BAB 2
LANDASAN TEORI

2.1 Teks dan Emosi

Teks merupakan media ekspresi linguistik yang memungkinkan individu menyampaikan pendapat, sikap, maupun kondisi emosional secara eksplisit maupun implisit. Pada platform media sosial, teks umumnya bersifat spontan, informal, dan tidak terstruktur, sehingga mengandung variasi bahasa yang luas, seperti kosakata nonbaku, singkatan, serta ekspresi khas pengguna. Karakteristik ini menjadikan teks media sosial sebagai sumber data yang kaya, namun menantang, dalam penelitian *Natural Language Processing* (NLP), khususnya pada tugas deteksi emosi [12, 13].

Dalam kajian psikologi, emosi didefinisikan sebagai respons afektif individu terhadap suatu stimulus atau pengalaman tertentu. Literatur klasik mengemukakan beberapa kerangka kategorisasi emosi, di antaranya teori emosi dasar oleh Ekman yang mencakup emosi dasar manusia, yaitu marah, sedih, bahagia, takut, cinta, dan jijik [14]. Selain itu, Plutchik mengembangkan model roda emosi yang memetakan delapan emosi primer beserta hubungan intensitas dan oposisi antar emosi [15]. Kerangka-kerangka ini menjadi landasan konseptual dalam berbagai penelitian deteksi emosi berbasis teks.

Penelitian ini memfokuskan klasifikasi emosi pada lima kategori, yaitu gembira, marah, sedih, takut, dan cinta. Pemilihan lima kelas emosi ini didasarkan pada dua pertimbangan utama. Pertama, kategori tersebut merupakan kelas emosi yang paling dominan dan konsisten muncul pada dataset berbahasa Indonesia yang digunakan dalam penelitian terdahulu [12, 13]. Kedua, pembatasan jumlah kelas dilakukan untuk menjaga keseimbangan distribusi data dan stabilitas proses pelatihan model, mengingat peningkatan jumlah kelas tanpa dukungan data yang memadai berpotensi menurunkan performa klasifikasi [1].

Meskipun sama-sama termasuk emosi bernuansa positif, kelas *gembira* dan *cinta* memiliki perbedaan konseptual yang jelas. Emosi gembira umumnya merepresentasikan respons afektif positif terhadap suatu peristiwa atau kondisi tertentu yang bersifat situasional dan sementara. Sebaliknya, emosi cinta lebih merepresentasikan afeksi interpersonal yang bersifat relasional, melibatkan keterikatan emosional yang lebih mendalam dan cenderung berjangka panjang.

Perbedaan ini sejalan dengan kerangka teori emosi dalam psikologi afektif dan digunakan sebagai dasar konseptual dalam interpretasi hasil klasifikasi [14, 15].

Dalam penelitian ini, definisi operasional setiap kelas emosi mengikuti pelabelan yang telah ditetapkan pada dataset sumber. Peneliti tidak melakukan anotasi ulang, melainkan memanfaatkan dataset publik yang telah melalui proses anotasi manual dan evaluasi kesepakatan antar-annotator (*inter-annotator agreement*) sebagaimana dilaporkan pada publikasi aslinya [12, 13]. Dengan demikian, konsistensi label tetap terjaga dan selaras dengan kerangka teori emosi yang digunakan.

2.2 Natural Language Processing

Natural Language Processing (NLP) merupakan bidang yang berfokus pada pemodelan komputasional bahasa manusia sehingga mesin mampu melakukan analisis, penalaran, maupun generasi teks secara otomatis. Perkembangan NLP modern sangat dipengaruhi oleh ketersediaan data dalam jumlah besar, teknik representasi bahasa yang semakin baik, serta kemajuan model berbasis deep learning [2, 1].

2.2.1 Pra-pemrosesan Teks

Pra-pemrosesan dilakukan untuk memastikan teks berada dalam format yang konsisten dan siap digunakan oleh model. Berbagai studi deteksi emosi berbahasa Indonesia menerapkan langkah seperti normalisasi teks, pembersihan karakter yang tidak relevan, serta penanganan variasi informal dan slang [16, 9]. Penggunaan kamus slang untuk menyelaraskan bentuk kata juga dianjurkan berdasarkan studi leksikon informal pada media sosial [17]. Tahap ini esensial karena variasi bahasa informal dapat mengubah pola distribusi kata yang berkaitan dengan pengungkapan emosi.

2.2.2 Tokenisasi

Tokenisasi merupakan tahap awal dalam pemrosesan bahasa alami yang bertujuan mengubah teks mentah menjadi urutan token yang dapat diproses oleh model. Pada model bahasa modern berbasis Transformer, tokenisasi umumnya tidak dilakukan pada tingkat kata (*word-level*), melainkan menggunakan pendekatan *subword tokenization*. Pendekatan ini dirancang untuk mengatasi

keterbatasan kosakata tetap (*fixed vocabulary*), menangani variasi morfologi, serta meminimalkan masalah *out-of-vocabulary* (OOV) [18, 19].

Secara umum, tokenisasi berbasis *subword* bekerja dengan membangun sebuah kosakata awal yang terdiri dari unit-unit sub-kata. Pada tahap inferensi, setiap kata akan dipecah menjadi rangkaian subword yang terdapat dalam kosakata tersebut. Dengan mekanisme ini, model tetap dapat merepresentasikan kata baru atau kata turunan melalui kombinasi *subword* yang lebih kecil, tanpa memerlukan proses tambahan seperti *stemming* atau *lemmatization* [19].

A WordPiece

WordPiece merupakan algoritma tokenisasi *subword* yang digunakan dalam keluarga model BERT, termasuk IndoBERT [18]. Algoritma ini bekerja berdasarkan kosakata tetap (*fixed vocabulary*) yang dibentuk pada tahap pra-pelatihan model. Dengan demikian, tokenisasi WordPiece tidak menciptakan token baru, melainkan memetakan teks masukan ke dalam kombinasi subword yang telah didefinisikan sebelumnya dalam kosakata.

Pada tahap pembentukan kosakata, WordPiece diawali dengan unit dasar berupa karakter individual dan simbol khusus. Selanjutnya, pasangan token yang sering muncul secara berurutan dalam korpus pelatihan akan digabungkan secara iteratif berdasarkan kriteria peningkatan probabilitas maksimum terhadap data. Proses ini menghasilkan kumpulan *subword* dengan ukuran kosakata yang tetap dan terbatas.

Pada tahap tokenisasi, WordPiece menerapkan algoritma *Greedy Longest-Match-First*. Proses ini bekerja dengan memecah setiap kata secara deterministik dari kiri ke kanan.

Sebagai ilustrasi, dengan asumsi kosakata WordPiece mengandung token `ber`, `##lari`, dan `##an`, maka proses tokenisasi kata *berlarian* dilakukan sebagai berikut:

`berlarian` → `[ber, ##lari, ##an]`

Apabila pada suatu tahap tidak ditemukan substring yang cocok dalam kosakata, maka seluruh kata tersebut akan direpresentasikan sebagai token khusus `[UNK]` (*unknown token*). Mekanisme ini menunjukkan bahwa kemampuan WordPiece dalam menangani kata tidak dikenal sepenuhnya bergantung pada keberadaan *subword* penyusunnya dalam kosakata. Oleh karena itu, kualitas data setelah pembersihan menjadi faktor penting dalam efektivitas tokenisasi WordPiece.

B SentencePiece

SentencePiece merupakan algoritma tokenisasi *subword* yang digunakan pada model XLM-RoBERTa [19]. Berbeda dengan WordPiece yang bergantung pada pemisahan kata berbasis spasi, SentencePiece dirancang untuk bekerja langsung pada teks mentah (*raw text*) dan memperlakukan spasi sebagai simbol eksplisit. Pendekatan ini memungkinkan SentencePiece dapat diterapkan secara konsisten pada berbagai bahasa tanpa memerlukan aturan linguistik khusus.

Pada tahap pembentukan kosakata, SentencePiece memodelkan proses tokenisasi sebagai permasalahan segmentasi probabilistik. Kosakata *subword* dibangun menggunakan algoritma berbasis statistik, seperti *Unigram Language Model* dengan tujuan memilih sekumpulan unit subword yang memaksimalkan *likelihood* data pelatihan. Setiap subword dalam kosakata diasosiasikan dengan probabilitas kemunculan tertentu, yang mencerminkan kontribusinya dalam merepresentasikan teks pelatihan.

Pada tahap tokenisasi, SentencePiece mencari segmentasi token yang memiliki probabilitas keseluruhan tertinggi berdasarkan model probabilistik yang telah dilatih. Dengan kata lain, sebuah kata atau kalimat dapat diuraikan menjadi beberapa kemungkinan kombinasi subword, dan SentencePiece memilih kombinasi yang paling optimal secara probabilistik. Proses ini memungkinkan segmentasi yang lebih fleksibel dibandingkan pendekatan deterministik seperti WordPiece.

Sebagai ilustrasi, dengan menganggap spasi direpresentasikan sebagai simbol khusus _, tokenisasi SentencePiece pada kalimat bahasa Indonesia dapat menghasilkan segmentasi sebagai berikut:

```
saya ingin berlarian → [_saya, _ingin, _ber, lari, an]
```

Pendekatan probabilistik ini membuat SentencePiece lebih toleran terhadap variasi morfologi, kesalahan ejaan, maupun kata yang jarang muncul. Apabila sebuah kata tidak dapat direpresentasikan secara utuh menggunakan subword berfrekuensi tinggi, SentencePiece akan memecahnya menjadi unit yang lebih kecil hingga mencapai tingkat karakter.

2.3 Representasi Teks dan Word Embedding

Sebelum berkembangnya model bahasa berbasis Transformer, representasi kata umumnya dilakukan menggunakan metode *word embedding* statis seperti

Word2Vec dan GloVe [20, 21]. Metode ini memetakan setiap kata ke dalam vektor berdimensi tetap berdasarkan pola kemunculan kata dalam korpus.

Keterbatasan utama dari embedding statis adalah ketidakmampuannya menangkap makna kontekstual, karena satu kata hanya memiliki satu representasi vektor. Untuk mengatasi keterbatasan ini, model seperti BERT memperkenalkan *contextualized word embedding*, di mana representasi token bergantung pada konteks kalimat secara keseluruhan [5]. Pendekatan ini sangat relevan untuk tugas deteksi emosi, karena ekspresi emosi dalam teks sering kali bersifat implisit dan bergantung pada hubungan antar kata.

2.4 Model Transformer

Arsitektur Transformer merupakan landasan utama pengembangan model bahasa modern, termasuk BERT, RoBERTa, IndoBERT, dan XLM-R. Model ini diperkenalkan oleh Vaswani et al. [4] dan menggantikan ketergantungan pada struktur berulang seperti *Recurrent Neural Network* (RNN) dengan mekanisme *self-attention* yang mampu memproses seluruh token secara paralel. Pendekatan ini memungkinkan Transformer mempelajari hubungan antar token dalam rentang jarak dekat maupun jauh secara lebih efisien.

Transformer terdiri atas dua komponen utama, yaitu *encoder* dan *decoder*. Untuk tugas klasifikasi teks seperti deteksi emosi, model hanya memanfaatkan bagian *encoder* yang bertugas menghasilkan representasi kontekstual untuk setiap token.

A Self Attention

Komponen inti dari Transformer adalah mekanisme *self attention*. Mekanisme ini bekerja dengan memetakan setiap token ke dalam tiga vektor, yaitu *query* (Q), *key* (K), dan *value* (V). Relevansi antar token dihitung melalui operasi *dot-product* antara Q dan K kemudian dinormalisasi menggunakan fungsi softmax.

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.1}$$

Pembagian dengan $\sqrt{d_k}$ bertujuan menstabilkan nilai *dot-product* agar tidak terlalu besar ketika dimensi vektor meningkat, sehingga proses softmax tetap berada pada rentang yang stabil.

B Multi-Head Attention

Untuk memungkinkan model menangkap berbagai jenis hubungan semantik secara paralel, Transformer menggunakan *multi-head attention*. Mekanisme ini menjalankan beberapa *attention head* sekaligus, masing-masing dengan parameter transformasi linear berbeda.

$$\text{MHA}(Q,K,V) = \text{Concat}(h_1,h_2,\ldots,h_H)W^O \tag{2.2}$$

dengan setiap head didefinisikan sebagai:

$$h_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{2.3}$$

di mana H menyatakan jumlah head, sedangkan W_i^Q, W_i^K, W_i^V, dan W^O merupakan matriks parameter yang dipelajari selama pelatihan.

Penggunaan banyak head memungkinkan model menangkap pola sintaksis dan semantik yang berbeda pada saat bersamaan.

C Feed-Forward Neural Network

Setelah melewati blok *multi-head attention*, representasi token diproses oleh *position-wise feed-forward network (FFN)*. Jaringan ini terdiri atas dua lapisan linear dengan fungsi aktivasi non-linear, biasanya ReLU atau GELU.

$$\text{FFN}(x) = \text{GELU}(xW_1 + b_1)W_2 + b_2 \tag{2.4}$$

Karena FFN diterapkan secara independen pada setiap token, proses ini efektif memperkaya representasi vektor tanpa mengubah struktur urutan.

D Residual Connection dan Layer Normalization

Untuk menjaga stabilitas pelatihan, setiap blok pada Transformer dilengkapi dengan mekanisme residual connection dan *layer normalization*. Residual connection membantu mencegah hilangnya gradien pada model berlapis banyak.

$$\text{Output} = \text{LayerNorm}(x + \text{Sublayer}(x)) \tag{2.5}$$

Penggunaan residual dan normalisasi ini terbukti meningkatkan kedalaman model yang dapat dilatih tanpa mengorbankan stabilitas.

2.5 Transfer Learning dan Fine-Tuning pada Model Bahasa

Transfer learning merupakan paradigma pembelajaran mesin di mana pengetahuan yang diperoleh dari suatu tugas atau domain sumber dimanfaatkan kembali untuk menyelesaikan tugas lain pada domain target yang berbeda, namun masih memiliki keterkaitan [22]. Dalam konteks pemrosesan bahasa alami, transfer learning umumnya diwujudkan melalui penggunaan model bahasa pralatih (*pretrained language models*) yang telah dilatih pada korpus teks berskala besar, kemudian diterapkan pada tugas *downstream* yang lebih spesifik.

Model bahasa pralatih mempelajari representasi linguistik umum melalui tugas pra-pelatihan seperti *Masked Language Modeling*. Representasi ini mencakup informasi sintaksis dan semantik yang bersifat umum dan tidak bergantung pada satu tugas tertentu. Melalui transfer learning, pengetahuan linguistik tersebut dapat digunakan kembali untuk berbagai tugas seperti klasifikasi sentimen, deteksi emosi, dan penjawaban pertanyaan tanpa harus melatih model dari awal [5].

Fine-tuning merupakan salah satu strategi implementasi dari transfer learning, di mana model pralatih disesuaikan secara langsung terhadap tugas target dengan melatih ulang parameter model menggunakan data berlabel pada tugas *downstream* [23]. Berbeda dengan pendekatan *feature extraction* yang hanya memanfaatkan keluaran encoder sebagai fitur statis, fine-tuning memperbarui bobot parameter model sehingga representasi yang dihasilkan menjadi lebih spesifik terhadap karakteristik tugas dan data target.

Pada praktiknya, fine-tuning dapat dilakukan secara parsial maupun penuh. Fine-tuning parsial hanya memperbarui parameter pada lapisan tertentu, seperti *classification head*, sementara lapisan encoder dibekukan. Sebaliknya, *full fine-tuning* memperbarui seluruh parameter model, termasuk seluruh lapisan encoder Transformer [24]. Pendekatan full fine-tuning umumnya memberikan performa yang lebih baik pada dataset berukuran menengah hingga besar, karena memungkinkan model menyesuaikan representasi internalnya secara optimal terhadap distribusi data tugas target.

Dalam penelitian ini, strategi yang digunakan adalah *full fine-tuning*, di mana seluruh parameter encoder pada model IndoBERT dan XLM-RoBERTa diperbarui selama proses pelatihan. Pemilihan strategi ini didasarkan pada

tujuan penelitian untuk membandingkan kemampuan adaptasi kedua model secara menyeluruh terhadap tugas klasifikasi emosi teks berbahasa Indonesia. Dengan menerapkan konfigurasi fine-tuning yang identik pada kedua model, perbedaan kinerja yang dihasilkan dapat dikaitkan secara langsung dengan karakteristik model pralatih yang digunakan, bukan dengan perbedaan strategi pelatihan.

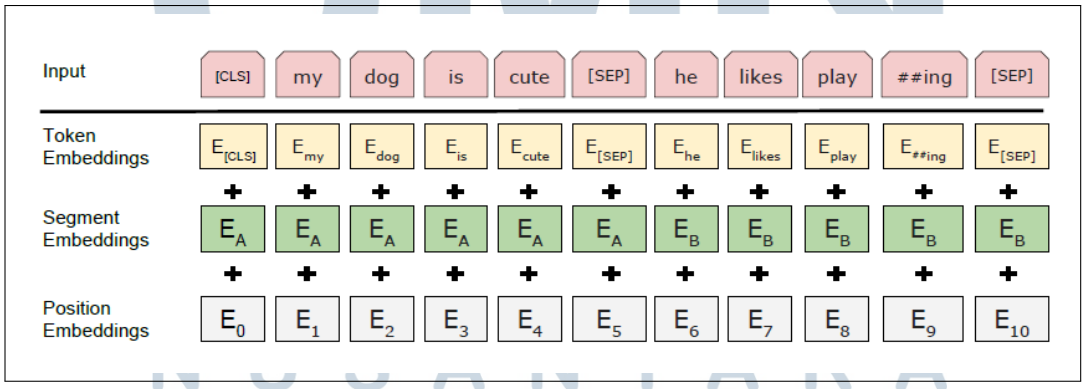
2.6 BERT

BERT (*Bidirectional Encoder Representations from Transformers*) adalah model pralatih berbasis arsitektur Transformer yang merevolusi pemrosesan bahasa alami sejak diperkenalkan oleh Devlin et al. [5]. Model ini dirancang untuk menghasilkan representasi kata yang kontekstual secara dua arah, sehingga mampu memahami makna kata berdasarkan konteks kalimat secara menyeluruh.

A Embedding pada BERT

Sebelum masuk ke lapisan *encoder*, teks yang telah ditokenisasi diubah menjadi representasi vektor melalui tiga jenis embedding yang dijumlahkan:

- **Token Embedding**: representasi untuk setiap token hasil *WordPiece*.
- **Segment Embedding**: identitas kalimat agar model mengetahui perbedaan antar kalimat ketika input terdiri atas pasangan kalimat.
- **Position Embedding**: informasi posisi token dalam urutan teks.



Gambar 2.1. Visualisasi *embedding* pada *BERT*

Sumber: [5]

B Tujuan BERT

BERT dilatih dengan dua tujuan utama [5]:

1. ***Masked Language Modeling* (MLM)**
   Sebagian token input ditutup menggunakan token `[MASK]`, kemudian model dilatih untuk menebak kata yang hilang. Pendekatan ini memaksa model memahami konteks dua arah karena kata yang hilang diprediksi berdasarkan kata di kiri dan kanan.

2. ***Next Sentence Prediction* (NSP)**
   Model dilatih untuk menilai apakah dua kalimat berurutan secara logis, sehingga BERT dapat memahami hubungan antar kalimat.

2.7 IndoBERT

IndoBERT merupakan model monolingual turunan dari BERT yang dikembangkan secara khusus untuk bahasa Indonesia oleh Wilie et al. [8]. Model ini dirancang untuk menghasilkan representasi bahasa yang lebih akurat terhadap karakteristik linguistik bahasa Indonesia, yang tidak sepenuhnya dapat ditangkap oleh model multibahasa seperti mBERT maupun XLM-R. Untuk membangun korpus pra-latih, Wilie et al. menggabungkan berbagai sumber teks Indonesia seperti Wikipedia, Kompas, Tempo, dan berita daring lain sehingga menghasilkan lebih dari 200 juta kata.

Secara arsitektural, IndoBERT mempertahankan struktur *Transformer encoder* yang sama dengan BERT [4]. Setiap lapisan *encoder* terdiri atas mekanisme *multi-head self-attention* dan jaringan *feed-forward* yang bekerja untuk menangkap hubungan semantik antar token dalam satu kalimat. Dengan demikian, setiap token di dalam teks akan merepresentasikan makna yang kontekstual, bukan sekadar makna leksikal.

2.8 RoBERTa

RoBERTa (Robustly Optimized BERT Pretraining Approach) merupakan model turunan langsung dari BERT yang dikembangkan oleh Liu et al. [6]. Secara struktural, RoBERTa tidak memperkenalkan arsitektur baru, melainkan mempertahankan keseluruhan desain *Transformer encoder* milik BERT [5].

Kesamaan arsitektur ini mencakup penggunaan *multi-head self-attention*, *feed-forward network*, *positional embeddings*, serta skema pengolahan token berbasis *WordPiece*. Dengan demikian, RoBERTa dapat dipandang sebagai versi BERT yang dioptimalkan, bukan model dengan mekanisme baru.

Perbedaan utama antara BERT dan RoBERTa terletak pada prosedur pra-latih, bukan pada struktur model. Liu et al. menunjukkan bahwa berbagai keputusan desain pada BERT—seperti penggunaan tugas *Next Sentence Prediction* (NSP), ukuran *batch* yang kecil, *static masking*, serta jumlah data pra-latih yang terbatas—membatasi performa BERT. RoBERTa memperbaiki keterbatasan tersebut melalui empat perubahan utama berikut:

1. **Penghapusan *Next Sentence Prediction* (NSP)**
   BERT menggunakan dua tujuan pelatihan, yaitu *Masked Language Modeling* (MLM) dan *Next Sentence Prediction* (NSP) [5]. RoBERTa menghilangkan NSP setelah ditemukan bahwa tugas tersebut tidak memberikan peningkatan yang signifikan terhadap pemahaman relasi antarkalimat [6]. Penghapusan NSP memungkinkan model memusatkan kapasitas representasinya pada pembelajaran konteks intra-kalimat melalui MLM.

2. **Jumlah Data yang Lebih Besar**
   RoBERTa dilatih menggunakan ukuran batch yang jauh lebih besar serta data yang lebih luas, termasuk BookCorpus, Wikipedia, CC-News, OpenWebText, dan Stories [6]. Jumlah ini menghasilkan stabilitas gradien yang lebih baik dan representasi bahasa yang lebih kuat.

3. **Dynamic Masking**
   Berbeda dari BERT yang menggunakan *static masking*, RoBERTa menerapkan *dynamic masking* sehingga token yang dimask berubah setiap iterasi. Strategi ini mendorong model melihat variasi konteks yang lebih kaya, sehingga meningkatkan performa pada tugas downstream.

4. **Penyesuaian *sequence length* secara bertahap.**
   RoBERTa melatih model pada *sequence* pendek terlebih dahulu sebelum beralih ke *sequence* yang panjang (*full-length sequence*). Pendekatan bertahap ini meningkatkan efisiensi pelatihan tanpa mengorbankan kemampuan menangkap dependensi jarak jauh.

2.9 XLM-R

XLM-R (*Cross-Lingual Language Model – RoBERTa*) merupakan model pralatih multilingual yang diperkenalkan oleh Conneau et al. [7]. Model ini dibangun sebagai pengembangan dari RoBERTa, yang merupakan penyempurnaan dari model BERT. Oleh karena itu, secara garis besar XLM-R tetap menggunakan *Transformer encoder* dengan mekanisme *self-attention* bertumpuk sebagaimana digunakan pada BERT [4, 6]. Namun, perbedaannya terletak pada strategi pelatihan, skala data, serta metode tokenisasi yang membuat model ini lebih unggul dalam pemahaman lintas bahasa.

XLM-R mengadopsi seluruh optimasi tersebut dan memperluas cakupannya ke skenario multibahasa melalui penggunaan korpus *CommonCrawl* (CC-100) sebesar 2,5 TB yang mencakup 100 bahasa, termasuk bahasa Indonesia [7]. Skala data yang luas memungkinkan XLM-R belajar pola linguistik lintas domain, variasi tata bahasa, serta struktur morfologi dari beragam bahasa.

Dari sisi tokenisasi, XLM-R tidak menggunakan *WordPiece* seperti BERT dan IndoBERT, melainkan *SentencePiece* dengan algoritma *Unigram Language Model* [19]. Tokenisasi ini bersifat *language-agnostic* karena memproses teks mentah tanpa bergantung pada spasi, sehingga lebih adaptif untuk menghadapi keragaman sistem tulisan dan bentuk kata.

Secara arsitektural, XLM-R tetap menggunakan blok *multi-head self-attention* dan jaringan *feed-forward* pada setiap lapisan encoder, sebagaimana pada BERT dan RoBERTa. Mekanisme ini memungkinkan model memahami ketergantungan jangka panjang antar token secara paralel. Namun, keunggulan XLM-R terutama berasal dari kombinasi skala pretraining yang sangat besar dan optimasi RoBERTa-style yang membuatnya lebih tahan terhadap perbedaan distribusi data antar bahasa. Studi Conneau et al. [7] menunjukkan bahwa XLM-R memberikan peningkatan substansial pada tugas klasifikasi lintas bahasa, termasuk pada bahasa dengan korpus berukuran kecil.

2.10 Cross-Entropy Loss

Pada tugas klasifikasi, fungsi loss yang umum digunakan adalah *Categorical Cross-Entropy* atau *Sparse Categorical Cross-Entropy*. Loss ini mengukur perbedaan antara distribusi prediksi dan distribusi label sebenarnya [25].

Untuk klasifikasi C kelas, persamaannya adalah:

$$L = -\sum_{c=1}^{C} y_c \log(\hat{y}_c) \tag{2.6}$$

dengan:

- y_c adalah label sebenarnya (one-hot),
- $\hat{y}_c$ adalah probabilitas prediksi dari model.

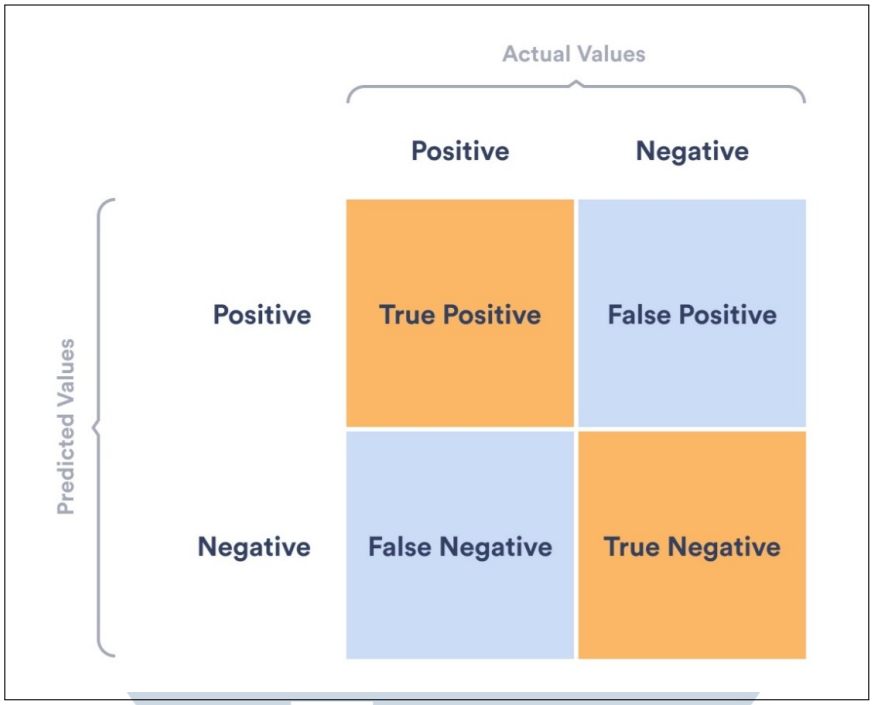
Loss ini mendorong probabilitas prediksi mendekati label sebenarnya dan stabil untuk pelatihan model besar seperti Transformer.

2.11 Metrik Evaluasi

Evaluasi kinerja model klasifikasi diperlukan untuk menilai seberapa baik model memprediksi label yang benar pada data uji. Proses ini penting agar hasil yang diperoleh tidak hanya dinilai secara subjektif, tetapi memiliki dasar kuantitatif yang dapat dibandingkan. Dalam penelitian ini, penilaian kinerja dilakukan dengan mengacu pada metrik evaluasi yang umum digunakan pada tugas klasifikasi teks.

A Confusion Matrix

Confusion matrix menunjukkan jumlah prediksi benar dan salah dari model klasifikasi untuk setiap kelas. Baris mewakili kelas aktual, sedangkan kolom mewakili kelas prediksi. Matriks ini memberikan gambaran distribusi kesalahan klasifikasi dan membantu mengidentifikasi kelas yang cenderung salah diprediksi [26].



Gambar 2.2. Visualisasi *confusion matrix*
Sumber: [26]

B Akurasi

Akurasi mengukur proporsi prediksi benar terhadap total data uji [27].

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.7}$$

C Precision

Precision mengukur ketepatan prediksi positif, yaitu rasio prediksi positif yang benar terhadap total prediksi positif [27].

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2.8}$$

D Recall

Recall mengukur kemampuan model menemukan seluruh contoh positif yang relevan, yaitu rasio prediksi positif yang benar terhadap total data positif sebenarnya [27].

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.9}$$

E F1-Score

F1-Score adalah rata-rata harmonik dari *Precision* dan *Recall*. Metrik ini memberikan keseimbangan antara keduanya, terutama ketika terdapat ketidakseimbangan kelas [27].

$$\text{F1-Score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.10}$$