BAB I
PENDAHULUAN

1.1 Latar Belakang

Web scraping merupakan sebuah proses pengambilan dokumen semi-terstruktur dari halaman website yang umumnya berupa dokumen yang disusun dalam bahasa Markup seperti HTML atau XHTML. Dengan adanya teknik ini tidak dipelukan proses pengumpulan data tradisional seperti survei atau kuesioner (Ayani et al., 2023). Hal ini dapat menghemat waktu pengumpulan data. Contoh penggunaan dari teknik *web scraping* adalah proses pengumpulan data teks berupa ulasan aplikasi pada Google Play Store. Seperti pada jurnal penelitian oleh Aldabbas et al. (2019) berjudul "*Google Play Content Scraping and Knowledge Engineering using Natural Language Processing Techniques with the Analysis of User Reviews"*. Dalam Artikel ini membahas salah satu teknik pengelolaan data teks adalah dengan memanfaatkan kemampuan kecerdasan buatan untuk membaca arah teks yang menunjukkan kepuasan pada aplikasi.

Kecerdasan buatan sendiri merupakan sebuah teknik dalam ilmu komputer dan matematika untuk menciptakan sebuah kecerdasan yang akan dimiliki oleh mesin untuk mengambil dan menentukan sebuah keputusan (Chahal et al., 2019). Kecerdasan buatan juga mampu menghasilkan data baru dengan menyesuaikan data acak menjadi informasi yang bermakna. Kemampuan kecerdasan buatan ini bisa dicapai dengan teknik atau proses yang dikenal dengan proses pembelajaran mesin (*machine learning*). Untuk data yang berbentuk teks secara khusus memiliki teknik pembelajaran yang agak berbeda dengan data numerik atau data terstruktur yang memanfaatkan teknik pembelajaran mesin, teknik ini dikenal dengan konsep pembelajaran mendalam (*deep learning*). Konsep pembelajaran mendalam ini biasanya menggunakan ANN (*Artificial Neural Network*) yang merupakan sebuah algoritma yang bekerja dengan meniru cara kerja otak untuk mengenali pola dari sebuah data yang diberikan (Chahal et al., 2019).

Ada banyak algoritma ANN yang telah berkembang untuk mempermudah proses pengenalan data sekuensial seperti data teks, salah satunya adalah RNN (*Recurrent Neural Network*). Sebagaimana Lipton et al. (2015) menyatakan bahwa *Recurrent Neural Networks* (RNN) telah banyak dikaji secara kritis untuk pembelajaran sekuensial. Salah satu contoh algoritma RNN yang sudah secara sukses salah satunya adalah LSTM (*Long Short-Term Memory*). Varian RNN ini dirancang untuk menangani ketergantungan jangka panjang pada data sekuensial. Namun dalam proses pemanfaatan algoritma ini diperlukan proses pelatihan dan
ga bisa menghasilkan model yang sesuai standar (Lipton et al.,
masalah yang umum adalah ketidakseimbangan data (*data*
ali menyebabkan model lebih condong mempelajari dan
oritas, sehingga kinerja pada kelas minoritas menurun (Li et al.,
ya masalah ini dapat diatasi dengan teknik *oversampling*
undersampling, atau metode adaptif lainnya. Namun, seperti

yang dijelaskan oleh Li et al. (2022). Metode tersebut tidak selalu efektif pada ketidakseimbangan yang ekstrem, dan terkadang dapat menyebabkan overfitting atau menghasilkan sampel yang kurang representatif.

Generative Adversarial Network (GAN) merupakan salah satu kerangka pembelajaran mesin yang terdiri atas dua jaringan saraf tiruan, yaitu *generator* dan *discriminator*, yang dilatih secara bersamaan dalam suatu proses kompetitif untuk menghasilkan data sintetis yang menyerupai distribusi data asli (Sharma, Singh, & Chandra, 2022). Awalnya, GAN dikembangkan untuk menghasilkan data realistis pada dataset berukuran besar. Arsitektur ini juga bisa dikembangkan dan dimanfaatkan dalam berbagai keperluan augmentasi data, termasuk untuk mengatasi permasalahan ketidakseimbangan kelas (*class imbalance*) yang dibahas sebelumnya dengan cara menghasilkan sampel minoritas yang lebih representatif.

Dalam penelitian ini, data pelatihan LSTM diperoleh melalui proses *web scraping* ulasan pengguna aplikasi JKN di Google Play Store. Pemanfaatan GAN diharapkan mampu menghasilkan data sintetis yang lebih berkualitas dibandingkan metode oversampling tradisional, sehingga dapat mengurangi ketidakseimbangan kelas sekaligus meningkatkan performa model dalam analisis kepuasan pengguna. Beberapa penelitian sebelumnya telah menunjukkan bahwa penerapan GAN pada kasus ketidakseimbangan data mampu meningkatkan metrik evaluasi model secara signifikan (Sharma, Singh, & Chandra, 2022). Oleh karena itu, penelitian ini tidak hanya memanfaatkan tetapi juga mengembangkan arsitektur GAN yang sesuai dengan karakteristik data teks ulasan berbahasa Indonesia pada aplikasi mobile JKN.

1.2 Rumusan Masalah

Berdasarkan uraian pada latar belakang, maka dapat diidentifikasi permasalahan yang perlu dikaji dalam penelitian ini, yakni sebagai berikut:

1. Bagaimana keunggulan *Generative Adversarial Network* (GAN) dibandingkan dengan metode tradisional seperti SMOTE dalam mengatasi ketidakseimbangan data teks ulasan pengguna?
2. Bagaimana cara meningkatkan kinerja GAN dalam proses analisis kepuasan pengguna aplikasi Mobile JKN?
3. Bagaimana Arsitektur yang dapat dikembangkan untuk mendukung pemanfaatan GAN dalam analisis kepuasan pengguna aplikasi Mobile JKN?

1.3 Batasan Masalah

Adapun batasan masalah dari penelitian ini adalah sebagai berikut:

1. Penelitian ini berfokus pada pengembangan dan pemanfaatan algoritma GAN (Generative Adversarial Network) untuk menghasilkan data sintetis kelas minoritas.
2. Penelitian ini berfokus pada penerapan GAN untuk meningkatkan metrik evaluasi dalam analisis kepuasan pengguna aplikasi Mobile JKN.
3. Penelitian ini tidak membahas metode lain di luar GAN dalam konteks analisis kepuasan aplikasi mobile JKN, kecuali metode perbandingan yang disebutkan.

1.4 Tujuan Penelitian

Adapun Tujuan dari penelitian ini adalah sebagai berikut:

1. Mengidentifikasi keunggulan GAN dibandingkan metode tradisional seperti SMOTE dalam menangani ketidakseimbangan data teks.
2. Mengembangkan arsitektur GAN yang mampu meningkatkan kinerja model LSTM pada analisis kepuasan pengguna aplikasi mobile JKN.
3. Merancang kerangka kerja pemanfaatan GAN yang sesuai untuk analisis kepuasan pengguna aplikasi mobile JKN.

1.5 Manfaat Penelitian

Adapun manfaat dari penelitian ini adalah sebagai berikut:

1. Memberikan Arsitektur yang tepat dalam pemanfaatan GAN untuk analisis kepuasan pengguna aplikasi mobile JKN.
2. Menghasilkan arsitektur GAN yang dikembangkan khusus untuk menghasilkan data teks sekuensial.
3. Menjadi landasan bagi penelitian selanjutnya dalam pengembangan dan pemanfaatan GAN untuk analisis data teks

1.6 Landasan Teori

1.6.1. Web scraping

Web scraping merupakan metode pengumpulan data dari situs web secara otomatis dengan bantuan program atau script yang dapat mengakses dan mengunduh konten menjadi dataset terstruktur. Proses ini umumnya terdiri dari dua tahap, yaitu *indexing* untuk menentukan target data yang akan diambil dan *fetching* yang bertugas mengekstrak informasi dari halaman web (Foerderer, 2023)

Metode ini banyak dimanfaatkan dalam berbagai bidang penelitian maupun industri karena mampu menyediakan data dalam jumlah besar dan beragam, seperti ulasan pengguna, harga produk, maupun konten media sosial. Meskipun demikian, *web scraping* memiliki tantangan, di antaranya volatilitas konten yang mudah berubah, personalisasi data yang menyebabkan perbedaan tampilan antarpengguna, serta keterbatasan akses terhadap data yang tidak terindeks secara langsung (Foerderer, 2023). Salah satu contoh penerapannya ditunjukkan oleh Aldabbas et al. (2020) yang melakukan scraping terhadap lebih dari 500 ribu ulasan aplikasi Google Play dari 14 kategori, kemudian menganalisisnya menggunakan algoritma pembelajaran mesin untuk mengklasifikasikan sentimen pengguna. Dengan demikian, *web scraping* berperan penting dalam mendukung penelitian dan analisis data berskala besar, meskipun harus tetap memperhatikan aspek teknis, etis, dan legalitasnya.

1.6.2. Kecerdasan buatan

Kecerdasan buatan (*Artificial Intelligence/AI*) didefinisikan sebagai kemampuan komputer digital atau robot yang dikendalikan komputer untuk melakukan tugas-tugas yang biasanya membutuhkan kecerdasan manusia, seperti pengambilan keputusan, pengenalan pola, dan pemecahan masalah (Mukhamediev et al., 2022) AI mencakup berbagai cabang ilmu, termasuk *machine learning* (ML), *natural language processing* (NLP), pengenalan suara, visi komputer, robotika, perencanaan, serta sistem pakar. Salah satu inti utama AI adalah ML, yaitu kumpulan algoritma yang memungkinkan sistem untuk belajar dari data dan melakukan prediksi, klasifikasi, atau analisis berdasarkan pola yang ditemukan. Seiring perkembangannya, muncul *deep learning* (DL) yang memanfaatkan jaringan saraf dengan banyak lapisan tersembunyi untuk menyelesaikan permasalahan kompleks, khususnya dalam pengolahan citra, teks, maupun suara. AI dapat dibedakan menjadi *weak AI* yang hanya berfokus pada penyelesaian tugas spesifik, serta *strong AI* atau ertujuan meniru kemampuan intelektual manusia secara raktiknya, penerapan AI lebih banyak menggunakan *weak AI* tif dalam berbagai aplikasi, seperti kesehatan, transportasi, perbankan, hingga pendidikan (Russell & Norvig, 2022). prospek yang menjanjikan, adopsi AI secara luas masih

menghadapi berbagai tantangan baik dari sisi teknis, sosial, maupun etika. Namun, AI tetap dipandang sebagai salah satu teknologi kunci dalam mendukung transformasi digital dan inovasi di berbagai bidang kehidupan (Mukhamediev et al., 2022)

1.6.3. Pembelajaran mendalam

Pembelajaran mendalam (deep learning) merupakan salah satu pendekatan utama dalam kecerdasan buatan yang memanfaatkan jaringan saraf buatan dengan banyak lapisan tersembunyi untuk mempelajari representasi data secara hierarkis. Setiap lapisan bertanggung jawab mengekstraksi fitur pada tingkat abstraksi yang semakin tinggi sehingga memungkinkan model memahami pola kompleks dari data mentah seperti teks, citra, maupun suara. Secara umum, pembelajaran mendalam mencakup tiga kategori model, yakni *generative models* seperti Deep Belief Network dan Auto-Encoders, *discriminative models* seperti Convolutional Neural Network, serta *hybrid models* seperti Deep Neural Network (Chahal & Gulia, 2019). Dalam konteks pengembangan model generatif modern seperti Generative Adversarial Networks (GAN), digunakan sejumlah parameter penting seperti *latent dimension* yang merepresentasikan ruang laten penghasil variasi data sintetis berupa *random noise* (Goodfellow et al., 2014), serta *embedding dimension* yang digunakan untuk memetakan elemen data—khususnya teks—ke dalam representasi vektor kontinu berdimensi tetap (Mikolov et al., 2013). Parameter lain seperti *hidden dimension* berfungsi menentukan jumlah unit tersembunyi dalam jaringan saraf, yang memengaruhi kapasitas model dalam mempelajari pola jangka panjang maupun dependensi sekuens (Hochreiter & Schmidhuber, 1997).

Selama proses pelatihan, parameter seperti *batch size* dan *epochs* berperan penting dalam mengatur dinamika pembaruan bobot model. *Batch size* mengontrol jumlah sampel yang diproses sebelum pembaruan parameter, yang dapat memengaruhi stabilitas gradien serta kemampuan generalisasi model (Keskar et al., 2017), sedangkan *epochs* menentukan berapa kali keseluruhan dataset dipelajari oleh model secara iteratif. Untuk mengoptimalkan parameter jaringan, algoritma Adam digunakan secara luas karena keunggulannya dalam stabilitas dan kecepatan konvergensi melalui estimasi momen pertama dan kedua yang adaptif (Kingma & Ba, 2015). Pada arsitektur GAN, laju pembelajaran (*learning rate*) antara generator dan discriminator sering dibedakan untuk menjaga stabilitas pelatihan, termasuk penggunaan konfigurasi *betas* seperti $\beta_1 = 0.5$ yang terbukti lebih efektif dalam mengurangi gejala osilasi (*oscillation*) selama proses *adversarial training* (Radford et

berbasis sekuens seperti LSTM, CNN berbasis teks, maupun
a, parameter seperti *input size*, *sequence length*, dan *hidden*
ang sangat mendasar. *Input size* menyatakan jumlah fitur atau
yang diterima model (Kim, 2014), sedangkan *sequence length*

digunakan untuk memastikan seluruh input memiliki panjang sekuens yang seragam melalui teknik *padding* atau *truncation* (Bahdanau et al., 2014). Sementara itu, *hidden size* menentukan kapasitas memori internal jaringan untuk menangkap hubungan temporal maupun semantik dalam data (Hochreiter & Schmidhuber, 1997). Selama proses pelatihan model sekuens, mekanisme seperti *learning rate decay* sering diterapkan untuk menurunkan nilai *learning rate* secara bertahap, misalnya melalui metode Exponential Decay yang meningkatkan stabilitas optimasi pada tahap pelatihan lanjutan (TensorFlow, 2023). Pada tugas klasifikasi, fungsi kerugian *binary crossentropy* digunakan untuk mengukur kesalahan antara prediksi model dan label target, khususnya dalam konteks klasifikasi biner (Goodfellow et al., 2016), sedangkan metrik seperti *validation accuracy* berfungsi mengevaluasi performa model pada data validasi. Untuk mengatasi ketidakseimbangan kelas pada dataset, teknik *class weighting* diterapkan dengan memberikan bobot lebih besar pada kelas minoritas agar kesalahan prediksi pada kelas tersebut berdampak lebih signifikan dalam proses pembaruan bobot, sehingga meningkatkan sensitivitas model terhadap kelas yang jarang muncul.

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

Long Short-Term Memory (LSTM) merupakan pengembangan dari *Recurrent Neural Networks* (RNN) yang dirancang oleh Hochreiter dan Schmidhuber (1997) untuk mengatasi masalah *vanishing gradient* pada RNN konvensional. Melalui arsitektur *memory cell* dan *gating mechanism*, LSTM mampu mempertahankan serta memanipulasi informasi dalam rentang waktu panjang tanpa kehilangan konteks yang relevan. Menurut Lipton et al. (2015), kemampuan LSTM dalam memahami urutan data sangat bergantung pada tahap tokenisasi dan embedding yang dilakukan sebelum data teks dimasukkan ke dalam jaringan. Tokenisasi berfungsi untuk memecah teks menjadi unit-unit kecil (kata atau sub-kata), yang kemudian direpresentasikan dalam bentuk numerik. Embedding memetakan token tersebut ke dalam ruang vektor berdimensi tetap, misalnya menggunakan *Word2Vec*, sehingga setiap kata direpresentasikan berdasarkan kemiripan semantik dan kontekstual antar kata.

Hasil embedding ini membentuk urutan vektor $(x_1, x_2, \dots, x_t)$ yang menjadi masukan bagi LSTM. Setiap vektor kata (x_t) merepresentasikan satu token dalam konteks kalimat, sehingga jaringan dapat mempelajari hubungan antar kata berdasarkan urutan kemunculannya. Proses ini penting karena LSTM memproses data secara sekuensial, menjaga informasi dari token sebelumnya untuk token selanjutnya — inilah yang menjadikannya efektif untuk *ning* seperti analisis sentimen atau klasifikasi teks . Secara ri dari tiga gerbang utama — input gate, forget gate, dan output gsi untuk mengontrol aliran informasi: *Input gate* menentukan ormasi baru yang akan disimpan, *Forget gate* menentukan

informasi lama yang akan dilupakan, dan *Output gate* menentukan informasi mana yang akan diteruskan ke lapisan berikutnya. Secara matematis, mekanisme internal LSTM pada setiap langkah waktu *t* dapat dijelaskan dengan persamaan (1) berikut:

$$\begin{aligned} i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ f_t &= \sigma\big(W_f \cdot [h_{t-1}, x_t] + b_f\big) \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ \tilde{C}_t &= \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \\ C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\ h_t &= o_t * \tanh(C_t) \end{aligned} \quad (1)$$

Keterangan:

x_t	=	Vektor embedding kata pada waktu ke-*t*, dihasilkan dari Word2Vec.
h_{t-1}	=	*Hidden state* pada langkah waktu sebelumnya.
h_t	=	*Hidden state* pada waktu ke-*t* sebagai keluaran utama LSTM.
C_{t-1}	=	*Cell state* pada langkah waktu sebelumnya.
C_t	=	*Cell state* pada waktu ke-*t* yang menyimpan informasi jangka panjang.
$\tilde{C}_t$	=	*Candidate cell state* yang dihasilkan dari fungsi tanh.
i_t	=	*Input gate* yang menentukan seberapa banyak informasi baru disimpan.
f_t	=	*Forget gate yang menentukan informasi mana dari cell state sebelumnya yang dilupakan.*
o_t	=	*Output gate* yang mengontrol keluaran dari LSTM.
W_i, W_f, W_o, W_C	=	Matriks bobot untuk masing-masing gerbang LSTM.
b_i, b_f, b_o, b_C	=	Vektor bias untuk masing-masing gerbang LSTM.
$\sigma(\cdot)$	=	Fungsi aktivasi sigmoid.
$\tanh(\cdot)$	=	Fungsi aktivasi tanh.
$[\ h_{t-1}, x_t\ ]$	=	Penggabungan (concatenation) antara *hidden state* sebelumnya dan input saat ini.

Dalam penelitian ini, proses *tokenisasi* dan *embedding* dilakukan terlebih dahulu menggunakan model Word2Vec, yang kemudian hasilnya dimasukkan ke dalam arsitektur LSTM dengan satu *embedding layer* berukuran 100 dimensi, satu *LSTM layer* berisi 128. Hasil keluaran dari LSTM diteruskan ke *dense layer* dengan aktivasi *softmax* untuk menghasilkan prediksi kelas performa. Dengan kombinasi proses tokenisasi, embedding, dan arsitektur LSTM yang dirancang secara khusus, model ini mampu memahami hubungan semantik antar kata serta konteks jangka erbahasa Indonesia secara efektif.

1.6.5. Word2vec

Word2Vec merupakan salah satu metode representasi teks berbasis *neural network* yang mengubah kata menjadi vektor numerik berdimensi tetap sehingga dapat diproses oleh algoritma pembelajaran mesin. Berbeda dengan metode tradisional seperti TF-IDF yang hanya mempertimbangkan frekuensi kemunculan kata, Word2Vec mampu menangkap hubungan semantik dan sintaktik antar kata melalui proses pelatihan (*training*) jaringan saraf (Mikolov et al., 2013). Word2Vec memiliki dua arsitektur utama, yaitu *Continuous Bag of Words* (CBOW) dan Skip-gram. Model CBOW memprediksi sebuah kata berdasarkan konteks di sekitarnya. Model Skip-gram melakukan hal sebaliknya, yaitu memprediksi kata konteks berdasarkan kata target.

Kedua pendekatan ini memungkinkan model untuk memahami hubungan makna antar kata, seperti sinonim, analogi, dan asosiasi semantik lainnya (Ahmad Hilman Dani et al., 2024). Misalnya, model Word2Vec dapat memahami bahwa hubungan antara *"raja"* dan *"ratu"* mirip dengan hubungan antara *"pria"* dan *"wanita"*. Kelebihan utama Word2Vec dibandingkan metode berbasis frekuensi seperti TF-IDF adalah menangkap makna semantik antar kata karena mempertimbangkan konteks, efisien secara komputasi, karena pelatihannya menggunakan *shallow neural network*. Dapat digunakan ulang (pre-trained) untuk berbagai tugas NLP. Memberikan representasi vektor kontinu, yang lebih sesuai untuk model deep learning seperti LSTM. Secara matematis, proses pembelajaran Word2Vec menggunakan fungsi objektif untuk memaksimalkan probabilitas kemunculan kata konteks w_c yang berdekatan dengan kata target w_t, yaitu:

$$\max \prod_{t=1}^{T} \prod_{-m \le j \le m, j \neq 0} P\left(w_{t+j} \mid w_t\right) \tag{2}$$

Keterangan:

T	=	Jumlah total kata dalam corpus pelatihan.
m	=	Ukuran jendela konteks (*window size*).
w_t	=	Kata target (kata pusat) pada posisi ke-*t*.
w_{t+j}	=	Kata konteks yang berada pada posisi relatif *j* dari kata target.
$P\left(w_{t+j} \mid w_t\right)$	=	Probabilitas kata konteks muncul berdasarkan kata target menurut model Word2Vec.

aan (2) m adalah ukuran jendela konteks. Pendekatan ini
t belajar distribusi makna kata berdasarkan kemunculannya
tian oleh Ahmad Hilman Dani dkk. (2024) juga menegaskan
kerja lebih baik dalam merepresentasikan hubungan semantik
an TF-IDF, karena memperhatikan konteks linguistik dalam

pelatihan model. Meski pada beberapa kasus TF-IDF dapat menghasilkan akurasi lebih tinggi tergantung pada dataset, Word2Vec tetap unggul dalam menghasilkan representasi kata yang lebih bermakna dan mendalam untuk tugas-tugas seperti *sentiment analysis*, *text classification*, dan *language modeling*.

1.6.6. Knowledge Distillation

Knowledge Distillation (KD) merupakan suatu metode kompresi model yang bertujuan mentransfer pengetahuan dari *teacher model* yang berukuran besar dan kaya parameter menuju *student model* yang lebih ringan dan efisien. KD memungkinkan model kecil memperoleh performa mendekati model besar melalui proses pembelajaran berbasis supervisi dari representasi yang dihasilkan teacher. Cooper et al. (2025) menjelaskan bahwa praktik KD pada umumnya dilakukan dengan dua cara, yaitu (*1*) meniru keluaran logits teacher melalui *vanilla knowledge distillation* (VKD) dan (*2*) meniru representasi fitur intermediate melalui *feature knowledge distillation* (FKD).

Menurut Cooper et al. (2025), logit-based losses memiliki keterbatasan karena bekerja pada ruang dimensi rendah sehingga tidak mampu mentransfer informasi representasional yang kaya dari teacher secara optimal. Sebaliknya, *feature-based losses* beroperasi pada ruang dimensi tinggi dan dinilai lebih efektif dalam menangkap struktur semantik dan informasi laten yang penting bagi proses pembelajaran student. Oleh karena itu, studi tersebut mengusulkan kerangka FKD baru yang sepenuhnya menghilangkan *logit losses* pada pelatihan *student backbone*, sehingga transfer pengetahuan dilakukan hanya melalui kesesuaian fitur antar lapisan Selain itu, Cooper et al. (2025) memperkenalkan knowledge quality metric (Q) untuk menentukan lapisan teacher mana yang mengandung kualitas pengetahuan terbaik untuk didistilasi. Metrik ini dibangun dari tiga karakteristik representasi laten, yaitu: separation (kemampuan memisahkan kelas), information (kekayaan informasi intrinsik), dan efficiency (efisiensi ukuran representasi). Lapisan dengan nilai Q tertinggi dipandang paling efektif untuk ditransfer kepada model student. Secara keseluruhan, Knowledge Distillation adalah pendekatan pembelajaran yang tidak hanya menyalin keluaran model besar, tetapi memindahkan struktur representasi internalnya secara terarah agar model ringan dapat memperoleh kemampuan serupa. Pendekatan modern seperti FKD berbasis *knowledge quality* mencerminkan evolusi KD dari metode imitasi logits menjadi strategi transfer pengetahuan yang lebih dalam dan berbasis geometri representasi.

1.6.7. NLP (*Natural Language Processing*)

Pemrosesan Bahasa Alami atau *Natural Language Processing (NLP)* merupakan salah satu cabang utama dari kecerdasan buatan (*Artificial Intelligence*) yang berfokus pada kemampuan komputer untuk memahami, memproses, dan menghasilkan bahasa manusia secara alami. Tujuan utamanya adalah memungkinkan mesin berinteraksi dengan manusia menggunakan bahasa sehari-hari, baik dalam bentuk teks maupun suara.

Menurut Simmons (1970), penelitian awal dalam bidang NLP berangkat dari pengembangan sistem *question-answering* yang berupaya memproses input berupa kalimat bahasa alami untuk menghasilkan jawaban yang relevan. Sistem-sistem generasi pertama seperti *ELIZA* (Weizenbaum, 1966) merupakan tonggak penting dalam sejarah NLP karena menunjukkan bahwa mesin dapat memahami struktur sintaksis dasar dan merespons secara semantik terhadap input manusia. Pada masa itu, paradigma utama NLP berkembang dari pendekatan berbasis aturan linguistik (*rule-based systems*) menuju pemrosesan yang melibatkan representasi formal dari makna, sintaksis, dan logika bahasa. Lebih lanjut, Simmons (1970) menjelaskan bahwa struktur bahasa dapat direpresentasikan melalui tiga komponen utama: sintaksis, semantik, dan fonologi. Sintaksis bertanggung jawab atas struktur kalimat, semantik menafsirkan makna dari struktur tersebut, dan fonologi berkaitan dengan representasi bunyi. Integrasi ketiga komponen ini memungkinkan pembentukan *deep structure* — yaitu representasi formal dari makna kalimat yang kemudian diubah menjadi *surface structure* untuk komunikasi manusia.

Dalam perkembangannya, NLP tidak hanya berfokus pada pemahaman sintaksis, tetapi juga pada konteks, relasi antar-kata, serta makna implisit yang muncul dalam percakapan. Pendekatan modern kemudian berkembang menuju paradigma *statistical NLP* dan *deep learning-based NLP*, di mana representasi teks diekstraksi melalui model berbasis vektor seperti *Word2Vec*, *GloVe*, hingga *transformer architectures* seperti *BERT* dan *GPT*. Pendekatan ini memungkinkan sistem memahami konteks semantik yang kompleks dengan melakukan pelatihan terhadap jumlah data yang besar. Konsep penting dalam NLP modern adalah kemampuan untuk memetakan teks menjadi representasi numerik atau *embedding*, yang memungkinkan algoritma pembelajaran mesin untuk melakukan analisis semantik, klasifikasi, dan prediksi konteks. Proses ini menjadi inti dalam berbagai aplikasi NLP seperti *text classification*, *sentiment analysis*, *machine translation*, *question answering*, dan *chatbot systems*. Dengan demikian, NLP tidak hanya

s untuk menerjemahkan teks, tetapi juga sebuah pendekatan memahami bahasa manusia secara konseptual dan pragmatis. kar pada teori linguistik klasik sebagaimana dijelaskan oleh api telah berevolusi menuju metode berbasis jaringan saraf ıenangkap makna secara kontekstual dan dinamis.

1.6.8. GAN (*Generative Adversarial Network*)

Generative Adversarial Network (GAN) diperkenalkan oleh Goodfellow et al. (2014) sebagai suatu kerangka baru untuk melatih model generatif melalui proses *adversarial*. Dalam kerangka ini terdapat dua model utama yang dilatih secara bersamaan, yaitu generator (G) dan discriminator (D). Generator berperan menghasilkan data baru yang menyerupai data asli, sedangkan discriminator bertugas membedakan antara data asli dari dataset pelatihan dengan data yang dihasilkan oleh generator. Tujuan pelatihan GAN adalah mencapai kondisi keseimbangan (*equilibrium*) di mana generator mampu menghasilkan data yang tidak dapat dibedakan dari data asli oleh discriminator. Proses ini diformulasikan sebagai permainan dua pemain (*two-player minimax game*) dengan fungsi nilai sebagai berikut:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \quad (3)$$

Keterangan:

G	=	*Generator*, yaitu model yang bertugas menghasilkan data sintetis menyerupai data asli.
D	=	*Discriminator*, yaitu model yang mengevaluasi apakah suatu sampel berasal dari data asli atau hasil generator.
$V(D,G)$	=	Fungsi objektif GAN pada skema permainan *two-player minimax*.
x	=	Sampel data asli yang diambil dari distribusi nyata.
z	=	Vektor noise acak yang dijadikan input Generator.
$p_{\text{data}}(x)$	=	Distribusi probabilitas data asli.
$p_z(z)$	=	Distribusi noise yang menjadi input untuk Generator.
$D(x)$	=	Probabilitas bahwa sampel $x$diklasifikasikan sebagai data asli oleh Discriminator.
$G(z)$	=	Data sintetis yang dihasilkan oleh Generator dari noise z.

Dalam persamaan (3), $p_{data}(x)$ merepresentasikan distribusi data asli, sedangkan $p_z(z)$merupakan distribusi acak (umumnya berdistribusi Gaussian) yang digunakan oleh *generator* untuk menghasilkan data sintetis yang direpresentasikan sebagai $G(z)$. Sementara itu, $D(x)$ adalah fungsi *discriminator* yang memberikan probabilitas apakah suatu data $x$berasal dari data asli atau data hasil generasi (*palsu*). Untuk memberikan ukuran performa atau "penghargaan" (*reward*) terhadap kedua model, digunakan dua komponen utama, yaitu $\log D(x)$ dan $\log(1 - D(G(z)))$.

dari fungsi nilai, yaitu $\mathbb{E}_{x\sim p_{data}(x)}[\log D(x)]$, merupakan nilai logaritma probabilitas dalam mengenali data asli. Nilai ini sebagai *reward* bagi *discriminator* ketika berhasil ata asli dengan benar. Komponen kedua, yaitu $\mathbb{E}_{z\sim p_z(z)}[\log(1 -$ an perhitungan rata-rata nilai logaritma probabilitas dalam

menolak data sintetis. Komponen ini menjadi *reward* bagi *discriminator* ketika mampu mengenali dan menolak data yang dihasilkan oleh *generator*. Dengan demikian, *discriminator* berupaya memaksimalkan kedua komponen tersebut agar dapat membedakan data asli dan data sintetis secara optimal. Sebaliknya, *generator* berusaha meminimalkan fungsi nilai tersebut dengan menghasilkan data sintetis yang membuat nilai $D(G(z))$ mendekati 1, sehingga *discriminator* kesulitan membedakan antara data asli dan data hasil generasi. Proses saling berlawanan ini membentuk suatu permainan *minimax*, di mana *generator* berusaha "menipu" *discriminator*, sedangkan *discriminator* berusaha mempertahankan kemampuannya dalam mengklasifikasikan data secara akurat.

Goodfellow et al. (2014) menunjukkan bahwa pada kondisi ideal—ketika kedua model memiliki kapasitas yang cukup besar dan proses pelatihan konvergen—distribusi yang dihasilkan oleh generator (p_g) akan sama dengan distribusi data sebenarnya (p_{data}). Hal ini berarti model generatif telah berhasil meniru distribusi data asli dengan sempurna. Keunggulan utama GAN dibandingkan pendekatan generatif tradisional seperti *Restricted Boltzmann Machines (RBM)* atau *Deep Belief Networks (DBN)* adalah bahwa GAN tidak memerlukan rantai Markov (Markov Chain) maupun inferensi aproksimasi dalam proses pembelajarannya. Seluruh proses pelatihan dapat dilakukan hanya dengan algoritma backpropagation, menjadikan GAN efisien secara komputasi dan fleksibel untuk berbagai arsitektur jaringan. Meski demikian, GAN juga memiliki kelemahan, terutama dalam menjaga keseimbangan pelatihan antara generator dan discriminator. Jika salah satu model terlalu dominan, maka pelatihan dapat menjadi tidak stabil dan menghasilkan fenomena seperti *mode collapse*, di mana generator menghasilkan variasi data yang terbatas. Secara keseluruhan, GAN telah membuka paradigma baru dalam pembelajaran generatif dan menjadi dasar bagi berbagai pengembangan model lanjutan, seperti *Conditional GAN (cGAN)*, *Deep Convolutional GAN (DCGAN)*, dan *Wasserstein GAN (WGAN)*.

1.6.9. *Data Imbalance* dan metode oversampling (SMOTE & SMOTEENN)

Ketidakseimbangan data (*data imbalance*) merupakan salah satu tantangan utama dalam penerapan algoritma pembelajaran mesin, terutama pada kasus klasifikasi di mana distribusi antar kelas tidak seimbang. Kondisi ini terjadi ketika jumlah sampel pada kelas mayoritas jauh lebih banyak dibandingkan kelas minoritas, sehingga model cenderung bias terhadap kelas dengan jumlah data yang lebih besar dan mengabaikan pola yang terdapat pada kelas minoritas. Akibatnya, meskipun akurasi keseluruhan dapat terlihat tinggi, performa model terhadap kelas minoritas biasanya

susnya pada metrik seperti *recall* dan *F1-score*. Masalah ni banyak dijumpai pada berbagai bidang, seperti deteksi medis, maupun analisis opini pengguna aplikasi, di mana ru sering muncul dalam kelas minoritas (Husain et al., 2025).

Untuk mengatasi masalah tersebut, berbagai pendekatan telah dikembangkan. Secara umum, metode penanganan *data imbalance* terbagi menjadi dua kategori utama, yaitu: (1) pendekatan pada tingkat data (*data-level approach*), yang berfokus pada penyesuaian distribusi kelas sebelum proses pelatihan model; dan (2) pendekatan pada tingkat algoritma (*algorithm-level approach*), yang mengubah *loss function* (fungsi yang menghitung kesalahan prediksi kelas) yang akan digunakan selama pelatihan agar lebih sensitif terhadap kesalahan prediksi pada kelas minoritas. Pendekatan berbasis data lebih banyak digunakan karena tidak memerlukan perubahan arsitektur model dan dapat diterapkan pada berbagai algoritma pembelajaran mesin. Salah satu metode oversampling paling populer pada kategori ini adalah SMOTE (*Synthetic Minority Oversampling Technique*). SMOTE dikembangkan untuk menambah jumlah sampel pada kelas minoritas dengan cara menghasilkan data sintetis baru, bukan sekadar menduplikasi data yang sudah ada. Metode ini bekerja dengan memilih setiap sampel minoritas, kemudian mencari beberapa tetangga terdekat menggunakan algoritma *k-nearest neighbors (KNN)*. Setelah itu, data sintetis dibuat dengan melakukan interpolasi antara sampel minoritas dan tetangganya menggunakan persamaan:

$$x_{new} = x_i + \lambda(x_j - x_i) \tag{4}$$

Keterangan:

x_i	=	Sampel dari kelas minoritas.
x_j	=	Salah satu tetangga terdekat dari sampel minoritas.
x_{new}	=	Sampel baru hasil sintesis yang dihasilkan oleh metode SMOTE.
λ	=	Bilangan acak pada rentang $0 \leq \lambda \leq 1$.

Pada persamaan (4) x_i merupakan sampel minoritas, $x_j$adalah salah satu tetangga terdekat, dan λadalah bilangan acak antara 0 dan 1. Melalui pendekatan ini, SMOTE mampu memperluas ruang fitur kelas minoritas, meningkatkan representasi data, serta mengurangi risiko *overfitting* yang biasanya muncul akibat duplikasi data secara langsung. Namun, kelemahan utama SMOTE terletak pada potensi tumpang tindih (*overlap*) antara kelas minoritas dan mayoritas di sekitar batas keputusan (*decision boundary*), yang dapat menyebabkan penurunan akurasi klasifikasi karena terciptanya area kelas yang tidak jelas.

Untuk memperbaiki kelemahan tersebut, dikembangkan metode hibrida SMOTEENN (*SMOTE + Edited Nearest Neighbors*). Pendekatan ini menggabungkan melakukan oversampling dengan SMOTE untuk menambah las minoritas; dan kedua, membersihkan data menggunakan rest Neighbors (ENN)*. Tahap ENN berfungsi untuk menghapus un sintetis) yang salah klasifikasi oleh mayoritas tetangganya, i *noise* dan memperjelas batas antar kelas. Dengan demikian,

SMOTEENN tidak hanya menambah representasi kelas minoritas, tetapi juga memperbaiki kualitas dataset melalui proses pembersihan data yang tidak konsisten.

Hasil penelitian yang dikemukakan oleh Husain et al. (2025) menunjukkan bahwa SMOTEENN mampu menghasilkan performa model yang lebih stabil dibandingkan SMOTE murni. Metode ini memberikan peningkatan pada metrik seperti akurasi, *recall*, dan *F1-score*, serta menurunkan tingkat kesalahan prediksi terhadap kelas minoritas. Selain itu, kurva pembelajaran (*learning curve*) yang dihasilkan oleh SMOTEENN lebih seimbang, menandakan kemampuan model untuk belajar secara stabil tanpa *overfitting* yang berlebihan. Pendekatan ini terbukti efektif terutama pada dataset dengan ketidakseimbangan tinggi, karena menggabungkan keunggulan dari dua teknik sekaligus: kemampuan SMOTE dalam menghasilkan variasi data minoritas dan kemampuan ENN dalam menghapus sampel yang tidak relevan.

1.6.10. *R&D (Research and Development)*

Metode *Research and Development (R&D)* dalam konteks eksperimen komputasional merupakan pendekatan penelitian yang berorientasi pada pengembangan model simulatif berbasis komputasi untuk memahami dan memprediksi perilaku sistem kompleks. Pendekatan ini berkembang dari integrasi antara ilmu kompleksitas, eksperimen virtual, dan inteligensi komputasional, dengan tujuan membangun sistem buatan (*artificial system*) yang mampu meniru dan menguji fenomena dunia nyata melalui simulasi terkontrol. Menurut Xue et al. (2021), *computational experiment* merupakan bentuk evolusi dari *computer simulation* yang tidak hanya berfungsi untuk deskripsi dan prediksi, tetapi juga untuk *prescription* — yaitu memberikan panduan atau intervensi terhadap sistem nyata. Melalui kombinasi antara teori sistem kompleks, teknologi simulasi, dan konsep *digital twin*, eksperimen komputasional memungkinkan pelaksanaan eksperimen berulang tanpa risiko nyata terhadap lingkungan fisik atau sosial. Hal ini menjadikannya sangat relevan dalam penelitian yang melibatkan sistem besar, berisiko tinggi, atau tidak dapat diuji secara langsung, seperti sistem sosial, ekonomi, dan ekologi.

Lebih lanjut, eksperimen komputasional bekerja melalui lima tahapan utama, yaitu: (1) pemodelan masyarakat buatan (*artificial society modeling*), (2) konstruksi sistem eksperimental, (3) desain eksperimen, (4) analisis hasil eksperimen, dan (5) verifikasi model (Xue et al., 2021). Proses ini berlangsung secara berulang (*feedback loop*) untuk memastikan validitas model serta kemampuan sistem dalam merepresentasikan fenomena kompleks secara akurat. Pendekatan ini menekankan *nt system* dan *bottom-up simulation*, di mana perilaku makro antar entitas mikro yang otonom. Sementara itu, Fernandez- menekankan bahwa eksperimen komputasional memerlukan (*experimental design*) yang kuat agar hasil yang diperoleh valid uksi. Dalam penelitian berbasis *computational intelligence*,

kesalahan dalam desain eksperimen dapat menyebabkan bias dan ketidakakuratan model. Oleh karena itu, diperlukan tahapan sistematis yang mencakup: (1) pembentukan dataset yang representatif, (2) pra-pemrosesan data, (3) pembelajaran model menggunakan algoritme pembelajaran mesin (*machine learning*), dan (4) pemilihan model terbaik berdasarkan validasi statistik. Pendekatan ini memastikan bahwa eksperimen komputasional memiliki dasar metodologis yang terukur dan dapat dipertanggungjawabkan. Secara konseptual, *computational experiment* dalam paradigma R&D bertindak sebagai laboratorium buatan (*artificial laboratory*). Dengan kemampuan untuk mengontrol variabel dan mengulang percobaan tanpa batas, metode ini mendukung pengembangan teori dan aplikasi baru secara efisien dan etis. Kombinasi antara *computational experiment* dan metodologi *experimental design* seperti yang diuraikan oleh Fernandez-Lozano et al. (2016) memberikan kerangka penelitian yang tidak hanya eksploratif, tetapi juga prediktif dan preskriptif — menjadikannya sangat sesuai untuk riset pengembangan berbasis simulasi dan pembelajaran mesin.

1.6.11. Confusion matrix

Confusion Matrix adalah tabel yang digunakan untuk menilai kinerja model klasifikasi dengan cara membandingkan hasil prediksi dengan label yang sebenarnya. Dalam tabel ini, setiap baris menunjukkan kelas aktual, sedangkan kolom menunjukkan kelas yang diprediksi. Dari *Confusion Matrix*, bisa menghitung metrik evaluasi seperti *akurasi*, *presisi*, *recall*, dan *F1-score* yang membantu memahami sejauh mana model dapat mengenali data dengan benar (Markoulidakis & Markoulidakis, 2024).

Evaluasi kinerja model klasifikasi dilakukan menggunakan metrik akurasi, presisi, dan recall yang diturunkan dari *Actual Label Probabilistic Confusion Matrix* (ALP-CM) sebagaimana dijelaskan oleh Markoulidakis & Markoulidakis (2024). Akurasi menggambarkan sejauh mana prediksi model sesuai dengan label aktual pada keseluruhan data. Dalam konteks ALP-CM, akurasi dihitung dengan mempertimbangkan probabilitas maksimum dari setiap kelas, dirumuskan sebagai pada persamaan (5) N adalah jumlah total sampel, $c_{ALP}(k,k)$ elemen diagonal ALP-CM yang merepresentasikan prediksi benar berbobot probabilitas, $p_{max}(k)$ adalah rata-rata probabilitas prediksi maksimum untuk kelas ke-k, dan $n(\hat{y} = L_k)$ jumlah sampel yang diprediksi sebagai kelas L_k. Selanjutnya, presisi menunjukkan proporsi prediksi positif yang benar dibandingkan seluruh prediksi positif. Untuk kasus multikelas, presisi dirumuskan pada persamaan (6). Sedangkan untuk kasus biner, presisi ... rasio *True Positives* terhadap jumlah *True Positives* dan *False* ...at pada persaman (7). Adapun recall mengukur kemampuan ...ali seluruh sampel positif secara benar. Untuk kasus multikelas, ...da persamaan (8). dan untuk kasus biner, recall dinyatakan ...*ositives* terhadap jumlah *True Positives* dan *False Negatives*, ...samaan (9)

$$\text{Acc}(\text{CM}_{ALP}) = \frac{1}{N}\sum_{k=1}^{M} c_{ALP}(k,k) = \frac{\sum_{k=1}^{M} p_{max}(k)\, n(\hat{y} = L_k)}{N} \quad (5)$$

$$\text{Prec}(\text{CM}_{ALP}) = \frac{1}{M}\sum_{k=1}^{M} \frac{c_{ALP}(k,k)}{n(\hat{y} = L_k)} = \frac{1}{M}\sum_{k=1}^{M} p_{max}(k) \quad (6)$$

$$\text{Prec}(\text{CM}_{ALP}) = \frac{TP}{TP + FP} = \frac{c_{ALP}(P,P)}{n(\hat{y} = P)} = p_{max}(P) \quad (7)$$

$$\text{Rec}(\text{CM}_{ALP}) = \frac{1}{M}\sum_{k=1}^{M} \frac{c_{ALP}(k,k)}{n(y = L_k)} = \frac{1}{M}\sum_{k=1}^{M} \frac{n(\hat{y} = L_k)}{n(y = L_k)}\; p_{max}(k) \quad (8)$$

$$\text{Rec}(\text{CM}_{ALP}) = \frac{TP}{TP + FN} = \frac{c_{ALP}(P,P)}{n(y = P)} = \frac{n(\hat{y} = P)}{n(y = P)}\, p_{max}(P) \quad (9)$$

Keterangan:

N	=	Jumlah total sampel.
M	=	Jumlah kelas.
L_k	=	Label untuk kelas ke-k.
$c_{\text{alp}}(k,k)$	=	Elemen diagonal dari ALP-CM yang merepresentasikan prediksi benar berbobot probabilitas.
$p_{\text{max}}(k)$	=	Probabilitas maksimum untuk kelas ke-k.
$n(\hat{y} = L_k)$	=	Jumlah sampel yang diprediksi sebagai kelas L_k.
$n(\text{y} = L_k)$	=	Jumlah sampel sebenarnya pada kelas L_k.
TP (True Positives)	=	Jumlah prediksi benar untuk kelas positif.
FP (False Positives)	=	Jumlah prediksi salah ketika model memprediksi positif.
FN (False Negatives)	=	Jumlah prediksi salah ketika model gagal memprediksi positif.
P	=	Kelas positif pada kasus klasifikasi biner.

Melalui pendekatan probabilistik ini, setiap metrik tidak hanya mempertimbangkan tetapi juga tingkat keyakinan model terhadap prediksinya. koulidakis (2024) menyatakan bahwa ketika kondisi teoretis ai akurasi, *presisi*, dan *recall* yang dihitung dari ALP-CM akan diperoleh dari *Confusion Matrix* reguler.

Selain bentuk biasa, penelitian terbaru juga memperkenalkan *Probabilistic Confusion Matrix* (PCM). Berbeda dengan versi reguler yang hanya melihat label prediksi akhir, PCM juga mempertimbangkan nilai probabilitas dari prediksi model. Salah satu bentuknya adalah *Actual Label Probabilistic Confusion Matrix* (ALP-CM), yang menggunakan *probabilitas* ini untuk memperkirakan label aktual. Keunggulannya, metode ini bisa tetap digunakan walaupun data uji tidak memiliki label lengkap, sehingga memberikan gambaran yang lebih fleksibel tentang kinerja model (Markoulidakis & Markoulidakis, 2024).

BAB II
METODOLOGI PENELITIAN

2.1 Pendekatan dan Jenis Penelitian

Metode Penelitian dan Pengembangan (R&D) adalah metodologi yang akan digunakan dalam penelitian ini. Dan lebih berfokus pada sebuah model eksperimen yang dikenal dengan nama eksperimen komputasi (*Computational Experiment*). Eksperimen komputasi menggunakan sistem simulasi komputer yang dikembangkan secara interaktif yang dilakukan secara berulang untuk mengukur berbagai parameter masukan yang menghasilkan keluaran tertentu. Perubahan lingkungan percobaan sendiri dipantau dalam proses pengulangan (*Repeatability*) yang menjadi satu dari ciri metode ini. Perbedaan utama dengan simulasi komputer biasa adalah pada pendekatan berbasis data, mekanisme yang muncul, serta interpretasi multi-dunia dan teori panduan yang digunakan (Xue et al., 2022).

Penggunaan metode eksperimen komputasi juga telah diterapkan dalam penelitian-penelitian sebelumnya, salah satunya oleh Fernandez-Lozano et al. (2016) dalam jurnalnya berjudul *"A methodology for the design of experiments in computational intelligence with multiple regression models"*. Penelitian tersebut menerapkan prinsip-prinsip eksperimen komputasi melalui tahapan-tahapan sistematis, mulai dari pemilihan dataset, pra-pemrosesan, pelatihan model dengan validasi silang ganda, hingga pemilihan model terbaik berdasarkan analisis statistik. Dengan demikian, pemilihan metode penelitian dan pengembangan berbasis eksperimen komputasi dianggap tepat karena dapat menghasilkan model yang teruji secara objektif dan dapat direplikasi dalam konteks penelitian tertentu (Fernandez-Lozano et al., 2016; Xue et al., 2022). Selain memanfaatkan model untuk mensimulasikan fenomena dunia nyata, penelitian ini berfokus pada pengembangan model itu sendiri. Dengan menyesuaikan kerangka kerja dan arsitektur model, diharapkan dapat ditemukan model atau arsitektur yang mampu menghasilkan prediksi sesuai dengan tujuan penelitian.

2.2 Sumber Data

2.2.1. Dataset Ulasan Aplikasi Mobile JKN

Dataset yang digunakan dalam penelitian ini diperoleh melalui proses *web scraping* dari laman aplikasi Mobile JKN yang tersedia di Google Play Store (BPJS Kesehatan, ...umpulkan berupa ulasan pengguna yang mencerminkan tingkat ... penggunaan aplikasi. Dataset tersebut kemudian disimpan ...n terdiri atas beberapa atribut utama.

Tabel 1. Atribut dan Fitur Dataset Aplikasi Mobile JKN

No	Atribut	Tipe Data	Keterangan
1.	rating	Numerik /Kategorikal	Dikategorikan sebagai positif (puas) jika bintang ≥ tiga, dan negatif (tidak puas) jika bintang < tiga.
2.	ulasan	Teks	Teks ulasan berbahasa Indonesia yang ditulis oleh pengguna.
3.	tanggal	Tanggal/Waktu	Waktu pengguna memberikan ulasan di Play Store.
4.	nama_pengguna	Teks	nama pengguna yang diambil dari akun Google.

Dataset ini kemudian dibagi menjadi dua versi pelabelan, yaitu:

a. Pelabelan pertama (berdasarkan rating bintang), digunakan karena rating merupakan satu-satunya indikator yang dapat diekstraksi secara otomatis dari Google Play Store tanpa proses anotasi manual. Pendekatan ini dipilih sebagai strategi awal untuk memperoleh label sentimen secara efisien dalam konteks web scraping berskala besar. Meskipun pemetaan rating ≥ 3 sebagai sentimen positif tidak sepenuhnya merepresentasikan polaritas teks secara akademik, pendekatan ini tetap dimanfaatkan sebagai dasar pelatihan awal untuk model LSTM dan arsitektur GAN.
b. Pelabelan kedua (berdasarkan hasil transformer), diterapkan untuk meningkatkan validitas akademik dataset. Model transformer digunakan untuk melakukan pembacaan ulang teks ulasan secara semantik, sehingga menghasilkan label sentimen yang lebih akurat, konsisten, dan sesuai dengan konteks linguistik. Kedua versi dataset tersebut digunakan secara bersamaan untuk memastikan proses analisis dan pelatihan GAN tidak bergantung pada satu pendekatan pelabelan tertentu, serta untuk mengevaluasi ketahanan model terhadap variasi kualitas label.

2.2.2. Dataset Ulasan Multi-Aplikasi

a ini diperoleh melalui web scraping dari 902 aplikasi di Google
akan terutama untuk melatih model GAN serta melatih ulang
r embedding yang dihasilkan lebih kaya dan relevan. Struktur
dataset pertama.

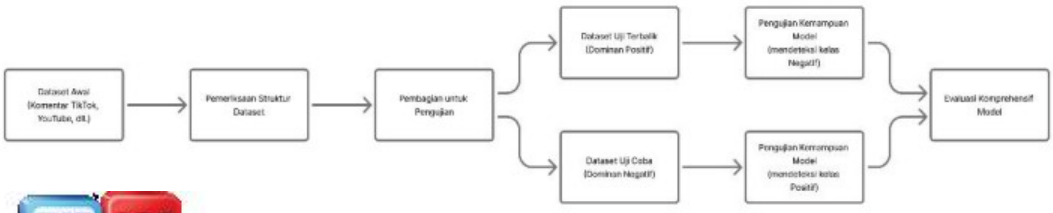
Sama seperti dataset pertama, dataset ini juga melalui dua proses pelabelan, yaitu pelabelan awal berbasis rating bintang dan pelabelan ulang menggunakan model Transformer. Penerapan dua metode pelabelan ini dilakukan untuk menjaga konsistensi proses anotasi di seluruh korpus. Penggunaan dataset kedua memberikan cakupan linguistik yang lebih luas sehingga model Word2Vec dapat mempelajari representasi teks yang lebih beragam. Hal ini sekaligus meningkatkan kemampuan model GAN dalam menghasilkan data sintetis yang lebih natural dan stabil karena dilatih pada embedding yang lebih kuat dan tidak terbatas pada satu domain aplikasi saja.

2.2.3. Dataset ulasan Uji Coba dan Uji Terbalik

Kedua dataset ini berasal dari dataset awal yang sama dan memiliki struktur identik. Dataset ini dikumpulkan dari komentar pengguna pada beberapa platform, seperti TikTok, YouTube, dan platform lainnya. Pembagian dataset dilakukan untuk tujuan pengujian yang berbeda, sebagai berikut:

a. Dataset Uji Coba. Dataset ini dibentuk dengan cara mengurangi sebagian besar data positif, sehingga menyisakan dominasi data negatif. Langkah ini dilakukan untuk menguji kemampuan model dalam mendeteksi atau menghasilkan data positif meskipun jumlahnya terbatas.
b. Dataset Uji Terbalik, Dataset ini dibentuk dengan cara mengurangi sebagian besar data negatif, sehingga mayoritas data yang tersisa merupakan data positif. Langkah ini dilakukan untuk menguji kemampuan model dalam mendeteksi atau menghasilkan data negatif meskipun jumlahnya terbatas.

Dengan strategi pengujian dua arah ini, performa arsitektur dapat diuji secara menyeluruh pada masing-masing kelas, sehingga hasil evaluasi model menjadi lebih komprehensif. Ilustrasi pembuatan dan penggunaan dari dataset ini ditunjukan pada Gambar 1 di bawah ini.



'enyusunan dan Pembuatan Dataset Uji Coba dan Uji Terbalik

2.3 Waktu dan Tempat

Penelitian ini dilakukan semenjak bulan Juli 2025 sampai dengan bulan Oktober 2025. Lokasi penelitian dilakukan di Laboratorium Rekayasa Perangkat Lunak Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Hasanuddin.

2.4 Tahapan Penelitian

Penelitian ini terbagi menjadi tahapan pra-penelitian dan tahapan penelitian, tahapan pra-penelitian biasa disebut sebagai tahapan persiapan yang mana dilakukan proses pengumpulan data. Sedangkan tahapan penelitian adalah bagian penting dari penelitian ini yang mana proses uji coba dan simulasi parameter dilakukan untuk bisa menghasilkan keluaran yang optimal.

2.5 Tahapan Pra-Penelitian (Persiapan)

Dalam tahapan ini beberapa hal yang dilakukan adalah melakukan perencanaan pelaksanaan secara konseptual, operasional, serta tahapan instrumental. Pada tahapan ini penelitian mengukuhkan tema penelitian dan prosedur penelitian yang sesuai dengan metode yang akan dilakukan. Pengumpulan jurnal pendukung penelitian serta proses pengumpulan data menggunakan metode *web scraping* menjadi bagian penting dalam tahapan pra penelitian ini.

2.6 Penelitian

Dalam tahapan ini dilakukan beberapa tahapan sebagai berikut:

2.6.1. Pra-Pemrosesan data teks

Pada tahapan ini dilakukan proses pengelolaan data teks menggunakan berbagai metode, dimulai dari pembersihan tanda baca, mengatur huruf kapital menjadi huruf kecil, serta penyesuaian teks dengan menggunakan library *stopwords*.

2.6.2. Proses simulasi berulang

Pada tahapan ini peneliti melakukan tiga langkah eksperimen komputasi secara berulang dengan berbagai parameter dan penyesuaian lingkungan. Adapun langkah yang dilakukan secara berulang adalah sebagai berikut:

edding, Pada tahapan ini dilakukan proses tokenisasi data ggunakan metode tfidf dan juga metode Word2vec Bahasa uk mengelola data teks menjadi berbentuk numerik. Serta pada a dilakukan pemiligan jenis model Word2vec seperti apa yang roses pengembangan GAN dan analisis kepuasan.

b. Simulasi penggunaan dan pengembangan arsitektur GAN, Dilakukan simulasi penggunaan arsitektur *Generative Adversarial Network* (GAN), yang selanjutnya akan disebut GAN, untuk menghasilkan data sintetis dalam mengatasi ketidakseimbangan data. Pada proses ini, arsitektur GAN akan disesuaikan dan dikembangkan, termasuk penyesuaian *hyperparameter* guna memperoleh kerangka kerja yang optimal.
c. Proses pelatihan Model LSTM dan Evaluasi, Pada tahapan ini dilakukan pengujian pada data sintetis yang dihasilkan oleh GAN yang sudah dikembangkan untuk melatih model LSTM (*Long Short-Term Memory*) yang selanjutnya akan disebut LSTM. Pada proses ini telah dilakukan penyesuaian parameter mulai dari jumlah *epoch* dan *hyperparameter* lainnya, penggunaan. Pada tahap ini pula akan dilakukan analisis pada nilai metrik pengukuran yang digunakan yakni (*Confusion Matrix*) untuk menentukan hasil penelitian.

Ketiga proses tersebut dilakukan secara berulang, namun tidak selalu harus mengikuti urutan yang sama. Penyesuaian dilakukan berdasarkan perubahan parameter dan kondisi lingkungan hingga diperoleh jawaban “iya” atas pertanyaan: “Apakah arsitektur GAN sudah lebih baik dibandingkan SMOTE dan SMOTEENN?”. Pada tahap evaluasi ini, penelitian menggunakan semua metrik *Confusion Matrix* sebagai metrik utama untuk menilai kualitas data sintetis yang dihasilkan oleh model dan melihat keunggulan GAN dibandingkan dengan SMOTE dan SMOTEEN dalam melaksananakan penyesuain data sintetis yang akan digunakan selama training untuk melaksanakan analisis kepuasan.

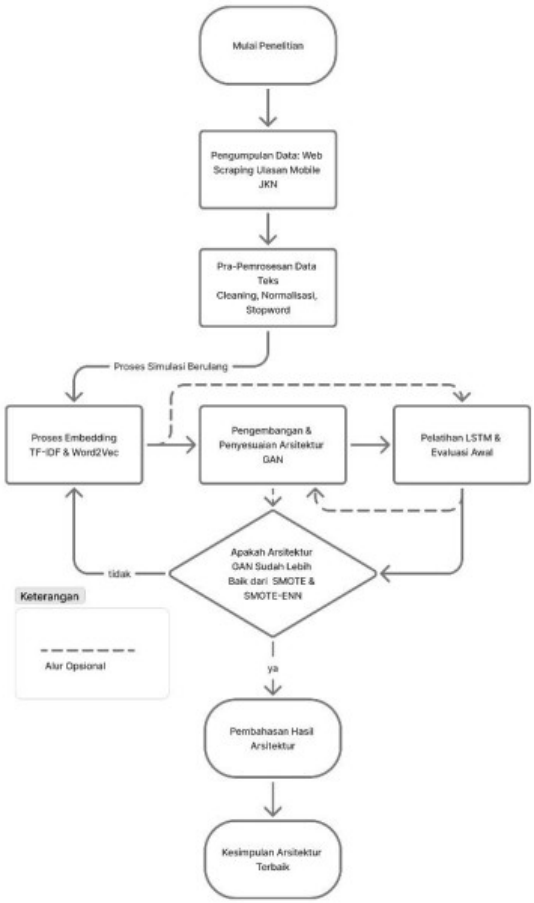
Pemilihan Seluruh metrik evaluasi *accuracy, precision, recall,* dan *F1-score* diturunkan langsung dari confusion matrix dan digunakan secara setara tanpa memberikan prioritas pada metrik tertentu. Pendekatan ini sejalan dengan literatur yang menegaskan bahwa confusion *matrix* merupakan representasi paling komprehensif dari perilaku prediksi model, karena seluruh metrik performa hanyalah fungsi turunan dari struktur kesalahan dan keberhasilan klasifikasi yang tercermin di dalamnya (Markoulidakis & Markoulidakis, 2024). Dengan demikian, penilaian kualitas model dan data sintetis dilakukan secara objektif berdasarkan keseluruhan distribusi prediksi, bukan berdasarkan satu metrik tunggal.

2.6.3. Pembahasan dan Uji Arsitektur

Pada tahap ini dilakukan pembahasan mengenai bagaimana tingkat keberhasilan dari Arsitektur yang dibuat dan dilakukan uji coba menggunakan dataset Uji Coba dan dataset Uji Terbalik.

2.7 Diagram Alur Penelitian

Diagram alur penelitian mencakup langkah-langkah pelaksanaan penelitian dari awal sampai akhir dapat dilihat pada Gambar 2.



Alur Penelitian