun, METEOR memiliki *kompleksitas computational* yang lebih tinggi dibandingkan BLEU karena memerlukan *linguistic resources* seperti *stemmer* dan *synonym database*, dan *language-specific* karena memerlukan adaptasi untuk setiap bahasa yang dievaluasi (meskipun versi untuk banyak bahasa telah tersedia).

Perbandingan karakteristik BLEU dan METEOR dapat dilihat pada Tabel 2. Kedua metrik memiliki filosofi yang berbeda dalam mengukur *translation quality*: BLEU fokus pada *precision* dengan *n-gram matching* untuk menangkap *fluency* dan *local coherence*, sementara METEOR menyeimbangkan *precision* dan *recall* dengan *unigram matching* yang lebih fleksibel untuk menangkap *adequacy* dan *semantic similarity*. Dalam praktik penelitian *NMT*, kedua metrik sering digunakan secara bersamaan karena mereka mengukur aspek kualitas terjemahan yang berbeda namun *komplementer* (Mathur et al., 2020).

Tabel 2. Perbandingan Karakteristik BLEU dan METEOR

Aspek	BLEU	METEOR
Strategi Pencocokan	Pencocokan *n-gram* secara eksak (n=1,2,3,4)	Pencocokan unigram (eksak, stem, sinonim)
Fokus Penilaian	Presisi	Rata-rata harmonik (dengan bobot pada *recall*)
Urutan N-gram	Multipel (1–4)	Hanya unigram
Sumber Daya Linguistik	Tidak diperlukan	Memerlukan stemmer dan basis data sinonim (WordNet)
Urutan Kata	Dipertimbangkan secara implisit melalui *n-gram* tingkat tinggi	Dipertimbangkan secara eksplisit melalui penalti berbasis *chunk*
Pengenalan Sinonim	Tidak	Ya (melalui WordNet)

Aspek	BLEU	METEOR
Penanganan Morfologi	Tidak	Ya (melalui proses *stemming*)
Biaya Komputasi	Rendah	Sedang
Ketergantungan Bahasa	Tidak bergantung pada bahasa (*language-agnostic*)	Bergantung pada sumber daya linguistik bahasa tertentu
Rentang Skor	0–100 (atau 0–1)	0–1
Korelasi dengan Penilaian Manusia	Baik (pada level korpus)	Lebih baik (pada level kalimat dan korpus)
Kelebihan Utama	Mengukur kelancaran dan kesesuaian *n-gram*	Mengukur kecukupan dan kesamaan semantik
Kelemahan Utama	Mengabaikan sinonim dan parafrasa	Kompleksitas lebih tinggi dan bergantung pada sumber daya linguistik
Kasus Penggunaan Terbaik	Perbandingan cepat, tolok ukur awal	Evaluasi mendalam, bahasa dengan sumber daya terbatas

(Sumber: Diadaptasi dari Papineni et al., 2002; Banerjee & Lavie, 2005; Mathur et al., 2020)

Meskipun BLEU dan METEOR menunjukkan *strong correlation* dengan *human judgment* pada *level korpus*, korelasinya pada *sentence-level* jauh lebih lemah dan dapat menghasilkan *misleading* skor (Mathur et al., 2020). Keterbatasan ini, seperti sistem yang *fluent* namun mengubah *meaning*, telah mendorong pengembangan metrik yang lebih *sophisticated* seperti *BERTScore* atau *COMET* (Rei et al., 2020; Zhang et al., 2019). Namun, BLEU dan METEOR tetap menjadi *de facto standard* dalam penelitian *NMT* karena *simplicity*, *interpretability*, dan *extensive usage*-nya.

Dalam konteks penelitian ini, penggunaan kombinasi keduanya memungkinkan evaluasi yang *comprehensive*: BLEU mengukur *fluency* (*n-gram sequences*), sementara METEOR menilai *adequacy* dan penggunaan *vocabulary* yang *semantically appropriate* (*recall component*) (Denkowski & Lavie, 2014). Kombinasi ini krusial untuk *domain-specific translation* (ornitologi) karena dapat mengungkap *trade-off* antara *fluency* dan *terminology accuracy*.

Interpretasi skor harus ditempatkan dalam konteks *low-resource language pairs* (English-Indonesian) dan *domain* yang dievaluasi. *Absolute scores* kurang *meaningful* dibandingkan *relative comparison between systems* pada *dataset* yang sama; *improvement* kecil (*e.g.*, 1-2 BLEU *points*) sudah dianggap *substantial* (Post, 2018). Oleh karena itu, selain *automatic metrics*, *qualitative analysis* melalui *manual inspection* terhadap terjemahan wajib dilakukan untuk mendapatkan *insight complementary* mengenai *strengths and weaknesses* model (Freitag et al., 2021).

1.2.7 Freeze Layer

Dalam pengembangan model *Neural Machine Translation* (NMT), teknik *freeze layer* merupakan strategi yang umum digunakan pada pendekatan *transfer learning* dan *fine-tuning* untuk meningkatkan efisiensi dan stabilitas pelatihan model. *Freeze layer* adalah proses membekukan parameter atau bobot pada lapisan tertentu dari jaringan saraf agar

tidak ikut diperbarui selama proses pelatihan lanjutan. Dengan demikian, lapisan yang dibekukan akan mempertahankan representasi atau pengetahuan umum (*general knowledge*) yang telah diperoleh selama tahap *pre-training* (Howard & Ruder, 2018).

Teknik ini memiliki beberapa manfaat penting. Pertama, *freeze layer* dapat menghemat waktu pelatihan dan sumber daya komputasi, karena hanya sebagian lapisan yang dihitung gradiennya. Hal ini sangat bermanfaat ketika menggunakan model besar seperti Transformer yang memiliki jutaan hingga miliaran parameter (Devlin et al., 2019). Kedua, pembekuan sebagian lapisan juga mengurangi risiko *overfitting* pada data *domain-specific*, terutama ketika ukuran data pelatihan terbatas (Houlsby et al., 2019). Lapisan awal dari model pralatih umumnya berfungsi sebagai *feature extractor* umum yang relatif stabil dan dapat ditransfer antar domain atau antar bahasa, sedangkan lapisan akhir dapat dibiarkan *trainable* untuk beradaptasi dengan karakteristik domain baru (Houlsby et al., 2019). Ketiga, pendekatan ini mempertahankan pengetahuan umum hasil *pre-training* sambil tetap memberikan fleksibilitas adaptasi pada lapisan yang lebih tinggi, sehingga model tidak kehilangan kemampuan semantik dasar yang telah diperoleh sebelumnya (Mosbach et al., 2021).

Dalam konteks NMT modern, penerapan *freeze layer* biasanya dilakukan pada bagian embedding, beberapa blok awal *encoder*, atau komponen tertentu dari *decoder*, tergantung pada tujuan adaptasi. Misalnya, pada model seperti T5 atau MarianMT, peneliti dapat membekukan lapisan *encoder* awal dan hanya melatih lapisan *decoder* untuk mempercepat proses konvergensi pada data target (Lee et al., 2023; Zoph et al., 2016). Strategi ini sejalan dengan *best practice* bahwa lapisan awal berfungsi untuk menangkap representasi linguistik umum (*low-level linguistic features*), sedangkan lapisan menengah dan akhir bertanggung jawab untuk representasi semantik yang lebih spesifik terhadap tugas (Ruder et al., 2019).

Beberapa penelitian empiris menunjukkan bahwa teknik *freeze layer* juga berperan sebagai regularizer implisit yang dapat meningkatkan generalisasi model dan stabilitas pelatihan. Dengan membatasi ruang parameter yang dioptimalkan, model cenderung beradaptasi secara lebih terkendali terhadap domain baru tanpa mengubah drastis representasi umum yang sudah mapan (Howard & Ruder, 2018). Namun, efektivitas teknik ini sangat bergantung pada posisi layer yang dibekukan, ukuran model (*overparameterization level*), dan kesesuaian data domain target dengan domain pre-training (Lee et al., 2023).

Dalam praktik penelitian *transfer learning* untuk NMT, strategi pembekuan lapisan sering dikombinasikan dengan pendekatan seperti *gradual unfreezing* di mana lapisan dibuka secara bertahap dari akhir ke awal model untuk mengontrol stabilitas pembelajaran dan mencegah *catastrophic forgetting* (Howard & Ruder, 2018). Oleh karena itu, pemilihan lapisan mana yang dibekukan harus disesuaikan dengan tujuan eksperimen dan karakteristik data, agar diperoleh keseimbangan antara efisiensi komputasi, stabilitas pelatihan, dan kualitas adaptasi domain.

1.3 Tujuan dan Manfaat

1.3.1 Tujuan

Tujuan dari penelitian ini adalah:

1. Mengimplementasikan model Neural Machine Translation (NMT) berbasis *attention* dengan model OPUS-MT dan FLAN-T5 untuk menerjemahan teks Inggris-Indonesia dalam domain ornitologi.
2. Mengevaluasi perbandingan performa model NMT berbasis *attention* dengan model OPUS-MT dan FLAN-T5 untuk penerjemahan teks Inggris-Indonesia dalam domain ornitologi.

1.3.2 Manfaat

1. Menjadi referensi bagi penelitian selanjutnya untuk mengembangkan model *Neural Machine Translation* (NMT), khususnya dalam penerjemahan dataset teknis yang berkaitan dengan domain ornitologi, dengan studi komparatif antara OPUS-MT dan FLAN-T5.
2. Mempermudah akses terhadap informasi dalam dataset CUB-200-2011 bagi peneliti yang menggunakan Bahasa Indonesia, dengan mempertimbangkan performa dan karakteristik model yang berbeda.
3. Menyempurnakan korpus data CUB-200-2011 paralel Bahasa Inggris-Indonesia yang memenuhi kaidah ornitologi.

BAB II
METODE PENELITIAN

2.1 Tempat dan Waktu

Proses penelitian dilakukan sejak bulan Maret 2025. Penelitian bertempat di Laboratorium *Cloud Computing and Information Syste*m (CCIS), Departemen Teknik Informatika, Fakultas Teknik, Universitas Hasanuddin.

2.2 Instrumen Penelitian

Instrumen yang digunakan dalam penelitian ini:

1. Perangkat Lunak:
    a. Windows 11
    b. Visual Studio Code
    c. Microsoft Word
    d. Microsoft Excel
2. Perangkat Keras:
    a. Intel® Core™ i9-11900F CPU 2.50GHz, RAM 32GB
    b. NVIDIA GeForce GTX 3070
3. Bahasa Pemrograman:
    a. Python v3.13.0
4. *Library*:
    a. Transformers v4.57.0, digunakan untuk memuat dan menjalankan model *pre-trained* NMT
    b. Torch v2.8.0, digunakan untuk membangun, melatih, dan menjalankan model deep learning berbasis tensor dengan dukungan GPU pada *framework* PyTorch.
    c. NLTK v3.9.1, digunakan untuk melakukan *preprocessing* teks seperti mengunduh *stopwords*
    d. Pandas v2.2.3, digunakan untuk mengelola dan memanipulasi data.
    e. NumPy v2.1.3, digunakan untuk operasi *array* dan komputasi numerik
    f. Scikit-learn v1.5.2, digunakan untuk data splitting dan *preprocessing*
    g. Matplotlib v3.9.2, digunakan untuk melakukan visualisasi data menjadi bentuk grafik atau diagram
    h. SacreBLEU v2.6.1, digunakan untuk implementasi standar metrik BLEU
    i. Evaluate v0.4.3, digunakan untuk *framework* metrik METEOR
5. Model *Pre-trained*:
    a. OPUS-MT
    b. FLAN-T5

2.3 Tahapan Penelitian

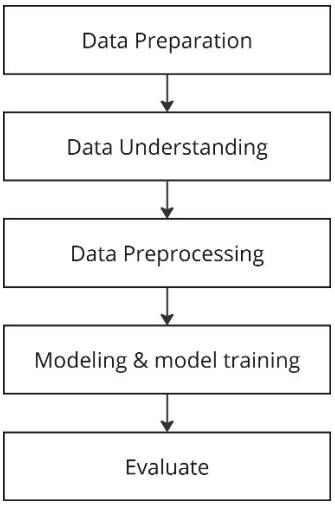
Penelitian ini dimulai dengan tahap *data preparation*, yaitu pengumpulan dataset CUB-200-2011 yang telah diterjemahkan ke Bahasa Indonesia, membentuk korpus paralel dengan lebih dari 117.000 pasangan kalimat deskripsi morfologi burung (Wah et al.,

2011). Selanjutnya, penyempurnaan terjemahan dataset dilakukan dengan validasi dan koreksi manual berdasarkan literatur ornitologi.

Selanjutnya *data understanding*, dilakukan analisis terhadap struktur dan karakteristik korpus paralel untuk memahami pola data sebelum proses *preprocessing* dan pelatihan model. Analisis mencakup identifikasi struktur data, panjang teks, serta distribusi frekuensi kata, termasuk istilah teknis ornitologi yang jarang muncul dalam teks umum. Analisis ini memberikan gambaran pola data sehingga dapat menentukan strategi *preprocessing* dan pelatihan model yang lebih tepat.

Setelah itu dilakukan *data preprocessing* untuk membersihkan teks dari karakter tidak relevan, menghapus duplikat, melakukan tokenisasi, dan membagi data ke dalam *train set* dan *test set*.Tahap berikutnya adalah *modeling & model training*, di mana dilakukan fine-tuning pada dua model NMT berbasis *attention* (OPUS-MT dan FLAN-T5) dengan penyesuaian *hyperparameter* untuk memperoleh performa terbaik

Model yang telah dilatih selanjutnya masuk ke tahap *evaluate*, dengan pengujian menggunakan metrik BLEU dan METEOR. Hasil evaluasi kemudian dianalisis pada tahap *analysis* untuk mengetahui efektivitas model dalam menerjemahkan teks ornitologi Inggris–Indonesia.

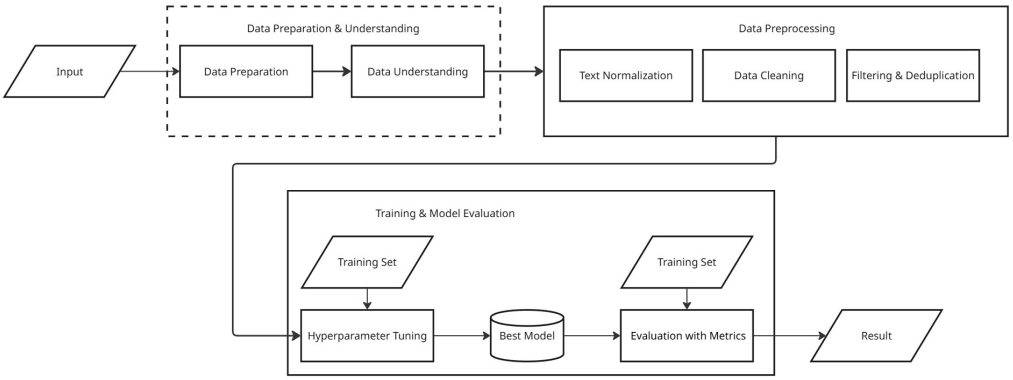


Gambar 5. Tahapan penelitian

2.4 Perancangan dan Implementasi Sistem

Pada penelitian ini, digunakan dua model utama yaitu *FLAN-T5* (*Instruction-Tuned Text-to-Text Transfer Transformer*) dan OPUS-MT (*Helsinki-NLP/opus-mt-en-id*). Kedua model tersebut di-*fine-tune* secara terpisah menggunakan dua skrip implementasi yang memiliki struktur *pipeline* yang sama, meliputi tahap input data, *preprocessing*, pelatihan (*fine-tuning*), dan evaluasi.

Gambar 5 merupakan diagram untuk perancangan alur sistem.



Gambar 6. Perancangan Alur Sistem

Tahap pertama dalam implementasi sistem adalah pengumpulan dan pemrosesan awal dataset yang akan digunakan sebagai data latih. Dataset yang digunakan dalam penelitian ini merupakan *parallel corpus* yang bersumber dari CUB-200-2011 *Captions Dataset*. Dataset tersebut berisi pasangan kalimat deskriptif mengenai berbagai spesies burung dalam Bahasa Inggris beserta terjemahannya dalam Bahasa Indonesia. Format dataset berbentuk JSON, di mana setiap entri memiliki pasangan kunci *english* untuk teks sumber dan indo untuk teks target. Data tersebut dibaca ke dalam program kemudian diubah menjadi pasangan teks bilingual yang siap digunakan pada tahap berikutnya. Tahap input data ini bertujuan untuk memastikan bahwa struktur data yang dimasukkan sesuai dengan kebutuhan model penerjemahan otomatis berbasis *sequence-to-sequence*.

2.4.1 Data Preparation & Understanding

Tahap Data Preparation difokuskan pada verifikasi dan penyempurnaan isi terjemahan dalam dataset CUB-200-2011. Proses ini diawali dengan penerjemahan ulang data menggunakan Google Translate sebagai baseline, mengingat hasil terjemahan sebelumnya masih mengandung banyak kesalahan istilah dan struktur. Setiap kata kemudian diperiksa satu per satu untuk memastikan kesesuaian istilah, memperbaiki kesalahan penulisan, serta menyesuaikan terminologi dan struktur frasa agar akurat secara morfologis dan biologis. Pemeriksaan ini mencakup koreksi beberapa istilah penting seperti *crown*, *secondaries*, *malar stripe*, dan *superciliary line* dengan merujuk pada literatur serta glosarium ornitologi. Seluruh revisi dicatat melalui log koreksi sehingga perubahan dapat ditelusuri dengan jelas. Tahap ini hanya berfokus pada

validasi ilmiah isi dataset dan tidak mencakup operasi teknis seperti penghapusan duplikat atau pembersihan teks.

Setelah penyempurnaan terjemahan, dilakukan *Data Understanding* untuk menganalisis karakteristik korpus paralel sebelum memasuki tahap preprocessing. Analisis ini meliputi identifikasi struktur data, panjang teks, dan distribusi frekuensi kata, termasuk istilah teknis ornitologi yang jarang muncul pada teks umum. Selain itu, tahap ini juga meninjau kemunculan artefak seperti karakter non-latin atau token tidak wajar. Hasil *Data Understanding* memberikan gambaran pola data yang diperlukan untuk menetapkan strategi *preprocessing* dan konfigurasi pelatihan model yang lebih tepat.

2.4.2 Preprocessing Data

Tahap preprocessing dilakukan untuk membersihkan dan menstandarkan data agar siap digunakan dalam proses pelatihan. Setiap teks dalam dataset melalui proses normalisasi karakter menggunakan standar Unicode (NFKC) untuk memastikan keseragaman encoding. Setelah itu, dilakukan konversi seluruh huruf menjadi huruf kecil (*lowercasing*), karakter non-alfanumerik, dan simbol asing yang tidak diperlukan. Proses ini juga mencakup penghapusan duplikasi kalimat dan penghapusan data yang memiliki panjang kurang dari 6 token agar model tidak belajar dari data yang terlalu pendek. Selain itu, dilakukan pula kompresi spasi berlebih dan penghapusan artefak teks tertentu yang muncul dalam dataset asli. Setelah seluruh data dibersihkan, dataset dibagi menjadi tiga bagian, yaitu 80% untuk data latih, 10% untuk data validasi, dan 10% untuk data uji. Pembagian data dilakukan secara acak menggunakan fungsi train_test_split() dari pustaka Scikit-learn dengan mempertahankan distribusi proporsional agar hasil pelatihan tetap representatif.

2.4.3 Fine-Tuning

Tahap *fine-tuning* merupakan bagian inti dari implementasi sistem, di mana model pralatih disesuaikan dengan domain dataset penelitian agar mampu menghasilkan terjemahan yang lebih kontekstual dan relevan. Pada penelitian ini digunakan dua model berbasis *sequence-to-sequence Transformer*, yaitu T5 dan OPUS-MT, yang masing-masing dijalankan melalui skrip pelatihan terpisah namun dengan alur implementasi yang identik.

Model FLAN-T5 yang digunakan merupakan model muvazana/flan-t5-base-opus-en-id-id-en, yaitu varian *instruction-tuned* dari FLAN-T5 berukuran *base* yang telah dilatih ulang menggunakan dataset bilingual OPUS untuk pasangan bahasa Inggris–Indonesia. Model ini menggabungkan kemampuan *instruction-based learning* dari T5 dengan kompetensi penerjemahan dua arah (EN→ID dan ID→EN) dari dataset OPUS, sehingga lebih adaptif terhadap perintah dan konteks terjemahan. Model diinisialisasi menggunakan AutoModelForSeq2SeqLM dan AutoTokenizer dari pustaka Transformers dengan tokenizer bertipe *SentencePiece*, yang juga digunakan pada model OPUS-MT untuk menjaga konsistensi pemrosesan teks.

Model OPUS-MT sendiri menggunakan arsitektur Helsinki-NLP/opus-mt-en-id yang merupakan model *Neural Machine Translation (NMT)* berbasis Transformer hasil pra-latih oleh tim Helsinki-NLP secara khusus untuk penerjemahan Bahasa Inggris ke Bahasa Indonesia. Meskipun memiliki basis pra-latih yang berbeda, kedua model ini

menggunakan struktur arsitektur dan mekanisme pembelajaran yang sama, sehingga pipeline pelatihan dapat diterapkan secara seragam pada keduanya.

Proses *fine-tuning* dilakukan dengan mengoptimalkan parameter model menggunakan *optimizer* AdamW, yang secara efektif mengatur pembaruan bobot dan regularisasi selama pelatihan. Untuk menjaga kestabilan laju pembelajaran, digunakan *learning rate scheduler* dari fungsi get_scheduler() yang mengatur penurunan *learning rate* secara bertahap di setiap *epoch*.

Untuk menentukan konfigurasi terbaik, dilakukan pencarian hyperparameter melalui metode *grid search*. Parameter yang diuji meliputi nilai *learning rate* (2e-5, dan 3e-5), jumlah *epoch* (3 dan 5), ukuran *batch size* (16), *weight decay* (0.01), dan *warmup ratio* (0.1). Kombinasi parameter tersebut diuji untuk menemukan konfigurasi yang paling stabil dan efisien selama pelatihan. Setiap kombinasi menghasilkan *checkpoint* model yang disimpan secara otomatis, sehingga model terbaik dapat dipilih berdasarkan hasil validasi.

Tabel 3. Parameter Training yang Digunakan.

Parameter	Nilai / Rentang	Keterangan
Optimizer	AdamW	*Optimizer* berbasis Adam dengan regularisasi bobot untuk menjaga stabilitas pembaruan parameter pada model Transformer.
Scheduler	Linear (get_scheduler)	Mengatur penurunan learning rate secara bertahap selama proses pelatihan.
Learning rate	2e-5 dan 3e-5	Nilai *learning rate* yang diuji melalui metode *grid search* untuk menentukan konfigurasi optimal.
Jumlah epoch	3 dan 5	Jumlah iterasi penuh proses pelatihan untuk setiap kombinasi parameter.
Batch size	8	Jumlah data per *batch* dalam satu langkah pelatihan.
Gradient accumulation steps	2	Akumulasi *gradien* setiap dua batch sebelum pembaruan parameter, sehingga *effective batch size* menjadi 16.
Weight decay	0.01	Nilai regularisasi untuk mengurangi risiko overfitting pada data pelatihan.
Warmup ratio	0.1	Proporsi langkah awal sebelum learning rate mulai menurun secara linear.
Freeze layer	Lapisan awal encoder dibekukan	Membekukan beberapa lapisan awal agar bobotnya tidak diperbarui selama *fine-tuning*, untuk mempertahankan representasi umum hasil pra-latih.

2.4.4 Modul Evaluasi

Modul evaluasi dirancang untuk mengukur performa model secara objektif menggunakan metrik standar machine translation. Modul ini menerima input berupa model yang telah

di-*fine-tune* dan *test set*, kemudian menghasilkan terjemahan untuk setiap kalimat dalam *test set*. Terjemahan yang dihasilkan dibandingkan dengan terjemahan referensi menggunakan dua metrik: BLEU dan METEOR. BLEU mengukur *precision n-gram* menggunakan *library* SacreBLEU (Post, 2018), sementara METEOR menggunakan library Evaluate dari Hugging Face yang mempertimbangkan synonym matching dan stemming (Denkowski & Lavie, 2014). Modul ini menghasilkan skor kuantitatif yang dapat dibandingkan antar model.

Setelah proses pelatihan selesai, model hasil *fine-tuning* dievaluasi menggunakan data uji untuk mengukur kemampuan penerjemahan yang dihasilkan. Evaluasi dilakukan dengan dua metrik utama, yaitu BLEU (Bilingual Evaluation Understudy) dan METEOR (*Metric for Evaluation of Translation with Explicit Ordering*). Metrik BLEU digunakan untuk menilai kesamaan *n-gram* antara hasil terjemahan model dan teks referensi (Papineni et al., 2002), sedangkan METEOR digunakan untuk mengevaluasi kesamaan semantik dengan mempertimbangkan sinonim, morfologi, serta urutan kata (Denkowski & Lavie, 2014). Proses evaluasi dilakukan menggunakan pustaka SacreBLEU dan Evaluate yang terintegrasi dengan *framework* Hugging Face. Hasil evaluasi kemudian disimpan dalam format CSV untuk setiap model agar dapat dianalisis dan dibandingkan secara kuantitatif.