BAB II
KAJIAN LITERATUR

2.1 *Natural Language Processing* (NLP)

Natural Language Processing (NLP) adalah cabang dari kecerdasan buatan dan linguistik yang bertujuan untuk membantu komputer memahami bahasa yang tertulis dalam bentuk kata, kalimat, atau paragraf [27], [28]. Linguistik merupakan ilmu dibalik *natural language* di mana bahasa dipelajari dengan tujuan untuk memahami bagaimana bahasa bekerja [29]. Dengan menggabungkan teknik komputasi bersama teori linguistik melalui pemodelan berbagai tingkatan bahasa seperti suara (fonologi), formasi kata (morfologi), struktur kalimat (sintaks), makna kata (semantik), dan konteks kata (pragmatik), NLP diharapkan dapat mendukung beragam tugas dari sebuah sistem atau algoritma untuk memproses bahasa manusia dengan lebih efektif [27].

Aplikasi NLP dapat diterapkan di berbagai area, termasuk alat translasi, ekstraksi informasi, dan *spam filtering*. Sebagai alat translasi, NLP mentranslasi kalimat dari satu bahasa ke bahasa lain tanpa mengubah makna dan tata bahasa. Dalam ekstraksi informasi, NLP mengambil data yang menarik dalam frasa teks seperti nama, tempat, atau kejadian untuk menyediakan informasi yang berguna dengan efisien. Dalam *spam filtering*, dengan mendapatkan seberapa banyak dan sering suatu kata digunakan bisa membantu mendeteksi potensi *spam* [27], [30].

Dalam metode NLP, setiap kata akan dibersihkan untuk mengurangi biaya komputasi dan meningkatkan performa model yang disebut *pre-processing*. Langkah yang umum digunakan untuk melakukan *pre-processing* adalah sebagai berikut [28]:

1. Tokenisasi, memecah teks menjadi token individual untuk dapat diidentifikasi oleh korpus.
2. *Lowercasing*, mengubah seluruh huruf di teks menjadi huruf kecil untuk mengurangi redundansi kata.
3. *Stopwords Removal*, menghapus kata tidak bermakna yang sering muncul dalam kalimat seperti “yang” ,”dan” ,”di” , dan lain-lain.
4. *Stemming*, mengubah kata menjadi bentuk bakunya untuk mengurangi redundansi makna kata.

NLP terbagi menjadi dua kategori, *Natural Language Understanding* (NLU) dan *Natural Language Generation* (NLG).

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

2.1.1 *Natural Language Understanding* (NLU)

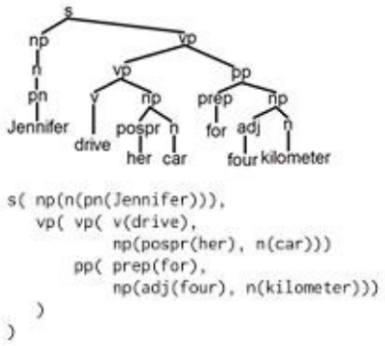
NLU merupakan cabang dari ilmu komputer yang menggunakan teknik komputasi untuk mempelajari dan memahami konten berbahasa manusia yang kontekstual, bermakna, dan akurat [31]. Secara teknis, NLU membantu komputer mengekstraksi data dari konten berbahasa untuk dapat diproses dan diaplikasikan dalam pelayanan manusia [32].

NLU telah mengalami perkembangan yang signifikan, banyak dari kemajuannya diberkati oleh meningkatnya teknik pembelajaran mesin terutama berbasis *deep learning*. Perkembangan ini memungkinkan mesin untuk memahami, menginterpretasi, dan menghasilkan bahasa manusia dengan tingkat akurasi yang tinggi [33]. NLU dapat berfungsi dalam berbagai tugas seperti membantu komunikasi antar manusia hingga komunikasi manusia dengan mesin [32]. Ini dapat mencakup tugas yang berhubungan dengan pemahaman bahasa manusia seperti identifikasi dialek, analisis sentimen, analisis struktur kata, dan *parsing* [27], [30]. Selain itu, NLU memungkinkan mesin untuk memahami bahasa manusia dengan mengekstraksi dan menganalisis informasi penting seperti konsep, entitas, dan kata kunci. Ini penting dalam pelayanan pelanggan untuk memahami masalah yang dialami [27]. Pentingnya NLU telah mendapatkan perhatian dari berbagai perusahaan IT seperti Facebook atau Google. Dengan penggunaan komputasi awan, platform NLU dapat dijangkau di mana saja dan akan selalu siap untuk digunakan tanpa mengkhawatirkan instalasi atau limitasi perangkat [31].

Umumnya NLU terdiri dari beberapa tugas berikut:

1. Tokenisasi, merupakan proses untuk menghapus tanda baca dan mengidentifikasi batasan antar kata. Hasil representasi token umumnya didampingi dengan kata bakunya [32]. Sebagai contoh, “Andi mengendarai mobilnya, sejauh empat kilometer.” akan ditokenisasi menjadi {[Andi, Andi], [mengendarai, kendara], [mobilnya, mobil], [sejauh, jauh], [empat, empat], [kilometer, kilometer]}.
2. Penandaan Kelas Kata, merupakan proses melabeli token dengan kategori seperti “kata benda” (N), “kata kerja” (V), “kata sifat” (ADJ) dan sebagainya. Fitur lain juga dapat ditambahkan berdasarkan morfologi dan tata bahasanya [32]. Contoh: {…, [mengendarai, kendara [V, awalan “meng-”]], [mobilnya, mobil [N, akhiran “-nya”],…}.
3. Analisis Sintaksis, proses ini mengecek apakah sebuah kalimat telah dibentuk dengan benar berdasarkan peraturan tata bahasa dan akan menghasilkan sebuah pohon turunan yang menunjukkan tingkatan struktur kalimat seperti di Gambar 2.1.

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.



Gambar 2.1 Pohon Turunan dan Tingkatan Struktur Kalimat [32]

Tipe dari struktur dalam pohon turunan berupa: s= “kalimat”, np= “frasa kata benda”, pn= “kata benda khusus”, pospr= “kata ganti posesif”, prep= “preposisi”, pp= “frasa preposisi”, vp= “frasa kata kerja”, dan lain-lain [32].

4. Disambiguasi Struktural, ambiguitas struktur terjadi jika tata bahasa yang digunakan membuat sebuah kalimat memiliki lebih dari satu makna. Metode untuk mengatasi masalah ini dapat dilakukan dengan melihat konteks dari kalimat di sekitarnya. Proses disambiguitas juga dapat dilakukan berdasarkan kepercayaan masyarakat. Misalnya dengan kalimat “Lihatlah anjing itu dengan teropong”. Masyarakat percaya bahwa anjing tidak dapat menggunakan teropong. Oleh karena itu, kalimat hanya memiliki satu makna yaitu melihat anjing dengan menggunakan teropong [32].
5. Disambiguasi Makna Kata, terdapat keadaan di mana sebuah kata memiliki lebih dari satu makna. Misalnya kata “bisa” dapat berarti kemampuan melakukan sesuatu atau sejenis racun. Proses disambiguitas juga dapat dilakukan dengan melihat konteks di sekitar kata atau melalui wawasan lainnya [32].
6. Representasi Semantik, satu dari beberapa metode untuk merepresentasi makna kalimat adalah melalui tata bahasanya. Metode ini menganalisis permukaan struktur sintaksis dari kalimat dengan mempelajari kombinasi peran semantik yang dapat berupa: Pelaku, Objek, Penerima, Lokasi, Alat, dan lain-lain. Sebagai contoh, kata kerja “memberi” memerlukan sebuah Pelaku, Objek, dan Penerima untuk menghasilkan kalimat seperti “Andi (Pelaku) memberi uang (Objek) kepada Budi (Penerima)” [32].
7. Resolusi Anafora, merupakan masalah dalam menentukan apa yang dirujuk oleh sebuah kata ganti atau frasa kata benda. Sebagai contoh, kalimat “Andi ingin menjenguk Budi. Dia adalah orang yang baik.” memiliki dua makna karena kalimat bisa menyatakan orang baik tersebut sebagai Andi atau Budi. Pendekatan resolusi anafora mengandalkan seperangkat faktor yang terdiri dari: jenis kelamin, batasan “c-command”, konsistensi

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

semantik, paralelisme sintaksis, paralelisme semantik, tonjolan kata, jarak antar kata, dan lain-lain.

Faktor-faktor tersebut kemudian dapat dilakukan proses menggunakan dua strategi [32]:

1. Eliminasi, di mana frasa kata benda tertentu akan diabaikan seperti jenis kelamin, batasan "c-command", konsistensi semantik [32]. Contoh kalimat "*Maria told John that she would help him*". Kata "*she*" merujuk kepada Maria karena dalam Bahasa Inggris kata ganti *she* digunakan untuk merujuk kepada seseorang berjenis kelamin perempuan sehingga nama yang bersifat jantan seperti "John" dieliminasi.
2. Preferensial, di mana faktor diberi peringkat berdasarkan faktor seperti jarak atau seberapa menonjol sebuah kata [32]. Contoh kalimat "Rina membantu Sari setelah dia menyelesaikan tugasnya". Kata "dia" lebih merujuk ke Sari karena jarak antara kata "Sari" dan "dia" lebih dekat dibandingkan kata "Rina".

NLU juga memiliki tugas opsional lainnya seperti [32]:

1. Komputasi Afikatif, bertugas menghubungkan teknologi dan emosi manusia dengan mendeteksi dan membalas perasaan yang diutarakan [32].
2. Analisis Wacana, memungkinkan sistem NLU untuk mengungkapkan motif dibalik sebuah kalimat. Analisis ini bertujuan untuk mendapatkan hubungan retoris antara kalimat, sebab akibat, dan waktu [32].
3. Pragmatik, berfokus dalam mempelajari tindakan ucapan dan implikasi dari sebuah kalimat [32].

2.1.2 *Natural Language Generation* (NLG)

NLG adalah proses untuk menghasilkan teks berbahasa alami yang dapat memenuhi tujuan komunikasi tertentu. Misalnya teks yang dihasilkan dapat berkisar antara jawaban singkat, komentar dan pertanyaan, hingga kalimat penjelasan sepanjang satu halaman [34].

Bidang NLG telah mengalami perubahan drastis selama 20 tahun terakhir, dengan munculnya beberapa metode *deep-learning* seperti model-model *encoder-decoder* neural yang dipelopori oleh *sequence-to-sequence* (Seq2Seq) telah diusulkan untuk mencapai tujuan dengan memetakan teks *input* ke teks *output* membuat NLG semakin luas digunakan. Baru-baru ini NLG mulai memanfaatkan kemajuan terkini melalui pendekatan berbasis data, pembelajaran mesin, dan *deep learning* sehingga evaluasi dari *output* NLG juga harus mendapatkan perhatian sistematis [34].

Sistem dan aplikasi NLG umumnya terdiri dari tiga atau empat tahap [32]:

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

1. Perencanaan, diperlukan bagi sistem untuk menentukan data dari *input* yang lebih menarik untuk diutarakan dalam percakapan dan bagaimana struktur kalimatnya akan dibuat [32].
2. Perencanaan Mikro, terdiri dari tugas berikut:
    a. Agregasi, dilakukan jika terdapat *input* data yang memiliki makna yang sama dengan data lainnya di mana data-data tersebut kemudian akan digabungkan [32].
    b. Generasi Anafora, dilakukan untuk mengurangi pengulangan nama entitas supaya hasil *output* tidak bersifat monoton atau buatan [32].
    c. Pemilihan item leksikal, merupakan cara untuk membuat sebuah kalimat tidak terdengar aneh dan mudah dimengerti dengan mengubah kata tertentu sesuai konteks [32].
    d. Struktur sintaksis, digunakan untuk memberikan struktur kepada item leksikal di sebuah kalimat karena konsep dapat diutarakan melalui lebih dari satu struktur [32].
3. Realisasi, bertugas memastikan hasil keluaran kalimat atau percakapan sudah benar secara morfologis dan ortografis. Ini meliputi tata bahasa, tanda baca, hingga ejaan [32].
4. Presentasi, merupakan tahap opsional yang berfokus dengan bagaimana konten akan disampaikan. Dalam teks tertulis, tahap ini berkaitan dengan pemberian judul, penekanan kalimat, dan tanda baca. Sedangkan untuk teks lisan, tahap ini berkaitan dengan intonasi dan jenis kalimat [32].

2.2 Analisis Sentimen

Analisis sentimen adalah studi komputasional yang mempelajari kepribadian seseorang terhadap suatu entitas yang bisa berupa sebuah individu, kejadian, atau topik. [35], [36]. Tugas dari analisis sentimen adalah mengidentifikasi jenis opini terhadap entitas tersebut untuk mengklasifikasi polaritas sentimen terkandung di dalamnya menjadi tiga kategori utama yaitu: positif, negatif, dan netral [36], [37]. Ini memungkinkan penggunaan analisis sentimen untuk mengetahui perasaan publik yang dapat digunakan sebagai data yang bisa ditindaklanjuti [28].

Meningkatnya penggunaan internet seperti situs ulasan produk membuatnya menjadi sumber informasi utama untuk mengetahui pandangan dan opini masyarakat mengenai suatu produk. Untuk dapat memantau opini publik secara terus menerus dan membantu pengambilan keputusan, perlu dilakukan analisis secara otomatis pada data yang dihasilkan pengguna. Sumber data dapat berasal dari ulasan produk yang bisa diekstraksi untuk membantu menjangkau pelanggan yang membutuhkan perhatian lebih. Dengan cara ini,

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

keputusan dapat diambil berdasarkan opini terhadap produk dan mampu meningkatkan kepuasan pelanggan. Hal ini juga membuat analisis sentimen menjadi topik penelitian yang populer pada beberapa tahun terakhir dan merupakan alat yang penting dalam berbagai bidang [35], [36], [37], [38].

Analisis sentimen umumnya terdiri dari beberapa tahap:

1. *Preprocessing*

Pada tahap ini, teks mentah dibersihkan dengan menghilangkan data yang tidak relevan. Ini bertujuan untuk mengurangi kompleksitas data sehingga teks bisa diolah dengan lebih mudah, berikut tahapannya [37], [39]:

a. Pembersihan Data, bertujuan untuk menghapus data duplikat dan atribut yang tidak digunakan.
b. *Case Folding*, adalah proses untuk mengubah semua huruf menjadi huruf kecil.
c. *Stopword*, di mana kata-kata yang kurang bermakna dan sering muncul dalam jumlah besar akan dihapus. Contoh kata stopword adalah ”yang”, ”dan”, ”di”, dan lain-lain.
d. Tokenisasi, merupakan proses memisahkan kata-kata didalam teks menjadi potongan token untuk dianalisis.
e. *Stemming*, untuk mengubah bentuk suatu kata menjadi bentuk dasarnya.

2. Ekstraksi Fitur

Setelah prapemrosesan, data akan diubah menjadi fitur yang dapat dimengerti model dengan teknik seperti [37], [40], [41]:

a. TF-IDF, bertujuan mengevaluasi seberapa penting suatu kata dalam suatu kalimat dengan mengukur seberapa sering kata tersebut muncul.
b. GloVe, adalah model *log-linear* yang bertugas mempelajari hubungan antar kata dengan menghitung seberapa sering kata tersebut muncul secara bersamaan di dalam kalimat.
c. Word2vec, merupakan algoritma yang memetakan setiap kata menjadi representasi vektor dengan memanfaatkan informasi lokal dari kata tersebut.
d. FastText, merupakan metode yang merepresentasikan kata sebagai karakter *n-gram* untuk memahami kata yang pendek.

3. Klasifikasi

Teks yang telah diekstraksi akan diklasifikasi berupa pengelompokan data ke dalam beberapa kelas berdasarkan fitur yang dimilikinya. Teknik klasifikasi menggunakan metode *machine learning* dengan model matematis seperti *Logic Regression*, *Naïve Bayes*, dan SVM

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

atau menggunakan metode *deep learning* seperti RNN dan LSTM yang memanfaatkan jaringan neural buatan [37].

Selain metode klasifikasi tradisional, telah muncul metode terbaru yang memanfaatkan arsitektur Transformer. Transformer adalah model bahasa yang menggunakan arsitektur berbasis perhatian dengan menerapkan mekanisme perhatian diri *multi-headed*. Model ini pertama kali diperkenalkan untuk tugas pemodelan bahasa dan telah menjadi arsitektur *deep learning* yang banyak digunakan di bidang NLP [42].

Dalam analisis sentimen, model transformer sangat berguna dalam menangkap terminologi dalam domain spesifik, pola negasi, pergeseran sentimen dan kalimat yang kompleks. Model ini menggunakan mekanisme perhatian diri yang mengukur relevansi kata sepanjang urutan kalimat. Fleksibilitas dari model ini telah menunjukkan kepraktisan dalam analisis umpan balik berskala besar [43].

Model Transformer terdiri dari lapisan *encoder* dan *decoder* di mana setiap lapisan *encoder* mengandung jaringan *position-wise feed-forward* dan perhatian diri *multi-headed* yang akan memberikan koneksi residual dan dilanjutkan dengan lapisan normalisasi, sedangkan lapisan *decoder* terdiri dari lapisan perhatian diri *multi-headed* tersembunyi (*masked*) yang akan menerima output dari *encoder* [42].Model harus menjalani proses *pre-training* sebelum dilakukan *fine-tuning*. Dalam *pre-training*, model akan dilatih dengan *dataset* besar secara *unsupervised* untuk memungkinkan pemodelan dalam skala tinggi. Setelah *pre-training*, model akan dilakukan *fine-tuning* dengan data di bidang tertentu untuk tugas *downstream* [42].

2.3 *Machine Learning*

Machine learning merupakan komponen dari kecerdasan buatan yang berupaya untuk mempelajari pola tersembunyi dalam sebuah data dengan tujuan menyelesaikan masalah berdasarkan informasi sebelumnya [44]. Disiplin ini bertujuan untuk membangun sistem komputer yang dapat berkembang secara otomatis melalui pengalaman [45].

Selama dua dekade terakhir, *machine learning* telah berkembang secara pesat mulai dari sebuah percobaan laboratorium menjadi teknologi komersial yang banyak digunakan [45]. Dalam kecerdasan buatan, *machine learning* telah menjadi metode andalan untuk mengembangkan aplikasi seperti visi komputer, pengenalan suara, NLP, robotik, dan lain-lain [45]. Banyak pengembang AI telah mengakui bahwa lebih mudah untuk melatih sebuah model dengan memperlihatkan contoh *input* dan *output* yang diinginkan dibandingkan memprogram respons terhadap semua kemungkinan *input* secara manual. Efek dari *machine*

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

learning telah mempengaruhi bidang ilmu komputer hingga berbagai industri yang berfokus pada data seperti diagnosis sistem atau logistik [45].

Machine learning terdiri dari metode *supervised* dan *unsupervised* dengan perbedaannya yang terletak pada ada tidaknya label dalam sebuah data pelatihan.

1. *Supervised Learning*

   Metode *supervised* melibatkan penggunaan atribut *output* yang telah ditentukan sebelumnya. *Output* akan diprediksi untuk mengukur akurasi dan performa algoritma berdasarkan jumlah prediksi yang benar atau salah. Proses pelatihan akan dihentikan setelah algoritma mencapai performa yang memuaskan. Secara teknis, algoritma *supervised* akan menganalisis data pelatihan dan membangun hubungan antara *output* dengan *input* untuk membangun fungsi pemetaan agar dapat memprediksi data baru [44].

2. *Unsupervised Learning*

   Sebaliknya, metode *unsupervised* melibatkan pengenalan pola tanpa penggunaan label. Semua data yang digunakan untuk analisis digunakan sebagai *input* membuatnya cocok untuk teknik *clustering* dan teknik penambangan asosiasi. Teknik *clustering* bekerja menemukan kelompok data tanpa label untuk dilabeli. Sedangkan teknik penambangan asosiasi bertugas mengidentifikasi aturan hubungan antar atribut dengan akurat [44].

2.4 *Web Scraping*

Web scraping adalah teknik mengekstraksi data dari internet secara manual atau otomatis untuk disimpan di dalam suatu berkas atau *database* guna keperluan analisis [46]. Besarnya jumlah di data yang dihasilkan oleh internet telah membuat *web scraping* diakui sebagai teknik yang efisien untuk mengumpulkan data. Untuk dapat diaplikasikan dalam berbagai tugas, *web scraping* telah disesuaikan untuk dapat mengubah situs web menjadi *dataset* yang terstruktur [46].

Web scraping dapat dilakukan beberapa cara:

1. Metode tradisional, di mana individu secara manual menyalin dan menempel (*copy and paste*) data dari sumber ke media lain [47].
2. *HTML Parsing*, memanfaatkan program *wrapper* yang dirancang untuk mengenali pola *template* dari halaman web dan mengekstraksi informasi yang ditampilkan untuk diubah menjadi bentuk relasional yang dapat diolah lebih lanjut [46].
3. *DOM parsing,* memanfaatkan program untuk mengambil konten dengan menjalankan *browser* otomatis seperti Mozilla atau Internet Explorer guna menjalankan skrip klien

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

dan mengubah halaman tersebut menjadi struktur DOM (*Document Object Model*) yang dapat diekstrak datanya [47].

4. Metode API (*Application Programming Services*), merupakan pendekatan yang dapat menyediakan data dalam format terstruktur seperti CSV, JSON, XML, atau Excel secara langsung oleh penyedia layanan melalui mekanisme autentikasi tertentu [16], [48].

Salah satu contoh dari alat *scraping* adalah Google-play-scraper. Google-Play-Scraper adalah sebuah *library* yang digunakan untuk mengambil dan mengekstraksi data di dalam Google Play Store [49]. Dalam *scraping* Google Play Store, data seperti ulasan, penilaian, dan informasi aplikasi akan diekstrak dengan menggunakan sebuah program untuk tugas *text mining*, analisis sentimen, dan klasifikasi [50].

Ulasan pengguna merupakan bagian yang penting dalam pasar aplikasi seluler seperti Google Play Store. Informasi dari ulasan tersebut sangat berguna bagi pengembang untuk meningkatkan kualitas aplikasi. Sayangnya, Google Play Store hanya memiliki alat analitik sederhana seperti histogram penilaian pengguna yang memberikan wawasan ulasan dalam bentuk ringkasan biasa. Ini membuat *scraping* diperlukan untuk mendapatkan data yang berguna [50].

Google-Play-Scraper menyediakan berbagai API untuk memudahkan proses crawling di Google Play Store. Berdasarkan *library* Google-Play-Scraper oleh JoMingyu yang berbasis Python, untuk mengambil data ulasan sebuah aplikasi dapat menggunakan fungsi *reviews* atau *reviews_all* [51].

```python
from google_play_scraper import Sort, reviews

result, continuation_token = reviews(
    'com.fantome.penguinisle',
    lang='en', # defaults to 'en'
    country='us', # defaults to 'us'
    sort=Sort.NEWEST, # defaults to Sort.NEWEST
    count=3, # defaults to 100
    filter_score_with=5 # defaults to None(means all score)
)

# If you pass `continuation_token` as an argument to the reviews function at this point,
# it will crawl the items after 3 review items.

result, _ = reviews(
    'com.fantome.penguinisle',
    continuation_token=continuation_token # defaults to None(load from the beginning)
)
```

Gambar 2.2 Fungsi *reviews* [51]

Dari Gambar 2.2, fungsi *reviews* akan mengembalikan nilai *result* dan *continuation_token*. Variabel *result* akan berisi data hasil *crawling* ulasan, sedangkan

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

continuation_token merupakan token *pagination* yang digunakan untuk melanjutkan proses pengambilan ke halaman berikutnya. Jika *continuation_token* digunakan kembali sebagai parameter pada pemanggilan fungsi *reviews* maka program akan melanjutkan proses *crawling* ke ulasan berikutnya tanpa mengulang data yang telah diambil [51].

Berikut parameter yang digunakan dalam fungsi *reviews*:

1. *app_id*, merupakan ID dari aplikasi Google Play Store yang akan di *scrap* (pada Gambar 2.2, *app_id* yang digunakan adalah "com.fantome.penguinisle").
2. lang, menentukan bahasa dari ulasan yang akan diambil.
3. *country*, menentukan negara Google Play Store yang akan diakses.
4. *sort*, menentukan urutan pengambilan ulasan.
5. *count*, menentukan jumlah ulasan yang akan diambil dalam satu kali permintaan.
6. *filter_score_with*, membatasi jenis ulasan berdasarkan penilaian.
7. *continuation_token*, digunakan untuk melanjutkan proses *crawling* berdasarkan token yang dihasilkan sebelumnya.

Fungsi *reviews_all* hampir mirip dengan fungsi *reviews*, perbedaannya terletak pada mekanismenya di mana fungsi *reviews_all* akan secara otomatis melakukan pemanggilan ulang menggunakan *continuation_token* hingga seluruh ulasan berhasil diambil [51].

2.5 *Data Labeling*

Data Labeling merupakan proses yang merujuk pada kegiatan seperti pembersihan data, klasifikasi, perbandingan, penandaan, dan verifikasi yang dilakukan untuk membuat *dataset* pelatihan bagi model *machine learning* agar dapat menghasilkan *output* yang cerdas [52].

Dalam pelatihan model *unsupervised*, penting untuk melabeli *dataset* yang digunakan untuk menentukan kategori yang sesuai kepada sebuah data (positif, netral, atau negatif) [53]. *Labeling* memudahkan model dalam proses klasifikasi karena input tekstual akan diubah menjadi bentuk numerik membuat data tekstual harus dilabeli [49].

Tugas NLP memiliki tiga pendekatan utama untuk *data labeling*, sebagai berikut [54]:

1. *Manual Labeling*, memanfaatkan tenaga manusia untuk memberi label pada data. Ini memerlukan pemahaman yang dalam terhadap konteks dan tujuan pelabelan.
2. *Automatic Labeling*, memanfaatkan algoritma atau model yang sudah dilatih untuk mengotomatisasi proses pelabelan. Metode yang banyak digunakan seperti algoritma *clustering* dan metode berbasis aturan.

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

3. *Semi-Automatic Labeling*, menggabungkan metode manual dan otomatis di mana model otomatis digunakan untuk memberi label, sedangkan manusia akan mengecek akurasi label.

Pendekatan tersebut dapat dilakukan dengan metode berikut:

1. Metode *crowd-based*, memanfaatkan kecerdasan dan upaya dari pekerja manusia untuk memberi label pada data. Metode ini dibagi menjadi dua jenis, *active learning* dan metode *crowdsourcing* [55].
   a. *Active Learning*, merupakan bagian dari *machine learning* di mana algoritma tersebut akan memilih data yang akan digunakan untuk pelatihan. Tujuan utama dari teknik ini adalah untuk menemukan contoh data yang lebih informatif untuk *labeling*, sehingga dapat mengurangi jumlah data berlabel yang diperlukan untuk meningkatkan performa model [55].
   b. *Crowdsourcing*, teknik ini melibatkan pembagian tugas *labeling* kepada pekerja manusia lainnya dalam jumlah yang besar, khususnya secara *online*. Metode ini dapat mengatasi kesulitan yang dihadapi jika tugas dilakukan secara individu [55].
2. *Weak Supervision*, merupakan pendekatan yang berfokus dalam generasi data berlabel dalam jumlah yang besar tapi menggunakan metode *labeling* yang kurang teliti. Metode ini melibatkan penggunaan teknik otomatisasi seperti *data programming atau fact extraction* untuk menghasilkan *label* lemah yang nantinya diperkuat dan digunakan untuk pelatihan [55].
   a. *Data Programming*, merupakan teknik dasar dari *weak supervision* dengan memanfaatkan banyak fungsi *labeling* untuk memberi label lemah pada data. Data tersebut akan digabungkan menggunakan model generatif untuk mengurangi *noise*, sebelum dilatih ke model *machine learning* dengan model diskriminatif untuk menghasilkan prediksi akurat [55].
   b. *Fact Extraction*, melibatkan ekstraksi informasi terstruktur dari data web seperti dokumen atau halaman web dengan menggunakan NLP, penyesuaian pola, dan *machine learning* untuk mengidentifikasi atribut dan hubungan yang dapat dimanfaatkan untuk membuat sebuah pengetahuan umum [55].
3. *Lexicon*, adalah korpus dengan bidang spesifik yang berisi kumpulan kata atau frasa yang merepresentasikan suatu label atau topik. Setiap label memiliki *lexicon* sendiri yang berfungsi sebagai sumber pengetahuan untuk mengklasifikasi teks. Dalam *labeling*, *lexicon* digunakan untuk mencocokkan kata-kata dalam dokumen dengan data yang ada

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

pada setiap *lexicon* label. Semakin sering sebuah kata muncul *dilexicon* suatu label, semakin besar kemungkinan relevansi label untuk dokumen tersebut [56].

2.6 *Valence Aware Dictionary and Sentiment Reasoner* (VADER)

VADER adalah model analisis sentimen berbasis aturan dan *lexicon* yang dirancang untuk memahami sentimen dalam teks pendek dan tidak baku seperti di media sosial. Model ini memiliki performa yang lebih tinggi dibandingkan model analisis sentimen terkemuka lainnya seperti LIWC, ANEW, GI, dan yang lainnya [57].

Model ini dibuat karena *lexicon* lama seperti LIWC, GI, atau ANEW umumnya kurang lengkap untuk menganalisis bahasa media sosial yang banyak berisi slang, emotikon, atau akronim. Selain itu, *lexicon* tersebut juga tidak mampu menangkap intensitas dari suatu sentimen. Dengan VADER, analisis sentimen dapat dilakukan dengan lebih akurat karena *lexicon* yang telah dilengkapi untuk penggunaan di media sosial dan kemampuannya untuk mengenali penekanan emosi dalam kalimat [57].

VADER menggunakan 5 aturan gramatikal dan sintaksis untuk mengatur perubahan intensitas dalam sentimen, aturan tersebut adalah [57]:

1. Tanda baca, terutama "!" menunjukkan peningkatan intensitas emosi tanpa mengubah urutan semantik. Contoh, kalimat "Makanan ini enak!!!" lebih intens dibandingkan "Makanan ini enak".
2. Kapitalisasi, menekankan kata yang lebih relevan secara sentimen di antara kata yang tidak dikapitalisasi. Contoh, kalimat "Makanan ini ENAK!" memiliki sentimen lebih intens dibandingkan "Makanan ini enak!".
3. *Degree Modifier*, untuk meningkatkan atau menurunkan intensitas sentimen berdasarkan penggunaan kata. Kalimat seperti "Layanan di sini sangatlah bagus" memberi lebih banyak intensitas daripada "Layanan di sini bagus". Sebaliknya, kalimat seperti "Layanan disini baik saja" menurunkan intensitas sentimen.
4. Konjungsi kontras, kata "tapi" menunjukkan perubahan polaritas sentimen untuk kalimat setelahnya. Kalimat seperti "Makanan di sini enak, tapi layanannya buruk" memiliki sentimen ganda di mana bagian akhir dari kalimat ("...layanannya buruk") akan menentukan hasil akhir sentimen.
5. Negasi, berperan membalikkan atau mengurangi polaritas jika kata seperti "bukan" atau "tidak" muncul dalam kalimat. Contoh kalimat yang telah dinegasi "Makanan di sini sebenarnya tidak begitu enak".

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

Untuk setiap kata dalam kalimat, VADER akan mengambil skor dasar dari *lexicon* dan menerapkan kelima aturan tersebut untuk mengatur valensi skor akhir. Setelah semua kata diberi skor, semua skor kata akan dijumlahkan untuk menghasilkan nilai sentimen untuk keseluruhan kalimat. Setelah mendapatkan jumlah total valensi, VADER akan melakukan normalisasi untuk menghasilkan skor *compound* yang menunjukkan intensitas sentimen dengan skor antara -1 dan +1. Proses normalisasi bisa ditulis dalam rumus (1) dimana α adalah konstanta skala dengan nilai bawaan sebesar 15.

$$compound = \frac{\sum valence}{\sqrt{\sum valence^2 + \alpha}} \quad (1)$$

Keterangan:

compound : skor compound

$\sum valence$: total valensi

α : konstanta skala

Selain *compound*, VADER juga menghasilkan tiga skor proporsi berupa pos (proporsi kata positif), *neu* (proporsi kata netral), dan *neg* (proporsi kata negatif) untuk menunjukkan sentimen yang dominan dalam kalimat [57], [58]. Contoh pemberian skor dapat dilihat melalui Gambar 2.3.

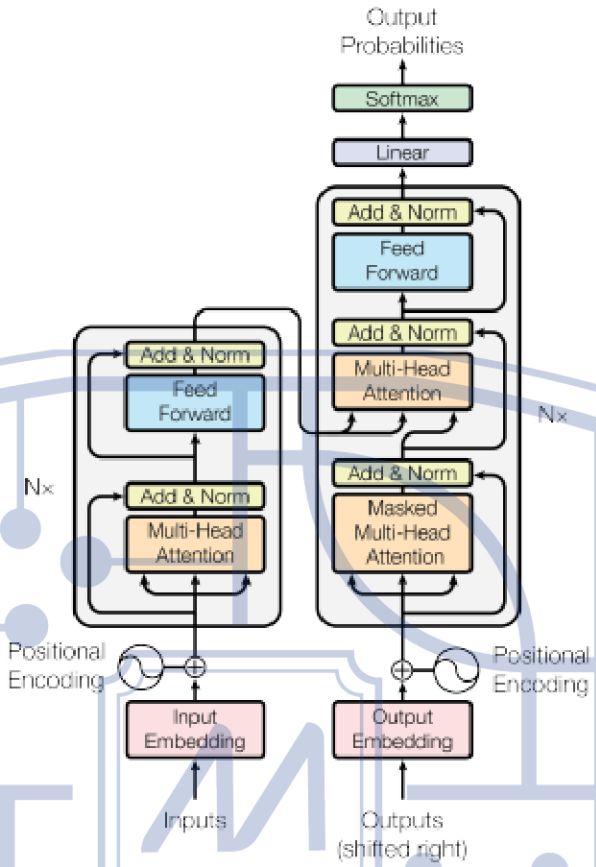
```
VADER is smart, handsome, and funny.---------------------------- {'pos': 0.746, 'compound': 0.8316, 'neu': 0.254, 'neg': 0.0}
VADER is smart, handsome, and funny!---------------------------- {'pos': 0.752, 'compound': 0.8439, 'neu': 0.248, 'neg': 0.0}
VADER is very smart, handsome, and funny.----------------------- {'pos': 0.701, 'compound': 0.8545, 'neu': 0.299, 'neg': 0.0}
VADER is VERY SMART, handsome, and FUNNY.----------------------- {'pos': 0.754, 'compound': 0.9227, 'neu': 0.246, 'neg': 0.0}
VADER is VERY SMART, handsome, and FUNNY!!!---------------------- {'pos': 0.767, 'compound': 0.9342, 'neu': 0.233, 'neg': 0.0}
VADER is VERY SMART, uber handsome, and FRIGGIN FUNNY!!!--------- {'pos': 0.706, 'compound': 0.9469, 'neu': 0.294, 'neg': 0.0}
VADER is not smart, handsome, nor funny.------------------------ {'pos': 0.0, 'compound': -0.7424, 'neu': 0.354, 'neg': 0.646}
The book was good.---------------------------------------------- {'pos': 0.492, 'compound': 0.4404, 'neu': 0.508, 'neg': 0.0}
At least it isn't a horrible book.------------------------------ {'pos': 0.363, 'compound': 0.431, 'neu': 0.637, 'neg': 0.0}
The book was only kind of good.--------------------------------- {'pos': 0.303, 'compound': 0.3832, 'neu': 0.697, 'neg': 0.0}
The plot was good, but the characters are uncompelling and the dialog is not great. {'pos': 0.094, 'compound': -0.7042, 'neu': 0.579, 'neg': 0.327}
Today SUX!------------------------------------------------------ {'pos': 0.0, 'compound': -0.5461, 'neu': 0.221, 'neg': 0.779}
Today only kinda sux! But I'll get by, lol---------------------- {'pos': 0.317, 'compound': 0.5249, 'neu': 0.556, 'neg': 0.127}
Make sure you :) or :D today!----------------------------------- {'pos': 0.706, 'compound': 0.8633, 'neu': 0.294, 'neg': 0.0}
Catch utf-8 emoji such as ♡ and ✍ and 😁------------------------ {'pos': 0.279, 'compound': 0.7003, 'neu': 0.721, 'neg': 0.0}
Not bad at all-------------------------------------------------- {'pos': 0.487, 'compound': 0.431, 'neu': 0.513, 'neg': 0.0}
```

Gambar 2.3 Contoh Hasil Skor Kalimat [58]

2.7 *Bidirectional Encoder Representations from Transformers* (BERT)

BERT adalah model representasi bahasa berbasis arsitektur *transformer* yang berhasil merevolusi cara pemrosesan bahasa alami (NLP) secara signifikan membuatnya menjadi opsi populer sebagai topik penelitian dan penggunaan industri [19]. BERT merupakan teknik penting yang berhasil memuaskan berbagai tugas NLP dengan menggunakan satu arsitektur model dasar [59]. Arsitektur dari model tersebut dapat diilustrasikan melalui Gambar 2.4.

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.



Gambar 2.4 Arsitektur *Transformer* [60]

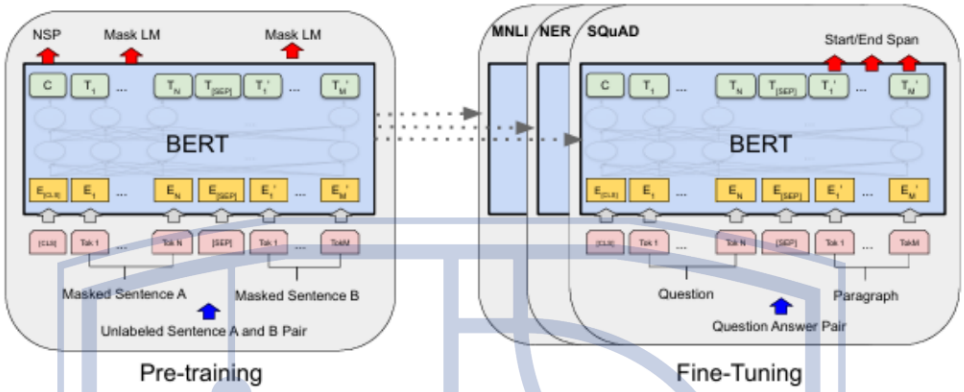
Kemampuan BERT dalam memahami hubungan kontekstual telah menetapkan standar baru dalam aplikasi NLP. Dalam proses *embedding*, BERT dapat mencerminkan makna kata berdasarkan penggunaan kontekstual, membuatnya lebih unggul dibandingkan *embedding* statis seperti Word2Vec atau GloVe.

Pelatihan *bidirectional* dan pemodelan bahasa yang menggunakan *mask* memungkinkan BERT memahami konteks dengan lebih dari satu arah membuatnya lebih akurat dibandingkan dengan model terdahulu yang bersifat *unidirectional*. Kemampuan ini menetapkan standar baru dalam aplikasi NLP menghasilkan performa tinggi pada tugas seperti bertanya jawab atau pemahaman komunikasi [19]. Hampir semua organisasi yang bekerja pada produk kebahasaan telah berfokus pada penggunaan BERT untuk kinerja terbaik karena model tersebut sudah digunakan ke dalam berbagai sistem AI seperti mesin pencarian, layanan tanya jawab, dan generasi teks untuk meningkatkan kemampuan bahasa alami secara signifikan [59].

Penggunaan BERT terdiri dari *pre-training* dan *fine-tuning*, seperti yang ditunjukkan di Gambar 2.5. Dalam *pre-training*, model akan dilatih dengan data tidak berlabel untuk

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

setiap tugas yang berbeda. Sedangkan untuk *fine-tuning*, model akan diinisialisasi dengan parameter yang telah dilatih dan disetel sebelumnya dengan data berlabel hasil *downstream* yang terpisah [61].

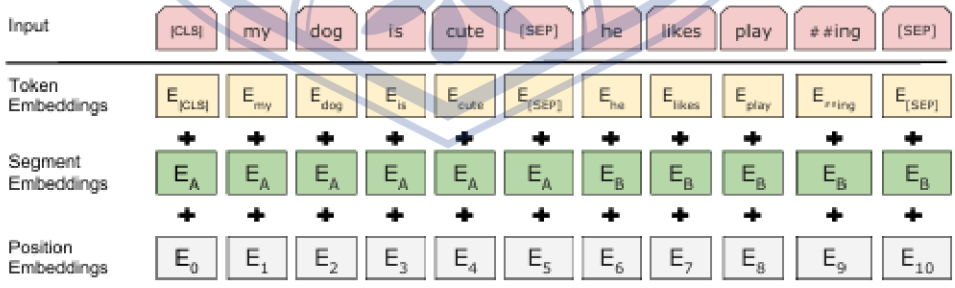


Gambar 2.5 Prosedur BERT dalam *Pre-Training* dan *Fine-Tuning* [61]

Berdasarkan Gambar 2.5, Token pertama untuk setiap urutan selalu merupakan sebuah token klasifikasi atau [CLS]. Pasangan kalimat digabung bersama menjadi satu urutan yang akan dibedakan melalui dua tahap. Pertama, dengan memisahkannya menggunakan token [SEP]. Selanjutnya, setiap token akan ditambahkan *embedding* yang telah dipelajari model dari pelatihan untuk menandakan dari mana kalimat tersebut berasal [61].

2.7.1 Representasi *Input*

Pertama kali, representasi *input* BERT dibuat agar mampu merepresentasi satu pasang kalimat atau dua pasang kalimat dalam satu urutan token untuk membantu menangani berbagai tugas *downstream* [61].



Gambar 2.6 Representasi *Input* BERT [61]

Berdasarkan Gambar 2.6, Token pertama untuk setiap urutan selalu merupakan sebuah token klasifikasi atau [CLS]. Pasangan kalimat digabung bersama menjadi satu urutan yang akan dibedakan melalui dua tahap. Pertama, dengan memisahkannya

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

menggunakan token [SEP]. Selanjutnya, setiap token akan ditambahkan *embedding E* yang telah dipelajari model dari pelatihan untuk menandakan dari mana kalimat tersebut berasal [61]. Cara *embedding* BERT dapat dirumuskan melalui persamaan (2) [59], [62].

$$E_{input} = E_{token} + E_{position} + E_{segment} \quad \text{.....(2)}$$

Keterangan:

E_{input} : representasi embedding

E_{token} : token embedding

$E_{position}$: posisi embedding

$E_{segment}$: segment embedding

Setelah hasil representasi *input* didapatkan, maka akan dilakukan normalisasi *layer* seperti yang dirumuskan melalui persamaan (3).

$$h_i = f\left(\frac{g_i}{\sigma_i}(a_i - \mu_i) + b_i\right) \quad \text{.....(3)}$$

Keterangan:

h_i : output neuron

f : fungsi aktivasi

g_i : parameter skala

σ_i : simpangan baku

a_i : nilai masukan

μ_i : nilai rata-rata

b_i : parameter bias

Dimana a_i sebagai input akan dinormalisasikan ke neuron melalui skalar μ dan σ, sekaligus mempelajari bias adaptif b dan peningkatan g untuk setiap neuron setelah normalisasi selesai [63]. Hasil normalisasi kemudian dapat diproses menggunakan *encoder* BERT dengan mekanisme perhatian diri untuk menghasilkan representasi kontekstual setiap token yang mempengaruhi makna token di sekitarnya [64]. Setelah itu, dilakukan *pooling* untuk menggabungkan seluruh representasi token menjadi satu vektor yang merepresentasikan seluruh kalimat [65]. Salah satu strategi *pooling* adalah *mean pooling* yang menghitung representasi rata-rata dari seluruh vektor sehingga hasil representasi menjadi lebih stabil seperti dirumuskan di persamaan (4).

$$z_{mean} = \frac{1}{N}\sum_t h_t^e \quad \text{.....(4)}$$

Keterangan:

z_{mean} : vektor representasi kalimat

N : jumlah token

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

$\sum_t$: penjumlahan vektor token t

h_t^e : representasi token t

Dimana *N* merupakan jumlah token dan h_t^e adalah vektor tersembunyi dari *encoder* pada langkah ke *t* [66].

2.7.2 *Pre-training*

Pre-training BERT tidak menggunakan model bahasa tradisional yang bersifat satu arah, melainkan menggunakan dua tugas *unsupervised* berikut:

1. *Masked LM* (MLM)

   Prosedur MLM bertujuan untuk melatih representasi secara dua arah dengan menyembunyikan (*mask*) beberapa kata dari *input* dengan token [MASK] untuk diprediksi nantinya. Ini dikarenakan pelatihan model secara dua arah memungkinkan setiap kata untuk "melihat dirinya sendiri" secara tidak langsung sehingga dapat dengan mudah memprediksi target kata di dalam beberapa lapisan. Kekurangan dari prosedur ini adalah token [MASK] tidak akan muncul dalam proses *fine-tuning*. Ini bisa menyebabkan ketidakcocokan antara *pre-training* dan *fine-tuning*. Untuk mencegahnya, kata yang disembunyikan tidak akan selalu diganti dengan token [MASK], melainkan melalui probabilitas berikut: kata akan diubah dengan token [MASK] dengan kemungkinan 80%, kata akan diganti menjadi kata acak dengan kemungkinan 10%, atau kata tidak akan dilakukan perubahan dengan kemungkinan 10% [61].

2. *Next Sentence Prediction* (NSP)

   Untuk melatih model agar dapat memahami hubungan antar kalimat, model BERT dilatih dengan tugas NSP terbinarisasi dari sebuah korpus satu bahasa. Dalam pelatihan, BERT memerlukan dua kalimat (misalnya kalimat A dan kalimat B) untuk melatih model apakah kalimat B memiliki hubungan dengan kalimat A. *Dataset* pelatihan akan dibentuk sebagai berikut:

   a. 50% kalimat B merupakan lanjutan sebenarnya dari kalimat A yang ada di korpus (dilabeli IsNext).

   b. Sisa 50% kalimat B merupakan kalimat acak dari korpus (dilabeli NotNext).

   Dalam prosedur *pre-training*, korpus yang digunakan berupa BooksCorpus dengan 800 juta kata dan Wikipedia berbahasa Inggris dengan 2,500 juta kata. Penting juga untuk menggunakan korpus tingkat dokumen untuk dapat mengekstraksi urutan yang lebih panjang [61].

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

2.7.3 *Fine-tuning*

Fine-tuning BERT dapat mudah dilakukan karena mekanisme perhatian diri *Transformer*, memungkinkan pemodelan tugas *downstream* yang banyak dengan menggantikan *input* dan *output* yang sesuai. Setiap tugas dilakukan dengan memasukkan *input* dan *output* dengan tugasnya tertentu ke BERT dan menyetel parameter di akhirnya. Bagian *input* diberikan kalimat A dan kalimat B dari *pre-training* yang dapat berupa pasangan kalimat yang dibandingkan kesamaannya dari proses parafrase, pasangan hipotesis dan premis, pasangan pertanyaan dan jawaban, dan pasangan di mana satu kalimat berisi teks dan kalimat lainnya kosong dalam klasifikasi teks. Untuk bagian *output*, representasi token diberikan kepada lapisan *output* untuk tugas tingkat token, seperti penandaan urutan atau tanya jawab dan representasi [CLS] digunakan untuk tugas klasifikasi seperti analisis sentimen [61]. Menurut Edwin, penelitiannya mengoptimasi klasifikasi gambar menyatakan bahwa *deep learning* memiliki tiga parameter yang penting dalam mengukur *dataset* pelatihan [67]:

1. *Epoch*, mengatur beberapa kali sebuah model akan memproses semua sampel yang ada di *dataset* dan memperbarui parameter berdasarkan kerugian yang dihitung.
2. *Learning Rate*, mengatur seberapa sering jaringan neural memperbarui ilmu yang dipelajari.
3. *Batch Size*, mengatur jumlah sampel latihan yang harus diproses sebelum parameter diperbarui.

Dalam pelatihan, model akan menghasilkan *output logits* $X_{b,i}$ dari lapisan *output* yang belum diubah menjadi probabilitas. Untuk memperjelas *logits* sebagai probabilitas setiap kelas $Y_{\{b,i\}}$, digunakan fungsi *softmax* yang bisa dirumuskan dengan persamaan (5) dimana $\sum_{j=1}^{K}$ adalah denominator, *e* adalah angka euler, *b* adalah *batch* serta *i* dan *j* sebagai indeks kelas. [68].

$$Y_{\{b,i\}} = \frac{e^{X_{b,i}}}{\sum_{j=1}^{K} e^{X_{b,j}}} \quad \text{.....(5)}$$

Keterangan:

$Y_{\{b,i\}}$: probabilitas kelas ke-i

$X_{b,i}$: nilai logits kelas ke-i

K : jumlah kelas

e : konstanta euler

i, j : indeks kelas

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

b : batch

Selisih dari prediksi model dan label asli kemudian akan dihitung dengan menggunakan fungsi *cross-entropy loss* $-\sum_{i=1}^{K}$ seperti di persamaan (6) dengan y^0 sebagai distribusi asli, y sebagai distribusi yang diprediksi, dan *i* sebagai indeks kelas [68].

$$CE(y^0, y) = -\sum_{i=1}^{K} y_i^0 \log y_i \quad \text{.....(6)}$$

Metode *backpropagation* dapat digunakan untuk mengoptimasi model dengan mempropagasi kesalahan *output* ke *layer* sebelumnya untuk diperbarui bobotnya. Gradien dari fungsi *loss* terhadap nilai *logits* akan dihitung dengan rumus (7) dimana X adalah nilai *input*, Y adalah nilai *ouput*, dan Y^0 adalah target hasil [68].

$$X = Y - Y^0 \quad \text{.....(7)}$$

Keterangan:

X : nilai input

Y : nilai output

Y^0 : target hasil

Nilai *gradien* $\nabla L(W, b)$ selanjutnya akan digunakan bersama dengan metode optimasi *AdamW* untuk memperbarui parameter bobot *W* dan bias *b* seperti ditunjukkan di persamaan (8) dan (9) dengan α sebagai *learning rate* [69].

$$W := W - \alpha \nabla L(W, b) \quad \text{.....(8)}$$

Selama proses *backpropagation*, rumus akan memperbarui bias *b* dengan menggunakan persamaan (9) [69].

$$b := b - \alpha \nabla L(W, b) \quad \text{.....(9)}$$

Keterangan (8) dan (9):

W : bobot model

b : bias model

$\nabla L(W, b)$: gradien fungsi kerugian

α : learning rate

Proses pembaruan parameter akan dilakukan secara berulang di setiap *epoch* hingga model mencapai performa terbaiknya untuk *dataset* yang ditentukan.

2.8 *Topic Modeling*

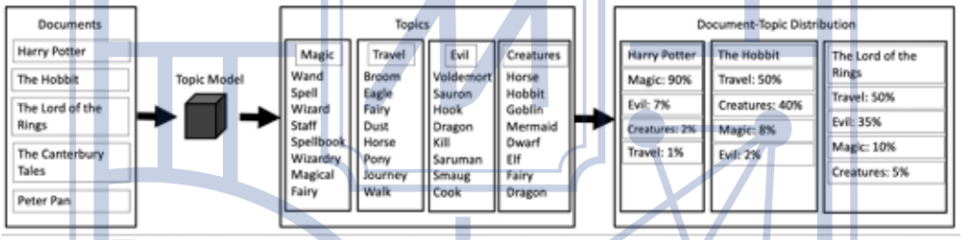
Model topik adalah metode *unsupervised machine learning* yang digunakan untuk mengidentifikasi tema utama dari suatu dokumen dengan merangkum sekumpulan kalimat menjadi ringkasan singkat yang disebut sebagai topik. Metode ini umum dirancang untuk mengklasifikasi dokumen seperti buku, makalah penelitian, dan artikel berita. Tema utama

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

dari suatu dokumen dapat diidentifikasi oleh model topik dengan mengenali pola kata yang berulang dan mengelompokkannya ke dalam topik yang mencerminkan isi dari dokumen [70].

Memahami tema utama dari sebuah koleksi dokumen telah menjadi tugas yang penting di era informasi saat ini [70]. Dengan meningkatnya jumlah data elektronik yang ada, diperlukan teknik yang mampu meringkas dan memahami informasi dalam jumlah yang besar [71]. Model topik menjadi teknik yang cocok untuk tugas ini karena dapat menyingkatkan ribuan dokumen menjadi sebuah ringkasan pendek sekaligus menangkap poin umum yang ada di dalam korpus. Ini membuatnya banyak digunakan dalam berbagai bidang yang perlu memahami teks yang berbeda, mulai dari buku hingga unggahan *tweets* [70].

Menurut definisi Churchill, model topik adalah model matematis *unsupervised* yang menerima sekumpulan dokumen sebagai *input* dan mengembalikan topik yang merepresentasikan isi dari dokumen dengan akurat [70].



Gambar 2.7 Diagram Proses Topik Model [70]

Berdasarkan Gambar 2.7 dalam melakukan pemodelan topik, dokumen ditentukan sebagai *input*. Kemudian, algoritma pemodelan topik akan mengembalikan serangkaian topik yang berhubungan dengan dokumen-dokumen yang di *input*. Dokumen-dokumen kemudian dilabeli dengan topik sehingga pengguna dapat memahami pentingnya topik di dokumen tersebut. Model topik mengidentifikasi sebuah tema dengan mengenali pola kata berulang yang kemudian dikelompokkan menjadi sebuah topik yang mencerminkan isi dokumen. Untuk menghasilkan hasil yang akurat dan konsisten, model topik bergantung pada pengulangan pola kata yang bisa diprediksi dan sering muncul dalam dokumen. Semakin tinggi prediksi pola kata, semakin berkualitas topik yang dihasilkan oleh model topik [70].

2.9 BERTopic

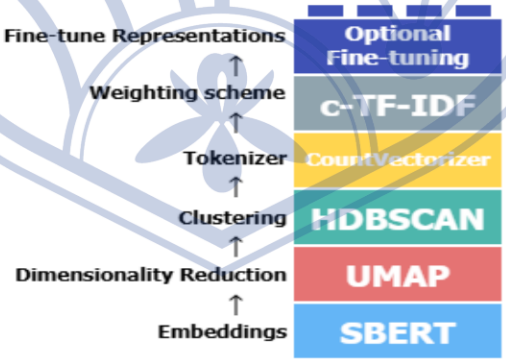
BERTopic adalah sebuah model topik yang digunakan untuk menemukan topik utama dari suatu dokumen yang memperluas pemodelan topik berbasis *clustering* dengan

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

menggunakan variasi dari TF-IDF berbasis kelas untuk mengekstrasi topik yang koheren [21]. Model ini dibangun berdasarkan cara kerja Top2Vec dan menggunakan BERT sebagai *embedder* dan dapat mengekstrak hasil *embedding* dokumen dengan menggunakan transformer kalimat yang mendukung lebih dari 50 bahasa [22].

Kemampuan BERTopic untuk tetap kompetitif terlepas dari model bahasa yang digunakan membuatnya konsisten untuk digunakan dalam berbagai konteks. Misalnya, pengguna dapat menggunakan Doc2Vec ketika akses ke GPU tidak tersedia. Pemisahan proses *embedding* dari representasi topik membuat BERTopic menjadi lebih fleksibel. Ini memungkinkan cara prapemrosesan yang berbeda untuk digunakan ketika menyematkan dokumen dan menghasilkan topik representasi. Penggunaan TF-IDF berbasis kelas membuat topik dapat direpresentasikan dengan berbagai kelas sebagai distribusi kata. Ini memungkinkan BERTopic untuk menggambarkan sifat topik yang dinamis dan berkembang tanpa perubahan yang signifikan [21].

Berdasarkan Gambar 2.8, BERTopic menghasilkan representasi topik melalui beberapa langkah. Pertama, dengan menggunakan model bahasa yang telah dilatih, setiap dokumen diubah akan menjadi representasi *embedding*. Selanjutnya, model dapat menggunakan UMAP untuk mengurangi dimensi dari *embedding* untuk mengoptimalisasi proses *clustering* yang akan dilakukan menggunakan HDBSCAN. Setelah itu, hasil *clustering* akan dilakukan vektorisasi dengan CountVectorizer. Terakhir, TF-IDF berbasis kelas akan mengekstraksi topik representasi dari kumpulan *embedding* dokumen yang telah diproses [21], [72].



Gambar 2.8 Alur kerja BERTopic [72]

1. *Embedding* Dokumen

   Embedding dokumen dilakukan untuk menciptakan sebuah representasi vektor yang bisa dibandingkan secara semantik dengan asumsi bahwa dokumen yang berisi topik serupa dapat mempunyai kesamaan semantik. Untuk melakukan *embedding*, BERTopic

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

menggunakan *Sentence-BERT* (SBERT) sebagai *framework* untuk mengubah kalimat menjadi representasi vektor dengan model bahasa yang sudah dilatih [21].

2. Pengurangan Dimensi

Sebelum melakukan *clustering*, dimensi *embedding* umumnya diperkecil. Ini dikarenakan semakin besar dimensi sebuah data, jarak dari suatu poin data ke lainnya ikut melebar. Akibatnya, perbedaan jarak relatif menjadi kurang berarti yang membuat proses *clustering* menjadi tidak efektif. BERTopic menggunakan metode UMAP untuk mengurangi dimensi karena dapat mempertahankan fitur dari data berdimensi besar ke dalam dimensi yang kecil lebih banyak dibandingkan metode lainnya. Selain itu, UMAP dapat digunakan dalam berbagai model bahasa dengan ruang dimensi yang berbeda [21]. Proses UMAP umumnya mencakup dua tahap yaitu pembentuk *graf* berbobot dan optimasi *graf* [73].

UMAP membentuk *graf* berbobot dengan memanfaatkan jarak antar data melalui kernel berbasis parameter ρ dan σ untuk menggambarkan hubungan lokal. Probabilitas hubungan dalam graf diturunkan dari probabilitas lokal baik pada ruang berdimensi tinggi maupun rendah. Peluang hubungan lokal antara data *i* dan *j* dalam dimensi tinggi dapat ditetapkan melalui persamaan (10) [73].

$$P_{i,j} = \exp\left(-\frac{\max(0, d(x_i x_j) - \rho_i}{\sigma_i}\right) \quad \text{.....(10)}$$

Keterangan:

$P_{i,j}$: probabilitas titik i dan j berdimensi tinggi

$d(x_i x_j)$: jarak antara titik data i dan j

ρ_i : paramater konektivitas

σ_i : parameter skala lokal

Dalam persamaan (8), $d(x_i x_j)$ merepresentasikan jarak antara data x_i dan x_j. Nilai ρ_i menunjukkan jarak minimum yang memastikan setiap titik memiliki setidaknya satu tetangga, sedangkan σ_i berfungsi sebagai parameter skala untuk menentukan keterhubungan lokal. Selain itu, terdapat probabilitas q yang menggambarkan hubungan antara data i dan j merupakan probabilitas dalam ruang berdimensi rendah setelah data direduksi menggunakan UMAP. Probabilitas tersebut bisa dijelaskan dalam persamaan (11) [73].

$$q_{i,j} = \frac{1}{1 + a\|y_i - y_j\|^{2b}} \quad \text{.....(11)}$$

Keterangan:

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

$q_{i,j}$: probabilitas titik i dan j berdimensi rendah

y_i, y_j : koordinat titik i dan j

a : parameter pengatur kurva fungsi keanggotaan

b : parameter pengonrol laju penurunan nilai hubungan jarak

Nilai $q_{i,j}$ digunakan untuk menunjukkan kesamaan hubungan antar data di ruang rendah dengan mempertimbangkan bahwa posisi data ruang rendah bisa dibuat lebih berjauhan dibandingkan dengan data ruang berdimensi tinggi. Titik data y_i dan y_j merepresentasikan posisi data *i* dan *j* dalam ruang berdimensi rendah setelah melalui proses *embedding*. Parameter a dan b berfungsi untuk mengatur distribusi jarak pada ruang dimensi rendah [73].

Setelah itu, UMAP akan memetakan data ke ruang rendah *Y* dengan mengurangi perbedaan antara *graf* yang terbentuk di ruang berdimensi tinggi *X* dan *graf* di ruang berdimensi rendah *Y* seperti dalam persamaan (12) [73].

$$C(Y) = \sum_{i<j}\left[P_{i,j} log \frac{p_{i,j}}{q_{i,j}} + \left(1 - p_{i,j}\right) log \frac{1-p_{i,j}}{1-q_{i,j}}\right] \quad (12)$$

Keterangan:

$C(Y)$: cross-entropy loss pada data ruang Y

$\sum_{i<j}$: penjumlahan titik i dan j

$p_{i,j}$: probabilitas titik i dan j berdimensi tinggi

$q_{i,j}$: probabilitas titik i dan j berdimensi rendah

Dalam persamaan (10), $P_{i,j}$ adalah probabilitas *graf* berdimensi tinggi, sedangkan $q_{i,j}$ adalah probabilitas *graf* berdimensi rendah [73].

3. *Clustering* Dokumen

Proses *clustering* kemudian dilakukan menggunakan HDBSCAN yang merupakan ekstensi dari DBSCAN menjadi algoritma *clustering* bertingkat untuk mencari kelompok *clustering* dengan kepadatan berbeda. HDBSCAN menggunakan pendekatan *soft-clustering* memungkinkan *noise* untuk diasingkan sehingga dapat mencegah *clustering* dari dokumen yang tidak relevan [21]. Kepadatan dihitung berdasarkan jarak antar titik dan jumlah titik terdekat di sekitar. Perhitungannya dapat dirumuskan dalam persamaan (13) [73].

$$\rho_i = \frac{1}{Volume(B_i)} \quad (13)$$

Keterangan:

ρ_i : densitas lokal

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

B_i : area sekitar titik

Nilai ρ_i menunjukkan densitas lokal pada titik i, sedangkan B_i adalah area sekitar titik tersebut yang ditentukan berdasarkan jarak tertentu. Dalam HDBSCAN, keterhubungan antara titik i dan j diukur berdasarkan kedekatan posisinya. Titik yang tidak memenuhi syarat keterhubungan dan kepadatan akan dikategorikan sebagai *noise*. Sebuah titik umumnya dianggap *noise* apabila kepadatan lokalnya lebih rendah dari batas yang telah ditentukan oleh parameter algoritma. Perhitungan jarak biasanya menggunakan metrik *Euclidean* atau *cosine distance* seperti yang dirumuskan dalam persamaan (14) [73].

$$d(i,j) = \|X_i - X_j\| \quad \text{.....(14)}$$

Keterangan:

$d(i,j)$: jarak antar titik i dan j

X_i, X_j : vektor posisi masing-masing titik i dan j

Nilai $d(i,j)$ merupakan jarak antar titik *i* dan *j*, sedangkan X_i dan X_j adalah vektor posisi masing-masing titik di dalam ruang fitur. Untuk mengidentifikasi kelompok titik yang membentuk *cluster*, HDBSCAN menggunakan pendekatan *hierarchical clustering* berbasis kepadatan dengan membangun struktur pohon turunan yang disebut *mutual reachability distance* untuk menggambarkan hubungan antara titik i dan j. Jarak tersebut dihitung menggunakan persamaan (15) [73].

$$d_{reach}(i,j) = \max\left(core_d istance(i), core_d istance(j), d(i,j)\right) \quad \text{.....(15)}$$

Keterangan:

$d_{\text{reach}}(i,j)$: jarak mutual reachability titik i dan j

$core_d istance(i)$: jarak minimum titik i untuk mendapat tetangga

$core_d istance(j)$: jarak minimum titik j untuk mendapat tetangga

$d(i,j)$: jarak euclidean antara titik i dan j

$d_{\text{reach}}(i,j)$ merupakan jarak *mutual reachability* antara titik i dan j. Selanjutnya, $core_d istance(i)$ dan $core_d istance(j)$ adalah jarak minimum yang diperlukan titik i dan j untuk mendapatkan tetangga terdekat sesuai parameter *min_samples*. Terakhir, $d(i,j)$ merupakan jarak *Euclidean* yang digunakan untuk mengukur jarak antara titik i dan j [73].

4. Vektorisasi

Pada tahap vektorisasi akan dilakukan proses tokenisasi dengan memanfaatkan CountVectorizer yang memecah dokumen menjadi kumpulan token. Token-token ini

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

akan direpresentasi dengan pendekatan *Bag-of-Words* untuk menghitung seberapa sering kemunculan kata [21].

5. Representasi Topik

Representasi topik dibuat berdasarkan dokumen masing-masing *cluster* yang nantinya akan ditentukan sebuah topik. Untuk BERTopic, proses representasi menggunakan TF-IDF yang telah dimodifikasi. TF-IDF adalah metode untuk mengukur seberapa penting sebuah kata terhadap dokumen yang memungkinkan metode ini untuk merepresentasikan topik berdasarkan kepentingan kalimat. Dalam TF-IDF modifikasi, setiap dokumen di dalam sebuah *cluster* akan digabung yang kemudian direpresentasikan sebagai satu dokumen. Selanjutnya, teknik TF-IDF berbasis kelas ini mengukur kepentingan kata di dalam *cluster* daripada dokumen secara individu, yang memungkinkan untuk menghasilkan distribusi topik untuk setiap *cluster*. Selain itu, dengan menggabungkan representasi TF-IDF berbasis kelas dari topik kurang umum dengan topik yang paling mirip, jumlah topik dapat dikurangi berdasarkan angka yang ditentukan pengguna. Untuk tetap mengikuti perkembangan NLP, BERTopic mampu melakukan *fine-tuning* terhadap representasi topik c-TF-IDF dengan model seperti GPT, T5, atau teknik lainnya [21], [72].

Untuk melakukan prosedur TF-IDF ke dalam bentuk *cluster*, setiap dokumen di dalam *cluster* akan dianggap sebagai satu dokumen yang di mana TF-IDF akan diatur untuk menghitung representasi baru tersebut dengan rumus (16) [21].

$$W_{t,c} = tf_{t,c} \cdot \log\left(1 + \frac{A}{tf_t}\right) \quad \text{.....(16)}$$

Keterangan:

$W_{t,c}$: bobot kata pada cluster

$tf_{t,c}$: jumlah kemunculan kata t pada cluster c

A : rata-rata jumlah kata setiap cluster

tf_t : total frekuensi kemunculan kata pada seluruh ckuster

Di rumus (14), *tf* adalah *term frequency* menunjukkan seberapa sering kata *t* muncul di dalam *cluster c*, yang dianggap sebagai satu dokumen besar yang terbentuk dari penggabungan semua dokumen dalam *cluster* tersebut. Selanjutnya, *inverse document frequency* diubah menjadi *inverse class frequency* untuk menghitung seberapa banyak informasi yang diberikan kata *t* terhadap *cluster c*, dengan menghitung logaritma dari rata-rata jumlah kata setiap *cluster A* dibagi dengan total frekuensi kemunculan kata *t*

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

di seluruh *cluster*. Penambahan 1 di dalam logaritma bertujuan untuk menghindari nilai negatif atau tak terdefinisi [21].

2.10 Evaluasi Model

Evaluasi Model bertujuan untuk menilai performa model yang diuji dengan mengukur akurasi, *precision*, *recall*, dan *F1-Score* dari model tersebut. *Confusion matrix* merupakan tabel yang digunakan untuk mengevaluasi kinerja model dengan membandingkan prediksi benar dan salah dari data yang dibanding dan data yang diuji seperti di tabel 2.1[74].

Tabel 2.1 Evaluasi *Confusion Matrix*

Data Asli	Data Prediksi			
	Positive	*Neutral*	*Negative*	Total
Positive	TP_1	FN_{12}	FN_{13}	$\sum B_1$
Neutral	FN_{21}	TP_2	FN_{23}	$\sum B_2$
Negative	FN_{31}	FN_{32}	TP_3	$\sum B_3$
Total	$\sum K_1$	$\sum K_2$	$\sum K_3$	N

Nilai evaluasi *confusion matrix* akan berupa:

1. *True Positive* (TP_1, TP_2, TP_3) menunjukkan data yang diprediksi dengan benar untuk masing-masing kelas.
2. *False Negative* (FN_{xy}) menunjukkan data aktual dengan kelas x tetapi salah diprediksi sebagai kelas y.
3. Total Kolom ($\sum K_1$, $\sum K_2$, $\sum K_3$) menunjukkan total data untuk masing-masing kolom kelas.
4. Total Baris ($\sum B_1$, $\sum B_2$, $\sum B_3$) menunjukkan total data untuk masing-masing baris kelas.
5. N menunjukkan total seluruh data.

Nilai akurasi, *precision*, *recall*, dan *F1-Score* dihitung menggunakan perumusan berikut:

1. Akurasi adalah hasil perbandingan dari data yang diprediksi dengan benar dengan total data dengan menggunakan rumus (17).

$$Accuracy = \frac{TP_1+TP_2+TP_3}{N} \quad (17)$$

2. *Precision* adalah hasil perbandingan dari data yang diprediksi dengan benar dengan data prediksi dengan menggunakan rumus(18).

$$Precision = \frac{TP_x}{\sum K_X} \quad (18)$$

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

3. *Recall* adalah hasil perbandingan dari data yang diprediksi dengan benar dengan data aktual dengan menggunakan rumus (19).

$$Recall = \frac{TP_x}{\sum B_X} \quad (19)$$

4. *F1-Score* adalah rata-rata dari *precision* dan *recall* dengan menggunakan rumus (20).

$$F1 - Score = 2 \times \frac{precision \times recall}{precision + recall} \quad (20)$$

Selain itu, terdapat evaluasi berupa *coherence score* atau metrik c_v yang digunakan untuk mengevaluasi interpretasi topik dari sebuah model. *Coherence score* mengukur kesamaan antara semantik dan keterbacaan kata terhadap setiap topik untuk memberikan dasar dalam menentukan jumlah topik optimal. *Coherence score* berkisar antara 1 yang menandakan coherence sempurna dan 0 yang berarti tidak ada *coherence* [75], [76]. Perhitungan skor c_v dapat ditampilkan berdasarkan persamaan (21).

$$c_v = \frac{1}{|T|} \sum_{t \in T} \frac{1}{|t|^2} \sum_{w_i, w_j \in t, i \neq j} NPMI(w_i, w_j) \quad (21)$$

Keterangan:

c_v : *coherence score*

T : kumpulan topik

t : topik

Wi, Wj : kata dalam topik

NPMI : *Normalized Pointwise Mutual Information*

Dimana *T* adalah sekumpulan topik. *t* menunjukkan topik yang terdiri dari kumpulan kata. *Wi* dan *Wj* merupakan kata dalam topik. Terakhir, NPMI (*Normalized Pointwise Mutual Information*) bertugas mengukur keterhubungan semantik antara pasangan kata [76].

2.11 Google Fit

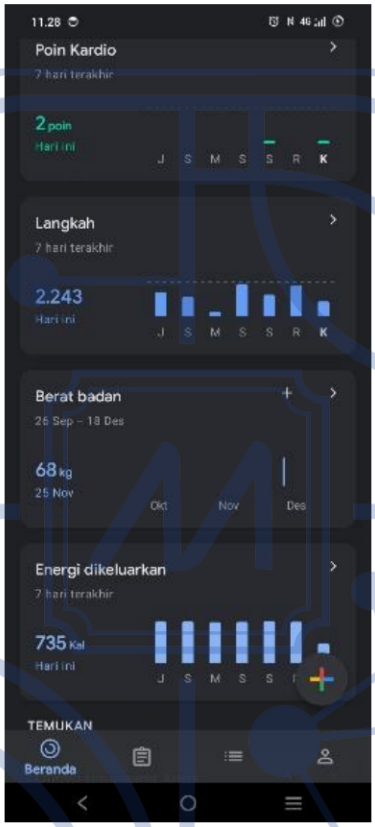
Google Fit adalah aplikasi pelacak kebugaran gratis yang dikembangkan oleh Google LLC untuk membantu memantau aktivitas dan kesehatan pengguna seperti informasi tidur, berat badan, detak jantung, dan laju pernapasan [5], [77], [78]. Aplikasi berjalan di latar belakang perangkat yang bertugas mencatat aktivitas fisik dan memberikan laporan terkait jenis kegiatan yang dilakukan pengguna [77].

Aplikasi ini mudah digunakan dan berpotensi untuk meningkatkan jumlah pengguna pelacak aktivitas. Umpan balik yang diberikan aplikasi dapat meningkatkan tingkat aktivitas fisik yang memungkinkan lebih banyak orang untuk memulai hidup yang lebih aktif.

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

Kemampuan aplikasi untuk memantau aktivitas secara otomatis sangat membantu bagi seseorang yang cenderung lupa untuk menulis aktivitas fisik sehari-hari [77].

Aplikasi ini mampu berjalan secara otomatis di latar belakang perangkat pengguna dan bisa mengenali beragam kegiatan fisik seperti berjalan, bersepeda hingga bermain ski dengan menghitung jumlah langkah dan lamanya aktivitas dilakukan menggunakan sensor seperti *accelerometer*, *gyroscope*, dan GPS.



Gambar 2.9 Tampilan Beranda Google Fit

Aplikasi juga memberikan laporan aktivitas dalam bentuk grafik untuk memudahkan pengguna untuk melihat data seperti jarak yang telah ditempuh, jumlah kalori yang dibakar, kondisi vital seperti detak jantung, dan lainnya seperti yang ditunjukkan di Gambar 2.9 [5], [77], [78]. Untuk melacak setiap aktivitas pengguna, perangkat yang memiliki Google Fit harus dibawa oleh pengguna setiap saat. Pengguna juga bisa menyunting data aktivitas yang telah tercatat. Google Fit bisa disinkronkan dengan aplikasi (seperti aplikasi pengecek nutrisi) atau perangkat (seperti jam tangan) pihak ketiga untuk memantau berbagai jenis kegiatan di satu tempat [77], [78].

Menurut Statista, jumlah pengguna di pasar kesehatan digital di Indonesia dinyatakan mencapai 37,85 juta pengguna pada tahun 2024. Tren yang meningkat menunjukkan pencapaian sebesar 32,49 juta pengguna sejak tahun 2017. Ditambah jumlah pengguna yang

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

juga akan meningkat sebanyak 14,41 juta orang di antara tahun 2024 dan 2029 menunjukkan peningkatan yang konsisten terhadap penggunaan aplikasi kebugaran digital [3].

2.12 Penelitian Terdahulu

Terdapat penelitian terdahulu terkait analisis sentimen yang menggunakan metode BERT dan BERTopic, seperti yang ada di Tabel 2.2.

Tabel 2.2 Penelitian Terdahulu

Sitasi	Peneliti	Tahun	Topik Penelitian	Metode	Hasil
[10]	J. N. Anggiyanti dan F. S. Papilaya	2026	Implementasi Pemodelan Topik BERTopic dan Analisis Sentimen BERT terhadap Ulasan Aplikasi SATUSEHAT	BERT + BERTopic	Analisis sentimen memperoleh akurasi sebesar 95% dengan nilai *coherence topic* 0,6101. Hasil menunjukkan mayoritas opini pengguna bersentimen negatif terutama pada aspek kebergunaan dan aksesibilitas aplikasi.
[11]	M. R. Nur, Y. Wibisono, dan R. Megasari	2025	Analisis sentimen dan pemodelan topik *post* tentang Xiaomi di media sosial X	*Fine-tuning* IndoBERT + BERTopic	Model memperoleh akurasi 79,8% dan menghasilkan 16 *cluster* topik dengan rata-rata *coherence score* sebesar 0,5437.

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

[12]	N. A. Adrielvino dan A. T. Ayunda	2026	Penerapan IndoBERT dan BERTopic dalam ABSA untuk Evaluasi Kualitas Aplikasi *E-Government* Indonesia	IndoBERT + BERTopic + ABSA	Model IndoBERT hasil fine-tuning memperoleh akurasi 91% dan menunjukkan sentimen pengguna cenderung negatif sebesar 62,71%, terutama pada aspek *system availability*, *fulfillment*, dan *privacy*.
[13]	W. M. Sihombing dan T. Widiyangnityas	2025	Analisis sentimen dan pemodelan topik isu kesehatan publik di platform X	IndoBERTweet + Bertopic	Model IndoBERTweet memperoleh *Weighted* F1-Score sebesar 0.7822 dan BERTopic berhasil menghasilkan 44 topik dengan *Coherence Score* 0.7154 serta *Diversity Score* 0.7139. Topik dominan meliputi kesehatan mental, kontroversi vaksinasi anak, penyakit kronis,

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.

					dan laporan penyakit berbasis wilayah.

Berdasarkan Tabel 2.2, Penelitian sebelumnya telah menerapkan metode BERT dan BERTopic pada analisis sentimen media sosial, layanan *e-government*, dan isu kesehatan publik. Namun, belum banyak penelitian yang menerapkan kombinasi metode tersebut pada ulasan pengguna aplikasi kebugaran digital seperti Google Fit di Google Play Store. Selain itu, penelitian sebelumnya belum secara spesifik mengidentifikasi topik-topik pengalaman pengguna terkait fitur pelacakan kesehatan, sinkronisasi perangkat *wearable*, dan performa aplikasi kesehatan berbasis *mobile*.

© Karya Dilindungi UU Hak Cipta
1. Dilarang menyebarluaskan dokumen tanpa izin.
2. Dilarang melakukan plagiasi.
3. Pelanggaran diberlakukan sanksi sesuai peraturan UU Hak Cipta.