BAB II
TINJAUAN PUSTAKA

1.1 *State of the Art*

Adanya penelitian sebelumnya bertujuan untuk membandingkan dengan penelitian yang sedang dilakukan. Dalam penelitian ini disertakan lima jurnal sebelumnya yang berhubungan dengan judul tugas akhir ini yaitu "Aplikasi Alih Suara ke Teks dan Perangkuman Kata dalam Konsultasi Kesehatan Menggunakan *Bidirectional Representation From Transformer* dan *Bidirectional and Auto-Regresive Transformer*". Jurnal tersebut antara lain;

Penelitian dengan judul "*Automatic Text Summarization of COVID-19 Medical Research Articles using BERT and GPT-2*" yang ditulis oleh Virapat Kieuvongngam, dkk. dan diterbitkan pada tahun 2020. Jurnal ini berasal dari arXiv Cornell University. Metode yang digunakan pada jurnal ini adalah model NLP seperti BERT dan GPT-2, yang berfungsi untuk merangkum literatur medis terkait COVID-19 melalui pendekatan ekstraktif dan abstraktif. Jurnal ini membahas penggunaan model NLP untuk menyederhanakan informasi medis terkait COVID-19 menjadi ringkasan yang dapat diinterpretasikan dan masuk akal. Hasil dari jurnal ini adalah meskipun ringkasan yang dihasilkan dapat diinterpretasikan dan masuk akal, kinerja model belum mencapai tingkat yang setara dengan manusia. Jurnal ini memiliki kelebihan dalam penggunaan inovatif model NLP untuk menangani kebutuhan mendesak selama pandemi dengan menyediakan pendekatan komprehensif melalui metode ganda ekstraktif dan abstraktif, tetapi jurnal ini memiliki kekurangan seperti keterbatasan data pelatihan spesifik domain, metrik evaluasi yang tidak sepenuhnya mencerminkan kualitas ringkasan abstraktif, keterbatasan sumber daya komputasi, dan proses ekstraksi kata kunci yang dapat ditingkatkan [4].

Penelitian dengan judul "*Application of Extractive Text Summarization Algorithms to Speech-to-Text Media*" yang ditulis oleh Domínguez M. Victor, dkk

dan diterbitkan pada tahun 2022. Jurnal ini berasal dari "*Journal of Computational Linguistics.*" Metode yang digunakan pada jurnal ini adalah algoritma ekstraksi, yang berfungsi untuk meringkas teks dari media hasil transkripsi suara ke teks. Jurnal ini membahas bagaimana algoritma ekstraksi dapat diterapkan pada media transkripsi suara untuk menghasilkan ringkasan teks yang efektif. Hasil dari jurnal ini adalah algoritma tersebut mampu meringkas teks dengan cukup baik. Jurnal ini memiliki kelebihan dalam mempercepat proses pemahaman informasi dari teks yang panjang dan kemudahan penerapannya pada berbagai jenis media, tetapi jurnal ini memiliki kekurangan dalam menangkap konteks yang lebih dalam dan menghadapi masalah dengan akurasi ketika teks yang diringkas memiliki struktur yang kompleks [5].

Penelitian dengan judul "Aplikasi Konversi Suara ke Teks Berbasis Android Menggunakan *Google Speech API*" yang ditulis oleh Supriyanta, dkk dan diterbitkan pada tahun 2014. Jurnal ini berasal dari jurnal Bianglala Informatika edisi September 2014. Metode yang digunakan pada jurnal ini adalah *Hidden Markov Model* (HMM), yang berfungsi untuk mengembangkan aplikasi berbasis Android yang dapat mengkonversi suara menjadi teks. Jurnal ini membahas mengenai pengembangan aplikasi yang mampu mengubah suara menjadi teks dalam Bahasa Indonesia menggunakan *API Google Speech* dan bahasa pemrograman Java. Hasil dari jurnal ini adalah aplikasi tersebut berhasil melakukan konversi suara ke teks dengan tingkat akurasi yang cukup baik. Jurnal ini memiliki kelebihan dalam memfasilitasi proses input teks yang lebih efisien, terutama bagi pengguna yang memiliki kesulitan dalam mengetik , tetapi jurnal ini memiliki kekurangan ketergantungan pada *API Google Speech* yang memerlukan koneksi internet serta keterbatasan dalam mengenali berbagai variasi aksen dan intonasi dalam Bahasa Indonesia [6].

Penelitian dengan judul "*Speech Recognition for Medical Conversations*" yang ditulis oleh Chung-Cheng Chiu, dkk dan diterbitkan pada tahun 2018. Jurnal ini berasal dari arXiv Cornell University. Metode yang digunakan pada jurnal ini adalah model berbasis fonem dengan *Connectionist Temporal Classification*

(CTC) dan model berbasis grafem dengan *Listen Attend and Spell (LAS)*, yang berfungsi untuk mengenali dan mentranskripsi percakapan antara dokter dan pasien secara otomatis. Jurnal ini membahas mengenai pengembangan sistem pengenalan suara yang dilatih menggunakan sekitar 14.000 jam percakapan anonim, dengan fokus pada proses pembersihan data, model CTC, dan LAS dalam menghadapi data yang memiliki banyak *noise*. Hasil dari jurnal ini adalah model CTC mencapai tingkat kesalahan kata sebesar 20,1%, sedangkan model LAS mencapai 18,3%, menunjukkan kemampuan kedua model dalam mengenali ungkapan medis penting. Jurnal ini memiliki kelebihan dalam memaparkan bagaimana model LAS lebih tahan terhadap kebisingan dalam transkrip dan penyelarasan, sehingga tidak membutuhkan model bahasa [7].

Penelitian dengan judul "Implementasi Aplikasi *Speech to Text* untuk Memudahkan Wartawan Mencatat Wawancara dengan Python" yang ditulis oleh I Komang Setia Buana dan diterbitkan pada tahun 2020. Jurnal ini berasal dari Jurnal Sistem dan Informatika. Metode yang digunakan pada jurnal ini adalah pengujian aplikasi yang dikembangkan dengan bahasa pemrograman Python dan modul *speech recognition*, yang berfungsi untuk mengonversi suara menjadi teks secara otomatis. Jurnal ini membahas mengenai tantangan yang dihadapi wartawan dalam mencatat wawancara secara manual dan bagaimana aplikasi yang dikembangkan dapat membantu proses tersebut. Hasil dari jurnal ini adalah aplikasi ini memiliki tingkat keberhasilan mencapai 94,75% dalam mengonversi suara menjadi teks. Jurnal ini memiliki kelebihan dalam menghemat waktu wartawan dan meningkatkan efisiensi dalam mencatat wawancara, tetapi jurnal ini memiliki kekurangan ketergantungan pada kualitas suara input dan kondisi noise yang dapat mempengaruhi akurasi hasil transkripsi [8].

1.2 *Rekam Medis Elektronik (RME)*

Rekam Medis Elektronik (RME) atau *Electronic Health Record* (EHR) adalah bentuk sistem informasi kesehatan berbasis elektronik yang diterapkan dalam konteks layanan kesehatan yang memberikan fokus pada aspek pasien.

RME mencakup aspek-aspek penting seperti diagnosis, Riwayat kesehatan sekarang, riwayat kesehatan dahulu dan rencana pengobatan. Sebagai catatan kesehatan yang komprehensif, RME berperan sebagai alat yang mendokumentasikan riwayat kesehatan seseorang, memfasilitasi para penyedia layanan kesehatan dalam memberikan perawatan yang efektif. Dalam skala global, rumah sakit telah beralih dari rekam kesehatan berbasis kertas menuju RME, dan Indonesia juga mengadopsi perubahan ini. Pada tanggal 12 September 2022, Kementerian Kesehatan Republik Indonesia menerbitkan Peraturan Menteri Kesehatan (Permenkes) Nomor 24 Tahun 2022 tentang Rekam Medis, menjadi dasar hukum untuk penerapan RME. Aturan ini mencerminkan dukungan terhadap transformasi teknologi di bidang kesehatan, sesuai dengan pilar ke-6 Transformasi Kesehatan [9][10]. Pada aplikasi ini rekam medis yang dilakukan adalah merekam hasil rangkuman percakapan antara pasien dengan dokter untuk mempermudah pasien dalam mengingat hasil konsultasi tersebut.

1.3 NLP *(Neural Language Processing)*

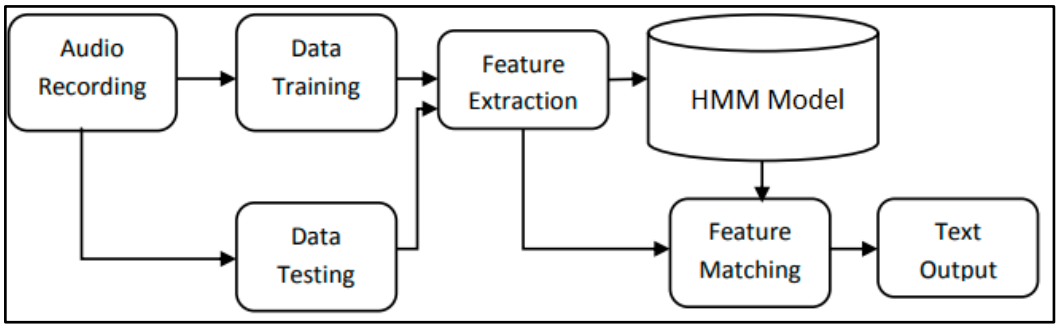
NLP atau *Neural Language Processing* adalah cabang kecerdasan buatan (*Artificial Intelligence*) yang berfokus pada pemrosesan bahasa alami. Proses ini bertujuan untuk menemukan pola atau mengekstrak informasi baru yang bertujuan untuk menemukan informasi-informasi penting yang ada dalam teks, dengan cara mengonversi teks menjadi data yang dapat digunakan untuk analisis lebih lanjut. Biasanya dalam proses NLP meliputi beberapa tahapan seperti pada tahap pertama adalah *case folding*, tokenisasi (*tokenization*), penyaringan (*Filtering*), *steeming* dan tahap terakhir adalah menganalisa proses analisis dari *steeming*, sejauh mana tingkat keterhubungan kata-kata dan antara dokumen-dokumen yang ada [11]. Adapun penjelasan dari *case folding, tokenization, filtering,* dan *steeming.*

- *Case folding* : adalah langkah yang mengubah semua huruf dalam dokumen menjadi huruf kecil. Hanya huruf 'a' sampai 'z' yang diterima. Karakter selain huruf dihilangkan dan dianggap sebagai pemisah [11].

- Tokenisasi : adalah proses memecah kalimat menjadi bagian-bagian kata yang disebut token. Selain itu, spasi digunakan untuk memisahkan kata-kata. Proses ini dilakukan agar sistem dapat memahami input dari pengguna [11].
- *Filtering* : adalah tahap mengambil kata-kata penting dari hasil proses tokenisasi. Gunakan algoritma stoplist (membuang kata-kata yang kurang penting) atau word list (menyimpan kata-kata penting). Dalam sistem ini, metode *stopword* digunakan untuk menghilangkan kata-kata yang tidak perlu dengan memeriksa kata-kata hasil tokenisasi. Jika ada *stopword*, maka kata tersebut akan dihapus sehingga kata-kata yang tersisa dianggap penting atau menjadi kata kunci (pola) [11].
- Stemming : bertujuan untuk mengembalikan kata-kata ke bentuk dasarnya (kata dasar) dengan menghilangkan semua imbuhan kata (afiks), termasuk prefiks (awalan), infiks (sisipan), sufiks (akhiran), dan/atau menghilangkan prefiks dan sufiks (konfiks) pada kata turunan [11].

1.4 *Speech Recognition*

Speech Recognition merupakan teknologi yang memungkinkan ucapan langsung dikonversi menjadi teks tertulis, dengan memanfaatkan sinyal suara manusia sebagai masukan untuk dikenali oleh sistem komputer [12][8]. Adapun alur gambaran umum dari perancangan *Speech to Text* yang digambarkan pada gambar 2.1 berikut.



Gambar 2.1 Diagram Perancangan *Speech to Text*

1) Sinyal Suara (*Speech Signal*) masuk melalui mikrofon
2) Fitur ekstraksi ciri (*Feature Extraction*) dari sinyal suara
3) *Training* dataset dari *Freature Extraction*
4) *Testing* dataset dari *Feature Extraction*
5) *Output* berupa teks dari sinyal yang dikenali

Pada proses pengenalan suara di aplikasi ini menggunakan *Google speech API,* sebuah kerangka kerja yang dibuat oleh *google* untuk bisa mengidentifikasi suara yang akan dirubah menjadi string (teks), dimana untuk cara mengidentifikasinya dengan menggunakan digitalisasi kata dan pencocokan sinyal digital tersebut dengan suatu pola yang disimpan di dalam *database* Google. Metode yang digunakan pada *Google Speech API* menggunakan algoritma *Hidden Markov Model (HMM)* [12]. Untuk mendapatkan akses dalam melakukan konversi suara ke teks menggunakan *Google Speech API*. *Developer* android dapat menggunakan *interface* dan *class* yang telah disediakan oleh *Google API* [13]

1.4.1 *Hidden Markov Model*

Hidden Markov Model (HMM) adalah model statistik di mana sistem yang dimodelkan dianggap sebagai proses Markov dengan parameter yang tidak diketahui. Tujuannya adalah menentukan parameter tersembunyi dari parameter yang dapat diamati. Parameter model yang diekstraksi kemudian dapat digunakan untuk analisis lebih lanjut, seperti pada aplikasi pengenalan pola atau ucapan. Dalam HMM, *state* adalah kondisi tersembunyi dari sistem yang dimodelkan. Setiap *state* dalam HMM tidak dapat diamati secara langsung, tetapi menentukan distribusi probabilitas keluaran yang dapat diamati. Dengan kata lain, *state* adalah variabel yang mengontrol bagaimana token observasi dihasilkan, dan transisi antar *state* mengikuti aturan probabilistik tertentu. Informasi mengenai urutan *state* diperoleh melalui urutan keluaran token yang diamati. *State* tidak terlihat secara langsung, namun variabel yang dipengaruhi oleh *state* dapat diamati. Setiap *state* memiliki distribusi probabilitas terhadap kemungkinan keluaran token. Transisi

antar *state* juga bersifat probabilistik. Oleh karena itu, urutan token yang dihasilkan oleh HMM memberikan informasi mengenai urutan *state* [14].

1.5 *Peringkasan Teks*

Peringkasan teks adalah sebuah metode untuk mengekstrak infromasi penting dari sebuah teks dan menyajikannya kepada pembaca dalam bentuk yang sederhana, singkat, dan padat, yang tetap mempertahankan isi bahasan yang relevan. Relevansi yang tinggi, redudansi yang sedikit, rasio kompresi yang sesuai, dan cakupan yang tinggi adalah faktor penting dari sebuah ringkasan teks yang baik. Dua klasifikasi umum pada peringkasan teks adalah peringkasan teks ekstraktif dan peringkasan teks abstrak [15].

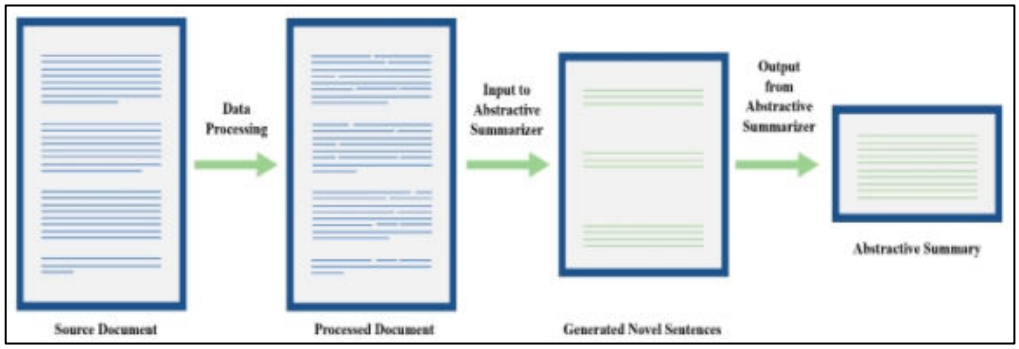
1.5.1 Peringkasan Teks Ekstraktif

Peringkasan teks ekstraktif akan mengekstrak kalimat-kalimat penting dari dokumen sumber dan menggabungkannya untuk membentuk ringkasan ekstraktif. Langkah awal yaitu untuk memilih kalimat-kalimat yang menonjol, dengan cara memberikan bobot pada semua kalimat pada dokumen, kemudian kalimat yang bobotnya mempunyai peringkat tinggi, akan diekstrasi dari dokumen sumber. Kalimat-kalimat yang telah di ekstrak, kemudian digabungkan untuk menghasilkan ringkasan kalimat. Gambar 2.2 mengilustrasikan representasi diagram umum pada model peringkasan teks ekstraktif. Garis biru pada dokumen atau teks yang ditunjukan pada gambar tersebut, menggambarkan teks asli dan garis kuning pada gambar tersebut mewakili kalimat-kalimat penting yang dipilih saat peringkasan ekstraktif [15].

Gambar 2.2 Perangkuman Teks Ekstraktif

1.5.2 Peringkasan Teks Abstrak

Dalam Teknik peringkasan teks abstrak, ringkasan yang akan dihasilkan terdiri dari kalimat-kalimat baru yang tidak diekstrak dari dokumen sumbernya. Kalimat-kalimat ini menyampaikan ide utama dari dokumen sumbernya dan teks disusun ulang sehingga akan menghasilkan ringkasan yang serupa dengan tulisan manusia. Ringkasan abstrak akan lebih kompleks, memakan waktu yang lebih banyak, dan juga membutuhkan analisis yang ekstensif sehingga metode peringkasan ini jarang digunakan orang-orang. Meski cenderung untuk meningkatakan koherensi, keterbacaan, dan kohesi, ringkasan yang dihasilkan terkadang terganggu oleh berbagai masalah seperti ketidaktepatan, ketidak masuk akalan, dan pengulangan kalimat. Gambar 2.3 mengilustrasikan representasi diagram umum pada model peringkasan teks abstrak. Garis biru pada dokumen atau teks yang ditunjukan pada gambar tersebut, menggambarkan teks asli dan garis hijau pada gambar tersebut mewakili menggambarkan kalimat baru yang dihasilkan oleh peringkas teks abstrak [15].



Gambar 2.3 Perangkuman Teks Abstraktif

1.6 *Named Entity Recognition (NER)*

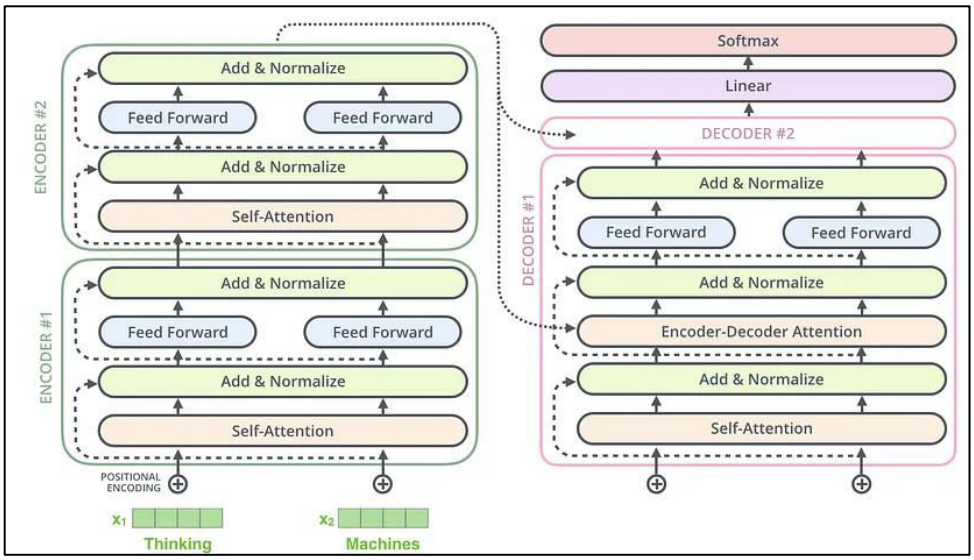
Named Entity Recognition (NER) adalah bagian dari ekstraksi informasi yang bertugas untuk mengklasifikasikan teks dalam dokumen atau korpus ke dalam kategori tertentu seperti nama orang, lokasi, organisasi, tanggal, dan waktu. NER banyak digunakan dalam berbagai bidang, termasuk *machine translation*, sistem tanya-jawab, *indexing* pada *information retrieval*, klasifikasi, dan *automatic summarization*. Tujuan utama NER adalah mengekstraksi dan mengklasifikasikan entitas dengan makna yang tepat [16].

Dalam NER, terdapat dua jenis model *machine learning* yang bisa digunakan: *supervised learning* dan *unsupervised learning*. *Supervised learning* melibatkan penggunaan program yang belajar mengklasifikasikan data berdasarkan label yang telah diberikan dengan jumlah fitur yang sama [16].

1.7 *Transformers*

Transformer adalah model yang digunakan dalam bidang *Neural Language Processing (NLP)* yang bertujuan untuk memahami dan menghasilkan teks dengan cara yang efisien dan efektif. *Transformer* sendiri adalah arsitektur model yang mengandalkan mekanisme perhatian (*attention*) sepenuhnya, menghilangkan penggunaan rekrusi dan konvolusi yang umumnya digunakan dalam model transduksi sekuensial [17].

Transformer terdiri dari dua komponen utama: *encoder* dan *decoder*, masing-masing terdiri dari beberapa tumpukan lapisan [17]. Pada gambar 2.4 terdapat arsitektur pada *transfomer* yang berisi *encoder* dan *decoder*. Berikut penjelasan dari *Encoder* dan *Decoder* pada arsitektur tersebut.



Gambar 2.4 Arsitektur *Transformer*

1) *ENCODER*
   a) Sub-lapisan :
      i) *Self-Attention*: Sub-lapisan ini digambarkan sebagai blok "*Self-Attention*" di dalam setiap *encoder*. Fungsi dari sub-lapisan ini adalah untuk memungkinkan setiap posisi dalam input memperhatikan semua posisi lainnya, yang membantu model dalam memahami hubungan antar kata dalam kalimat. [17].
      ii) *Feed-Forward* : Terlihat sebagai blok "*Feed Forward*" dalam setiap *encoder*, sub-lapisan ini adalah jaringan saraf sederhana yang bekerja secara individual pada setiap posisi dalam input, tanpa adanya interaksi antar posisi [17].
   b) Koneksi Residual dan Normalisasi Lapisan : Pada setiap sub-lapisan (baik *Self-Attention* maupun *Feed-Forward*), hasil akhirnya adalah kombinasi dari input asli ditambah dengan hasil sub-lapisan tersebut, yang kemudian

dinormalisasi melalui blok "*Add & Normalize*" untuk menjaga stabilitas dan membantu model belajar lebih cepat dan lebih baik. [17].

2) *DECODER*
   a) Sub-lapisan :
      i) *Self-attention* dengan *Masking* : Digambarkan sebagai blok "*Self-Attention*" dalam *decoder*, sub-lapisan ini mirip dengan yang ada di *encoder*, tetapi dengan tambahan mekanisme *masking* yang mencegah model memperhatikan posisi kata yang belum dihasilkan. [17].
      ii) *Multi-Head Attention* pada Keluaran *Encoder* : Sub-lapisan ini digambarkan sebagai "*Encoder-Decoder Attention*". Fungsinya adalah untuk memungkinkan setiap posisi dalam *decoder* memperhatikan semua posisi dalam keluaran *encoder*, yang membantu model memahami konteks dari input [17].
      iii) *Feed-Forward* : Sama seperti di *encoder*, blok "*Feed Forward*" dalam *decoder* adalah jaringan saraf sederhana yang bekerja secara individual pada setiap posisi dalam input [17].
   b) Koneksi Residual dan Normalisasi Lapisan : Sama seperti pada *encoder*, setiap sub-lapisan memiliki koneksi residual dan normalisasi lapisan untuk menjaga stabilitas dan efisiensi dalam pembelajaran modelnya [17].
   c) Output : Hasil dari lapisan terakhir *decoder* diubah menjadi skor untuk setiap kata dalam kosakata melalui lapisan linier dan fungsi *softmax*, yang menentukan kata berikutnya yang akan dihasilkan berdasarkan probabilitas tertinggi [17].

1.8 *BERT (Bidirectional Encoder Representations from Transformers)*

BERT adalah singkatan dari *Bidirectional Encoder Representations from Transformers*. BERT berfungsi untuk melatih representasi mendalam yang bidireksional (kemampuan model agar memahami konteks dari kedua arah) dari teks yang tidak memiliki label dengan cara mempertimbangkan konteks baik dari sebelah kiri maupun sebelah kanan dalam semua lapisan. Sehingga, model BERT yang telah dilatih dapat disesuaikan lebih lanjut dengan cara menambahkan satu

lapisan keluaran tambahan untuk menciptakan model terkini dalam berbagai tugas [18].

1.8.1 Arsitektur BERT

Model BERT, atau *Bidirectional Encoder Representations from Transformers*, adalah model pembelajaran mesin yang dirancang untuk meningkatkan representasi bahasa dengan pelatihan dua arah. BERT mengatasi keterbatasan model bahasa searah dengan menggunakan metode pelatihan "*masked language model*" (MLM), di mana beberapa token dari input secara acak disembunyikan dan model dilatih untuk memprediksi token yang disembunyikan tersebut berdasarkan konteks di sekitarnya. Ini memungkinkan BERT untuk mempelajari representasi yang menggabungkan konteks dari kedua arah (kiri dan kanan), yang sangat penting untuk tugas-tugas pemrosesan bahasa alami seperti menjawab pertanyaan dan pengenalan entitas [18]. BERT menggunakan dua ukuran yang digunakan yaitu:

1) *BERTBASE* : 12 lapisan, ukuran tersembunyi 768, 12 kepala *self-attention* [18].
2) *BERTLARGE* : 24 lapisan, ukuran tersembunyi 1024, 16 kepala *self-attention* [18].

1.8.2 *Pre-Training* BERT

Pada proses *pre-training*, model BERT dilatih menggunakan data tidak berlabel melalui dua tugas utama, yaitu:

1) *Masked Language Model (MLM)* : Beberapa token dalam input disembunyikan secara acak dan meminta model untuk memprediksi token yang disembunyikan tersebut, hal ini akan melatih model untuk memahami konteks dari kedua arah (kiri dan kanan) secara bersamaan [18].
2) *Next Sentence Prediction (NSP)* : Model dilatih untuk memprediksi apakah dua kalimat yang diberikan secara berurutan benar-benar berurutan dalam teks

asli. Tugas ini membantu model dalam memahami hubungan antar kalimat [18].

1.8.3 *Fine-Tuning* BERT

Pada aplikasi ini, model BERT di-tuning untuk dapat menganalisis teks terhadap topik keluhan, penggunaan obat, dan saran penyembuhan. Proses *training* model BERT dimulai dengan persiapan dataset yang terdiri dari teks yang telah dilabeli menjadi tiga kategori: keluhan, obat, dan saran, masing-masing label memiliki 100 data dengan total keseluruhan dataset sebesar 300 data. Dataset ini berasal dari diskusi tanya jawab di web alodokter.com dan data yang dibuat sendiri, yang berisi teks keluhan, penggunaan obat, dan saran. Dataset ini diacak dan dibagi menjadi *training* dan test set dengan rasio 80:20 (*training* 240 data dan data tes 60 data) untuk memastikan model dapat dilatih dan diuji dengan baik. Model BERT yang digunakan adalah *bert-base-uncased*, yang di-*finetune* untuk melakukan klasifikasi teks menjadi tiga label berbeda. Proses tokenisasi dilakukan menggunakan *BertTokenizer*, yang memastikan setiap teks memiliki panjang yang seragam dengan menerapkan padding dan truncation sesuai kebutuhan. Adapun parameter pelatihan model ini sebgai berikut:

- Proporsi Data : 80% untuk data latih, 20% untuk data uji
- Jumlah *Epoch* Pelatihan : 100
- *Loss* Evaluasi : 0.1199
- Waktu Evaluasi : 1.888 detik
- Kecepatan Evaluasi :
    - Sampel per Detik: Sekitar 31.776
    - Langkah Evaluasi per Detik: Sekitar 15.888

1.9 *BART (Bidirectional and Auto-Regressive Transformers)*

BART adalah singkatan dari *Bidirectional and Auto-Regressive Transformers*. Ini adalah model pemrosesan bahasa yang dirancang untuk memulihkan teks yang telah diubah atau dirusak. BART bekerja dengan cara

mengubah teks asli dengan menambahkan *noise*, seperti mengganti beberapa kata dengan simbol tertentu. Setelah itu, model ini mencoba mengembalikan teks yang rusak tersebut ke bentuk aslinya.

Selama proses pelatihan, BART melalui dua tahap utama: pertama, teks asli diubah dengan menambahkan gangguan, dan kedua, model dilatih untuk memperbaiki teks yang rusak tersebut. Keunikan BART adalah kemampuannya untuk memodifikasi dan memperbaiki berbagai jenis gangguan pada teks, karena input yang diterima oleh *encoder* tidak harus persis sama dengan *output* dari *decoder*.

Sebagai contoh, jika sebuah dokumen dirusak dengan mengganti beberapa kata dengan simbol *mask*, BART akan menggunakan model *bidirectional* untuk menganalisis dokumen tersebut dan kemudian menggunakan *decoder autoregresif* untuk mencoba menebak kata-kata yang hilang dan mengembalikannya ke bentuk aslinya. Ketika dilakukan penyesuaian lebih lanjut, dokumen yang tidak diubah dimasukkan ke dalam model untuk mendapatkan representasi terbaik dari teks tersebut. [19]. Model BART dikenal dengan kemampuannya dalam menghasilkan teks secara generatif, yang membuatnya cocok untuk berbagai tugas pemrosesan bahasa alami seperti penringkasan teks, penerjemahan, pemulihan kata, parafrase, dan banyak lagi.

1.9.1 Arsitektur BART

BART menggunakan arsitektur *transformer* standar yang terdiri dari dua bagian: *encoder bidirectional* dan *decoder autoregresif*. Pada model *base* BART, baik *encoder* maupun *decoder* masing-masing memiliki 6 lapisan, sementara pada model *large* BART, keduanya memiliki 12 lapisan. Arsitektur BART mirip dengan BERT, tetapi terdapat perbedaan sebagai berikut:

1) Setiap lapisan pada *decoder* BART melakukan *cross-attention* pada lapisan tersembunyi terakhir dari *encoder*, mirip dengan cara kerja model *sequence-to-sequence Transformer*.

2) BERT memiliki jaringan tambahan sebelum melakukan prediksi kata, sedangkan BART tidak menggunakan jaringan ini.

Secara keseluruhan, BART memiliki sekitar 10% lebih banyak parameter dibandingkan dengan model BERT yang memiliki ukuran yang sama, yang berarti BART sedikit lebih kompleks.

1.9.2 *Pre-Training* BART

BART dilatih dengan cara merusak dokumen dan kemudian memperbaikinya, mengukur seberapa baik model bisa mengembalikan dokumen asli. Ini dilakukan dengan menghitung perbedaan (*cross-entropy*) antara hasil dari model dan dokumen asli. Berbeda dari model lain yang biasanya dirancang untuk memperbaiki jenis kerusakan tertentu, BART bisa menangani berbagai jenis kerusakan dokumen. Jika semua informasi dalam dokumen hilang, BART berfungsi seperti model bahasa biasa. Model BART memiliki sekitar 10% lebih banyak parameter daripada model BERT dengan ukuran yang sama.

1.9.3 *Fine-Tuning* BART

Pada aplikasi ini digunakan model *fine-tuning* BART *Mr-Vicky-01/Bart-Finetuned-conversational-summarization* dari *website HuggingFace*. Model ini awalnya merupakan model *BART Large* dengan arsitektur *BART Large* CNN yang telah di *pre-trained* oleh facebook, kemudian disesuaikan lebih lanjut untuk tugas peringkasan teks menggunakan dataset XSum dan SAMSum. Dengan parameter pelatihan sebagai berikut:

- *Num train epochs* : 1,
- *Warmup steps* : 500,
- *Per device train batch size* : 4,
- *Per device eval batch size* : 4,
- *Weight decay* : 0.01,

- *Gradient accumulation steps* :16

1.10 *Word Error Rate*

Word Error Rate (WER) adalah matrik yang digunakan untuk mengevaluasi hasil dari transkripsi pada sistem *Automatic Speech Recognition (ASR)*. Nilai WER didapatkan dari perbandingan teks referensi degan teks hasil dari transkripsi dan menghitung jumlah *insertion* (sistem menambahkan kata yang tidak ada dalam teks referensi), *deletion* (sistem menghilangkan kata yang ada dalam teks referensi), *substitution* (sistem mengganti satu kata yang dalam teks referensi) yang terjadi. WER adalah rasio jumlah kesalahan terhadap jumlah kata pada transkripsi teks referensi. Adapun rumus WER sebagai berikut [21] :

$$WER = \frac{\mathrm{S} + \mathrm{D} + \mathrm{I}}{\mathrm{N}} \times 100 \tag{2.1}$$

Untuk rumus akurasi :

$$Accuracy = 100 - WER \tag{2.2}$$

1.11 *ROUGE-N*

Rouge-N adalah salah satu matriks dalam *toolkit ROUGE* yang digunakan untuk mengevaluasi kualitas ringkasan otomatis dengan menghitung N-Gram yang tumpang tindih antara ringkasan yang dihasilkan dan ringkasan referensi. *ROUGE-N* bisa untuk berbagai nilai N, misalnya ROUGE-1 menghitung *overlap*

unigram (kata tunggal), ROUGE-2 menghitung *overlap* bigram (pasangan kata) [20].

ROUGE sendiri terdiri dari *Precision, Recal* dan F1. *Precision, Recal* dan *F1-Measure* [21] adalah matriks evaluasi yang sering digunakan dalam pengambilan informasi dan sistem klasifikasi, termasuk dalam penilaian kualitas rangkuman otomatis.

1. ***Precision* (P)**

   Precision adalah metode untuk mengukur jumlah prediksi relevan dengan cara menghitung jumlah kata yang sama, dibagi dengan keseluruhan kata pada ringkasan sistem [21].

$$Rouge - 1\ Precisio = \frac{TP}{TP + FP} \tag{2.3}$$

2. ***Recall* (R)**

   Recall adalah metode untuk mengukur jumlah prediksi yang relevan dengan cara menghitung jumlah kata yang dibagi dengan keseluruhan kata pada ringkasan manusia [21].

$$Rouge - 1\ Recall = \frac{TP}{TP + FN} \tag{2.4}$$

3. ***F1-Measure* (F1)**

   F1- Measure atau *F-score* adalah metode untuk mengukur nilai rata-rata harmonok (*harmonic mean*) antara *recall* dan *precision* [21].

$$f1 - scores = 2 \times \frac{precision \times recall}{precision + recall} \quad (2.5)$$

1.12 *Flask*

Flask adalah sebuah microframework yang memungkinkan pengembang untuk membuat aplikasi web yang tertata dan terstruktur dengan baik. Kelebihan utama *Flask* adalah kemampuannya untuk dengan mudah mengintegrasikan berbagai ekstensi dan pustaka pihak ketiga yang diperlukan dalam pengembangan aplikasi web [22]. *Flask* memiliki sebuah *file* dengan nama __init__.py berisi inti dari *flask* itu sendiri yang nantinya akan menampung *routes* yang diperlukan.