BAB II
KAJIAN LITERATUR

2.1 Deteksi Komentar Spam

Deteksi komentar spam merupakan proses identifikasi pesan atau tanggapan yang bersifat tidak relevan, berbentuk promosi, mengandung tautan berbahaya, atau dilakukan secara otomatis oleh *bot* untuk tujuan tertentu, seperti penyebaran iklan ilegal dan *online scam* [7]. Dalam konteks media sosial seperti YouTube, komentar spam sering kali muncul dalam bentuk promosi situs judi online, tautan ke platform judi, atau ajakan yang disamarkan dengan penggunaan huruf acak, spasi berlebih, dan simbol agar tidak terdeteksi oleh sistem filter sederhana [8]. Tujuan utama dari deteksi komentar spam adalah menjaga integritas ruang diskusi, meningkatkan pengalaman pengguna, serta melindungi pengguna dari konten berbahaya. Secara umum, komentar spam dapat dikelompokkan menjadi beberapa kategori [9], yaitu:

1. Spam promosi adalah Komentar yang berisi ajakan atau tautan ke situs komersial, seperti perjudian atau penjualan produk ilegal.
2. Phishing merupakan Komentar yang mengandung tautan palsu dengan tujuan mencuri data pribadi pengguna.
3. Spam bot adalah jenis Komentar yang dihasilkan secara otomatis menggunakan skrip atau bot, yang berulang kali menempelkan pesan sama di berbagai video.
4. Spam terselubung biasanya menggunakan istilah disamarkan, campuran bahasa, huruf acak, atau simbol tertentu agar sulit dikenali oleh sistem deteksi otomatis.

Komentar spam menimbulkan berbagai dampak negatif terhadap kualitas interaksi dan reputasi platform media sosial. Spam tidak hanya mengganggu kenyamanan pengguna, tetapi juga dapat menurunkan *engagement rate* dan kredibilitas *Content Creator* [10]. Konten spam juga kerap menurunkan efektivitas algoritma rekomendasi, karena sistem menjadi kesulitan membedakan interaksi asli dengan interaksi palsu. Selain itu, spam yang berisi ajakan ke situs judi online dapat mendorong perilaku berisiko dan aktivitas ilegal di kalangan pengguna [8]. Oleh karena itu, kebutuhan akan sistem deteksi otomatis yang mampu bekerja secara real-time dan kontekstual menjadi semakin penting.

Metode deteksi spam dapat dikelompokkan menjadi tiga pendekatan utama, yaitu:

1. *Rule-Based*

   Pendekatan ini menggunakan pola teks seperti kata kunci atau *regular expressions*

untuk mengenali komentar yang mengandung kata tertentu, misalnya “slot”, “bonus”, atau “gacor” [2]. Metode ini memiliki keunggulan dari sisi kecepatan, namun kelemahannya terletak pada ketidakmampuannya menghadapi variasi ejaan dan modifikasi karakter yang sering digunakan *spammer* untuk mengelabui sistem [6].

2. *Machine Learning*

   Model pembelajaran mesin tradisional seperti *Naïve Bayes*, *Support Vector Machine (SVM)*, *Random Forest*, dan *Logistic Regression* bekerja dengan representasi teks menggunakan *Term Frequency–Inverse Document Frequency (TF-IDF)* atau *n-gram*. Pendekatan ini relatif ringan, mudah diimplementasikan, serta telah banyak digunakan untuk deteksi spam pada berbagai platform media sosial seperti Instagram dan YouTube [11]. Namun, metode ini bergantung pada kualitas ekstraksi fitur dan kurang mampu menangani konteks semantik yang kompleks dalam bahasa [12].

3. *Deep Learning*

   Pendekatan ini menggunakan arsitektur jaringan saraf seperti *Convolutional Neural Network (CNN)*, *Long Short-Term Memory (LSTM)*, dan *Bidirectional LSTM (BiLSTM)*. Model-model tersebut lebih adaptif dalam memahami konteks teks yang tidak baku dan dinamis [13].

2.2 *Natural Language Processing* (NLP)

Natural Language Processing (NLP) merupakan cabang dari kecerdasan buatan (*Artificial Intelligence*) yang berfokus pada kemampuan komputer untuk memahami, memproses, dan menghasilkan bahasa manusia secara alami [14]. NLP menggabungkan konsep linguistik komputasional, statistika, dan pembelajaran mesin untuk mengubah teks menjadi bentuk representasi numerik yang dapat dipahami oleh sistem [15]. Dalam konteks deteksi komentar spam, NLP berperan penting untuk memahami struktur dan makna kalimat dalam komentar pengguna, sehingga sistem mampu membedakan antara komentar yang relevan dengan konten dan komentar yang bersifat promosi atau mengandung unsur spam.

Tahapan umum dalam NLP mencakup proses *text preprocessing* yang meliputi *case folding*, *tokenization*, *stopword removal*, dan *stemming*. Langkah-langkah ini berfungsi untuk menyederhanakan teks agar dapat dianalisis oleh model secara lebih efisien[16]. Sebagai contoh, tahap *tokenization* digunakan untuk memecah kalimat menjadi unit-unit kata, sedangkan *stopword removal* menghapus kata umum seperti “yang”, “dan”, atau “di” yang tidak memiliki makna kontekstual signifikan. Proses *stemming* juga sangat penting

untuk bahasa Indonesia karena memiliki variasi imbuhan seperti “bermain”, “memainkan”, dan “dimainkan” yang semuanya memiliki akar kata yang sama, yaitu “main”.

Dalam pengembangan sistem deteksi komentar spam, hasil dari pra-pemrosesan teks kemudian diubah menjadi representasi numerik melalui berbagai pendekatan seperti *Bag-of-Words (BoW)*, *Term Frequency–Inverse Document Frequency (TF-IDF)*, dan *word embeddings*. Pendekatan BoW menghitung frekuensi kemunculan kata dalam teks, sedangkan TF-IDF memberikan bobot lebih tinggi pada kata yang jarang muncul namun penting dalam konteks tertentu [17]. Sementara itu, representasi berbasis *word embeddings* seperti Word2Vec atau GloVe menghasilkan vektor yang menggambarkan makna semantik antar kata sehingga model dapat mempelajari hubungan makna dalam teks secara lebih mendalam [18].

Klasifikasi teks merupakan salah satu aplikasi utama NLP yang berfungsi untuk mengelompokkan dokumen atau kalimat ke dalam kategori tertentu, seperti spam dan non-spam. Pendekatan tradisional menggunakan *statistical text classification*, di mana teks direpresentasikan dalam bentuk fitur numerik dan diproses menggunakan algoritma pembelajaran mesin seperti *Naïve Bayes* atau *Support Vector Machine (SVM)* [19]. Namun, metode ini memiliki keterbatasan dalam memahami konteks semantik yang kompleks, terutama pada teks pendek seperti komentar YouTube yang sering kali tidak mengikuti struktur gaya penulisan formal.

2.3 Model *Machine Learning* untuk Deteksi Spam

Machine Learning merupakan bidang studi yang befokus pada analisis dan desain algoritma yang memungkinkan komputer memecahkan masalah dengan cara belajar seperti manusia [3]. Dalam konteks penelitian ini, model pembelajaran mesin digunakan untuk mengklasifikasikan komentar berbahasa Indonesia dari platform YouTube menjadi dua kelas, yaitu spam iklan judi online dan non-spam.

Machine Learning dapat dikelompokkan menjadi tiga, yaitu [20]:

1. *Supervised Learning* adalah pembelajaran terarah atau terawasi, dimana proses pembelajaran, komputer dan mesin akan mempelajari data training yang berisi label. Adapun contoh – contoh algoritma yang termasuk ke dalam *supervised learning* adalah *Linear Regression, Logistic Regression, Linear Discriminant Analysis, k-Nearest Neighbors, Support Vector Machines, Random Forests,* dan *Neural networks.*

2. *Unsupervised Learning* adalah proses pembelajaran tanpa petunjuk. Algoritma dalam komputer yang akan menemukan pola-pola di dalam data. *Unsupervised learning* terjadi ketika memiliki jumlah data input (x) dan tanpa variabel output yang berhubungan. Unsupervised Learning ini dibagi menjadi dua kelas asosiasi dan clustering.
3. *Reinforcement Learning* berbeda dengan *supervised* dan *unsupervised learning*. Komputer akan diminta untuk memecahkan suatu tugas dengan berinteraksi dengan suatu lingkungan. Mesin akan mempelajari cara mengambil keputusan yang spesifik berdasarkan lingkungan yang dinamis. Salah satu contoh implementasi metode ini adalah dengan menggunakan reward and punishment.

Model *Machine Learning* tradisional telah lama digunakan untuk mendeteksi komentar spam pada media sosial. Secara umum, model ini bekerja dengan mempelajari pola dari kumpulan data yang sudah diberi label sebagai *"spam"* dan *"non-spam"*, lalu menggunakan pola tersebut untuk memprediksi kelas pada komentar baru [21]. Beberapa metode populer yang sering digunakan adalah *Naïve Bayes, Support Vector Machine (SVM), Random Forest,* dan *Logistic Regression.*

Model-model ini bekerja dengan merepresentasikan teks dalam bentuk numerik, seperti melalui metode *Term Frequency–Inverse Document Frequency (TF-IDF)* atau *n-gram*, untuk kemudian dianalisis oleh algoritma klasifikasi [22]. Keunggulan dari metode ini adalah kecepatan pelatihan dan kebutuhan komputasi yang rendah. Namun, model klasik memiliki keterbatasan dalam menangkap konteks semantik atau makna yang bergantung pada urutan kata, sehingga kinerjanya menurun pada komentar yang memiliki variasi bahasa, ejaan tidak baku, atau bentuk penyamaran kata.

Untuk mengatasi kelemahan tersebut, metode *Deep Learning* mulai banyak diterapkan. Arsitektur seperti *Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM),* dan *Bidirectional LSTM (BiLSTM)* digunakan untuk memahami urutan dan konteks kata dalam teks secara otomatis. Model BiLSTM, misalnya, mampu membaca konteks dari dua arah (kiri ke kanan dan kanan ke kiri), sehingga lebih efektif dalam mengenali makna utuh dari sebuah komentar [23]. Airlangga [24] menunjukkan bahwa model BiLSTM memberikan hasil yang unggul dibandingkan metode klasik karena dapat menyesuaikan terhadap pola bahasa tidak baku pada komentar YouTube.

Seiring perkembangan teknologi, penelitian terbaru mulai beralih ke model berbasis *Transformer*, yang secara signifikan meningkatkan kemampuan model dalam memahami

hubungan antar kata di seluruh kalimat. Pendekatan ini melandasi perkembangan model modern seperti BERT dan IndoBERT, yang menjadi fokus dalam penelitian ini.

2.4 *Transformer* dan BERT

Model *Transformer* merupakan inovasi besar dalam bidang *Natural Language Processing* (NLP) karena mampu memproses konteks kalimat secara paralel dan efisien [25]. Tidak seperti model berbasis RNN atau LSTM yang membaca teks secara berurutan, *Transformer* menggunakan mekanisme self-attention, yang memungkinkan model memahami hubungan antara semua kata dalam kalimat sekaligus [26].

Secara umum, arsitektur *Transformer* terdiri atas dua komponen utama, yaitu *encoder* dan *decoder*. Bagian *encoder* berfungsi untuk memproses dan memahami konteks dari input teks, sedangkan *decoder* digunakan untuk menghasilkan output berupa teks atau prediksi berdasarkan representasi yang telah dibentuk oleh *encoder* [27]. Dalam setiap lapisan *encoder* terdapat tiga komponen penting: self-attention layer, feed-forward neural network, dan normalization layer. *Self-attention* memungkinkan model menentukan kata mana yang relevan dengan kata lain dalam kalimat, sementara feed-forward layer memperkaya representasi semantic . *Decoder* memiliki struktur serupa tetapi menambahkan *masked self-attention*, agar prediksi setiap kata hanya mempertimbangkan kata sebelumnya, bukan kata yang akan datang [28].

Berdasarkan arsitektur *Transformer*, model BERT (*Bidirectional Encoder Representations from Transformers*) dikembangkan oleh *Google Research* untuk meningkatkan pemahaman konteks kalimat dari dua arah — depan ke belakang dan sebaliknya [29]. Dengan pendekatan bidirectional, BERT dapat memahami makna kata berdasarkan posisi dan konteks sekitarnya, bukan hanya berdasarkan urutan [13].

Berbeda dari *Transformer* asli yang memiliki dua komponen (*encoder–decoder*), BERT hanya menggunakan bagian *encoder* dari *Transformer*, karena tujuannya bukan untuk menghasilkan teks baru seperti pada penerjemahan otomatis, melainkan untuk memahami dan merepresentasikan makna teks secara mendalam. *Encoder* dalam BERT terdiri atas beberapa lapisan *Transformer* (biasanya 12 atau 24), di mana setiap lapisan belajar memahami hubungan kontekstual antar kata melalui mekanisme *multi-head self-attention* [30].

2.5 IndoBERT

IndoBERT merupakan adaptasi dari arsitektur *Bidirectional Encoder Representations from Transformers* (BERT) yang secara khusus dilatih menggunakan korpus teks berbahasa Indonesia. Model ini dikembangkan untuk menjawab kebutuhan NLP bahasa Indonesia, yang memiliki struktur morfologi, afiksasi, dan ragam kosakata berbeda dibandingkan bahasa Inggris [30]. Karena dilatih dengan teks asli berbahasa Indonesia, IndoBERT lebih memahami bentuk kata tidak baku, afiks, serta konteks kalimat khas bahasa Indonesia, termasuk penggunaan kata serapan dan frasa percakapan informal yang sering muncul di komentar YouTube [27].

2.5.1 Dasar Arsitektur IndoBERT

IndoBERT dibangun berdasarkan arsitektur Transformer, khususnya bagian *encoder-only*. Arsitektur *encoder-only*, seperti BERT, dirancang untuk tugas pemahaman bahasa (NLU) dengan menganalisis seluruh konteks kalimat secara dua arah [31]. Arsitektur ini berbeda dengan model *decoder-only* seperti GPT (*Generative Pre-trained Transformer*) yang dirancang untuk tugas generative dengan melihat konteks satu arah (dari kiri ke kanan), atau model *encoder-decoder* seperti T5 (Text-to-Text Transfer Transformer) yang dirancang untuk tugas sekuens-ke-sekuens seperti translasi [32]. Arsitektur ini terdiri atas beberapa lapisan *Transformer* (biasanya 12 untuk versi *base*) dengan beberapa *self-attention heads* di tiap lapisan [33]. Setiap lapisan memiliki tiga komponen utama [34]:

1. *Self-Attention Layer*, menggunakan mekanisme *Multi-Head Self-Attention*, komponen ini menentukan tingkat keterkaitan antar kata dalam kalimat. Mekanisme ini sangat penting untuk menangani polisemi atau kata-kata ambigu, di mana makna sebuah kata sangat bergantung pada kata-kata lain dalam kalimat. Dengan beberapa 'kepala' (*heads*), model dapat menangkap berbagai jenis relasi kontekstual secara bersamaan (misalnya, satu *head* fokus pada relasi subjek-predikat, *head* lain fokus pada relasi kata ganti).
2. *Feed-Forward Neural Network*, adalah jaringan *fully connected* sederhana yang diterapkan secara independen pada setiap posisi (token). Fungsinya adalah untuk memproyeksikan representasi hasil *attention* ke ruang dimensi yang lebih tinggi lalu mengembalikannya, yang secara efektif menambah kapasitas dan kedalaman komputasi model untuk mempelajari representasi yang lebih kompleks.
3. *Normalization dan Residual Connection*, Komponen ini krusial untuk melatih jaringan yang sangat dalam. *Residual connection* adalah 'jalan pintas' di mana *input*

ke sebuah sub-lapisan (misalnya, *input* ke *Self-Attention*) ditambahkan secara langsung ke *output* dari sub-lapisan tersebut. Mekanisme ini memungkinkan gradien mengalir langsung saat *backpropagation* dan mencegah masalah *vanishing gradient*. Setelah itu, *Layer Normalization* diterapkan untuk menstabilkan distribusi aktivasi antar lapisan, membuat proses pelatihan lebih mulus dan cepat.

2.5.2 Proses Embedding

Sebelum teks diproses oleh lapisan *Transformer*, IndoBERT mengubah setiap token menjadi representasi numerik atau *embedding* . Proses *embedding* ini terdiri atas tiga bagian utama [27]:

1. *Token Embedding*, adalah representasi vektor dari setiap token hasil tokenisasi. IndoBERT menggunakan metode WordPiece Tokenization untuk memecah kata menjadi *sub-word*. Metode ini sangat relevan untuk morfologi Bahasa Indonesia yang bersifat aglutinatif (kaya akan imbuhan seperti *me-, -kan, ber-, -an*). WordPiece memecah kata kompleks menjadi *sub-word* yang lebih bermakna memungkinkan model mengenali akar kata yang sama dan mengurangi jumlah kata yang tidak dikenal (*Out-of-Vocabulary*) [35].
2. *Segment Embedding*, adalah vektor yang dipelajari untuk membedakan antara dua segmen teks (Kalimat A dan Kalimat B) yang dimasukkan secara bersamaan. Mekanismenya adalah menambahkan vektor konstan (Segmen A) ke semua token pada kalimat pertama, dan vektor konstan lain (Segmen B) ke semua token pada kalimat kedua, memberikan sinyal yang jelas kepada model untuk mengetahui batas antar kalimat.
3. *Positional Embedding*, adalah vektor yang dipelajari yang menambahkan informasi urutan posisi kata. Karena arsitektur Transformer memproses semua token secara paralel dan tidak memiliki pemahaman urutan secara inheren, embedding ini sangat penting agar model tetap memahami struktur sintaksis dan urutan kata.

Gabungan ketiga embedding ini menghasilkan representasi vektor yang kaya dan kontekstual, siap diproses oleh setiap lapisan *Transformer*.

2.5.3 Proses Pretraining

Pada tahap *pretraining*, IndoBERT dilatih dengan dua tugas utama, yaitu [36]:

1. *Masked Language Modeling* (MLM) — sebagian token (sekitar 15%) dari input diacak dan disembunyikan (*masking*), lalu model dilatih untuk menebak token yang hilang berdasarkan konteks sekitarnya.
2. *Next Sentence Prediction* (NSP) — Model diberi pasangan kalimat (A dan B) dan harus memprediksi apakah B adalah kalimat yang secara alami mengikuti A dalam korpus, atau apakah B adalah kalimat acak. Tugas NSP ini melatih model untuk memahami koherensi dan hubungan logis antar kalimat, yang relevan saat memproses teks media sosial, seperti komentar youtube, yang seringkali tidak memenuhi unsur gramatikal standar atau terfragmentasi

Kedua proses ini membuat model mampu memahami hubungan semantik antar kata dan antar kalimat secara mendalam. Dataset *pretraining* IndoBERT berisi lebih dari 220 juta kata, yang dikumpulkan dari korpus berita dan Wikipedia Indonesia [30]

2.5.4 Proses Fine-Tuning

Setelah melalui *pretraining*, IndoBERT dapat disesuaikan untuk berbagai tugas spesifik melalui proses *fine-tuning*. Pada tahap ini, representasi yang dihasilkan oleh token [CLS] (yang mewakili keseluruhan kalimat) akan diteruskan ke lapisan klasifikasi sederhana, biasanya berupa *fully-connected layer* untuk melakukan prediksi kelas [37]. Semua bobot model dapat di-update agar sesuai dengan domain data baru, seperti komentar YouTube atau ulasan produk [25].

Kemampuan beradaptasi ini merupakan inti dari *Transfer Learning*. *Transfer learning* memanfaatkan 'pengetahuan' yang telah dipelajari model dari *dataset pretraining* berskala besar untuk diterapkan pada tugas baru. Karena IndoBERT sudah memiliki pemahaman mendalam tentang semantik dan sintaksis Bahasa Indonesia, ia tidak perlu belajar bahasa dari awal. Inilah sebabnya IndoBERT tetap dapat mencapai performa tinggi meskipun dengan *dataset fine-tuning* yang relatif kecil, karena model hanya perlu menyesuaikan bobotnya [38].

Parameter pelatihan yang umum digunakan pada fine-tuning IndoBERT mencakup *learning rate* kecil (2e-5 – 5e-5), *batch size* 16 – 32, dan 2 – 4 *epochs* untuk mencegah *overfitting*. Karena telah dilatih pada data berbahasa Indonesia yang beragam, model ini relatif lebih mudah beradaptasi bahkan dengan dataset kecil namun spesifik [5].

Overfitting terjadi ketika model terlalu 'hafal' data latih sehingga performanya buruk pada data baru. Selain pemilihan *learning rate* dan *epoch* yang cermat, beberapa strategi umum untuk mencegah *overfitting* pada tahap *fine-tuning* meliputi [39]:

1. *Early Stopping*: Menghentikan proses pelatihan ketika performa model pada *validation set* mulai menurun, meskipun performa pada *training set* masih meningkat. Dalam implementasi Trainer HuggingFace, ini dapat dicapai dengan mengatur load_best_model_at_end=True, yang secara otomatis akan memuat model dari *epoch* dengan *validation loss* terendah.
2. *Dropout*: Menerapkan lapisan *dropout* pada *output layer* untuk menonaktifkan sebagian neuron secara acak selama pelatihan, sehingga mengurangi ko-adaptasi antar neuron.
3. *Weight Decay* (*L2 Regularization*): Memberikan penalti pada bobot (*weights*) yang bernilai besar (seperti parameter *weight_decay*=0.01) untuk menjaga model tetap sederhana dan mengurangi kompleksitas.

2.5.5 Keunggulan IndoBERT

Kelebihan utama IndoBERT dibandingkan model klasik maupun multilingual antara lain [26]:

1. Kemampuan memahami konteks dua arah (*bidirectional*) dalam bahasa Indonesia, sehingga lebih akurat dalam menafsirkan makna kalimat.
2. Daya tahan terhadap variasi ejaan dan gaya bahasa informal, yang umum dijumpai di media sosial.
3. Kemudahan penerapan (*fine-tuning*) untuk berbagai tugas klasifikasi teks, termasuk deteksi komentar spam, analisis sentimen, dan deteksi ujaran kebencian .
4. Efisiensi dan skalabilitas, karena tersedia dalam berbagai versi (*base*, *large*, *lite*) yang dapat disesuaikan dengan kebutuhan sumber daya komputasi.

Dengan demikian, penggunaan IndoBERT dalam penelitian ini dipilih karena memberikan keseimbangan antara keakuratan tinggi, adaptabilitas terhadap teks informal, serta kemudahan integrasi ke dalam sistem aplikasi deteksi komentar spam YouTube.

2.5.6 Keterbatasan IndoBERT

Di samping keunggulannya, IndoBERT memiliki beberapa keterbatasan yang melekat pada arsitekturnya yang perlu dipertimbangkan [26]:

1. Tidak Optimal untuk Tugas Generatif: Sebagai model *encoder-only*, IndoBERT tidak dirancang untuk tugas-tugas yang memerlukan pembuatan teks baru, seperti meringkas teks secara abstrak atau membangun *chatbot* percakapan.

2. Keterbatasan Panjang Sekuens: Model ini memiliki batas panjang sekuens input. Teks yang lebih panjang dari batas ini, seperti dokumen penuh atau artikel panjang, harus dipotong (*truncation*) atau diproses menggunakan metode *windowing*, yang berpotensi menyebabkan hilangnya konteks global.
3. Tantangan pada Neologisme Ekstrem: Meskipun tangguh terhadap bahasa informal, model mungkin masih kesulitan menafsirkan bentuk bahasa *slang* yang sangat baru dan belum pernah muncul dalam data *pretraining*-nya.

2.6 Confusion Matrix

Confusion Matrix Adalah tabel yang digunakan dalam klasifikasi *machine learning* untuk mengevaluasi kinerja model. Kinerja model diukur menggunakan berbagai metrik evaluasi, yaitu *accuracy*, *precision*, *recall*, dan *F1-score* [40].

Confusion matrix terdiri dari empat komponen utama [41] :

1. *True Positive* (TP): Komentar spam yang berhasil terdeteksi dengan benar.
2. *True Negative* (TN): Komentar non-spam yang dikenali dengan benar.
3. *False Positive* (FP): Komentar non-spam yang keliru diklasifikasikan sebagai spam.
4. *False Negative* (FN): Komentar spam yang tidak terdeteksi.

Bentuk matriksnya dapat digambarkan sebagai berikut:

Tabel 2.1 *Confusion Matrix*

	Prediksi Spam	Prediksi Non-Spam
Aktual Spam	TP	FN
Aktual Non-Spam	FP	TN

Berdasarkan nilai-nilai di atas, dihitung metrik evaluasi dengan rumus [42]:

1. *Accuracy* menunjukkan proporsi prediksi benar dari seluruh data, yang dapat di hitung menggunakan persamaan 1:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

2. *Precision* menggambarkan seberapa tepat model dalam mengidentifikasi komentar spam tanpa kesalahan pada komentar normal, yang dapat di hitung menggunakan persamaan 2:

$$\text{Precision} = \frac{TP}{TP+FP} \quad (2)$$

3. *Recall* menunjukkan kemampuan model dalam menemukan seluruh komentar spam yang benar, yang dapat di hitung menggunakan persamaan 3:

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3)$$

4. *F1-Score* merupakan harmonisasi antara precision dan recall, digunakan ketika dataset tidak seimbang, yang dapat di hitung menggunakan persamaan 4:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

2.7 Chrome Extension

Google Chrome Extension merupakan modul perangkat lunak kecil yang dirancang untuk memodifikasi dan meningkatkan fungsionalitas *web browser* Google Chrome [43]. *Extension* ini dibangun menggunakan teknologi web standar seperti *HyperText Markup Language* (HTML), *Cascading Style Sheets* (CSS), dan JavaScript, namun memiliki akses ke kemampuan khusus peramban yang tidak dimiliki oleh halaman web biasa. *Browser extension* menyediakan mekanisme yang efektif untuk memfilter konten secara *real-time* langsung pada antarmuka pengguna, memberikan lapisan perlindungan tambahan tanpa bergantung sepenuhnya pada sistem moderasi platform penyedia layanan [44].

Arsitektur pengembangan extension modern mengacu pada standar Manifest V3, yang terdiri dari beberapa komponen inti yang saling berinteraksi. Komponen paling vital dalam konteks manipulasi halaman web adalah *Content Scripts*. *Content Scripts* adalah berkas JavaScript yang dieksekusi dalam konteks halaman web yang sedang dikunjungi pengguna, memberikan kemampuan untuk membaca dan memodifikasi *Document Object Model* (DOM). Dengan akses ke DOM, extension dapat memindai elemen spesifik pada halaman, seperti kolom komentar YouTube, untuk mengekstraksi teks dan menyuntikkan gaya visual baru, seperti menyembunyikan atau memburamkan elemen yang teridentifikasi berbahaya [45].

Tantangan utama dalam menerapkan extension pada platform modern seperti YouTube adalah sifat dinamis dari *Single Page Application* (SPA), di mana konten dimuat secara asinkron tanpa memuat ulang seluruh halaman. Untuk menangani hal ini, teknologi *MutationObserver* digunakan sebagai mekanisme pemantauan perubahan struktur HTML secara *real-time*. *MutationObserver* memungkinkan extension untuk mendeteksi penambahan *node* baru pada DOM, seperti komentar yang baru muncul saat pengguna

melakukan *scrolling*, dan memicu proses deteksi secara otomatis hanya pada elemen baru tersebut [46].

Mengingat keterbatasan sumber daya komputasi pada peramban untuk menjalankan model *Deep Learning* yang kompleks seperti IndoBERT, implementasi *Extension* sering kali mengadopsi arsitektur *Client-Server* [47]. Dalam pengembangan aplikasi deteksi spam judi online, extension berfungsi sebagai antarmuka pengguna (*frontend*) yang mengirimkan data teks melalui protokol HTTP (*Fetch API*) ke server pemrosesan (*backend*). Pendekatan ini memungkinkan pemrosesan berat dilakukan di server terpisah, sementara extension hanya bertugas menerima label prediksi dan memanipulasi tampilan antarmuka pengguna berdasarkan respons tersebut.