SKRIPSI

IMPLEMENTASI *NEURAL MACHINE TRANSLATION* DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

OLEH
Fiska Nabila F. Risal
07352011108

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS KHAIRUN
TERNATE

2024

LEMBAR PENGESAHAN

IMPLEMENTASI *NEURAL MACHINE TRANSLATION* DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

Oleh
Fiska Nabila F. Risal
07352011108

Skripsi ini telah disahkan
Tanggal 29 Juli 2024

Menyetujui
Tim Penguji

Ketua Penguji

Dr. MUHAMMAD RIDHA ALBAAR, S.Kom., M.Kom.
NIP. 198504232008031001

Pembimbing I

YASIR MUIN, S.T., M.Kom.
NIDN. 9990582796

Anggota Penguji

ROSIHAN, S.T., M.Cs.
NIP. 197607192010121001

Pembimbing II

Ir. AMAL KHAIRAN, S.T., M.Eng., IPM.
NIP. 197401112003121003

Anggota Penguji

HAIRIL KURNIADI SIRAJUDDIN, S.Kom., M.Kom.
NIP. 198204272023211009

Mengetahui/Menyetujui

Koordinator Program Studi
Informatika

ROSIHAN, S.T., M.Cs.
NIP. 197607192010121001

Dekan Fakultas Teknik
Universitas Khairun

Ir. ENDAH HARISUN, S.T., M.T., CRP.
NIP. 197511302005011013

LEMBAR PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama	:	Fiska Nabila F.Risal
NPM	:	07352011108
Fakultas	:	Teknik
Jurusan/Program Studi	:	Informatika
Judul Skripsi	:	Implementasi *Neural Machine Translation* Dengan Menggunakan Metode LSTM Pada Proses Penerjemahan Bahasa Indonesia Ke Bahasa Ternate

Dengan ini menyatakan bahwa penulisan Skripsi yang telah saya buat ini merupakan hasil karya sendiri dan benar keasliannya. Apabila ternyata di kemudian hari penulisan Skripsi ini merupakan hasil plagiat terhadap karya orang lain, maka saya bersedia mempertanggung jawabkan sekaligus bersedia menerima sanksi berdasarkan aturan tata tertib di Universitas Khairun.

Demikian pernyataan ini saya buat dalam keadaan sadar dan tidak dipaksakan.

Penulis

Fiska Nabila F. Risal

HALAMAN PERSEMBAHAN

Bismillahirrahmannirrahim atas segala Rahmat Allah *Subhanahu Wa Ta'ala* yang Maha Pengasih lagi Maha Penyayang serta mengucapkan rasa Syukur *Alhamdulillah* atas nikmat yang di berikan tanpa hentinya, penulis persembahkan skripsi ini kepada:

1. Bapak Faisal Risal Yusuf dan Ibu Umiyati Alkatiri, kedua orang tua hebat yang selalu menjadi cahaya penuntun dalam setiap langkah penulis. Segalanya tidak terlepas dari doa Ummi dan Abba yang selalu menjadi kekuatan dan alasan penulis bisa bertahan sejauh ini, terima kasih atas segala cinta dan pengorbanan kalian yang tiada henti kepada penulis sehingga dapat menyelesaikan penulisan skripsi ini. Terima kasih Ummi dan Abba.
2. Kepada saudara dan saudari saya, Sugandi Syarif, Faika Yuliarti, dan Firly F. Risal serta keluarga besar yang tidak dapat disebut satu persatu, terima kasih atas segala doa dan dukungan yang diberikan kepada penulis.
3. Terima kasih kepada ponakan tersayang, Prayudha dan Pradhipta. Kehadiran kalian adalah sinar terang yang selalu menyinari hari-hari penulis, menghadirkan senyum dan memberi kekuatan di saat-saat yang penuh tantangan. Terima kasih kesayangan hale.
4. Terima kasih kepada dosen dosen yang telah membantu dan mengarahkan saya dalam penulisan skripsi ini sehingga dapat menyelesaikan skripsi ini.
5. Terima kasih untuk teman-teman seperjuangan karena telah membantu dan menyemangati saya dalam menyelesaikan skripsi ini.
6. Terakhir,Terima kasih kepada diri sendiri telah mampu bertahan di saat-saat sulit, mampu berjuang sejauh ini meskipun jalan terasa sangat berat. Dalam setiap langkah dan Keputusan, aku telah belajar untuk lebih menghargai diri sendiri.

MOTTO

"Allah tidak akan membebani seseorang melainkan sesuai dengan kesanggupannya."

(Q.S Al-Baqarah: 286)

KATA PENGANTAR

Puji syukur saya panjatkan kepada Allah *Subhanahu Wata'ala* yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga penulis dapat menyusun skripsi ini dengan judul "Implementasi *Neural Machine Translation* Dengan Menggunakan Metode LSTM Pada Proses Penerjemahan Bahasa Indonesia Ke Bahasa Ternate", ini dapat diselesaikan untuk memenuhi salah satu persyaratan penyelesaian pendidikan sarjana Informatika Strata Satu (S1) pada Program Studi Informatika Fakultas Teknik Universitas Unkhair. Untuk menyelesaikan skripsi ini penulis sepenuhnya mendapat dukungan dari banyak pihak, oleh karena itu dengan rendah hati penulis mengucapkan terimakasih kepada:

1. Bapak Dr. Ridha Ajam, M.Hum., selaku Rektor Universitas Khairun Ternate.
2. Bapak Ir. Endah Harisun, S.T., M.T., CRP., selaku dekan Fakultas Teknik Universitas Khairun.
3. Bapak Rosihan, S.T., M.Cs., selaku Koordinator Program Studi Informatika Fakultas Teknik Universitas Khairun dan penguji II yang telah membantu penulis dalam penyelesaian skripsi ini.
4. Bapak Yasir Muin, S.T., M.Kom., sebagai Pembimbing I, terima kasih atas bimbingannya dalam penyelesaian skripsi ini.
5. Bapak Ir. Amal Khairan, S.T., M.Eng., IPM., sebagai Pembimbing II, terima kasih atas bimbingannya dalam penyelesaian skripsi ini.
6. Bapak Dr. Muhammad Ridha Albaar, S. Kom., M. Kom., sebagai Penguji I, terima kasih telah membantu dalam menyelesaikan skripsi ini. .
7. Bapak Hairil Kurniadi Sirajuddin, S. Kom., M.Kom., sebagai Penguji III, terima kasih telah membantu dalam menyelesaikan skripsi ini.
8. Kepada kedua orang tua yang senantiasa memberikan doa, dukungan dan kasih sayang kepada penulis hingga saat ini.
9. Kepada keluarga yang telah memberikan doa dan dukungan kepada penulis hingga saat ini.
10. Kepada teman-teman seperjuangan yang telah membantu memberikan saran dan kritik atas laporan skripsi ini.

11. Dan terima kasih untuk diri sendiri karena telah mampu berjuang sejauh ini.

Walaupun demikian dalam skripsi ini, penulis menyadari masih belum sempurna. oleh karena itu harapan penulis dalam memberikan kritik dan saran dari semua pihak yang bersifat membangun

Ternate, 12 Maret 2024

Penulis

DAFTAR ISI

Halaman

HALAMAN JUDUL ..... **i**
HALAMAN PENGESAHAN ..... **ii**
HALAMAN PERNYATAAN KEASLIAN ..... **iii**
HALAMAN PERSEMBAHAN ..... **iv**
KATA PENGANTAR ..... **v**
DAFTAR ISI ..... **vii**
DAFTAR GAMBAR ..... **x**
DAFTAR TABEL ..... **xi**
ABSTRAK ..... **xii**
BAB I PENDAHULUAN
1.1. Latar Belakang 1
1.2. Rumusan Masalah 3
1.3. Batasan Masalah 3
1.4. Tujuan Penelitian 3
1.5. Manfaat Penelitian 4
1.6. Sistematika Penulisan 4
BAB II TINJAUAN PUSTAKA
2.1. Penelitian Terkait 6
2.2. *Machine Learning* (ML) 9
2.3. *Deep learning* 10
2.4. *Neural Machine Translation* (NMT) 10
2.5. *Recurrent Neural Network* (RNN) 11
2.6. *Long Short-Term Memory* (LSTM) 11
2.7. Fungsi Aktivasi 13
2.8. Terjemahan Mesin 14
2.9. Bahasa 15
2.10. Bahasa Nasional dan Bahasa Daerah 15
2.11. Bahasa Indonesia 15

2.12. Bahasa Ternate 16
2.13. Aplikasi *Mobile* 16
2.14. *Bilingual Language Understudy* (BLEU) 16
2.15. *Android* 17
2.16. *Android Kotlin* 17
2.17. *Python* 18
2.18. *Flask* 18
2.19. *Prototype* 19
BAB III METODE PENELITIAN
3.1. Objek dan Waktu Penelitian 20
3.2. Pengembangan Perangkat Lunak 20
3.3. Alur Penelitian 21
3.4. Metode Pengumpulan Data 21
3.5. Alat dan Bahan Penelitian 22
3.6. Implementasi NMT 23
3.6.1. *Preprocessing Text* 23
3.6.2. Pemodelan Arsitektur LSTM 26
3.6.3. *Deploy Model* 27
3.7. Contoh Perhitungan LSTM 27
3.8. Evaluasi Hasil Terjemahan dengan BLEU 30
BAB IV HASIL DAN PEMBAHASAN
4.1. Analisis Data 31
4.2. *Preprocessing* Data 31
4.2.1. *Case Folding* 32
4.2.2. *Special Tokens* 33
4.2.3. Tokenisasi dan *Padding* 35
4.3. Latih Model 37
4.4. *Deploy* Model 45
4.5. Implementasi Sistem 46
4.5.1. Tampilan Beranda 46
4.5.2. Tampilan Utama Terjemahan Bahasa Indonesia – Bahasa Ternate 46

4.5.3. Tampilan Hasil Terjemahan Bahasa Indonesia – Bahasa Ternate 47
4.6. Hasil Analisis 48
4.7. Pembahasan 49
BAB V PENUTUP
5.1. Kesimpulan 50
5.2. Saran 50
DAFTAR PUSTAKA
LAMPIRAN

DAFTAR GAMBAR

Halaman

Gambar 3.1. Model *Prototype* Sistem Terjemahan 20
Gambar 3.2. Alur Penelitian 21
Gambar 3.3. Tahapan Metode NMT 23
Gambar 3.4. Arsitektur LSTM 26
Gambar 3.5. Jaringan LSTM 27
Gambar 4.1. Kode Program Sebelum *Case Folding* 32
Gambar 4.2. Kode Program *Special Tokens* 33
Gambar 4.3. Membaca data 35
Gambar 4.4. Tokenisasi dan *Padding* Untuk Bahasa Indonesia 35
Gambar 4.5. Tokenisasi dan *Padding* Untuk Bahasa Indonesia 36
Gambar 4.6. Menghitung Ukuran Kosakata 36
Gambar 4.7. Pembagian Dataset dan Arsitektur *Encoder* NMT 37
Gambar 4.8. Implementasi *Bahdanau Attention Layer* 38
Gambar 4.9. Arsitektur Model *Decoder* NMT 39
Gambar 4.10. Proses Pelatihan 39
Gambar 4.11. Demonstrasi Fungsi Prediksi NMT 40
Gambar 4.12. Proses Pelatihan Model NMT 42
Gambar 4.13. Hasil *Running* 43
Gambar 4.14. Fungsi Terjemahan NMT 44
Gambar 4.15. Hasil Terjemahan 45
Gambar 4.16. API *Flask* 45
Gambar 4.17. Tampilan Beranda 46
Gambar 4.18. Tampilan Utama Terjemahan Bahasa Indonesia – Bahasa Ternate 47
Gambar 4.19. Tampilan Hasil Terjemahan Bahasa Indonesia – Bahasa Ternate 48

DAFTAR TABEL

Halaman

Tabel 2.1. Penelitian Terkait 6
Tabel 3.1. Contoh proses *Case Folding* Bahasa Indonesia – Bahasa Ternate 22
Tabel 3.2. Contoh proses *Tokenization* Bahasa Indonesia – Bahasa Ternate 22
Tabel 3.3. Data *Input* 24
Tabel 3.4. Bobot 24
Tabel 3.5. Spesifikasi *Hardware* 28
Tabel 3.6. Spesifikasi *Software* 28
Tabel 3.7. Hasil BLEU *Scores* 30
Tabel 4.1. *Korpus paralel* Bahasa Indonesia – Bahasa Ternate 31
Tabel 4.2. Contoh Data Sebelum *Case Folding* 32
Tabel 4.3. Contoh Data Setelah *Case Folding* 33
Tabel 4.4. Sebelum Dan Sesudah *Special Tokens* BOS 34
Tabel 4.5. Sebelum Dan Sesudah *Special Tokens* EOS 34

ABSTRAK

IMPLEMENTASI METODE *NEURAL MACHINE TRANSLATION* DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

Fiska Nabila[1], Yasir Muin[2] , Amal Khairan[3]
Program Studi Informatika, Fakultas Teknik, Universitas Khairun
Jl. Jati Metro, Kota Ternate Selatan
*Email: [1]nabilarizmy12@gmail.com, [2]yasirmuin@unkhair.ac.id, [3]amalkhairan@unkhair.ac.id

Di Maluku Utara terdapat 19 Bahasa Daerah yang sudah dipetakan daya hidupnya. Namun, daya hidup Bahasa Daerah ini ada yang mengalami kemunduran dan terancam punah. Salah satu Bahasa yang terancam punah adalah Bahasa Ternate, ini disebabkan karena penutur jatinya tidak lagi mewariska Bahasa Daerah Ternate kepada generasi berikutnya. Saat ini, *machine translation* merupakan salah satu opsi yang dapat digunakan untuk membantu permasalahan terkait dengan Bahasa. *Machine translation* adalah alat penerjemah otomatis dari sebuah bahasa sumber ke bahasa target. Penelitian ini membangun sebuah penerjemah bahasa Indonesia ke bahasa Ternate dengan menggunakan arsitektur *encoder* dan *decoder Long short-term memory* (LSTM). Metode penelitian dimulai dengan pengumpulan *dataset*, *preprocessing* data, pemodelan dan pelatihan LSTM, evaluasi sistem menggunakan BLEU, dan implementasi sistem. *Dataset* yang diperoleh berjumlah 6.925 data dan 1732 data *training*. Dari hasil evaluasi skor BLEU diperoleh nilai 64,92%. Meskipun skor ini menunjukkan bahwa model mampu menghasilkan terjemahan yang cukup baik, masih terdapat beberapa kekurangan yang perlu diperhatikan. Salah satu kendala utama yang mempengaruhi akurasi hasil terjemahan adalah keterbatasan data paralel Bahasa Indonesia - Bahasa Ternate. Jumlah data yang sedikit menghambat kemampuan model untuk belajar pola dan struktur bahasa secara efektif. Keterbatasan data ini berdampak pada kualitas model karena model tidak memiliki cukup data untuk mempelajari berbagai variasi dan konteks kalimat dalam bahasa Ternate, sehingga hasil terjemahan kurang natural dan kadang-kadang tidak akurat.

Kata Kunci: *Machine Translation,* LSTM, BLEU, Bahasa Ternate.

IMPLEMENTATION NEURAL MACHINE TRANSLATION USING THE LSTM METHOD IN THE PROCESS OF TRANSLATING INDONESIAN TO TERNATE LANGUAGE

In North Maluku*, 19 regional languages have been mapped in terms of their vitality. However, some of these languages are experiencing a decline and are at risk of extinction. One of the endangered languages is* Ternate*, largely because native speakers are no longer passing it down to the next generation. Machine translation has become one option to help address language-related issues. Machine translation is an automatic translation tool that translates a source language into a target language. This study builds an Indonesian-to-*Ternate *language translator using the Long Short-Term Memory (LSTM) encoder-decoder architecture. The research method includes dataset collection, data preprocessing, LSTM modeling and training, system evaluation using BLEU scores, and system implementation. The dataset obtained consists of 6,925 entries, with 1,732 used for training. The evaluation results show a BLEU score of 64.92%. Although this score indicates that the model can produce fairly good translations, there are still some shortcomings to consider. One of the main challenges affecting translation accuracy is the limited amount of parallel data in Indonesian-*Ternate*. The small amount of data hampers the model's ability to effectively learn the language patterns and structures. This limitation impacts the model's quality, as it does not have enough data to learn*

various sentence variations and contexts in the Ternate language, resulting in translations that are less natural and sometimes inaccurate.

***Keywords**: Machine learning, LSTM, BLEU,* Ternate *language*.

BAB I

PENDAHULUAN

1.1. Latar belakang

Keanekaragaman etnis di Indonesia menciptakan beragamnya bahasa yang berasal dari masing-masing kelompok etnis. Bahasa-bahasa ini, umumnya disebut sebagai bahasa daerah, mencerminkan kekayaan budaya yang ada di negara ini. Situasi dan kedudukan bahasa daerah dalam spektrum komunitas beragam, sebagian besar diterima dan digunakan luas, sementara yang lain hanya tersebar di antara kelompok minoritas. Tantangan utama yang dihadapi oleh bahasa daerah yang berada dalam kelompok minoritas adalah jumlah penuturnya yang terbatas. Kondisi ini berpotensi memicu perubahan bahasa atau penurunan penggunaanya jika tidak dipedulikan (Inun, 2021).

Hal ini juga terjadi di Maluku Utara yang dimana terdapat 19 bahasa daerah yang sudah dipetakan daya hidupnya. Namun, daya hidup bahasa daerah ini ada yang mengalami kemunduran dan terancam punah. Menurut Kepala Kantor Bahasa Provinsi Maluku Utara, saat ini Bahasa Ternate dan Bahasa Makean Timur terancam punah. Penyebab utama kepunahan bahasa daerah adalah penutur jatinya tidak lagi menggunakan dan mewariskan bahasanya kepada generasi berikutnya (Kusnadi, 2022).

Bahasa Ternate, kebanyakan penuturnya yaitu masyarakat yang dari kalangan tua namun dengan jumlah yang sedikit, merosotnya jumlah penutur ini dapat menyebabkan bahasa Ternate lama kelamaan akan punah. Selain jumlah penutur yang sedikit, kaum milenial lebih sering menggunakan bahasa umum yang gaul dibandingkan dengan bahasa Ternate sendiri, bahkan kalangan tua cenderung menggunakan bahasa melayu ternate jika berinteraksi dengan kalangan muda, karena kebanyakan kalangan muda tidak mengerti

bahasa Ternate. Oleh karena itu, dibutuhkan sistem untuk membantu menerjemahkan. Salah satu pendekatan yang dapat digunakan untuk membantu permasalahan tersebut yaitu dengan *Machine Translation*. *Machine Translation* (mesin penerjemah) adalah alat penerjemah otomatis dari sebuah bahasa sumber ke bahasa lain atau bahasa tujuan (Aristyanto, 2021).

Mesin penerjemah jaringan saraf tiruan, *Neural Machine Translation* (NMT) adalah metode terjemahan mesin yang memanfaatkan jaringan saraf tiruan yang kompleks untuk memprediksi urutan kata dalam konteks kalimat. NMT bertugas mengonversi bahasa sumber ke dalam vektor dengan panjang yang tetap (Gunawan, 2021). Pendekatan NMT tidak bisa begitu saja melakukan suatu penerjemahan bahasa, pada sebuah NMT berisi susunan model, lapisan proses yang terdiri dari metode yang digunakan. NMT menggunakan sebuah pemodelan yang disebut dengan *Sequence To Sequence* (*SeqToSeq*) yang dapat menunjang terjadinya proses terjemahan bahasa. Pada model *SeqToSeq* terdapat dua buah tahapan yaitu *encoder* dan *decoder,* lapisan *encoder* dan *decoder* menerapkan LSTM sebagai arsitekturnya, dimana *encoder* merupakan sebuah lapisan proses yang akan menjadi masuknya bahasa sumber dan *decoder* merupakan lapisan proses hasil terjemahan dari lapisan *encoder* yang diubah menjadi sebuah bahasa target atau bahasa terjemah (Ramadhan, 2022).

Metode yang sama juga digunakan oleh (Fauziyah, 2022) dengan judul Implementasi *Neural Machine Translation* pada bahasa Inggris – Bahasa Sunda Dengan Menggunakan Memori Jangka Pendek Panjang (LSTM). Dalam penelitian ini menggunakan *Recurrent Neural Network* yang dimodifikasi sebagai model yaitu *Long Short-Term Memory* (LSTM). Model LSTM baik untuk menangkap informasi suatu *sequence* karena kemampuannya

menyimpan sebagian memori sehingga dapat menyelesaikan masalah *vanishing gradien*t yang terjadi pada *Plain Recurrent Neural Network*. Masukkan dari model ini akan berupa kalimat Bahasa Inggris yang divektorkan dan keluarannya juga berupa kalimat Bahasa Sunda yang divektorkan. Performa model ini cukup baik dengan akurasi 0,99 pada pelatihan dan pengujian serta kerugian kurang dari 0,1 pada pelatihan dan pengujian. Model ini juga mencapai skor BLEU rata-rata 0,8 untuk data pelatihan dan pengujian.

Berdasarkan uraian di atas, maka penelitian ini bertujuan untuk bagaimana mengimplementasikan Metode LSTM untuk menerjemahkan Bahasa Indonesia ke Bahasa Ternate.

1.2. Rumusan Masalah

Berdasarkan temuan pertanyaan pada latar belakang diatas peneliti merumuskan permasalahan yaitu bagaimana mengimplementasi metode *Long Short-Term Memory* (LSTM) untuk menerjemahkan Bahasa Indonesia ke Bahasa Ternate.

1.3. Batasan Masalah

Batasan masalah dalam penelitian ini yaitu sebagai berikut:

1. Penelitian ini hanya berfokus pada hasil terjemahan bahasa Indonesia ke bahasa Ternate.
2. Penelitian ini akan membatasi perhatian pada implementasi algoritma ke dalam perangkat mobile yaitu pada platform *mobile* (*android*).
3. Sumber data yang dipakai berasal dari kantor Bahasa Provinsi Maluku Utara.

1.4. Tujuan Penelitian

Berdasarkan masalah yang dirumuskan diatas, tujuan dari penelitian ini yaitu untuk mengimplementasi metode *Long Short-Term memory* (LSTM) untuk menerjemahkan

Bahasa Indonesia ke Bahasa Ternate.

1.5. Manfaat Penelitian

Manfaat yang di dapat dari penelitian ini yaitu:

1. Dapat menjadi alat yang sangat berguna untuk memfasilitasi komunikasi antar-bahasa. Ini dapat membantu orang-orang yang tidak fasih dalam bahasa Ternate untuk berkomunikasi lebih mudah dengan masyarakat yang menggunakan bahasa tersebut.
2. Melalui penelitian ini dapat membantu memperkenalkan dan melestarikan warisan budaya dan keunikan bahasa Ternate, memastikan bahwa nuansa dan makna lokal tetap terwakili.
3. Sebagai media pembelajaran.

1.6. Sistematika Penulisan

Sistematis penulisan skripsi ini merupakan pembahasan singkat dari setiap bab yang menjelaskan hubungan antara bab satu dengan bab yang lainnya, yaitu sebagai berikut:

BAB I PENDAHULUAN

Terdiri dari latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematis penulisan.

BAB II TINJAUAN PUSTAKA

menerapkan teori-teori yang didapat dari sumber-sumber relevan untuk digunakan sebagai panduan dalam penelitian serta menyusun skrispi ini.

BAB III METODE PENELITIAN

Bab ini membahas tentang metode penelitian yang telah dilakukan oleh penulis dengan permasalahan yang diangkat.

BAB IV HASIL DAN PEMBAHASAN

Bab ini membahas tentang hasil dari penelitian yang telah dilakukan oleh penulis dengan menggunakan metode LSTM.

BAB V PENUTUP

Memuat kesimpulan dari hasil penelitian yang telah dilakukan, dan saran untuk penelitian selanjutnya mengenai topik terkait.

BAB II

TINJAUAN PUSTAKA

2.1. Penelitian Terkait

Dalam penyusunan skripsi ini, penulis terinspirasi dan mereferensi dari penelitian-penelitian sebelumnya yang terkait dengan latar belakang masalah pada skripsi ini. Adapun penelitian yang berhubungan dengan skripsi ini dapat dilihat pada tabel 2.1.

Tabel 2.1 Penelitan Terkait

No	Nama dan Tahun	Judul	Hasil Penelitian
1	Yustiana Fauziyah, Ridwan Ilyas (2022)	Mesin penterjemah bahasa indonesia-bahasa sunda menggunakan *recurrent neural networks*	Penelitian ini mengusulkan sistem penerjemah Bahasa Indonesia ke Bahasa Sunda menggunakan *Neural Machine Translation* (NMT) dengan RNN. Dengan pengujian, mereka menemukan bahwa model GRU memberikan akurasi terbaik 99.17%, sementara penggunaan *Attention* meningkatkan menjadi 99.94%. Metode *optimasi Adam* memberikan hasil terbaik dengan akurasi 99.35%. BLEU *Score* menghasilkan nilai optimal 92.63% dengan *brevity penalty* 0.929. Ini menunjukkan bahwa NMT dengan arsitektur RNN, *Attention*, dan metode optimasi yang tepat efektif untuk terjemahan Bahasa Indonesia-Bahasa Sunda.
2	Leni Purwaningsih, Endang Wahyu Pamungkas (2021)	Perbandingan kinerja sistem *Neural Machine Translation OpenNMT* dan THUMT dalam eksperimen	penelitian ini telah menghasilkan *dataset* penerjemahan bahasa Jawa ngoko - krama sebanyak 8 berkas, yang

		menerjemahkan bahasa Jawa Ngoko - Krama	terdiri dari total 2000 kalimat data latih, 1000 kalimat data validasi, 200 kalimat data uji 1, dan 100 kalimat data uji 2. Kemudian, penelitian ini juga 15 menghasilkan skor BLEU dari pengukuran hasil terjemahan *OpenNMT* sebesar 98,55 untuk data uji 1 dan 2,01 untuk data uji 2, serta untuk hasil terjemahan THUMT sebesar 99,47 untuk data uji 1 dan 1,6 untuk data uji 2. Terakhir, penelitian ini juga menghasilkan aplikasi penerjemahan bahasa Jawa ngoko – krama bernama "*Nerjemahake*" yang dibuat menggunakan bahasa pemrograman *Python* dan pustaka *Python* yaitu *Streamlit*.
3	Muhammad Yusuf Aristyanto, Robert Kurniawan (2021)	Pengembangan metode *Neural Machine Translation* Berdasarkan *Hyperparameter Neural Network*	Hasil pengembangan model NMT yang akan terbagi menjadi 4 model, antara lain model dasar dari *Bahdanau*, kemudian model yang dioptimalkan dengan metode *Callbaks*, lalu model dengan penambahan *layer Dropout*, dan terakhir adalah model yang dioptimalkan dengan metode *Cross Validation*. Model yang telah dikembangkan dengan penambahan *callbacks*, *layer dropout*, dan metode *cross validation* dapat mengatasi *overfitting* pada model dasar. Sehingga mendapatkan akurasi sebesar 71,95% dan skor

			BLEU sebesar 43,11% pada data *test*. Kemudian hasil dari simulasi *hyperparameter* antara lain mendapatkan ukuran dari masing-masing *hyperparameter* sebagai berikut, *batch size* (64), *epoch* (10), *optimizer* (SGD), *activation function* (*SoftMax*), dan *dropout rate* (0,2). Yang terakhir, implementasi model dan *hyperparameter* terbaik pada *dataset corpus parallel* Bahasa Spanyol-Inggris menghasilkan skor BLEU yang sangat mirip dengan dataset *corpus parallel* Bahasa Jerman-Inggris, yakni 0,453146 pada data *train* dan 0,457280 data *test*.
4.	Fadel Razsiah (2023)	Aplikasi penerjemah Bahasa Bangka - Indonesia - Inggris berbasis *website* dengan *Neural Machine Translation*	Penelitian ini menghasilkan aplikasi berbasis *website* untuk melakukan proses penerjemahan berdasarkan model. Evaluasi pada model menggunakan BLEU *Score*. Dari proses evaluasi mendapatkan hasil 55,32% untuk model Bangka – Indonesia, Bangka – Inggris sebesar 64,73%, 54,5% untuk model Indonesia – Bangka, model Indonesia – Inggris sebesar 55,83%, 40,2% untuk model Inggris – Bangka dan model Inggris – Indonesia sebesar 37,32%. Dari hasil tersebut didapatkan bahwa nilai BLEU *Score* tertinggi didapatkan oleh model Bangka – Inggris yaitu sebesar 64,73% dan nilai

			BLEU *Score* terendah didapatkan oleh model Inggris – Indonesia 37,32%.
5.	Khen Dedes (2022)	*Neural Machine Translation of* Spanish-English *Food Recipes Using* LSTM	Metode LSTM dapat menerjemahkan resep makanan dari bahasa Inggris ke bahasa Spanyol atau sebaliknya dari Bahasa Spanyol ke bahasa Inggris berdasarkan hasil penelitian. *Encoder-decoder* LSTM berhasil memperoleh BLEU dengan nilai terbaik yaitu pada saat penerjemahan Bahasa Spanyol ke Bahasa Inggris dengan 1000 *epoch* dengan menggunakan 70%:30%. komposisi data sebesar 0.99842 dengan BLEU 1 karena bahasa inggris lebih pendek dari bahasa Spanyol. Selanjutnya, satu kata dalam bahasa Inggris mungkin memiliki lebih dari satu terjemahan bahasa Spanyol, mengurangi akurasi terjemahan. Untuk penelitian selanjutnya peneliti dapat menggunakan data di *filter* agar hanya muncul satu kali dan tidak berulang dirinya sendiri dalam kumpulan data.

2.2. *Machine Learning* (ML)

Machine Learning atau pembelajaran mesin merupakan cabang dari kecerdasan buatan AI yang berfokus pada pengembangan algoritma dan model dan memungkinkan komputer untuk belajar, membuat prediksi atau keputusan berdasarkan data. Pada *machine*

learning dilengkapi sejumlah aturan program yang dijalankan oleh algoritma, oleh karena itu pada teknik mesin belajar dapat dikategorikan sebagai instruksi yang dijalankan dan dipelajari secara otomatis untuk menghasilkan *output* yang optimal, hal ini dilakukan secara otomatis tanpa ada campur tangan manusia sedikitpun. Semua dilakukan secara otomatis untuk mengubah data menjadi beberapa pola dan diinputkan ke dalam *system* untuk mendeteksi masalah otomatis (Raup, 2022).

2.3. *Deep Learning*

Deep Learning adalah bagian dari kecerdasan buatan dan *machine learning* yang merupakan pengembangan dari *neural network multiple layer* untuk memberikan ketepatan tugas seperti deteksi objek, pengenalan suara, terjemahan bahasa dan lain–lain. *Deep Learning* berbeda dari teknik *machine learning* yang tradisional, karena *deep learning* secara otomatis melakukan representasi dari data seperti gambar, video atau teks tanpa memperkenalkan aturan kode atau pengetahuan domain manusia. lain–lain. *Deep Learning* berbeda dari teknik *machine learning* yang tradisional, karena *deep learning* secara otomatis melakukan representasi dari data seperti gambar, video atau teks tanpa memperkenalkan aturan kode atau pengetahuan domain manusia. Selain itu, *Deep learning* merupakan subbidang *machine learning* yang algoritmanya terinspirasi dari struktur otak manusia. Dalam teknologi *deep learning*, menurut Zailani ada beberapa jenis algoritma yang digunakan, di antaranya adalah *Convolutional Neurat Network* (CNN), *Recurrent Neural Network* (RNN), *Long Short-Term Memory* (LSTM), dan lain lain (Raup,2022).

2.4. *Neural Machine Translation* (NMT)

Neural Machine Translation (NMT) merupakan metode penerjemahan bahasa yang menggunakan *Neural Network*. *Neural Network* adalah suatu metode yang mengolah

informasi atau data yang terinspirasi dari cara kerja jaringan otak. *Neural Network* memiliki susunan berupa *node* dan relasi. Ada tiga jenis *node* diantaranya adalah *input layer*, *hidden layer*, dan *output layer*. Sedangkan fungsi dari relasi adalah untuk menggabungkan lebih dari satu *node* dengan suatu bobot yang memiliki arah aliran dalam suatu proses tertentu. *Neural Network* bermanfaat dalam memecahkan masalah terkait dengan pengenalan tren atau pola, prediksi, data *mining*, dan klasifikasi. Salah satu keuntungannya adalah keandalannya dalam mengolah data (Rasziah, 2023).

2.5. *Reccurrent Neural Network* (RNN)

RNN atau *Recurrent Neural Network* merupakan suatu metode dalam *deep learning* yang digunakan untuk memproses data sekuensial dengan pemanggilan berulang. Pada arsitektur RNN memiliki beberapa jaringan *neuron* yang terbaik dari dirinya sendiri atau ke *neuron* di *layer input* sehingga jaringan dapat menyimpan nilai yang memberi dampak pada cara *input* untuk menghasilkan nilai sebelumnya ke dalam jaringan. Lapisan pertama memiliki bobot dari lapisan *input*, lalu lapisan selanjutnya akan menerima bobot dari lapisan sebelumnya (Rasziah, 2023).

2.6. *Long Short-Term Memory* (LSTM)

LSTM atau *Long Short-Term Memory* merupakan salah satu pendekatan terbaik dalam pembelajaran mesin untuk tugas terjemahan bahasa. LSTM berhasil diterapkan secara efektif dalam berbagai konteks pemodelan NMT sebagai varian dari jaringan saraf rekurensi (RNN). Keunggulan metode ini terletak pada kemampuannya untuk menyimpan dan mengakses informasi memori dalam jangka waktu yang panjang. LSTM mampu memahami hubungan yang kompleks antara data, serta memberikan kontribusi informasi yang sangat bernilai dalam proses menentukan hasil terjemahan.

RNN memiliki kelemahan yaitu pada kemampuan dalam mengingat, sehingga pada saat dihadapakan dengan data yang panjang, maka RNN akan kehilangan beberapa informasi pada data *input*. Hal ini disebabkan karena pada RNN terjadi *vanishing gradient* yang mengakibatkan nilai *weight* pada sebuah data menjadi sangat besar ataupun sangat besar, sehingga memperlambat proses *training* pada RNN (Razsiah, 2023).

LSTM telah menjadi solusi untuk RNN dengan menyediakan *cell state* sebagai cadangan atau *back-up* dari *hidden state* untuk melakukan pemrosesan penyimpanan secara berurutan. Kedua *state* tersebut dapat meningkatkan kemampuan memori RNN. Pada langkah t, *cell state* Ct berinteraksi dengan data dan *hidden state* melalui empat tahap gerbang. Gerbang- gerbang tersebut merupakan komponen utama dari LSTM.

Berikut beberapa komponen utama dalam LSTM beserta rumusnya:

1. *Forget Gate* (f_t):

   Forget gate memutuskan informasi apa saja yang harus dihapus atau dilupakan dari sel memori sebelumnya. Rumus dapat dilihat pada persamaan 2.1.

   $$f_t = \sigma\ (U_f \cdot x_t + W_f \cdot h_{t-1} + b_f \text{.....(2.1)}$$

2. *Input Gate* (i-t):

   Input gate menentukan informasi baru apa yang akan disimpan dalam sel memori. Ini melibatkan dua langkah: pertama, menentukan nilai yang akan di *update* (C~) dan kedua memutuskan seberapa banyak informasi baru yang akan diintegrasikan. Rumus dapat dilihat pada persamaan 2.2.

   $$i_t = \sigma\ (U_i .\ x_t + W_i\ .\ h_{t-1} + \ b_i) \text{.....(2.2)}$$

3. *Cell State* (C_t)

   Cell State menggabungkan informasi lama dari sel memori sebelumnya dengan

informasi baru yang dipilih oleh *input gate*. Rumus dapat dilihat pada persamaan 2.3.

$$C_t = f_t \cdot c_{t-1} + i_t \cdot c_t \quad \text{.....(2.3)}$$

4. *Output Gate* (O_t)

   Output gate menentukan *output* aktual dari sel memori dan mendefinisikan *output* dari *unit* LSTM. Rumus dapat dilihat pada persamaan 2.4.

$$o_t = \sigma\ (U_o\,.\,x_t\ +\ W_o\,.\,h_{t-1}\ +\ b_o)$$

$$h_t = o_t \cdot tanh\,(c_t) \quad \text{.....(2.4)}$$

Keterengan:

1. ht adalah *output* t,
2. xt adalah *output* t,
3. W dan b adalah parameter yang perlu dipelajari,
4. σ adalah fungsi *sigmoid* dan
5. *tanh* adalah fungsi *tangen hiperbolik*.

2.7. Fungsi Aktivasi

Fungsi aktivasi adalah suatu fungsi yang digunakan pada *neural network* untuk mengaktifkan dan menonaktifkan suatu *neuron* yang berguna untuk membantu *neural network* dalam mempelajari pola kompleks dalam data (Kurniawan, 2024). Ada beberapa fungsi aktivasi yang sering digunakan dalam berbagai aplikasi jaringan saraf, antara lain:

1. Fungsi logistik *sigmoid* digunakan untuk mengubah *input* apa pun menjadi *output* yang berada dalam rentang 0 sampai dengan 1. Fungsi logistik *sigmoid* digunakan dalam lapisan-lapisan jaringan saraf untuk menambahkan non-linearitas, yang memungkinkan jaringan untuk mempelajari dan memodelkan data yang kompleks (Kurniawan, 2024).

2. Fungsi aktivasi *tanh* adalah fungsi yang digunakan untuk membantu dalam mengatur nilai - nilai yang mengalir dalam jaringan. Fungsi ini mengubah *input* menjadi nilai antara -1 dan 1, sehingga memberikan *output* yang terpusat pada nol (Kurniawan, 2024).
3. Fungsi ReLu

   Fungsi ReLu (*rectified liner unit*) merupakan fungsi non-linier dimana pengaktifan *neuron* tidak dilakukan secara bersamaan, dan hanya ketika *output* dari transformasi linier bernilai nol (Kurniawan, 2024).
4. Fungsi *Softmax*

   Fungsi *softmax* merupakan fungsi aktivasi yang digunakan pada *layer output*. Fungsi aktivasi *softmax* digunakan untuk mendapatkan hasil klasifikasi. Fungsi aktivasi menghasilkan nilai yang diinterpretasi sebagai probabilitas yang belum dinormalisasi untuk tiap kelas. Nilai kelas dihitung dengan menggunakan fungsi aktivasi *softmax* (Romario, 2020).
5. Fungsi *swish*

   Fungsi aktivasi *swish* adalah salah satu fungsi yang digunakan dalam jaringan saraf tiruan (*neural networks*) untuk membantu model dalam belajar pola-pola dari data. *Swish* dikembangkan oleh para peneliti di *Google* sebagai alternatif untuk fungsi aktivasi ReLu (Kurniawan, 2024).

2.8. Terjemahan Mesin

Terjemahan mesin merupakan suatu proses menerjemahkan teks dari satu bahasa ke bahasa lain. Dalam konteks ini, pengguna menyampaikan teks dalam bahasa sumber kepada perangkat lunak terjemahan mesin, yang secara otomatis melakukan transfer teks

tersebut ke dalam bahasa target yang telah ditentukan (Aristyanto, 2021).

2.9. Bahasa

Bahasa merupakan sarana komunikasi esensial dalam kehidupan manusia. Melalui penggunaan bahasa, manusia dapat berinteraksi dan memperoleh informasi yang diperlukan. Selain itu, bahasa digunakan individu untuk mengkomunikasikan ide, gagasan, konsep, atau perasaan kepada pihak lain (Mailani, 2022).

2.10. Bahasa Nasional dan Bahasa Daerah

Bahasa nasional adalah bahasa yg menjadi bahasa standar atau *lingua franca* di negara yg mempunyai banyak bahasa karena perkembangan sejarah, kesepakatan bangsa, atau ketetapan perundang-undangan. Dalam kedudukannya sebagai bahasa nasional, bahasa Indonesia berfungsi sebagai lambang kebanggaan kebangsaan, indentitas nasional, media penghubung antarwarga, antardaerah dan antarbudaya, serta media pemersatu suku, budaya dan bahasa di Nusantara (Nasution, 2022).

Dalam rumusan Seminar Politik Bahasa (2003) disebutkan bahwa bahasa daerah adalah bahasa yang dipakai sebagai bahasa perhubungan intradaerah atau intramasyarakat di samping bahasa Indonesia dan yang dipakai sebagai sarana pendukung sastra serta budaya daerah atau masyarakat etnik di wilayah Republik Indonesia. Bahasa Indonesia, bahasa rumpun Melayu, dan bahasa asing tidak masuk dalam kategori bahasa daerah (Adi Budiwiyanto, 2022).

2.11. Bahasa Indonesia

Bahasa Indonesia adalah bahasa nasional Indonesia yang digunakan secara luas di seluruh wilayah negara Indonesia. Hal ini menjadikan bahasa Indonesia sebagai alat komunikasi yang umum digunakan di seluruh Indonesia, meskipun ada juga banyak bahasa

daerah yang dipertuturkan (Rif'at, 2019).

2.12. Bahasa Ternate

Bahasa Ternate dituturkan oleh masyarakat Ternate di Kota Ternate. Dalam perjalanan sejarahnya, bahasa ini pernah menjadi *lingua franca* atau bahasa pengantar di Maluku Utara. Untuk itu, dapat dilihat adanya kantong-kantong bahasa ini pada beberapa daerah, seperti di Pulau Kayoa, Bacan, Obi, dan juga tersebar di sepanjang pantai barat Halmahera (Duwila, 2021).

2.13. Aplikasi *Mobile*

Aplikasi *mobile* (*Mobile Apps*) adalah aplikasi yang dibuat khusus untuk perangkat bergerak seperti *Smartphone*, *SmartWatch*, Tablet, dan perangkat serupa. Umumnya, pembuatan dan perancangan aplikasi *mobile* melibatkan penggunaan bahasa pemrograman khusus. Perangkat lunak aplikasi, juga dikenal sebagai *software* aplikasi, merupakan hasil dari kegiatan pemrograman *mobile* yang dilakukan dengan menggunakan bahasa pemrograman tertentu (Agusti, 2022).

2.14. BLEU (*Bilingual Evaluation understudy)*

BLEU (*Bilingual Evaluation Understudy*) digunakan untuk menilai kualitas terjemahan mesin dari satu bahasa alami ke bahasa lain. Skor BLEU mengukur presisi n-gram yang diubah antara terjemahan otomatis dan referensi, menggunakan konstanta penalti singkat. Nilai BLEU dihitung dengan mengalikan penalti singkat dengan rata-rata geometrik dari skor presisi yang diubah, yang mencapai 292. Semakin tinggi nilai BLEU, semakin akurat referensinya. Rentang nilai BLEU berada antara 0 hingga 1. Sebuah terjemahan memperoleh nilai 1 jika identik dengan terjemahan referensi (Dedes, 2022). Rumus dapat dilihat pada persamaan 2.5 dibawah ini.

$$P_B = \begin{cases} 1, c > r \\ e\left(1 - \frac{r}{c}\right), c \leq r \end{cases} \text{.....(2.5)}$$

1. BP = Simbol *brevity penalty*
2. *c* = Jumlah kata-kata dari hasil penerjemahan mesin
3. *r* = Jangka panjang terjemahan rujukan yang efektif.
4. *p* = Rata-rata geometris dari presisi *n-gram* yang dimodifikasi
5. *N* yang digunakan adalah *N* = 4 dan *w* = $\frac{1}{N}$ *[8]*

2.15. *Android*

Android adalah sistem operasi (OS) paling populer saat ini datang ke *platform* seluler. Menurut Statistik *Global*, OS Android menikmati hampir 75% pangsa pasar di Industri OS Seluler, diikuti oleh iOS dengan pangsa 25% di Juni 2020. Pengguna lebih memilih *Android* karena gratis dan alam sumber terbuka dengan dukungan untuk banyak aplikasi. Pengembang juga pilih Android daripada iOS yang kompetitif karena bersifat *opensource*. Aplikasi untuk *Android* ditulis pertama kali menggunakan Bahasa pemrograman java dan biasanya disebut sebagai 'aplikasi (Mishra, 2022).

2.16. *Android Kotlin*

Kotlin adalah Bahasa pemrograman yang merupakan “penyempurnaan” dari Bahasa pemrograman Java untuk pengembangan aplikasi *Android*. *Kotlin* awalnya dikembangkan oleh *JetBrains*, perusahaan dibalik IntelliJ IDEA. Setelah melalui banyak perkembangan, *JetBrains* merilis *Kotlin* secara *open source* dan kini setelah perkembangannya semakin maju, *Google* mendukung penuh Kotlin untuk pengembangan aplikasi *Android*. Bahasa pemrograman yang relatif baru ini mengedepankan produktifitas, oleh sebab itu *Kotlin* memudahkan dalam pembuatan kode program (Uzayr, 2022).

2.17. *Python*

Pemrograman menggunakan *Python* bersifat *open-source*, yang artinya terbuka untuk umum dan dapat dikembangkan oleh siapa saja. *Python* dikategorikan sebagai bahasa tingkat tinggi yang mempermudah pemrograman berorientasi objek melalui pendekatan yang disederhanakan. *Python* sangat relevan bagi para ilmuwan data karena mendukung berbagai aplikasi ilmu data serta memiliki kemampuan yang sangat baik dalam menangani masalah matematika, statistik, dan fungsi ilmiah lainnya. Salah satu alasan utama popularitas *Python* dalam penelitian adalah kemudahan penggunaannya dan sintaksis yang sederhana, yang membuatnya mudah dipelajari oleh mereka yang tidak memiliki latar belakang teknik (Muhammad, 2023).

2.18. *Flask*

Flask adalah *framework web* berbasis *Python* yang termasuk dalam kategori *microframework*. *Flask* adalah kerangka aplikasi dan tampilan berbasis *web*. Pengembang dapat lebih mudah mengelola perilaku *web* dan membuat *web* terstruktur dengan memanfaatkan *flask* dan bahasa pemrograman *Python*. *Flask* adalah *microframework* karena tidak memerlukan alat atau pustaka khusus untuk bekerja. Sebagian besar fungsi dan komponen umum, seperti validasi *form* dan *database*, tidak ter-*install* secara *default* di *Flask*. Ini karena *Flask* dapat menggunakan ekstensi untuk membuat *fitur* dan komponen ini tampak diimplementasikan oleh *Flask* sendiri karena disediakan oleh pihak ketiga. Selain itu, hanya karena *Flask* disebut sebagai *microframework* tidak berarti *Flask* tidak memiliki fungsionalitas. *Microframework* ini mengacu untuk membuat inti aplikasi berpotensi sambil tetap membuat instalasi *add-on* menjadi sederhana. Dengan cara ini, *Flask* mengungguli *framework* lain dalam hal skalabilitas dan fleksibilitas (Rasziah, 2023).

2.19. *Prototype*

Model *prototyping* merupakan suatu teknik untuk mengumpulkan informasi tertentu mengenai kebutuhan-kebutuhan informasi pengguna secara cepat. Berfokus pada penyajian dari aspek-aspek perangkat lunak tersebut yang akan nampak bagi pelanggan atau pemakai. Prototipe tersebut akan dievaluasi oleh pelanggan/pemakai dan dipakai untuk menyaring kebutuhan pengembangan perangkat lunak. *Prototype* didefinisikan sebagai alat yang memberikan ide bagi pembuat maupun pemakai potensial tentang cara sistem berfungsi dalam bentuk lengkapnya (Pricillia, 2021).

BAB III

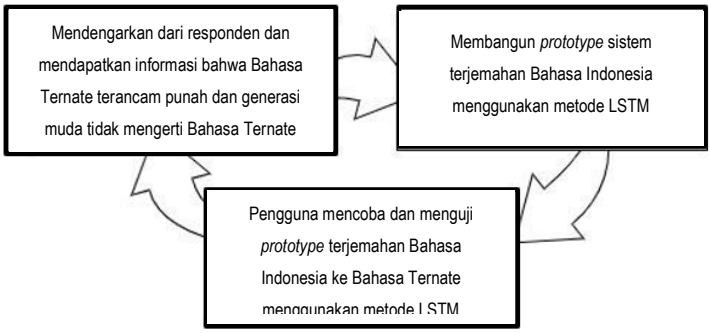
METODE PENELITIAN

3.1. Objek dan Waktu Penelitian

Objek yang diteliti pada penelitian ini yaitu Bahasa Ternate, yang dimana pada penelitian ini adalah implementasi *Neural Machine Translation* (NMT) dengan menggunakan metode *Long Short-Term Memory* (LSTM) pada proses penerjemahan dari bahasa Indonesia ke bahasa Ternate. Waktu penelitian yang dilakukan pada semester ganjil tahun ajaran 2022/2023.

3.2. Pengembangan Perangkat Lunak

Tahapan pengembangan sistem ini menggunakan metode pengembangan *prototype*. *Prototyping* merupakan proses yang digunakan untuk membantu pengembangan perangkat lunak dalam membentuk model dari perangkat lunak yang harus dibuat. Hal ini digunakan sebagai tahapan perancangan dalam memberi gambaran sistem yang dibuat nantinya, dapat dilihat pada gambar 3.1.



Gambar 3.1. Pemodelan *prototype* terjemahan Indonesia – Ternate

Tahapan-tahapan dalam model *prototyping* yaitu:

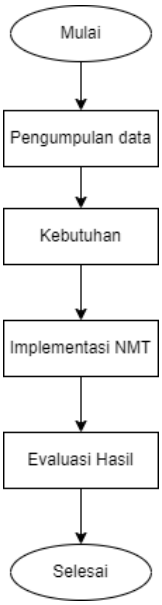
1. Mendengarkan responden dimana pada tahapan ini termasuk kegiatan pengumpulan data untuk dijadikan sampel. Peneliti mengambil data dari kantor Bahasa Provinsi

Maluku Utara dari buku yang berjudul “Kamus Praktis Indonesia – Ternate.

2. Pembuatan rancangan NMT pada proses penerjemahan Bahasa Indonesia ke Bahasa Ternate menggunakan metode LSTM, dimana pada tahapan ini termasuk analisis sistem, perancangan sistem, dan pembuatan aplikasi.
3. *Prototype* diuji coba oleh pengguna (orang yang menggunakan aplikasi) terhadap sistem yang telah dibuat, jika telah memenuhi maka pembuatan aplikasi ini telah *final* dan dapat digunakan oleh pengguna, jika belum maka akan dilakukan evaluasi dan perbaikan lagi.

3.3. Alur Penelitian

Berikut merupakan alur dari penelitian yang akan diteliti. Dapat dilihat pada gambar 3.2.



Gambar 3.2. Alur Penelitian

Berikut adalah penjelasan dari gambar 3.2 di atas:

3.4. Metode pengumpulan Data

Data yang digunakan dalam penelitian ini merupakan *korpus parallel* dari Bahasa

Indonesia – Bahasa Ternate yang bersumber dari buku dengan judul "Kamus Praktis Indonesia - Ternate" yang diambil dari Kantor Bahasa Provinsi Maluku Utara. Total data yang dikumpulkan dan dimasukkan ke dalam *database* berasal dari kamus tersebut mencakup 6.925 data, serta mencakup 24 fenom, yakni /a, b, c, d, e, f, g, h, I, j, k, l, m, n, o, p, r, s, t, u, v, w, y, z/.

3.5. Alat dan Bahan Penelitian

Dalam melakukan penelitian ini, ada beberapa spesifikasi alat penelitian yang harus dipenuhi. Spesifikasi alat penelitian maksudnya adalah standar minimal dari alat (*tools*) yang digunakan sebagai wadah utama dalam melakukan penelitian ini. Adapun spesifikasi minimum atau standar dapat dilihat pada tabel 3.1 dan 3.2.

Tabel 3.1 Spesifikasi *Hardware*

No	Jenis	Spesifikasi
1.	Laptop	HP Spectre x360 13
2.	*Processor*	Intel Core i5 gen 8
3.	Memori (RAM)	4 GB
4.	SSD	512 GB
5.	*System Type*	64-bit

Berikut merupakan spesifikasi *software* tabel 3.2.

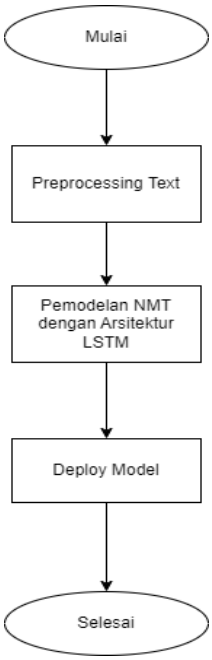
Tabel 3.2 Spesifikasi *Software*

No	Jenis	Spesifikasi	Keterangan
1.	Sistem Operasi	*Windows* 11	Sistem Operasi yang digunakan selama penelitian
2.	Aplikasi *Editor text*	*Python* Versi 3.9 *jupyter Notebook & Visual Studio Code*.	Digunakan untuk menulis dan menjalankan *script python*
3.	*Kotlin*	Versi 1.6	Digunakan untuk membuat atau mengembangkan aplikasi *mobile* yang dapat berjalan pada *device android*.
4.	Figma	Versi 116.14.7	Digunakan untuk membuat desain aplikasi.
5.	*Android Studio*	Versi 4.1. 2	Digunakan untuk menulis dan

			menjalankan *script python* dan membuat *interface android*.

3.6. Implementasi NMT menggunakan metode LSTM

Penelitian ini menggunakan arsitektur *Long Short-Term Memory* (LSTM) untuk menerjemahkan Bahasa Indonesia ke Bahasa Ternate. Dalam implementasi metode ini ada beberapa tahapan yang akan dilakukan seperti pada gambar 3.3.



Gambar 3.3 Tahapan NMT

Berdasarkan gambar 3.3 diatas menjelaskan mengenai tahapan NMT adalah sebagai berikut:

3.6.1. *Preprocessing text*

Tahapan *preprocessing* yaitu proses untuk menjadikan data yang sebelumnya tidak terstruktur menjadi terstruktur. Melalui proses *preprocessing* data mentah yang tersedia dalam dataset diolah dan di standarkan sehingga dapat diproses oleh algoritma yang akan digunakan.

1. *Case folding*

Case folding adalah proses dalam pemrosesan teks yang bertujuan untuk mengubah semua huruf dalam teks menjadi bentuk yang sama, yaitu huruf kecil. Misalnya, frasa "Saya", "SaYa", atau "Saya" dapat ditemukan dalam sebuah *dataset*. "Aku" memiliki arti yang sama bagi semua orang. Komputer, di sisi lain, mengartikan beberapa dari kata-kata ini sebagai kata yang berbeda meskipun memiliki arti yang sama karena kasus karakter yang berbeda. Sehingga untuk menghindari hal tersebut pada penilitian ini menggunakan satu standar yaitu dengan menyamakan semua data. Untuk lebih jelasnya dapat dilihat pada tabel 3.3.

Tabel 3.3. Contoh proses *Case Folding* pada bahasa Ternate – Indonesia

Sebelum	
Bahasa Ternate	**Bahasa Indonesia**
Ake oke	Minum air
Jujaru mina ge mu jang foloi	Gadis itu cantik sekali
Sesudah	
Bahasa Ternate	**Bahasa Indonesia**
ake oke	minum air
jujaru mina ge mu jang foloi	gadis itu cantik sekali

2. *Tokenization*

Proses *tokenization* adalah proses memisahkan teks menjadi potongan – potongan berupa token, bisa berupa potongan huruf, kata, atau kalimat. Entitas yang bisa disebut sebagai token misalnya kata, angka, simbol dan lain sebagainya. Dalam proses *tokenizer* menggunakan *word tokenizer* yaitu untuk memecah kalimat dalam bentuk kata per kata. Untuk lebih jelasnya dapat dilihat pada tabel 3.4.

Tabel 3.4. Contoh proses *Tokenization* pada Bahasa Indonesia – Bahasa Ternate

Sebelum	
Bahasa Ternate	**Bahasa Indonesia**
Ake oke	Minum air
Jujaru mina ge mu jang foloi	Gadis itu cantik sekali

Sesudah	
Bahasa Ternate	**Bahasa Indonesia**
[‘ake’, ‘oke’]	[‘minum’, ‘air’]
[‘jujaru’, ‘mina’, ‘ge’, ‘mu’,’jang’, ‘foloi’]	[‘gadis, ‘itu’, ‘cantik’, ‘sekali’]

3. *Indexing*

   Proses *indexing* merujuk pada langkah di mana setiap kata atau *token* dalam teks diubah menjadi representasi numerik. Tahap ini memiliki peran krusial dalam mengkonversi setiap kata dalam teks ke bentuk bilangan, memungkinkan pemrosesan oleh model. Hal ini disebabkan karena model *machine learning* atau *deep learning* memerlukan data yang bersifat numerik untuk dapat diolah secara efektif.

4. Konversi teks ke urutan

   Pada tahap ini yaitu mengganti setiap kata dalam kalimat indeks yang bersesuaian.

5. *Padding*

   Tahap berikutnya melibatkan proses penyesuaian panjang *sequences*, yang sering dikenal sebagai proses *padding*. Proses ini memiliki tujuan untuk menciptakan konsistensi panjang *sequence* di dalam setiap dokumen, memungkinkan penggunaan efisien pada model jaringan saraf tiruan. Konsep *padding* sendiri adalah memangkas *sequence* yang terlalu panjang dan menambahkan nilai 0 pada tiap *sequence* yang terlalu pendek, baik di bagian akhir maupun awal, sehingga sesuai dengan panjang maksimal *sequence* yang telah ditetapkan.

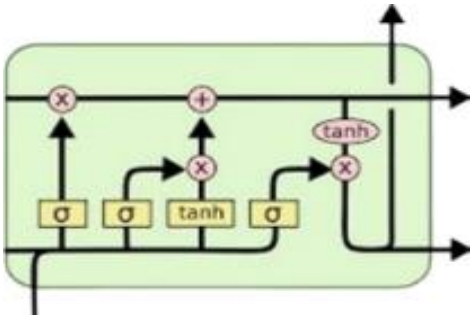
6. *Embedding*

   Proses *embedding* mengubah indeks numerik dari kata – kata menjadi vektor berdimensi tetap yang kaya dengan informasi semantik. Vektor ini berisi nilai – nilai *float* yang dapat dipelajari melalui pelatihan model atau diambil dari model yang sudah

terlatih.

3.6.2. Pemodelan Arsitektur LSTM

Pada tahap ini akan dilakukan pemodelan arsitektur LSTM seperti pada gambar berikut:



Gambar 3.4. Arsitektur LSTM

Untuk pemodelan LSTM ini terdiri dari 3 tahapan yaitu *encoder*, *mechanism attention*, dan *decoder.* Berikut ini merupakan penjelasan dari tahapan tersebut:

1. *Encoder*

   Pada tahap *encoder*, kalimat input yang merupakan kalimat bahasa sumber diubah menjadi vektor *embedding*. Vektor *embedding* kemudian diproses oleh LSTM untuk menghasilkan *sequence of hidden states.*

2. *Attention mechanism*

   Selanjutnya, tahap *attention mechanism* ini memungkinkan model untuk fokus pada bagian relevan dari kalimat sumber saat menghasilkan setiap kata di kalimat target. Pada tahap ini yang dilakukan yaitu menghitung skor antara *hidden state encoder* dan *hidden state decoder*, kemudian menggunakan *softmax* pada skor untuk mendapatkan *weights* yang menunjukan seberapa penting setiap *hidden state encoder*, dan menghitung *context vector* sebagai *weighted sum* dari *hidden state encoder* berdasarkan *alignment weights*.

3. *Decoder*

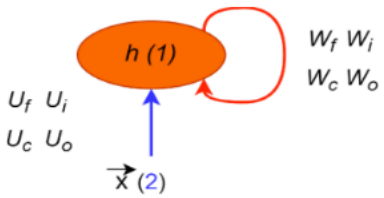
Pada tahap *decoder* ini yaitu menghasilkan terjemahan dalam bahasa target dengan memproses urutan token dan *context vector*, di setiap langkah waktu *input decoder* menggabungkan *context vector* dari mekanisme *attention*, *hidden state* sebelumnya, dan kata sebelumnya (dimulai dengan token <bos>). *Decoder* LSTM kemudian memproses *input* tersebut untuk menghasilkan *hidden state* yang baru dan memprediksi kata berikutnya sampai mencapai kata akhir khusus (EOS yang berarti "*end of sentence*").

3.6.3. *Deploy model*

Deploy model digunakan untuk menerapkan proses yang telah di latih oleh model agar dapat digunakan secara berkala tanpa perlu melatih kembali, disini model di *deploy* dengan *flask* pada *port* tertentu kemudian dihubungkan dengan *ngrok* agar nanti dari *interface android* dapat mengirim kalimat yang ingin diterjemahkan secara *online* ke domain yang di sediakan oleh *ngrok* yang terhubung dengan model yang telah di *deploy*, yang dimana nantinya kalimat tersebut terjemahannya kemudian dikembalikan ke *interface android*, setelah menangkap maka akan ditampilkan hasil terjemahan di tampilan (kolom hasil terjemahan).

3.7. Contoh perhitungan LSTM

Dibawah ini merupakan contoh simulasi perhitungan algoitma LSTM.



Gambar 3.5 Jaringan LSTM

Pada gambar 3.5 jaringan LSTM terdapat 2 dimensi input dengan 1 *unit* sel LSTM. Pada contoh kasus ini diasumsikan bahwa kita suda memiliki model LSTM berupa kumpulan matriks bobot U, W dan b. untuk lebih jelasnya dapat dilihat pada tabel 3.5 dan 3.6. Berikut adalah tabel 3.5 data *input.*

Tabel 3.5 Data *Input*

A1	A2	Target
1	2	0,5
0,5	3	1

Pada tabel 3.5 merupakan contoh data *input* dengan terdapat 2 fitur yaitu A1 dan A2 serta terdapat target dari setiap baris data.

Berikut adalah tabel 3.6 bobot.

Tabel 3.6 Bobot

Bobot			
U_f		W_f	b_f
0,700	0,450	0,100	0,150
U_i		W_i	b_i
0,950	0,800	0,800	0,650
U_c		W_c	b_c
0,450	0,250	0,150	0,200
U_o		W_o	b_o
0,600	0,400	0,250	0,100

Tabel diatas memuat himpunan bobot U, W, dan b. Bobot-bobot ini dihasilkan dari pelatihan model jaringan saraf.

Diketahui h_{t-1} = 0 dan c_{t-1} = 0

1. *Forget Gate*

Kita menghitung nilai *forget gate* (f_t).

$f_t = \sigma\ (U_f \cdot x_t + W_f \cdot x_{h_{t-1}} + b_f)$

$= \sigma\ ([0{,}700\ \ 0{,}450]) \times \begin{bmatrix}1\\2\end{bmatrix} + (0{,}100\ \times 0) + 0{,}150)$

$= \sigma\ (1{,}750)$

$= \frac{1}{1 + e^{(-1{,}750)}}$ = 0.852

2. *Input Gate*

Selanjutnya menghitung nilai *input gate* (i_t).

$i_t = \sigma\ (U_i\ .\ x_t + W_i\ .\ h_{t-1} + b_i)$

$= \sigma\left([0{,}950 \quad 0{,}800] \times \left[\frac{1}{2}\right] + (0{,}800 \times 0) + 0{,}650\right)$

$= \sigma\ (3{,}200)$

$= \frac{1}{1 + e^{(-3{,}200)}}$ = 0,961

$\tilde{c}$ = *Tanh* $(U_c\ .\ x_t + W_c\ .\ h_{t-1} + b_{\square})$

= *Tanh* $\left([0{,}450 \quad 0{,}250] \times \left[\frac{1}{2}\right] + (0{,}150 \times 0) + 0{,}200\right)$

= *Tanh* $(1{,}150)$

$= \frac{e^{2\ x\ 1{,}150} - 1}{e^{2\ x\ 1{,}150} + 1}$ = 0,818

3. *Cell State*

Setelah mendapatkan nilai *Input Gate* dan *Forget Gate*, maka kita dapat menghitung memori sel *new state*.

$C_t = f_t\ .C_{t-1} + i_t\ .\tilde{C}_t$

$= (0{,}852\ x\ 0) + (0{,}961\ x\ 0{,}818)$

= 0,786

4. *Output Gate*

$o_t = \sigma\ (U_o\ .\ x_t + W_o\ .\ h_{t-1} + b_o)$

$= \sigma\left([0{,}600 \quad 0{,}400] \times \left[\frac{1}{2}\right] + (0{,}250 \times 0) + 0{,}100\right)$

$= \sigma\ (1{,}500)$

$= \frac{1}{1 + e^{(-1{,}500)}}$ = 0,818

$h_t = o_t\ .\ Tanh\ (c_t)$

= 0,818 × *Tanh* $(0{,}786)$ = 0,53

3.8. Evaluasi Hasil Terjemahan dengan BLEU

Pengujian hasil terjemahan dilakukan dengan otomatis. Pengujian dilakukan untuk mendapatkan keakuratan mesin penerjemah berdasarkan nilai akurasi. Proses pengujian secara otomatis pada penelitian ini menggunakan skor BLEU. BLEU mengukur nilai *modified n-gram precision score* antara hasil terjemahan mesin dengan terjemahan referensi dan menggunakan konstanta yang disebut *brevity penalty.* Nilai BLEU adalah hasil dari perkalian antara rata rata geometri dari *modified precision score dengan brevity penalty*. Evaluasi hasil BLEU dapat dilihat pada tabel 3.7.

Tabel 3.7 Hasil BLEU *Scores* Bahasa Indonesia – Bahasa Ternate

Hasil BLEU *Scores*	
Jumlah *Dataset*	6925
Epoch	200
Batch Size	64
Bleu *Scores*	64.92%

BAB IV

HASIL DAN PEMBAHASAN

4.1. Analisis data

Data yang digunakan pada penelitian ini bersumber langsung dari Kantor Bahasa Provinsi Maluku Utara, *dataset* ini berupa bahasa Indonesia dan bahasa Ternate, untuk lebih jelasnya bisa dilihat pada tabel 4.1.

Tabel 4.1 *Korpus parallel* Bahasa Indonesia - Bahasa Ternate

Bahasa Indonesia	Bahasa Ternate
Ayah	Baba, aba
Satu abad	Taong ratu moi
Dua abad	Taong ratu romdidi
Hidup kekal di akhirat	Ahu kekal ge toma akhirat
Sebelum shalat kita harus membersihkan diri	Kalu sari sabea harus sigofi nadiri
Rakyat jelata mengabdi pada sultan	Bala ge co'ou se kolano
Jangan putus asa	Afa tola harapang
Orang itu berasal dari jawa	Mancia nagee na asal toma jawa
Dia sangat cantik	Mina jang foloi
Masjid itu kubahnya bagus	Sigi ena ge ma kuba jang
Harga minyak naik	Goroho ma ija fere
Wanita muslim harus memakai kerudung	Foheka muslim musti pake gugudu
Jangan khawatir	Khawatir afa
Sekarang ini musim hujan	Kanang ne musim besa
Nenek saya pergi ke pasar	Ri ere mutagi toma butu
Wajahnya mirip ayahnya	Igunaga doka mababa
Hari minggu sekolah libur	Ahad mawange skolah ifrei
Dia seorang ahli Sejarah	Una ge u-ahli sajarah
Saya tidak setuju	Fangare tonyiha ua
Anak itu akhlaknya baik	Ngofa una ge i-akhlak jang

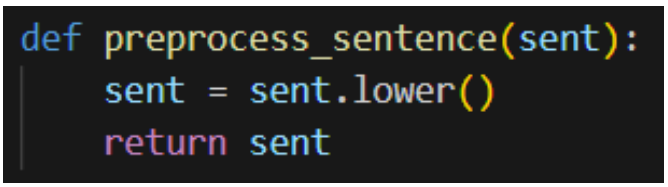
Data yang diperoleh oleh peneliti berupa data dalam bentuk buku kemudian disalin kedalam *excel* dengan total 6925 data bahasa indonesia dan bahasa ternate.

4.2. *Preprocessing text*

Dalam proses ini akan dilakukan *preprocessing text* Bahasa Indonesia dan Bahasa ternate.

4.2.1. *Case Folding*

Proses ini adalah proses untuk menyamakan data dengan kata yang sama tetapi kasus yang berbeda. Karena proses ini untuk mengubah semua huruf dalam teks yang huruf besar menjadi huruf kecil. Misalnya Kata “Ternate”, “TerNate” diubah menjadi “ternate”. Berikut adalah kode programnya bisa dilihat pada gambar 4.1.



Gambar 4.1 Kode Program *Case Folding*

Fungsi ‘*preprocess_sentence’* menerima sebuah kalimat sebagai *argument* ‘*sent’,* kemudian menggunakan metode ‘*.lower()’* untuk mengubah semua huruf dalam kalimat tersebut menjadi huruf kecil. Hasilnya, yaitu kalimat yang telah diubah ke huruf kecil, dikembalikan fungsi. Berikut tabel 4.2 contoh data sebelum *case folding*.

Tabel 4.2 Contoh Data Sebelum *Case Folding*

Bahasa Indonesia	Bahasa Ternate
Fandi adalah admiral dewan kerajaan Ternate	jou Fandi ge kapita lau bobato karajaan Tarnate
baginda Sultan Ternate	joou kolano Tarnate
di Bacan banyak dukuh	toma Bacan dofu duku
jalan Trans Halmahera panjang sekali	ngako toma Trans Halmahera igila foloi
Indonesia membeli tank di Amerika	Indonesia ifodi oto waja toma Amerika

Tabel 4.2 diatas merupakan tabel contoh data sebelum melewati proses *case folding*, dapat dilihat bahwa data tersebut masih bercampur dengan huruf besar dan kecil. Setelah melakukan *case folding* maka datanya akan menjadi huruf kecil semua. Untuk lebih jelasnya bisa dilihat pada tabel 4.3.

Tabel 4.3 Contoh Data Setelah *Case Folding*

Bahasa Indonesia	Bahasa Ternate
fandi adalah admiral dewan kerajaan ternate	jou fandi ge kapita lau bobato karajaan tarnate
baginda sultan ternate	joou kolano tarnate
di bacan banyak dukuh	toma bacan dofu duku
jalan trans halmahera panjang sekali	ngako toma trans halmahera igila foloi
indonesia membeli tank di amerika	indonesia ifodi oto waja toma amerika

Tabel 4.3 diatas merupakan tabel data yang telah melewati proses *case folding*, dan dapat dilihat pada data diatas sudah tidak bercampur dengan huruf besar lagi, huruf-huruf pada data diatas telah diubah menjadi huruf kecil semua.

4.2.2. *Special tokens*

Pada tahap ini kita akan mengimplementasikan *preprocessing* data *special tokens* 'BOS' dan 'EOS', dan *special tokens* ini hanya ditambahkan kedalam Bahasa Ternate saja, dan tidak ditambahkan ke Bahasa Indonesia. *Token* 'BOS' merupakan singkatan dari *Beginning of Sentence* yang ditambahkan di awal setiap kalimat dalam data Bahasa ternate untuk menandai permulaan kalimat. Sebaliknya *token* 'EOS' yang berarti *End of Sentence,* ditempatkan diakhir setiap kalimat untuk menandai akhir kalimat. Penggunaan kedua *special token* ini memungkinkan model untuk lebih efisien dalam mempelajari dan menghasilkan kalimat yang struktural dan sesuai, dengan memahami di mana kalimat dimulai dan berakhir. Untuk lebih jelasnya berikut adalah gambar 4.2 kode program untuk menambah *special tokens.*

```
indo_sents.append(indo_sent)
ternate_sents_in.append("BOS " + ternate_sent)
ternate_sents_out.append(ternate_sent + " EOS")
```

Gambar 4.2 Kode Program *Special Tokens*

Gambar 4.2 diatas merupakan kode untuk menambahkan *special tokens* pada dataset, dan fungsi dari '*ternate_sents_in.append("BOS " + ternate_sent)*' Pada baris ini, untuk setiap kalimat dalam Bahasa Ternate yang telah diproses untuk dijadikan huruf kecil, kode akan menambahkan token "BOS" di awal kalimat tersebut. Sedangkan fungsi dari *'ternate_sents_out.append(ternate_sent + " EOS")'* Pada baris ini, kode menambahkan token "EOS" pada akhir setiap kalimat Bahasa Ternate. Dengan demikian, model dapat mengenali kapan seharusnya menghentikan proses generasi kata dalam tugas-tugas yang memerlukan *output* teks, seperti penerjemahan atau penyusunan narasi. Berikut adalah contoh sebelum dan sesudah dari proses *special tokens*. Dapat dilihat tabel 4.4 sebelum dan sesudah *special tokens* BOS.

Tabel 4.4 Sebelum dan Sesudah *Special Tokens* BOS

Sebelum *Special Tokens* BOS	Sesudah *Special Tokens* BOS
baba, aba	'BOS baba, aba'
taong ratu moi	'BOS taong ratu moi'
taong ratu romdidi	'BOS taong ratu romdidi'
aku kekal ge toma akhirat	'BOS aku kekal ge toma akhirat'
una u master ua guru na parentah	'BOS una u master ua guru na parentah'

Tabel 4.4 diatas merupakan tabel contoh data sebelum dan sesudah melewati proses *special tokens* 'BOS'. Dan berikut adalah tabel 4.5 data sebelum dan sesudah melewati *special tokens* 'EOS'.

Tabel 4.5 Sebelum dan Sesudah *Special Tokens* EOS

Sebelum *Special Tokens E*OS	Sesudah *Special Tokens* EOS
taong ratu moi	'taong ratu moi EOS'
taong ratu romdidi	'taong ratu romdidi EOS'
aku kekal ge toma akhirat	'aku kekal ge toma akhirat EOS'

una u master ua guru na parentah	'una u master ua guru na parentah EOS'

Dengan menanamkan 'BOS' dan 'EOS' dalam data latih, model dapat lebih efektif dalam mengatur aliran informasi dan struktur kalimat selama proses penerjemahan. Implementasi ini memperbaiki kualitas *output* yang dihasilkan

4.2.3. Tokenisasi dan *Padding*

Pada tahap ini kita akan melakukan proses persiapan data untuk model *Neural Machine Translation*. Berikut adalah penjelasan setiap langkahnya:

Berikut adalah gambar 4.3 membaca data .

```
NUM_SENT_PAIRS = 1000
FILENAME = 'indo_ternate.csv'
sents_en, sents_fr_in, sents_fr_out = read_data(FILENAME, NUM_SENT_PAIRS)
```

Gambar 4.3 membaca data

Kode diatas berfungsi untuk Membaca 1000 pasang kalimat dari file CSV. *'indo_ternate.csv'*, di mana '*sents_en'* adalah kalimat-kalimat dalam bahasa Indonesia, '*sents_fr_in'* adalah kalimat-kalimat Bahasa Ternate sebelum pemrosesan, dan '*sents_fr_out'* adalah kalimat-kalimat Bahasa Ternate setelah pemrosesan.

Berikut adalah gambar 4.4 tokenisasi dan *padding* untuk Bahasa Indonesia

```
tokenizer_en = tf.keras.preprocessing.text.Tokenizer(filters="", lower=False)
tokenizer_en.fit_on_texts(sents_en)
data_en = tokenizer_en.texts_to_sequences(sents_en)
data_en = tf.keras.preprocessing.sequence.pad_sequences(data_en, padding="post")
```

Gambar 4.4 Tokenisasi dan *Padding* Untuk Bahasa Indonesia

Kode di atas berfungsi untuk tokenisasi dan *padding* urutan token untuk kalimat-kalimat dalam bahasa indonesia. Pertama, *tokenizer* '*tokenizer_en'* dibuat menggunakan *'tf.keras.preprocessing.text.Tokenizer'* untuk mengonversi kalimat-kalimat teks menjadi urutan token. Kemudian, kalimat-kalimat tersebut di-tokenisasi menjadi urutan token dengan

'*tokenizer_en. texts to sequences(sents_en)*'. Selanjutnya, dilakukan *padding* urutan token menggunakan '*tf. keras. preprocessing. sequence. pad_sequences (data_en, padding="post")',* yang memastikan bahwa semua urutan token memiliki panjang yang sama dengan *padding* di bagian belakang. Hasilnya adalah '*data_en'*, yang berisi urutan token kalimat-kalimat dalam bahasa indonesia yang siap untuk digunakan dalam pembangunan model NMT.

Berikut adalah gambar 4.5 tokenisasi dan *padding* untuk Bahasa Ternate.

```
tokenizer_fr = tf.keras.preprocessing.text.Tokenizer(filters="", lower=False)
tokenizer_fr.fit_on_texts(sents_fr_in)
tokenizer_fr.fit_on_texts(sents_fr_out)
data_fr_in = tokenizer_fr.texts_to_sequences(sents_fr_in)
data_fr_in = tf.keras.preprocessing.sequence.pad_sequences(data_fr_in, padding="post")
data_fr_out = tokenizer_fr.texts_to_sequences(sents_fr_out)
data_fr_out = tf.keras.preprocessing.sequence.pad_sequences(data_fr_out, padding="post")
```

Gambar 4.5 Tokenisasi dan *Padding* Untuk Bahasa Ternate

Kode diatas memiliki fungsi yang sama dengan kode sebelumnya yang ada pada gambar 4.4, hanya saja pada kode diatas berfungsi untuk bahasa Ternate.

Berikut adalah gambar 4.6 menghitung ukuran kosakata.

```
vocab_size_en = len(tokenizer_en.word_index)
vocab_size_fr = len(tokenizer_fr.word_index)
word2idx_en = tokenizer_en.word_index
idx2word_en = {v: k for k, v in word2idx_en.items()}
word2idx_fr = tokenizer_fr.word_index
idx2word_fr = {v: k for k, v in word2idx_fr.items()}
print("vocab size (id): {:d}, vocab size (tte): {:d}".format(vocab_size_en, vocab_size_fr))
maxlen_en = data_en.shape[1]
maxlen_fr = data_fr_out.shape[1]
print("seq len (id): {:d}, (tte): {:d}".format(maxlen_en, maxlen_fr))
```

Gambar 4.6 Menghitung Ukuran Kosakata

Fungsi dari kode diatas adalah untuk menghitung ukuran kosakata (jumlah kata unik) untuk setiap bahasa dan menyimpan kamus yang memetakan kata ke indeks dan sebaliknya. Juga menghitung panjang maksimum urutan token untuk setiap bahasa. Kemudian, dilakukan pencetakan jumlah kata unik (*vocab size*) untuk Bahasa Indonesia dan Bahasa Ternate, serta panjang maksimum urutan token (*seq len*) untuk Bahasa Indonesia

dan Bahasa Ternate, yang akan digunakan dalam konfigurasi model NMT.

4.3. Latih Model

Tahap ini model akan dilatih sebelum menyimpan modelnya agar bisa di *load* kembali tanpa perlu di latih ulang, berikut adalah langkah-langkah untuk melatih model. Berikut adalah gambar 4.7 pembagian *dataset* dan arsitektur *encoder* NMT.

```
batch_size = 64
dataset = tf.data.Dataset.from_tensor_slices((data_en, data_fr_in, data_fr_out))
dataset = dataset.shuffle(10000)
test_size = NUM_SENT_PAIRS // 4
test_dataset = dataset.take(test_size).batch(batch_size, drop_remainder=True)
train_dataset = dataset.skip(test_size).batch(batch_size, drop_remainder=True)

class Encoder(tf.keras.Model):
    def __init__(self, vocab_size, embedding_dim, num_timesteps, encoder_dim, **kwargs):
        super(Encoder, self).__init__(**kwargs)
        self.encoder_dim = encoder_dim
        self.embedding = tf.keras.layers.Embedding(vocab_size, embedding_dim)
        self.rnn = tf.keras.layers.GRU(encoder_dim, return_sequences=True, return_state=True)

    def call(self, x, state):
        x = self.embedding(x)
        x, state = self.rnn(x, initial_state=state)
        return x, state

    def init_state(self, batch_size):
        return tf.zeros((batch_size, self.encoder_dim))
```

Gambar 4.7 Pembagian *Dataset* dan *Arsitektur Encoder* NMT

Kode di atas merupakan implementasi dari kelas *Encoder* dalam *TensorFlow* untuk model *Encoder* dalam arsitektur *Neural Machine Translation* (NMT). Kelas *Encoder* ini bertanggung jawab untuk menerjemahkan urutan token *input* menjadi representasi yang lebih abstrak yang disebut dengan "*context*". Pada konstruktor (__init__), kelas ini menerima beberapa parameter, seperti '*vocab_size*' (ukuran kosakata), '*embedding_dim*' (dimensi vektor embedding), '*num_timesteps*' (panjang maksimum urutan token), dan '*encoder_dim*' (dimensi *encoder*). Dalam metode *call*, *input* pertama kali diubah menjadi *vektor embedding* menggunakan *layer embedding*. Selanjutnya, *vektor embedding* tersebut dilewatkan ke lapisan GRU (*Gated Recurrent Unit*), yang menghasilkan *output* urutan serta *state* terakhir

dari *encoder*. Metode '*init_state'* digunakan untuk menginisialisasi *state* awal *encoder*. Kemudian, *dataset* dibagi menjadi *dataset* pelatihan dan *dataset* pengujian menggunakan '*tf.data. Dataset'* dan dipisahkan menggunakan metode *skip* dan *take*.

Berikut adalah gambar 4.8 implementasi *bahdanau attention layer*.

```
class BahdanauAttention(tf.keras.layers.Layer):
    def __init__(self, num_units, phase='training'):
        super(BahdanauAttention, self).__init__()
        self.W1 = tf.keras.layers.Dense(num_units)
        self.W2 = tf.keras.layers.Dense(num_units)
        self.V = tf.keras.layers.Dense(1)

    def call(self, d_state, e_states):
        d_state_extend = tf.expand_dims(d_state, axis=1)
        attention_score = self.V(tf.keras.activations.tanh(self.W1(e_states) + self.W2(d_state_extend)))
        attention_weight = tf.nn.softmax(attention_score, axis=1)
        context = tf.reduce_sum(tf.linalg.matmul(tf.transpose(attention_weight, perm=[0, 2, 1]), e_states), axis=1)
        return context, attention_weight
```

Gambar 4.8 Implementasi *Bahdanau Attention Layer*

Kelas '*BahdanauAttention'* yang didefinisikan di atas merupakan lapisan *attention* dengan metode '*Bahdanau'*, salah satu jenis *attention* yang umum digunakan dalam model NMT (*Neural Machine Translation*). Dalam konstruktor *'(__init__)'*, lapisan ini menginisialisasi tiga lapisan *Dense*: W1, W2, dan V, yang digunakan untuk menghitung skor *attention*. Pada metode *call*, lapisan ini menerima dua *input*: '*d_state'* (state dari decoder pada langkah waktu tertentu) dan '*e_states'* (*state-state* dari *encoder* pada semua langkah waktu). Pertama, *state* dari *decoder* diperluas dimensinya untuk disesuaikan dengan *state-state* dari *encoder*. Kemudian, skor *attention* dihitung menggunakan lapisan *Dense* dan aktivasi *tanh*. Skor *attention* kemudian dinormalisasi menggunakan *softmax* untuk mendapatkan bobot *attention*. Akhirnya, *weighted contex* dihitung sebagai hasil perkalian tensor antara bobot *attention* dan *state-state* dari *encoder*, lalu dijumlahkan. Metode ini mengembalikan konteks dan bobot *attention* sebagai *output*. Lapisan ini menghasilkan konteks berdasarkan *state* dari *decoder* dan *state-state* dari *encoder*.

Berikut adalah gambar 4.9 arsitektur model *decoder* NMT.

```
class Decoder(tf.keras.Model):
    def __init__(self, vocab_size, embedding_dim, num_timesteps, decoder_dim, attention_type='Bahdanau'):
        super(Decoder, self).__init__()
        self.decoder_dim = decoder_dim
        self.attention = BahdanauAttention(decoder_dim)
        self.embedding = tf.keras.layers.Embedding(vocab_size, embedding_dim)
        self.rnn = tf.keras.layers.GRU(decoder_dim, return_sequences=False, return_state=True)
        self.Wc = tf.keras.layers.Dense(decoder_dim, activation='tanh')
        self.Ws = tf.keras.layers.Dense(vocab_size)

    def call(self, x, state, encoder_out):
        x = self.embedding(x)
        h, state = self.rnn(x, state)
        context, attention_weight = self.attention(h, encoder_out)
        h = tf.concat([h, context], axis=1)
        h = self.Wc(h)
        logits = self.Ws(h)
        return logits, state, attention_weight

embedding_dim = 256
encoder_dim, decoder_dim = 100, 100
encoder = Encoder(vocab_size_en+1, embedding_dim, maxlen_en, encoder_dim)
decoder = Decoder(vocab_size_fr+1, embedding_dim, maxlen_fr, decoder_dim, 'Bahdanau')
```

Gambar 4.9 Arsitektur Model *Decoder* NMT

Kode di atas mendefinisikan kelas *Decoder* untuk model NMT menggunakan *TensorFlow*. Kelas ini memiliki beberapa lapisan, termasuk lapisan *attention*, *embedding*, GRU, serta lapisan-lapisan *Dense* untuk menghasilkan *output* terjemahan. Fungsi *call* untuk melakukan proses *decoding* dengan memasukkan *input*, *state*, dan *output* dari *encoder*.

Berikut adalah gambar 4.10 menghitung nilai *loss*.

```
def loss_fn(y_true, y_pred):
    scce = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True)
    mask = tf.math.logical_not(tf.math.equal(y_true, 0))
    mask = tf.cast(mask, dtype=tf.int64)
    loss = scce(y_true, y_pred, sample_weight=mask)
    return loss

optimizer = tf.keras.optimizers.Adam()

@tf.function
def train_step(encoder_in, decoder_in, decoder_out, encoder_state):
    with tf.GradientTape() as tape:
        encoder_out, encoder_state = encoder(encoder_in, encoder_state)
        decoder_state = encoder_state
        loss = 0

        for t in range(decoder_out.shape[1]):
            decoder_in_t = decoder_in[:, t]
            decoder_in_t = tf.reshape(decoder_in_t, [-1, 1])
            decoder_pred_t, decoder_state, _ = decoder(decoder_in_t, decoder_state, encoder_out)
            loss += loss_fn(decoder_out[:, t], decoder_pred_t)
        variables = encoder.trainable_variables + decoder.trainable_variables
        gradients = tape.gradient(loss, variables)
        optimizer.apply_gradients(zip(gradients, variables))
    return loss / decoder_out.shape[1]
```

Gambar 4.10 Proses Pelatihan

Kode tersebut merupakan bagian dari proses pelatihan (*training*) model, fungsi '*loss_fn'* digunakan untuk menghitung nilai kerugian (*loss*) antara prediksi dan label sebenarnya, dengan menerapkan fungsi '*SparseCategoricalCrossentropy'* yang memperhitungkan nilai-nilai yang tidak *valid* dengan menggunakan maska. Selanjutnya, *optimizer* Adam digunakan untuk mengoptimalkan bobot model berdasarkan *loss* yang dihitung. Fungsi '*train_step*' menentukan satu langkah pelatihan pada setiap iterasi, dimana model menerima satu *batch* data pelatihan dan menghitung *loss*. Dengan menggunakan '*tf.GradientTape'*, *gradien loss* terhadap bobot model direkam dan selanjutnya digunakan untuk memperbarui bobot model melalui *optimizer*. Kemudian, fungsi mengembalikan rata-rata *loss* per token untuk *batch* tersebut.

Berikut adalah gambar 4.11 prediksi model NMT.

```
def predict(encoder, decoder, batch_size, sents_en, data_en, sents_fr_out, word2idx_fr, idx2word_fr):
    random_id = np.random.choice(len(sents_en)) #=
    print("input :", "".join(sents_en[random_id]))
    print("label :", "".join(sents_fr_out[random_id]))
    encoder_in = tf.expand_dims(data_en[random_id], axis=0)
    decoder_out = tf.expand_dims(sents_fr_out[random_id], axis=0)
    encoder_state = encoder.init_state(1)
    encoder_out, encoder_state = encoder(encoder_in, encoder_state)
    decoder_state = encoder_state
    decoder_in = tf.expand_dims([word2idx_fr["BOS"]], axis=0)
    pred_sent_fr = []
    decoding_step = 0

    while decoding_step < maxlen_fr:
        decoder_pred, decoder_state, _ = decoder(decoder_in, decoder_state, encoder_out)
        decoder_pred = tf.argmax(decoder_pred, axis=-1)
        pred_word = idx2word_fr[decoder_pred.numpy()[0]]
        pred_sent_fr.append(pred_word)
        if pred_word == "EOS":
            break
        decoder_in = tf.reshape(decoder_pred, [1, -1])
        decoding_step += 1

    print("predicted :", " ".join(pred_sent_fr))
```

Gambar 4.11 Demonstrasi Fungsi Prediksi NMT

Fungsi '*predict'* dalam kode diatas bertujuan untuk mengevaluasi dan memprediksi hasil terjemahan dari Bahasa Indonesia ke Bahasa Ternate menggunakan model *neural machine translation* (NMT) yang telah dilatih sebelumnya, fungsi ini menerima beberapa

parameter:

1. '*Encoder',* model *encoder* yang mengubah kalimat Bahasa Indonesia menjadi representasi kontekstual.
2. '*decoder',* model ini menggunakan *output* dari *encoder* untuk menghasilkan terjemahan dalam Bahasa Ternate.
3. *'batch_size',* ukuran *batch* yang digunakan selama prediksi, meskipun pada fungsi ini, '*batch_size'* tidak secara eksplisit digunakan karena prediksi dilakukan pada satu contoh pada satu waktu.
4. *'sents_en'* yaitu *array* dari kalimat-kalimat dalam Bahasa Indonesia.
5. *'data_en'* yaitu data tokenisasi dari kalimat-kalimat Bahasa Indonesia.
6. '*sents_fr_out'* berisi *array* dari kalimat-kalimat target dalam Bahasa Ternate yang digunakan untuk evaluasi.
7. '*word2idx_fr'* kamus yang memetakan kata-kata Bahasa Ternate ke indeksnya.
8. '*idx2word_fr'* kamus yang memetakan indeks kembali ke kata-kata Bahasa ternate.

Fungsi ini pertama-tama memilih secara acak sebuah kalimat dari '*sents_en'* dan menampilkannya bersama dengan terjemahan targetnya dari '*sents_fr_out'* sebagai "*input*" dan "*label*". Kemudian, kalimat Bahasa Indonesia tersebut diubah menjadi *tensor* dan diberikan ke *encoder* untuk mendapatkan representasi kontekstualnya. *State* awal untuk *decoder* diinisialisasi menggunakan *state* terakhir dari *encoder*, dan decoder mulai menghasilkan terjemahan kata per kata dalam Bahasa Ternate.

Prediksi dilakukan dalam sebuah *loop*, di mana setiap kata dalam Bahasa Ternate diprediksi berdasarkan konteks yang diberikan oleh *encoder* dan kata-kata sebelumnya yang telah dihasilkan oleh *decoder*. Proses ini berlanjut sampai token "*EOS*" (*end of*

sentence) muncul atau mencapai panjang maksimum kalimat target (*maxlen_fr*).

Hasil prediksi kemudian diubah menjadi *string* kata-kata dan dicetak. Hal ini memungkinkan kita untuk membandingkan kualitas dan akurasi terjemahan yang dihasilkan oleh model dengan terjemahan yang sebenarnya.

Berikut adalah gambar 4.12 pelatihan model NMT.

```
def train_all():
    num_epochs = 200
    for e in range(num_epochs):
        encoder_state = encoder.init_state(batch_size)
        for batch, data in enumerate(train_dataset):
            encoder_in, decoder_in, decoder_out = data
            loss = train_step(encoder_in, decoder_in, decoder_out, encoder_state)
        print("Epoch {:d}, Loss: {:.4f}".format(e + 1, loss.numpy()))
        predict(encoder, decoder, batch_size, sents_en, data_en, sents_fr_out, word2idx_fr, idx2word_fr)

    # Calculate BLEU score after training
    references = []
    hypotheses = []
    for batch, (encoder_in, decoder_in, decoder_out) in enumerate(test_dataset):
        for i in range(encoder_in.shape[0]):
            sentence = " ".join([idx2word_en.get(token, "") for token in encoder_in[i].numpy() if token != 0])
            reference = " ".join([idx2word_fr.get(token, "") for token in decoder_out[i].numpy() if token not in [0, word2idx_fr['EOS']]])
            hypothesis = translate_sentence(sentence, encoder, decoder, tokenizer_en, tokenizer_fr, maxlen_en, maxlen_fr, idx2word_fr)
            references.append(reference)
            hypotheses.append(hypothesis)

    bleu_score = compute_bleu_score(references, hypotheses)
    print(f"Final BLEU score: {bleu_score}")

train_all()
```

Gambar 4.12 Proses Pelatihan Model NMT

Kode diatas merupakan fungsi '*train_all()*' yang bertanggung jawab untuk melatih model *neural machine translation* (NMT) menggunakan data latih. Berikut adalah fungsi kode diatas:

1. *Loop Epoch*, Fungsi ini mengiterasi sebanyak '*num_epochs*', yang mewakili jumlah iterasi pelatihan yang diinginkan.
2. Inisialisasi *State Encoder*, Pada awal setiap *epoch*, *state* awal *encoder* diinisialisasi menggunakan fungsi '*init_state*' dari *encoder*. Ini adalah langkah yang diperlukan sebelum setiap iterasi pelatihan untuk memastikan *state encoder* diatur ulang.
3. Iterasi *Batches*: Selama setiap iterasi pelatihan, fungsi ini mengiterasi melalui setiap *batch* dalam *dataset* latih ('*train_dataset*'). Setiap *batch* berisi data masukan untuk

encoder (‘*encoder_in’*), data masukan untuk *decoder* (‘*decoder_in’*), dan data keluaran yang benar (‘*decoder_out’*). Selama iterasi ini, fungsi memanggil ‘*train_step()’* untuk melakukan satu langkah pelatihan pada *batch* saat ini.

4. Pelatihan dan Perhitungan *Loss*, Di dalam setiap iterasi *batch*, fungsi ‘*train_step()’* dipanggil untuk melakukan satu langkah pelatihan.
5. *Monitoring* Pelatihan, Setelah satu *epoch* selesai, *loss* terakhir dari iterasi *batch* terakhir dicetak ke konsol untuk memantau kemajuan pelatihan model.
6. Evaluasi Prediksi, setelah setiap *epoch*, fungsi ‘*predict()*’ dipanggil untuk mengevaluasi kualitas terjemahan yang dihasilkan oleh model saat ini. Evaluasi ini melihat kualitas terjemahan yang dihasilkan oleh model sepanjang pelatihan.

Berikut adalah hasil terjemahan saat *running*, dapat dilihat pada gambar 4.13.

```
Epoch 1, Loss: 1.6705
input : nasi amis
label : bira mahami EOS
predicted : EOS
Epoch 2, Loss: 1.3245
input : dawa
label : kalaki EOS
predicted : EOS
Epoch 3, Loss: 1.4839
input : curam
label : oko oko EOS
predicted : EOS
Epoch 4, Loss: 1.5438
input : arus menghanyutkan perahu
label : bao ge siruru oti EOS
predicted : EOS
Epoch 5, Loss: 1.4546
input : cuaca
label : wer EOS
predicted : EOS
Epoch 6, Loss: 1.3682
input : diploma
label : surat tamat EOS
predicted : EOS
Epoch 7, Loss: 1.4858
...
Epoch 200, Loss: 0.0493
input : pedagang
label : dibo dibo EOS
predicted : dibo dibo EOS
```

Gambar 4.13 Hasil *Running*

Gambar diatas merupakan hasil *running* setelah di ‘*train_all’* yang diukur selama 200

epoch. Dari data tersebut kita dapat melihat evolusi *loss* dari awal pelatihan dimulai dari 1.6705 turun menjadi 0.0493 di *epoch* ke 200, yang menunjukkan peningkatan kemampuan model untuk menghasilkan terjemahan yang akurat.

Berikut adalah gambar 4.14 fungsi terjemahan NMT.

```
def translate_sentence(sentence, encoder, decoder, tokenizer_en, tokenizer_fr, maxlen_en, maxlen_fr, idx2word_fr):
    sentence = preprocess_sentence(sentence)
    tokens = tokenizer_en.texts_to_sequences([sentence])
    encoder_input = tf.keras.preprocessing.sequence.pad_sequences(tokens, maxlen=maxlen_en, padding='post')
    encoder_state = encoder.init_state(1)
    encoder_out, encoder_state = encoder(encoder_input, encoder_state)

    decoder_state = encoder_state
    decoder_input = tf.expand_dims([tokenizer_fr.word_index['BOS']], axis=0)
    translated_sentence = []

    for i in range(maxlen_fr):
        decoder_output, decoder_state, _ = decoder(decoder_input, decoder_state, encoder_out)
        predicted_id = tf.argmax(decoder_output, axis=-1).numpy()[0]
        translated_sentence.append(idx2word_fr.get(predicted_id, '<UNK>'))
        if idx2word_fr.get(predicted_id) == 'EOS':
            break
        decoder_input = tf.expand_dims([predicted_id], axis=0)

    return ' '.join(translated_sentence)

# Contoh penggunaan:
sentence = "gadis itu cantik sekali"
translated_sentence = translate_sentence(sentence, encoder, decoder, tokenizer_en, tokenizer_fr, maxlen_en, maxlen_fr, idx2word_fr)
print("Terjemahan:", translated_sentence)
```

Gambar 4.14 Fungsi Terjemahan NMT

Kode di atas mendefinisikan fungsi '*translate_sentence*' yang bertugas menerjemahkan satu kalimat dari bahasa sumber (dalam contoh ini, bahasa Indonesia) ke bahasa target (dalam contoh ini, bahasa Ternate). Fungsi ini menerima sebagai masukan kalimat yang akan diterjemahkan, model *encoder-decoder* yang telah dilatih (*encoder* dan *decoder*), tokenisasi untuk bahasa sumber dan bahasa target ('*tokenizer_en*' dan '*tokenizer_fr*'), panjang maksimum dari kalimat dalam bahasa sumber dan bahasa target ('*maxlen_en*' dan *'maxlen_fr*'), serta kamus yang memetakan indeks ke kata untuk bahasa target ('*idx2word_fr*'). Dan pada kode diatas saya berikan contoh kalimat "gadis itu cantik sekali", dan model berhasil menerjemahkan ke dalam Bahasa Ternate, untuk hasil *running* dapat dilihat pada gambar 4.15.

```
Terjemahan: jujaru mina ge mu jang foloi EOS
```

Gambar 4.15 Hasil Terjemahan

4.4. *Deploy* Model

Pada tahap ini akan dibuat implementasi API menggunakan *Flask*, dan pada kodenya menggunakan kode yang sama seperti sebelumnya hanya saja ada tambahan kode untuk *flask*. Untuk lebih jelasnya dapat dilihat pada gambar 4.16.

```python
@app.route('/translate', methods=['POST'])
def translate():
    # Ambil teks input dari payload JSON
    input_json = request.get_json()
    input_text = input_json.get('text', '')

    if not input_text:
        return jsonify({'error': 'No input text provided'}), 400

    translated_sentence = translate_sentence(input_text, encoder, decoder,
tokenizer_en, tokenizer_fr, maxlen_en, maxlen_fr, idx2word_fr)

    kata_yang_dihapus = "EOS"

    # Menghapus kata dari kalimat
    translate = translated_sentence.replace(kata_yang_dihapus, "")

    return jsonify({'terjemahan': translate})

if __name__ == '__main__':
    app.run(port=4000)
```

Gambar 4.16 API *Flask*

Kode di atas adalah implementasi API terjemahan menggunakan *Flask*, di mana *endpoint* /*translate* menerima teks melalui metode *POST*, mengekstrak teks tersebut dari *payload* JSON, dan memvalidasi keberadaannya. Jika teks disediakan, fungsi '*translate_sentence'* dipanggil untuk menerjemahkan teks dari bahasa sumber ke bahasa target, menghapus *token* 'EOS' yang menandakan akhir kalimat dalam *output* terjemahan, dan kemudian mengembalikan hasil terjemahan melalui JSON *response*. Aplikasi ini dijalankan pada *port* 4000, dan hanya akan aktif jika skrip dijalankan secara langsung sebagai program utama, memastikan bahwa API hanya tersedia melalui *server Flask* saat dijalankan secara eksplisit.

4.5. Implementasi Sistem

Implementasi sistem merupakan tahap penerapan sistem yang telah didesain atau dirancang, sehingga sistem yang telah dibuat dapat dioperasikan. Sistem ini dibuat dengan menggunakan *android studio* sebagai dasar pembuatan aplikasi *android* menggunakan Bahasa Pemrograman *Kotlin.* Berikut ini hasil implementasi perangkat lunak:

4.5.1. Tampilan Beranda

Tampilan beranda merupakan halaman pertama yang dimana pada tampilan ini ada kalimat pengucap selamat datang Terjemahan Bahasa Indonesia – Bahasa Ternate. Tampilan beranda dapat dilihat pada gambar 4.17.

Gambar 4.17 Tampilan beranda

4.5.2. Tampilan Utama

Pada tampilan utama ini, terdapat kolom terjemahan bahasa Indonesia untuk meng-

input kalimat dari bahasa Indonesia, tombol terjemahkan untuk menerjemahkan dari bahasa Indonesia ke bahasa Ternate, dan kolom bahasa ternate untuk hasil terjemahan dari bahasa Indonesia. Tampilan utama dapat dilihat pada gambar 4.18.

Gambar 4.18 Tampilan utama terjemahan Bahasa Indonesia – Ternate

4.5.3. Tampilan Hasil Terjemahan Bahasa Indonesia – Bahasa Ternate

Selanjutnya tampilan hasil terjemahan bahasa Indonesia – bahasa Ternate, yang dimana pada tampilan ini di kolom terjemahan bahasa Indonesia sudah terdapat kalimat yang akan di terjemahkan ke bahasa target (Bahasa Ternate), dan saat menekan tombol "terjemahkan" maka kalimat pada bahasa Indonesia akan di *translate* ke bahasa Ternate. Tampilan hasil terjemahan Bahasa Indonesia – Bahasa Ternate dapat dilihat pada gambar 4.19.

Gambar 4.19 Tampilan hasil terjemahan Bahasa Indonesia – Bahasa Ternate

4.6. Hasil Analisis

Dalam penelitian ini, evaluasi hasil terjemahan dilakukan menggunakan BLEU (*Bilingual Evaluation Understudy*) untuk mengukur akurasi terjemahan yang dihasilkan oleh metode LSTM. BLEU *score* digunakan sebagai indikator untuk mengukur kesesuaian antara teks terjemahan yang dihasilkan oleh model dengan teks referensi yang benar. Skor BLEU berkisar antara 0 hingga 100, di mana skor yang lebih tinggi menunjukkan tingkat kesesuaian yang lebih baik. Dalam penelitian ini, hasil pengujian menunjukkan bahwa penerapan metode LSTM mencapai skor BLEU sebesar 64.92%. Meskipun skor ini menunjukkan bahwa model mampu menghasilkan terjemahan yang cukup baik, masih terdapat beberapa kekurangan yang perlu diperhatikan. Salah satu kendala utama yang mempengaruhi akurasi hasil terjemahan adalah keterbatasan data paralel bahasa Indonesia - bahasa Ternate. Jumlah data yang sedikit menghambat kemampuan model untuk belajar

pola dan struktur bahasa secara efektif. Keterbatasan data ini berdampak pada kualitas model karena model tidak memiliki cukup data untuk mempelajari berbagai variasi dan konteks kalimat dalam Bahasa Ternate, sehingga hasil terjemahan tidak akurat dan jauh dari harapan.

4.7. Pembahasan

Dalam penelitian ini menggunakan LSTM pada arsitektur *encoder-decoder* dengan BLEU *Score* sebesar 64,92%. Sistem NMT hanya mampu menerjemahkan kalimat yang sama dengan yang terdapat di data latih atau kalimat yang sudah tersimpan di model, hal ini dikarenakan data yang digunakan kurang sehingga model sulit untuk mempelajari kosakata serta struktur kalimat. Pada penelitian sebelumnya yang dilakukan oleh (Ramadhan, 2022) dengan judul Implementasi *Neural Machine Translation* Bahasa Inggris – Bahasa Sunda menggunakan *Long Short-Term Memory* dengan total data 20.000 kalimat *parallel*, Penelitian mencapai akurasi 0,99 pada pelatihan dan pengujian serta nilai kerugian kurang dari 0,1 pada data pengujian dan pelatihan. Model ini mencapai skor BLEU rata-rata lebih dari 0,8 untuk data pelatihan dan pengujian. Penelitian ini mencapai hasil yang baik dalam tugas terjemahan mesin dikarenakan data yang digunakan cukup dalam membantu model mempelajari pola serta struktur kalimat.

BAB V

PENUTUP

5.1. Kesimpulan

Penelitian ini menggunakan arsitektur *Long Short-Term Memory* (LSTM) dalam *Neural Machine Translation* (NMT) untuk penerjemahan bahasa Indonesia ke bahasa Ternate, dengan data yang bersumber dari buku "Kamus Praktis Indonesia - Ternate" yang terdiri dari 6,925 data. Proses penelitian mencakup pengumpulan *dataset*, *preprocessing* data, pemodelan dan pelatihan LSTM, evaluasi sistem menggunakan skor BLEU (*Bilingual Evaluation Understudy*), dan implementasi dalam bentuk aplikasi *Android*. Evaluasi menunjukkan bahwa model NMT dengan LSTM mencapai skor BLEU sebesar 64.92%, yang menunjukkan kemampuan model dalam menghasilkan terjemahan yang cukup baik meskipun dengan beberapa kekurangan. Salah satu kendala utama adalah keterbatasan data paralel bahasa Indonesia - Ternate yang menyebabkan hasil terjemahan kurang natural dan kadang tidak akurat.

5.2. Saran

Adapun saran yang dapat diberikan untuk mengembangkan penelitian ini adalah:

1. Menggunakan data yang lebih banyak dan besar agar bisa mendapatkan hasil yang lebih maksimal dan akurasi yang lebih tinggi
2. Dan untuk penelitian selanjutnya diharapkan menggunakan metode yang berbeda agar dapat mengetahui kualitas dari masing masing metode.

DAFTAR PUSTAKA

Inun, A. A. 2022. Revitalisasi Bahasa Minoritas Di Indonesia. *Etnolingual*, ISSN:2580-0280, Vol. 6 *Issue*.2 Desember, 2022

Neural, H., Studi, N., & Bahasa, K. 2021. Pengembangan Metode *Neural Machine Translation* Berdasarkan *Hyperparameter Neural Network* (Studi Kasus : Bahasa Jerman – Inggris). November, 2021

Gunawan, W., Sujaini, H., & Tursina. 2021. Analisis Perbandingan Nilai Akurasi Mekanisme *Attention Bahdanau* dan *Luong* pada *Neural Machine* Translation Bahasa Indonesia ke Bahasa Melayu Ketapang dengan Arsitektur *Recurrent Neural Network*. JEPIN (Jurnal Edukasi dan Penelitian Informatika), ISSN:2548-9364, Vol. 7 *Issue*. 3 Desember, 2021

Fauziyah, Y., Ilyas, R., Kasyidi, F. 2022. Mesin Penterjemah Bahasa Indonesia-Bahasa Sunda Menggunakan *Recurrent Neural Networks.* Jurnal Teknoinfo, ISSN:1693-0010, Vol. 16 Issue. 2 Juli, 2022

Ramadhan, T. I., Ramadhan, N. G., & Supriatman, A. 2022. *Implementation of Neural Machine Translation for English-Sundanese Language using Long Short Term Memory* (LSTM). *Building of Informatics, Technology and Science* (BITS), Vol. 4 Issue. 3 Desember, 2022

Razsiah, F. (2023). Aplikasi Penerjemah Bahasa Bangka – Indonesia – Inggris Berbasis *Website* Dengan *Neural Machine Translation* (NMT). Januari, 2023

Raup, A., Ridwan, W., Khoeriyah, Y., Yuliati Zaqiah, Q. 2022. *Deep Learning* dan Penerapannya dalam Pembelajaran. JIIP(Jurnal Ilmiah Ilmu Pendidikan), Vol. 5 *Issue.* 9 September, 2022

Krama, J. N., Purwaningsih, L., Pamungkas, E. W. 2021. Perbandingan Kinerja Sistem *Neural Machine Translation Open*NMT Dan THUMT Dalam Eksperimen Menerjemahkan Bahasa Jawa Ngoko-Krama, 1-16

Mailani, O., Nuraeni, I., Syakila, S. A., Lazuardi, J. 2022. Bahasa Sebagai Alat komunikasi Dalam Kehidupan Manusia. Kampret *Journal*, Vol. 1 *Issue*. 2 Januari, 2022

Nasution, A. S., Wani, S. A., Syahputra, E. 2022. Sejarah Perkembangan Bahasa Indonesia. Jurnal Multidisiplin *Dehasen*, ISSN:2828-1799, Vol. 1 *Issue*. 3 Juli, 2022

Azizah, A. R. Penggunaan Bahasa Indonesia Dan Bahasa Gaul Di Kalangan Remaja. Vol. 5 *Issue*. 2 September, 2019

Duwila, E. 2021. Bahasa Ternate Dan Bahasa Tidore (Kajian *Linguistik Historis Komparatif* Dan *Dialektologi*). ISSN:1693-1041, Vol. 19, *Issue*. 2

Agusti, E. Perancangan Aplikasi *Invoice* berbasis *Mobile* Studi Kasus UMKM. Jurnal Ilmiah Teknik, Vol. 1 *Issue*. 1 Februari, 2022

Dedes, K., Utama, A. B. P., Wibawa, A. P., Afandi, A. N., Handayani, A. N., & Hernandez, L. *Neural Machine Translation of Spanish-English Food Recipes Using* LSTM. *International Journal On Informatics Visualization Journal*, Juni, 2022

Mishra, B., Agarwal, A., Goel, A., Ansari, A. A., Gaur, P., Singh, D., & Lee, H. N. (2022). *Privacy Protection Framework for Android*. *IEEE Access*, Vol. 10 *Issue*. 2 Januari, 2022

Bin Uzayr, S. (2022). *Kotlin The Ultimate Guide*, 4-18

Muhammad, R., & Yulianto, S. 2022. Penerapan Pemrograman *Python* Dalam Menentukan Waktu *Overhoul* Kondensor Turbin Uap . Jurnal Konversi Energi Dan Manufaktur, ISSN:2339-2029, Vol. 8 *Issue*. 1 Januari, 2023

Romario, M. H., Ihsanto, E., & Kadarina, T. M. 2020. Sistem Hitung dan Klasifikasi Objek dengan Metode *Convolutional Neural Network*. Jurnal Teknologi Elektro, ISSN:2086-9479, Vol. 11 *Issue*. 2 Mei, 2020

Kurniawan, K., Ceasaro, B., & Sucipto. 2024. Perbandingan Fungsi Aktivasi Untuk Meningkatkan Kinerja Model LSTM Dalam Prediksi Ketinggian Air Sungai. JEPIN (Jurnal Edukasi Dan Penelitian Informatika), ISSN:2548-9364, Vol. 10 *Issue*. 1 April, 2024

Pricillia, T., & Zulfachmi. 2021. Perbandingan Metode Pengembangan Perangkat Lunak (*waterfall*, *prototype*, RAD). Jurnal Bangkit Indonesia, ISSN:2337-4055, Vol. 10 *Issue*. 1 Maret, 2021

Kusnadi., 15 November 2022, Kantor Bahasa Malut Petakan 19 Bahasa Daerah, https://infopublik.id/kategori/nusantara/685303/index.html

Budiwiyanto, A, 22 Januari 2022, Kontribusi Kosakata Bahasa Daerah Dalam Bahasa Indonesia, https://badanbahasa.kemdikbud.go.id/artikel-detail/792/kontribusi-kosakata-bahasa-daerah-dalam-bahasa-indonesia.

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET DAN TEKNOLOGI
UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK PROGRAM STUDI INFORMATIKA
Kampus III Universitas Khairun, Kelurahan Jati Kota Ternate Selatan
http://if.unkhair.ac.id, http://unkhair.ac.id Group FB: if.unkhair

KARTU BIMBINGAN HASIL

Nama Mahasiswa : Fiska Nabila F. Risal
NIM : 07352011108
Dosen Pembimbing I : Yasir Muin, S.T ., M. Kom.
Judul : Implementasi *Neural Machine Translation* dengan Menggunakan Metode LSTM Pada Proses Penerjemahan Bahasa Indonesia – Bahasa Ternate

NO	Tanggal	Uraian	Paraf
1.	9/6/24	- perbaiki Bab III sesuai revisi	
2.		- perbaiki Bab III	
3	12/6/24	- Bab 3 diperbaiki - tambahkan Arsitektur LSTM	
		ok siap ujian ACC tgl 13/6/24	

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET DAN TEKNOLOGI
UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK PROGRAM STUDI INFORMATIKA
Kampus III Universitas Khairun, Kelurahan Jati Kota Ternate Selatan
http://if.unkhair.ac.id, http://unkhair.ac.id Group FB: if.unkhair

KARTU BIMBINGAN HASIL

Nama Mahasiswa : Fiska Nabila F. Risal
NIM : 07352011108
Dosen Pembimbing II : Ir. Amal Khairan, S.T., M.Eng.
Judul : Implementasi *Neural Machine Translation* Menggunakan Metode LSTM Pada Proses Penerjemahan Bahasa Indonesia – Bahasa Ternate

NO	Tanggal	Uraian	Paraf
1	3 6 2024	- Sistematika penulisan tunning - P - Bab 4. Blm melakukan evaluasi & model yg dibuat. - Data testing yg digunakan. - Bab 4 dan Bab 5 blm menjawab tujuan yang ada pd rumusan masalah & tujuan penelitian juga blm tercapai. (Bandingkan antara Rumusan masalah/tujuan penelitian dan apa yg sdh ditulis pd bab 4 dan 5) -	

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET DAN TEKNOLOGI
UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK PROGRAM STUDI INFORMATIKA
Kampus III Universitas Khairun, Kelurahan Jati Kota Ternate Selatan
http://if.unkhair.ac.id, http://unkhair.ac.id Group FB: if.unkhair

NO	Tanggal	Uraian	Paraf
2	10/6 2024	- Fakai daftar isi otomatis - Bab 2 perlu diatur ukuran posisi tiap sub bab dan tambah teori Machine Learning	
		- Bab 4 tambah sub bab Analisis u/ memberikan hasil analisa dr kinerja model yg di peroleh (Evaluasi model)	
		- Kelengkapan pelun menjawab pertanyaan pd Rumusan masalah.	
3	14/06 2024	- Tambah abstrak (liat cara pd naskah)	
		- Beberapa format naskah/penggetikan perlu diperbaiki. Cek naskah.	
		- Siapkan power point, hafalkan cerpt penelitian.	
		- Jika sudah siap datanya dty menghadap - Bawa script & power point nya.	
4	26/06 2024	Acc Seminar Hasil	

UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK
PROGRAM STUDI INFORMATIKA

DAFTAR PERBAIKAN SEMINAR HASIL SKRIPSI

Dengan ini dinyatakan bahwa pada

Hari / tanggal	:	RABU, 17 JULI 2024
Pukul	:	07:30 - 09:30
Tempat	:	RUANG SIDANG

telah berlangsung Seminar Hasil Skripsi dengan Peserta:

Nama Mahasiswa	:	FISKA NABILA F. RISAL
NPM	:	07352011108
Judul	:	IMPLEMENTASI NEURAL MACHINE TRANSLATION DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

dinyatakan HARUS menyelesaikan perbaikan, yaitu:

- Revisi sesuai yang diminta oleh penguji

ACC 26/7/24 sup ujian ole

Dosen Pembimbing I,

YASIR MUIN, S.T., M.Kom.
NIDN. 9990582796

UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK
PROGRAM STUDI INFORMATIKA

DAFTAR PERBAIKAN SEMINAR HASIL SKRIPSI

Dengan ini dinyatakan bahwa pada

Hari / tanggal	:	RABU, 17 JULI 2024
Pukul	:	07:30 - 09:30
Tempat	:	RUANG SIDANG

telah berlangsung Seminar Hasil Skripsi dengan Peserta:

Nama Mahasiswa	:	FISKA NABILA F. RISAL
NPM	:	07352011108
Judul	:	IMPLEMENTASI NEURAL MACHINE TRANSLATION DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

dinyatakan HARUS menyelesaikan perbaikan, yaitu:

- Segera lakukan perbaikan sebagaimana yg diminta para penguji.
-

Ternate, 25 Juli 2024

Perbaikan dan masukan saat seminar hasil ujian tutup

Dosen Pembimbing II,

E. AMAL KHAIRAN, S.T., M.Eng., IPM
NIP. 197401112003121003

UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK
PROGRAM STUDI INFORMATIKA

DAFTAR PERBAIKAN SEMINAR HASIL SKRIPSI

Dengan ini dinyatakan bahwa pada

Hari / tanggal	:	RABU, 17 JULI 2024
Pukul	:	07:30 - 09:30
Tempat	:	RUANG SIDANG

telah berlangsung Seminar Hasil Skripsi dengan Peserta:

Nama Mahasiswa	:	FISKA NABILA F. RISAL
NPM	:	07352011108
Judul	:	IMPLEMENTASI NEURAL MACHINE TRANSLATION DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

dinyatakan HARUS menyelesaikan perbaikan, yaitu:

Tambahkan Pembahasan pada Bab IV

Kesimpulan belum menjawab batasan masalah

Abstrak tambahkan data set dan data training

ACC
23-07-24

Dosen Penguji I,

Dr. MUHAMMAD RIDHA ALBAAR, S.Kom., M.Kom.
NIP. 198504232008031001

UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK
PROGRAM STUDI INFORMATIKA

DAFTAR PERBAIKAN SEMINAR HASIL SKRIPSI

Dengan ini dinyatakan bahwa pada

Hari / tanggal	:	RABU, 17 JULI 2024
Pukul	:	07:30 - 09:30
Tempat	:	RUANG SIDANG

telah berlangsung Seminar Hasil Skripsi dengan Peserta:

Nama Mahasiswa	:	FISKA NABILA F. RISAL
NPM	:	07352011108
Judul	:	IMPLEMENTASI NEURAL MACHINE TRANSLATION DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

dinyatakan HARUS menyelesaikan perbaikan, yaitu:

- Ikuti format penulisan
- Belajar pemrograman dengan baik
- Tambahkan teori fungsi aktivasi (sigmoid, tanh, ReLu, dll)
- Pelajari cara kerja penggunaan data (dataset) serta data uji pada rumus2 yang ada pada LSTM
- Cari tahu lebih mendalam fungsi/procedure yang dipakai dari 3 library
- Lihat isi dari masing2 fungsi/procedure yang dipakai tersebut

Dosen Penguji II,

ROSIHAN, S.T., M.Cs.
NIP. 197607192010121001

UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK
PROGRAM STUDI INFORMATIKA

DAFTAR PERBAIKAN SEMINAR HASIL SKRIPSI

Dengan ini dinyatakan bahwa pada

Hari / tanggal	:	RABU, 17 JULI 2024
Pukul	:	07:30 - 09:30
Tempat	:	RUANG SIDANG

telah berlangsung Seminar Hasil Skripsi dengan Peserta:

Nama Mahasiswa	:	FISKA NABILA F. RISAL
NPM	:	07352011108
Judul	:	IMPLEMENTASI NEURAL MACHINE TRANSLATION DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

dinyatakan HARUS menyelesaikan perbaikan, yaitu:

1. Perbaiki format penulisan paragraf (cek naskah)
2. Perbaiki latar belakang masalah

ASC 22/07/2024

Dosen Penguji III,

HAIRIL KURNIADI SIRAJUDDIN, S.Kom., M.Kom.
NIP. 198204272023211009

UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK
PROGRAM STUDI INFORMATIKA

DAFTAR PERBAIKAN UJIAN SKRIPSI/TUTUP

Dengan ini dinyatakan bahwa pada

Hari / tanggal	:	SENIN, 29 JULI 2024
Pukul	:	09:00 - 10:30
Tempat	:	RUANG PRODI

telah berlangsung Ujian Skripsi/Tutup dengan Peserta:

Nama Mahasiswa	:	FISKA NABILA F. RISAL
NPM	:	07352011108
Judul	:	IMPLEMENTASI NEURAL MACHINE TRANSLATION DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

dinyatakan HARUS menyelesaikan perbaikan, yaitu:

Revisi sesuai permintaan Pengujia

Dosen Pembimbing I,

YASIR MUIN, S.T., M.Kom.
NIDN. 9990582796

UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK
PROGRAM STUDI INFORMATIKA

DAFTAR PERBAIKAN UJIAN SKRIPSI/TUTUP

Dengan ini dinyatakan bahwa pada

Hari / tanggal	:	SENIN, 29 JULI 2024
Pukul	:	09:00 - 10:30
Tempat	:	RUANG PRODI

telah berlangsung Ujian Skripsi/Tutup dengan Peserta:

Nama Mahasiswa	:	FISKA NABILA F. RISAL
NPM	:	07352011108
Judul	:	IMPLEMENTASI NEURAL MACHINE TRANSLATION DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

dinyatakan HARUS menyelesaikan perbaikan, yaitu:

- Silahkan lakukan perbaikan sebagaimana yang diminta oleh para penguji.

Dosen Pembimbing II,

Ir. AMAL KHAIRAN, S.T., M.Eng., IPM
NIP. 197401112003121003

UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK
PROGRAM STUDI INFORMATIKA

DAFTAR PERBAIKAN UJIAN SKRIPSI/TUTUP

Dengan ini dinyatakan bahwa pada

Hari / tanggal	:	SENIN, 29 JULI 2024
Pukul	:	09:00 - 10:30
Tempat	:	RUANG PRODI

telah berlangsung Ujian Skripsi/Tutup dengan Peserta:

Nama Mahasiswa	:	FISKA NABILA F. RISAL
NPM	:	07352011108
Judul	:	IMPLEMENTASI NEURAL MACHINE TRANSLATION DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

dinyatakan HARUS menyelesaikan perbaikan, yaitu:

Belajar Dasar Informatika

Acc
31-07-24

Dosen Penguji I,

Dr. MUHAMMAD RIDHA ALBAAR, S.Kom., M.Kom
NIP. 198504232008031001

UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK
PROGRAM STUDI INFORMATIKA

DAFTAR PERBAIKAN UJIAN SKRIPSI/TUTUP

Dengan ini dinyatakan bahwa pada

Hari / tanggal	:	SENIN, 29 JULI 2024
Pukul	:	09:00 - 10:30
Tempat	:	RUANG PRODI

telah berlangsung Ujian Skripsi/Tutup dengan Peserta:

Nama Mahasiswa	:	FISKA NABILA F. RISAL
NPM	:	07352011108
Judul	:	IMPLEMENTASI NEURAL MACHINE TRANSLATION DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

dinyatakan HARUS menyelesaikan perbaikan, yaitu:

- Ikuti format penulisan
- Tambahkan metode pengambangan perangkat lunak (saran: prototype), teorinya di BAB II, penjelasannya di BAB III
- Belajar program untuk membuat inisialisasi matriks, perhitungan matriks (perkalian) dengan python
- Buat program input biodata secara dinamis dan berulang berbasis web

Dosen Penguji II,

ROSIHAN, S.T., M.Cs.
NIP. 197607192010121001

UNIVERSITAS KHAIRUN
FAKULTAS TEKNIK
PROGRAM STUDI INFORMATIKA

DAFTAR PERBAIKAN UJIAN SKRIPSI/TUTUP

ngan ini dinyatakan bahwa pada

Hari / tanggal	:	SENIN, 29 JULI 2024
Pukul	:	09:00 - 10:30
Tempat	:	RUANG PRODI

ah berlangsung Ujian Skripsi/Tutup dengan Peserta:

Nama Mahasiswa	:	FISKA NABILA F. RISAL
NPM	:	07352011108
Judul	:	IMPLEMENTASI NEURAL MACHINE TRANSLATION DENGAN MENGGUNAKAN METODE LSTM PADA PROSES PENERJEMAHAN BAHASA INDONESIA KE BAHASA TERNATE

nyatakan HARUS menyelesaikan perbaikan, yaitu:

1. Perbaiki kesimpulan
2. pelajari dasar HTML, PHH, Phython dan Ms.Excel
3. Perbaiki aplikasi/ batasi data yang tidak ada dalam database

Acc 9/8/2024

Dosen Penguji III,

MAIRIL KURNIADI SIRAJUDDIN, S.Kom., M.Kom.
NIP. 198204272023211009