Tugas-2-Memperbaiki Performance Analisis Sentimen dengan Transformer
Nama : Alfa Renaldo Aluska
NRP : 5026221**144**

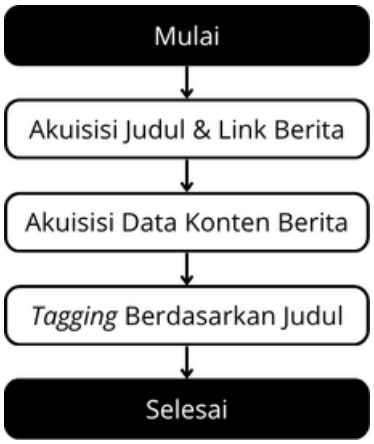
Anda diminta untuk merancang, melatih, dan menganalisis model Transformer dari nol (*from-scratch*) untuk tugas klasifikasi sentimen menggunakan *dataset* anda sendiri.

A. **Persiapan data**. Jelaskan *preprocessing* yang diperlukan (tokenisasi/*subword*, *max length*, *padding*, *train/val/test split*, dsb.)

Jawaban Bagian A:

Menyiapkan *dataset* untuk tugas analisis sentimen adalah langkah krusial dalam proyek NLP, karena kualitas *dataset* sangat memengaruhi performa model. Proses ini mencakup pengumpulan, pembersihan, pelabelan, dan format data sehingga dapat digunakan untuk pelatihan dan evaluasi model.

Tahap akuisisi data merupakan langkah awal dalam penelitian ini yang bertujuan untuk menghimpun, menyiapkan, dan menyusun data berita agar dapat diproses dan dianalisis secara lebih mendalam. Prosedur ini dilaksanakan secara sistematis melalui serangkaian tahapan berurutan guna memastikan bahwa data yang dikumpulkan bersifat relevan, terstruktur, dan siap untuk tahap pra-pemrosesan serta analisis. Secara garis besar, proses akuisisi dimulai dengan pengambilan judul dan tautan berita dari situs resmi daring, yang kemudian diikuti oleh ekstraksi isi berita berdasarkan tautan tersebut. Setelah seluruh konten berhasil dikumpulkan, dilakukan proses penandaan awal (*tagging*) untuk memberikan label kategori pada masing-masing berita, sesuai dengan kerangka penelitian yang telah ditentukan. Rangkaian lengkap tahapan akuisisi data ditampilkan dalam diagram berikut.



Gambar Diagram Alir Tahap Scraping Berita

A.1 Akuisisi Judul dan Tautan Berita

Pada tahap ini, dilakukan proses pengambilan data berita dari Google News secara otomatis menggunakan bahasa pemrograman Python dan pustaka Selenium. Langkah awal melibatkan instalasi sejumlah dependensi penting seperti wget, curl, unzip, serta pustaka Python seperti Selenium, Chromedriver-Autoinstaller, dan Dateparser. Untuk mendukung otomasi peramban dalam mode tanpa tampilan grafis (headless), juga diperlukan instalasi

Google Chrome dan Chromedriver. Pengambilan artikel dibatasi hingga tanggal 30 September 2025, dengan total 1194 artikel berhasil dikumpulkan. *Dataset* yang telah diperoleh kemudian menjalani tahap pra-pemrosesan dan disiapkan untuk analisis, sebagaimana dijelaskan pada tabel berikut.

Tabel Jumlah artikel dalam *dataset* per tahapan

No.		Tahapan	Dihapus	Jumlah Akhir
1	Hanya judul dan tautan berita	Akuisisi judul dan tautan berita	-0	1194
2		Hapus baris *scraping* judul dan tautan berita kosong karena gagal di-*scrap*	-191	1003
3		Filter bahasa Inggris judul berita dengan formula Google Speadsheets, lalu dihapus dengan Python	-389	614
4	Beserta dengan isi konten berita	*Scraping* isi konten berdasarkan tautan yang telah di-*scrap* sebelumnya,	-0	614
5		Penghapusan baris hasil *scraping* isi konten yang kosong	-81	533
6		Penghapusan baris hasil *scraping* isi konten berbahasa Inggris dengan fungsi detect_language()	-38	495
7		Filter dan hapus konten berbahasa Inggris manual melalui Microsoft Excel	-26	469

Setelah seluruh dependensi terpasang, dilakukan konfigurasi opsi Chrome agar dapat dijalankan secara aman di lingkungan kerja. Selanjutnya, dibuat struktur data berupa *set* untuk menyimpan hasil scraping agar tidak terjadi duplikasi tautan.

Fungsi utama yang digunakan adalah scrape_google_news_link(). Fungsi ini bertugas untuk mengakses hasil pencarian berita di Google News berdasarkan kata kunci tertentu, yaitu “Garuda Indonesia” dan “Pesawat Garuda”. Melalui Selenium, *browser* otomatis membuka halaman-halaman hasil pencarian, kemudian mengambil sejumlah artikel per batch (setiap 10 artikel), dan mengekstrak beberapa komponen penting dari masing-masing berita, meliputi: tautan berita (*link*), judul artikel, tanggal publikasi, dan nama portal berita. Data yang telah dikumpulkan disimpan dalam variabel all_articles yang berisi kumpulan data berita unik.

```
Mengakses URL: https://www.google.com/search?q=garuda+indonesia
Halaman artikel sudah tidak tersedia
Proses selesai. Total link artikel unik yang berhasil diambil: 1194
----------------------------------------------------------
Mengakses URL: https://www.google.com/search?q=pesawat+garuda
Halaman artikel sudah tidak tersedia
Proses selesai. Total link artikel unik yang berhasil diambil: 1194
----------------------------------------------------------
```

Gambar Hasil proses *scraping*

Setelah proses pengumpulan selesai, sebagian data ditampilkan ke layar sebagai contoh untuk memastikan hasil *scraping* berjalan dengan baik. Bagian ini memperlihatkan beberapa entri awal berupa tautan, judul, tanggal, dan portal berita.

```
('https://www.travelandtourworld.com/news/article/garuda-indonesia-now-expands-flight-network-to-bali-yogyakarta-surabaya-and
-other-main-tourism-hub-by-2029-heres-what-you-need-to-know/', 'Garuda Indonesia Now Expands Flight Network to Bali, Yogyakar
ta, Surabaya, and Other Main Tourism Hub by 2029: Here’s What You Need To Know', '2025-07-31', 'Travel And Tour World')
('https://www.ft.com/content/45ff1892-b4fe-4caf-9d07-2e6a0a189fed', 'Islamic bonds come under microscope after Garuda Indones
ia default', '2021-08-17', 'Financial Times')
('https://ulasan.co/beredar-video-ban-pesawat-garuda-menggelinding-di-landasan-pacu-tanjungpinang/', 'Beredar Video Ban Pesaw
at Garuda Menggelinding di Landasan Pacu Tanjungpinang', '2025-04-16', 'Ulasan.co')
('https://www.cnnindonesia.com/ekonomi/20240824164025-92-1137068/135-penumpang-pesawat-tangki-bocor-garuda-sudah-diterbangkan
-kembali', '135 Penumpang Pesawat Tangki Bocor Garuda Sudah Diterbangkan Kembali', '2024-08-24', 'CNN Indonesia')
('https://tirto.id/cara-pilih-kursi-garuda-terbaru-dan-besaran-tarifnya-g65a', 'Cara Pilih Kursi Garuda Terbaru dan Besaran T
arifnya', '2024-12-31', 'Tirto.id')
```

Gambar Entri awal hasil *scraping*

Tahap berikutnya adalah mengonversi data hasil scraping ke dalam bentuk DataFrame menggunakan pustaka Pandas. DataFrame ini memiliki empat kolom utama yaitu *link*, *judul*, *tanggal*, dan *portal*. Kolom tanggal kemudian dikonversi ke format DateTime dan diurutkan dari tanggal terbaru ke tanggal terlama agar memudahkan proses analisis di tahap berikutnya.

	link	judul	tanggal	portal
0	https://kumparan.com/kumparanbisnis/garuda-ind...	Garuda Indonesia Kembali RUPSLB di Tengah Isu ...	2025-09-30	Kumparan
1	https://voi.id/en/economy/519004	Commission V DPR Will Investigate Allegations ...	2025-09-29	VOI.ID
2	https://www.prnewswire.com/news-releases/garud...	Garuda Indonesia Goes Digital: Air Cargo Capac...	2025-09-29	PR Newswire
3	https://in.investing.com/news/company-news/gar...	Garuda Indonesia adds air cargo capacity to We...	2025-09-29	Investing.com India
4	https://jambi.pikiran-rakyat.com/info-data/pr-...	Jadwal Kedatangan Pesawat di Bandara Sultan Th...	2025-09-29	Jambian
...	...	...	...	...
1189	https://www.thejakartapost.com/indonesia/2024/...	Garuda Indonesia flight makes emergency landin...	NaT	The Jakarta Post
1190	https://finance.detik.com/bursa-dan-valas/d-81...	Gelar RUPSLB Lagi, Anak Usaha Garuda Tunjuk Di...	NaT	detikFinance
1191	https://djsaviation.net/garuda-indonesia-aircr...	Garuda Indonesia Aircraft Are Grounded	NaT	Dj's Aviation
1192	https://www.aviacionline.com/qatar-airways-and...	Qatar Airways and Garuda Indonesia expand part...	NaT	Aviacionline
1193	https://www.flightglobal.com/airlines/garuda-s...	Garuda secures key payment from Indonesian gov...	NaT	FlightGlobal

1194 rows × 4 columns

Gambar *Preview dataset* hasil *scraping*

Hasil akhir dari proses ini disimpan ke dalam dua format file, yaitu CSV dan Excel, dengan nama *link_berita_garudaindonesia.csv* dan *link_berita_garudaindonesia.xlsx*. File tersebut berisi kumpulan tautan berita lengkap dengan judul, tanggal publikasi, serta nama portal berita yang menjadi hasil akhir tahap akuisisi.

Setelah *scraping* menggunakan Python, didapatkan data sebanyak 1004 baris, dengan 1003 artikel dan 1 *header*. Akan tetapi, pada data tersebut, terdapat beberapa judul berita dalam bahasa Inggris. Berita dalam bahasa Inggris kemudian di-*detect* melalui Google Spreadsheets dan dilabeli dengan kode ‘EN’. Setelah itu, dilakukan *filtering* untuk menghapus semua baris berita berbahasa Inggris.

```
=IF(AND(LEN(TRIM(A2))>=10, DETECTLANGUAGE(A2)="en"), "EN","KEEP")
```

Gambar Formula deteksi bahasa Inggris

A.2 *Tagging* Berdasarkan Judul

Penentuan *tag* dilakukan berdasarkan kecocokan ekspresi reguler (*regex*) dari judul berita (setelah diubah menjadi huruf kecil) dengan daftar kata kunci yang telah ditentukan menggunakan fungsi =ARRAYFORMULA() di Google Spreadsheets.

```
=ARRAYFORMULA(
IF(A2:A="","",
  IFS(
    REGEXMATCH(LOWER(A2:A),"(laba|rugi|pendapatan|utang|restrukturisasi|pkpu|obligasi|kuartal|laporan keuangan)"),"Keuangan",
    REGEXMATCH(LOWER(A2:A),"(rute|penerbangan baru|buka rute|tutup rute|frekuensi|penerbangan|operasional|insiden|delay|batal|keselamatan)"),"Rute/Operasional",
    REGEXMATCH(LOWER(A2:A),"(direktur|komisaris|manajemen|rups|ceo|pergantian|pengurus)"),"Manajemen",
    REGEXMATCH(LOWER(A2:A),"(kasus|pengadilan|kpk|dugaan|suap|hukum|sidang|sanksi|denda|izin)"),"Hukum/Regulasi",
    TRUE,"Lainnya"
  )
))
```

Gambar Formula untuk *tagging* berdasarkan judul

Keterangan:

- Tag "Keuangan" diberikan jika teks mengandung salah satu kata berikut: *laba*, *rugi*, *pendapatan*, *utang*, *restrukturisasi*, *pkpu*, *obligasi*, *kuartal*, atau *laporan keuangan*.
- Tag "Rute/Operasional" diberikan jika teks mengandung salah satu kata berikut: *rute*, *penerbangan baru*, *buka rute*, *tutup rute*, *frekuensi penerbangan*, *operasional*, *insiden*, *delay*, *batal*, atau *keselamatan*.
- Tag "Manajemen" diberikan jika teks mengandung salah satu kata berikut: *direktur*, *komisaris*, *manajemen*, *rups*, *ceo*, *pergantian*, atau *pengurus*.
- Tag "Hukum/Regulasi" diberikan jika teks mengandung salah satu kata berikut: *kasus*, *pengadilan*, *kpk*, *dugaan*, *suap*, *hukum*, *sidang*, *sanksi*, *denda*, atau *izin*.
- Tag "Lainnya" diberikan jika teks tidak memenuhi kriteria kata kunci dari salah satu kategori di atas.

A.3 Akuisisi Isi Konten Berita

Tahap ini bertujuan untuk mengekstrak isi berita dari daftar tautan yang telah diperoleh pada tahap sebelumnya. Data awal berupa berkas data_link_berita.csv sudah berisi kolom judul dan tautan berita. Pada tahap ini, sistem hanya berfokus untuk mengunduh serta mengekstrak teks utama dari setiap tautan tersebut.

Proses diawali dengan instalasi beberapa pustaka pendukung seperti Pandas, Tqdm, Requests, Newspaper3k, Trafilatura, dan Readability-LXML. Pustaka-pustaka ini digunakan untuk membaca data, melakukan permintaan ke situs berita, serta mengekstrak isi utama halaman web secara otomatis.

Setiap tautan dalam *dataset* diproses satu per satu menggunakan beberapa metode ekstraksi yang disusun secara berlapis. Metode pertama adalah Newspaper3k, yang berusaha mengambil judul dan isi artikel secara langsung. Jika hasilnya terlalu pendek atau gagal, sistem beralih ke Trafilatura, yang mengekstrak teks berdasarkan struktur HTML halaman. Bila kedua metode tersebut masih belum menghasilkan teks yang memadai, digunakan Readability-LXML untuk menyaring bagian teks yang paling relevan dari halaman tersebut.

Selama proses pengambilan konten, sistem menggunakan mekanisme *timeout*, *retry* otomatis, serta jeda acak antar-*request* untuk mencegah kegagalan akibat pembatasan

akses (error 429). Setiap teks hasil ekstraksi juga dibersihkan dari spasi berlebih dan disusun ulang agar tampil lebih rapi.

Setelah seluruh tautan berhasil diproses, hasil ekstraksi berupa konten berita ditempatkan bersebelahan dengan kolom judul di dalam dataframe. Data akhir kemudian disimpan kembali dalam berkas keluaran berformat .xlsx, dengan nama yang sama seperti berkas input namun ditambahkan akhiran _with_content.xlsx. Hasil akhir tahap ini adalah dataset yang telah berisi teks lengkap dari masing-masing berita, siap digunakan pada tahap pra-pemrosesan teks berikutnya.

```
Mengambil konten artikel: 100%|██████████| 614/614 [42:42<00:00,  4.17s/it]
Selesai! Tersimpan: data_link_berita_with_content.csv
```

Gambar Hasil scraping isi konten berita

Meskipun sudah dilakukan tahapan *filtering* judul artikel berbahasa Inggris, beberapa artikel berbahasa Inggris tetap masuk dalam *dataset*. Oleh karena itu, dilakukan tahapan untuk membersihkan dan memfilter data hasil *scraping* sebelumnya agar hanya tersisa data yang valid dan relevan untuk proses analisis berikutnya. Proses diawali dengan membaca *file* hasil akuisisi, yaitu *data_link_berita_with_content.csv*, menggunakan pustaka Pandas. File tersebut berisi data mentah berupa tautan, judul, tanggal, portal, serta konten berita yang telah berhasil dikumpulkan dari Google News.

Apabila *file* berhasil dimuat, program akan menampilkan beberapa baris pertama untuk memastikan struktur data telah sesuai. Jika file tidak ditemukan, sistem akan memberikan pesan kesalahan agar pengguna dapat memperbaiki jalur file yang digunakan.

```
Successfully loaded data from data_link_berita_with_content.csv
```

	link	judul	konten	tanggal	portal	tag
0	https://kumparan.com/kumparanbisnis/garuda-ind...	Garuda Indonesia Kembali RUPSLB di Tengah Isu ...	Garuda Indonesia Kembali RUPSLB di Tengah Isu ...	2025-09-30 00:00:00	Kumparan	Manajemen
1	https://www.bloombergtechnoz.com/detail-news/8...	Garuda Gelar RUPSLB di Tengah Isu Masuknya Dir...	Garuda Gelar RUPSLB di Tengah Isu Masuknya Dir...	2025-09-29 00:00:00	Bloomberg Technoz	Manajemen
2	https://voi.id/ekonomi/519004/komisi-v-dpr-bak...	Komisi V DPR Bakal Dalami Dugaan Mafia Jual Be...	JAKARTA - Ketua Komisi V DPR Lasarus mengataka...	2025-09-29 00:00:00	VOI.ID	Rute/Operasional
3	https://in.investing.com/news/company-news/gar...	Garuda Indonesia adds air cargo capacity to We...	NaN	2025-09-29 00:00:00	Investing.com India	Lainnya
4	https://www.kompasiana.com/zainularifin2714/68...	Rencana Merger Garuda Indonesia - Pelita Air: ...	Latar Belakang\nPada pertengahan 2023, wacana ...	2025-09-29 00:00:00	Kompasiana.com	Lainnya

Gambar *Preview* hasil *scraping* mentah

Langkah pertama dalam pembersihan data adalah menghapus baris yang tidak memiliki nilai pada kolom “konten”. Hal ini dilakukan menggunakan fungsi dropna() agar hanya berita yang memiliki isi lengkap yang tersisa dalam dataset. Berdasarkan hasil eksekusi, jumlah data awal sebanyak 614 baris berkurang menjadi 533 baris setelah pembersihan, yang berarti terdapat 81 data tanpa isi konten yang dihapus.

```
Original DataFrame shape: (614, 6)
Cleaned DataFrame shape: (533, 6)
```

	link	judul	konten	tanggal	portal	tag
0	https://kumparan.com/kumparanbisnis/garuda-ind...	Garuda Indonesia Kembali RUPSLB di Tengah Isu ...	Garuda Indonesia Kembali RUPSLB di Tengah Isu ...	2025-09-30 00:00:00	Kumparan	Manajemen
1	https://www.bloombergtechnoz.com/detail-news/8...	Garuda Gelar RUPSLB di Tengah Isu Masuknya Dir...	Garuda Gelar RUPSLB di Tengah Isu Masuknya Dir...	2025-09-29 00:00:00	Bloomberg Technoz	Manajemen
2	https://voi.id/ekonomi/519004/komisi-v-dpr-bak...	Komisi V DPR Bakal Dalami Dugaan Mafia Jual Be...	JAKARTA - Ketua Komisi V DPR Lasarus mengataka...	2025-09-29 00:00:00	VOI.ID	Rute/Operasional
4	https://www.kompasiana.com/zainularifin2714/68...	Rencana Merger Garuda Indonesia - Pelita Air: ...	Latar Belakang\nPada pertengahan 2023, wacana ...	2025-09-29 00:00:00	Kompasiana.com	Lainnya
5	https://www.cnnindonesia.com/ekonomi/202509292...	Dony Oskaria Pastikan Merger Pelita Air-Garuda...	--\nPlt Menteri Badan Usaha Milik Negara (BUMN...	2025-09-29 00:00:00	CNN Indonesia	Lainnya

Gambar *Preview* hasil *scraping* setelah penghapusan baris kosong

Selanjutnya dilakukan deteksi bahasa pada kolom “judul” menggunakan pustaka LangDetect. Fungsi khusus detect_language() dibuat untuk mengidentifikasi bahasa setiap judul berita dengan menangani kemungkinan error, misalnya jika teks kosong atau bukan bertipe string. Setelah bahasa terdeteksi, data yang berbahasa Inggris (kode “en”) dihapus agar hanya berita berbahasa Indonesia yang dipertahankan dalam dataset.

```
df_cleaned['detected_language'] = df_cleaned['judul'].apply(detect_language)
```

	link	judul	konten	tanggal	portal	tag
0	https://kumparan.com/kumparanbisnis/garuda-ind...	Garuda Indonesia Kembali RUPSLB di Tengah Isu ...	Garuda Indonesia Kembali RUPSLB di Tengah Isu ...	2025-09-30 00:00:00	Kumparan	Manajemen
1	https://www.bloombergtechnoz.com/detail-news/8...	Garuda Gelar RUPSLB di Tengah Isu Masuknya Dir...	Garuda Gelar RUPSLB di Tengah Isu Masuknya Dir...	2025-09-29 00:00:00	Bloomberg Technoz	Manajemen
2	https://voi.id/ekonomi/519004/komisi-v-dpr-bak...	Komisi V DPR Bakal Dalami Dugaan Mafia Jual Be...	JAKARTA - Ketua Komisi V DPR Lasarus mengataka...	2025-09-29 00:00:00	VOI.ID	Rute/Operasional
4	https://www.kompasiana.com/zainularifin2714/68...	Rencana Merger Garuda Indonesia - Pelita Air: ...	Latar Belakang\nPada pertengahan 2023, wacana ...	2025-09-29 00:00:00	Kompasiana.com	Lainnya
5	https://www.cnnindonesia.com/ekonomi/202509292...	Dony Oskaria Pastikan Merger Pelita Air-Garuda...	--\nPlt Menteri Badan Usaha Milik Negara (BUMN...	2025-09-29 00:00:00	CNN Indonesia	Lainnya

Gambar *Preview dataset* setelah penghapusan artikel berbahasa Inggris

Data berita yang berbahasa Inggris kemudian disimpan dalam *dataframe* terpisah. Tujuannya adalah untuk menampilkan daftar berita yang terdeteksi menggunakan bahasa Inggris untuk diperiksa sebelum dihapus dari *dataset* utama.

Rows removed (English titles):

	link	judul	konten	tanggal	portal	tag	detected_languag
19	https://www.ch-aviation.com/news/158639-citili...	Citilink, Garuda Indonesia to reactivate more ...	Aviation Intelligence for your everyday use\nO...	2025-09-27 00:00:00	ch-aviation	Lainnya	en
33	https://www.ch-aviation.com/news/158613-garuda...	Garuda Indonesia says Pelita Air merger in ear...	Aviation Intelligence for your everyday use\nO...	2025-09-25 00:00:00	ch-aviation	Lainnya	en
48	https://www.kompas.id/artikel/en-berat-sebelah...	The One-Sided Merger of Garuda Indonesia and P...	The issue of merging PT Garuda Indonesia (Pers...	2025-09-24 00:00:00	Kompas.id	Lainnya	en
62	https://en.tempo.co/read/2050959/garuda-indone...	Garuda Indonesia to Launch New Halim-Palembang...	TEMPO.CO, Jakarta - PT Garuda Indonesia (Perse...	2025-09-23 00:00:00	Tempo.co	Lainnya	en

Gambar *Preview dataset* berbahasa Inggris yang dihapus

Setelah proses penyaringan selesai, data bersih disimpan ke dalam file baru dengan nama *data_link_berita_with_content_cleaned.csv*. File ini berisi berita dengan konten lengkap dan berbahasa Indonesia yang siap digunakan pada tahap analisis berikutnya.

Sebagai tahap akhir, dilakukan analisis deskriptif awal terhadap dataset hasil pembersihan. Analisis ini mencakup informasi statistik dasar, jumlah nilai unik pada setiap kolom kategorikal, serta distribusi data berdasarkan portal berita dan tag yang ada. Selain itu, kolom *tanggal* dikonversi ke format waktu standar untuk melihat rentang waktu publikasi berita yang berhasil dikumpulkan. Hasil analisis menunjukkan persebaran artikel dari berbagai portal dengan periode publikasi tertentu yang akan menjadi dasar untuk analisis lanjutan.

Descriptive Statistics:

	link	judul	konten	tanggal	portal	tag
count	495	495	495	495	495	495
unique	495	494	494	191	104	5
top	https://www.tempo.co/ekonomi/tahun-depan-garud...	Danantara injects US$405 million into Garuda I...	Aviation Intelligence for your everyday use\nO...	2025-06-10 00:00:00	Kompas.com	Lainnya
freq	1	2	2	19	66	347

Gambar Distribusi hasil *scraping* konten setelah *cleaning*

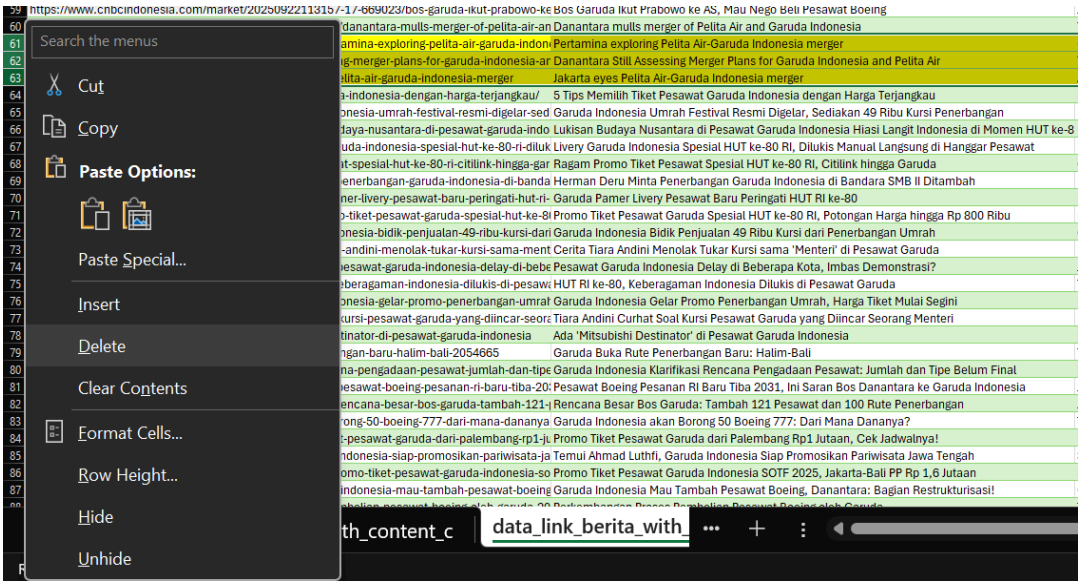
portal	count
Kompas.com	66
Tempo.co	34
Bisnis.com	30
Liputan6.com	26
CNBC Indonesia	26
CNN Indonesia	25
detikFinance	25
ANTARA News	25
detikTravel	13
Suara.com	12

tag	count
Lainnya	347
Rute/Operasional	97
Keuangan	22
Hukum/Regulasi	19
Manajemen	10

```
Date Range of Articles:
Earliest Date: 2012-12-07 00:00:00
Latest Date: 2025-09-30 00:00:00
```

Gambar Distribusi hasil *scraping* konten setelah *cleaning* per portal dan tag

Selanjutnya, isi *file* diperiksa secara manual melalui Microsoft Excel. Dari hasil pemeriksaan, ditemukan bahwa, meskipun judul-judul bahasa Inggris sudah dihapus sebelum dilakukan akuisisi isi konten berita, beberapa judul dan konten berbahasa Inggris masih saja tetap masuk ke dalam *dataset*. Oleh karena itu, dilakukan pembersihan secara manual dengan penandaan dan penghapusan dalam Microsoft Excel.



Gambar Flag manual dengan Excel

Setelah pembersihan awal tersebut dilakukan, dilakukan pemeriksaan statistika deskriptif dasar terhadap data. Dari hasil pemeriksaan tersebut, didapatkan pengurangan jumlah baris data dari yang awalnya berjumlah 495 menjadi 469.

Descriptive Statistics:

	link	judul	konten	tanggal	portal	tag
count	469	469	469	469	469	469
unique	469	469	469	176	90	5
top	https://www.tempo.co/ekonomi/tahun-depan-garud...	Tahun Depan Garuda Datangkan 24 Pesawat Baru	TEMPO.CO, Jakarta - Maskapai penerbangan pelat...	10/6/2025	Kompas.com	Lainnya
freq	1	1	1	19	66	322

Gambar Distribusi hasil *scraping* konten setelah *cleaning manual*

portal	count
Kompas.com	66
Bisnis.com	30
Tempo.co	30
CNBC Indonesia	26
Liputan6.com	26
detikFinance	25
CNN Indonesia	25
ANTARA News	21
detikTravel	13
MetroTVNews.com	12

tag	count
Lainnya	322
Rute/Operasional	97
Keuangan	22
Hukum/Regulasi	19
Manajemen	9

Date Range of Articles:
Earliest Date: 2012-12-07 00:00:00
Latest Date: 2025-09-30 00:00:00

Gambar Distribusi hasil *scraping* konten setelah *cleaning manual* per portal dan tag

A.4 *Tagging* dengan OpenAI API

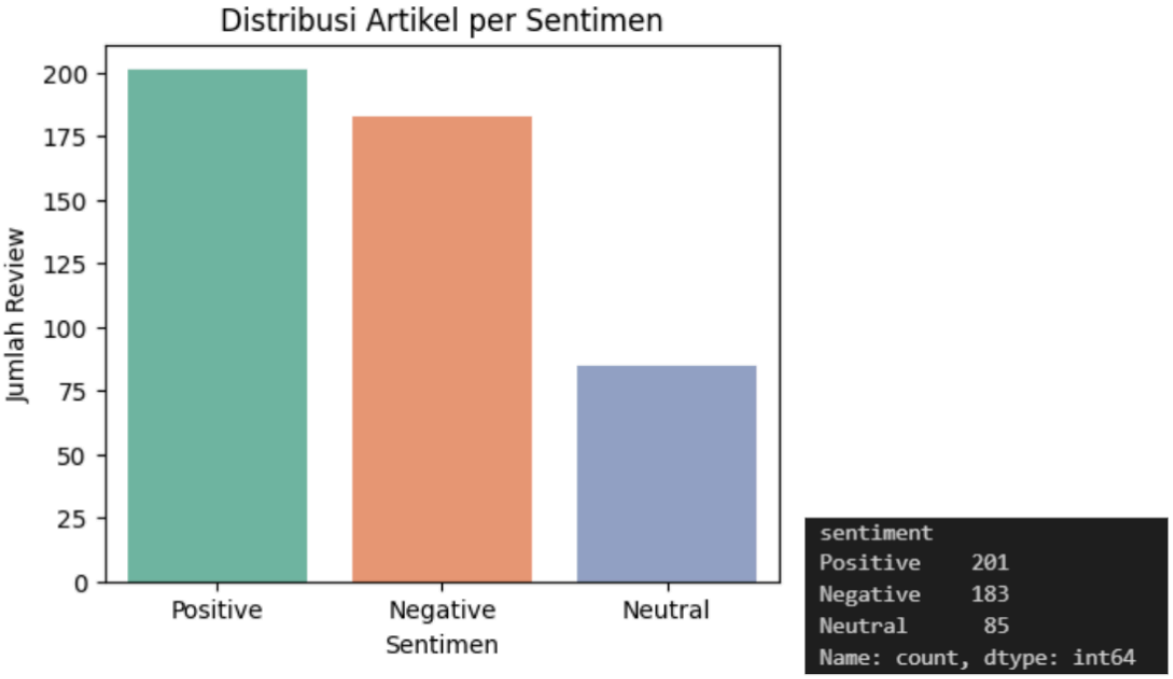
Untuk meningkatkan akurasi klasifikasi, dilakukan proses penandaan ulang (*re-tagging*) dengan bantuan model bahasa besar (LLM) melalui OpenAI API. Model yang digunakan adalah GPT-4.1, dengan pertimbangan keseimbangan antara harga dan efektivitas pemrosesan teks dalam skala besar.

Selain menghasilkan kategori topik baru (tag_new), tahap ini juga menambahkan dimensi analisis sentimen, yaitu mengidentifikasi polaritas emosi dari teks (*positive*, *neutral*, atau *negative*). Proses pelabelan sentimen dilakukan menggunakan model bahasa GPT-4.1 yang diintegrasikan melalui API OpenAI pada Google Colab. Sebelum menjalankan pelabelan, dilakukan proses instalasi dependensi seperti openai, pandas, dan google-colab, serta inisialisasi variabel API agar koneksi dapat dijalankan secara otomatis. Setelah itu, dilakukan pengaturan nama file input dan output, penentuan kolom teks yang akan dianalisis, serta daftar kategori berita yang valid, yaitu *Kinerja & Keuangan*, *Operasional & Pelayanan*, *Regulasi & Kebijakan*, *Krisis & Kontroversi*, dan *Industri & Pariwisata Nasional*.

```
system_instruction = (
"Kamu adalah AHLI analisis sentimen yang sangat AKURAT. "
"Untuk setiap teks, kamu harus mengeluarkan output dalam format JSON lengkap seperti contoh berikut:\n\n"
'Contoh:\n'
'{"sentiment": "Positive", "tag_new": "Operasional & Pelayanan"}\n'
'{"sentiment": "Neutral", "tag_new": "Kinerja & Keuangan"}\n'
'{"sentiment": "Negative", "tag_new": "Krisis & Kontroversi"}\n\n'
"Selalu isi kedua field dengan nilai yang sesuai.\n"
"Nilai sentiment hanya boleh: Positive, Neutral, atau Negative.\n"
"Nilai tag hanya boleh salah satu dari lima kategori berikut:\n"
"1. Kinerja & Keuangan\n"
"2. Operasional & Pelayanan\n"
"3. Regulasi & Kebijakan\n"
"4. Krisis & Kontroversi\n"
"5. Industri & Pariwisata Nasional\n\n"
"Jangan kosongkan field apapun. Jangan tulis apapun di luar JSON."
)
```

Gambar *Prompt labelling* API OpenAI

Fungsi utama dalam tahap ini adalah analyze_sentiment_openai(). Fungsi ini menerima input berupa teks berita, kemudian mengirimkannya ke model GPT-4.1 untuk dianalisis. Model diminta menghasilkan output dalam format JSON yang berisi dua atribut, yaitu sentiment dan tag_new. Atribut sentiment hanya memiliki tiga kemungkinan nilai, yakni *Positive*, *Neutral*, atau *Negative*, sedangkan tag_new menentukan kategori berita sesuai konteks isi teks. Untuk memastikan hasil yang valid, fungsi ini juga mencakup proses validasi dan *fallback* agar sistem tetap memberikan hasil standar meskipun terjadi kesalahan pemrosesan. Kemudian, didapatkan distirbusi artikel per sentimen sebagai berikut:



Gambar Distribusi Artikel per Sentimen

A.5 Praproses Data

Berikut ini langkah-langkah yang akan dilakukan pada tahapan persiapan dan praproses data:

1. Mulai: Data mentah masuk ke dalam proses.

2. *Lowercasing*: Seluruh teks diubah menjadi huruf kecil (*lowercase*) untuk menyeragamkan data dan menghindari perbedaan yang tidak perlu antar kata (misalnya, "Data" dan "data" dianggap sama).
3. Hapus Emoji dan Karakter: Karakter non-teks seperti emoji, simbol, atau karakter khusus yang tidak relevan dengan analisis dihapus.
4. Koreksi Ejaan: Dilakukan perbaikan kesalahan pengetikan atau ejaan untuk memastikan konsistensi dan akurasi kata.
5. Selesai: Data teks telah bersih dan siap untuk tahap analisis berikutnya.

A.6 *Lowercasing* dan Hapus Emoji-Karakter

Pada tahap ini, dilakukan proses pemrosesan awal terhadap teks konten berita untuk memastikan bahwa seluruh data berada dalam format yang seragam dan bebas dari karakter yang tidak relevan. Proses ini mencakup beberapa langkah utama yaitu konversi huruf menjadi huruf kecil (*lowercasing*), penghapusan emoji serta karakter non-alfanumerik, dan perlindungan angka yang mengandung makna finansial agar tidak terhapus selama pembersihan.

Langkah pertama dimulai dengan memuat dataset hasil pembersihan sebelumnya, yaitu *data_link_berita_with_content_cleaned_manual.csv*, menggunakan pustaka Pandas. *Dataset* ini memuat berita yang telah diseleksi dan siap untuk tahap pra-pemrosesan teks lebih lanjut.

Selanjutnya ditentukan beberapa pola (*regular expression*) untuk mendeteksi dan menghapus karakter yang tidak diinginkan, meliputi emoji dan karakter "*zero-width*", yang sering muncul akibat salinan teks dari laman berita atau media sosial; dan tanda baca dan simbol non-alfanumerik, agar teks yang tersisa hanya terdiri atas huruf, angka, dan spasi.

Namun, oleh karena data ini mencakup konteks finansial (misalnya laporan laba rugi, nilai saham, atau angka mata uang), maka disusun pula pola proteksi numerik. Pola ini menjaga agar angka-angka penting yang berkaitan dengan istilah finansial seperti "Rp 2 miliar", "naik 5%", atau "laba 120 juta" tidak terhapus selama proses pembersihan. Perlindungan dilakukan dengan membungkus angka-angka tersebut menggunakan placeholder khusus yang kemudian dikembalikan ke bentuk aslinya setelah seluruh proses selesai.

Setelah semua pola ditetapkan, dibuat dua fungsi utama, yakni protect_financial_numbers(), berfungsi membungkus angka-angka finansial dengan placeholder agar tidak ikut terhapus; dan clean_text_with_numeric_rules(), berfungsi menghapus emoji, tanda baca, dan angka yang tidak relevan, menormalisasi huruf menjadi huruf kecil, serta merapikan spasi ganda. Kedua fungsi tersebut diterapkan pada kolom "konten", lalu hasilnya disimpan ke kolom baru bernama "konten_hapus_karakter". Dengan demikian, setiap berita kini memiliki dua versi: versi asli dan versi yang telah dibersihkan dari karakter berlebih namun tetap mempertahankan informasi numerik penting.

Dataset hasil pembersihan ini kemudian disimpan dalam *file* baru bernama *garudaindonesia_news_lower_hapus_karakter.csv* untuk digunakan pada tahap analisis berikutnya.

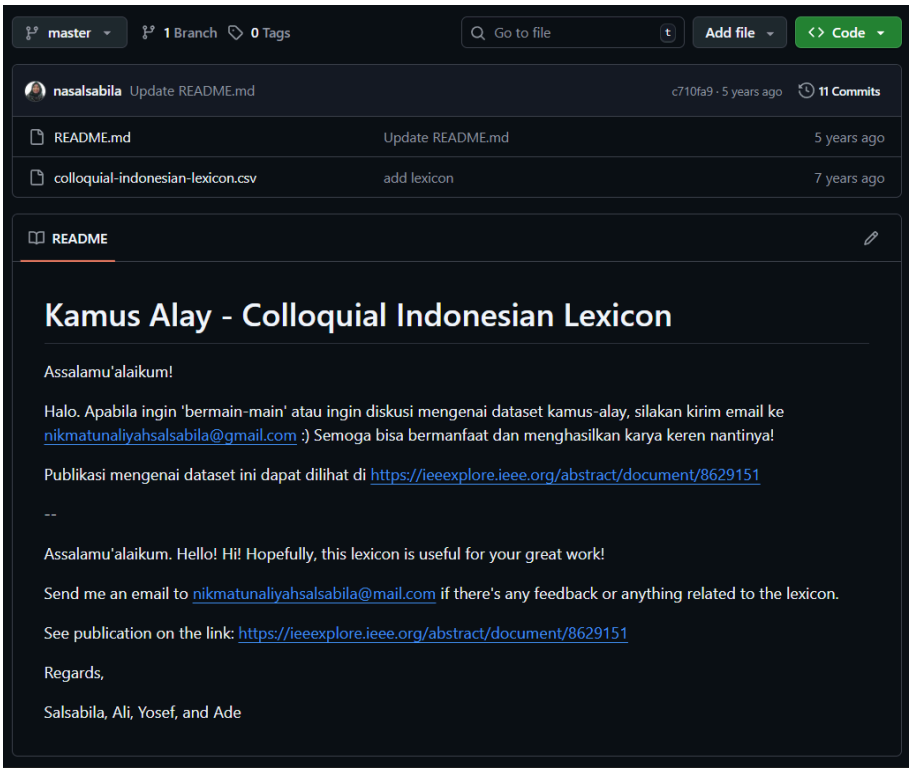
	konten	konten_hapus_karakter
0	Garuda Indonesia Kembali RUPSLB di Tengah Isu ...	garuda indonesia kembali rupslb di tengah isu ...
1	Garuda Gelar RUPSLB di Tengah Isu Masuknya Dir...	garuda gelar rupslb di tengah isu masuknya dir...
2	JAKARTA - Ketua Komisi V DPR Lasarus mengataka...	jakarta ketua komisi v dpr lasarus mengatakan ...
3	Latar Belakang\nPada pertengahan 2023, wacana ...	latar belakang pada pertengahan wacana konsoli...
4	--\nPlt Menteri Badan Usaha Milik Negara (BUMN...	plt menteri badan usaha milik negara bumn seka...

Gambar *Preview Dataset* hasil pembersihan karakter

A.7 Koreksi Ejaan

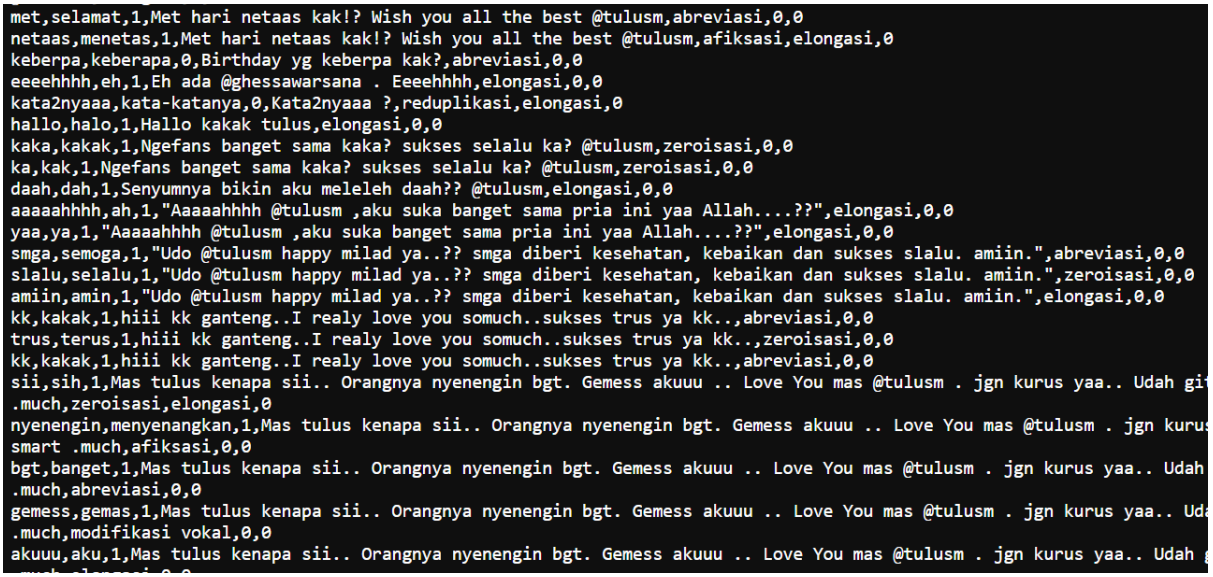
Tahap ini bertujuan untuk melakukan koreksi atau normalisasi ejaan terhadap teks konten berita agar bentuk katanya menjadi lebih baku dan konsisten. Proses ini penting karena data hasil scraping sering kali mengandung bahasa tidak formal, singkatan, atau bentuk *slang* yang perlu disesuaikan sebelum analisis lebih lanjut dilakukan.

Langkah pertama adalah memuat dua *file* utama, yaitu *garudaindonesia_news_lower_hapus_karakter.csv* yang berisi kolom *konten_lowercase*, dan *colloquial-indonesian-lexicon.csv* dari repositori github.com/nasalsabila/kamus-alay yang berisi daftar padanan kata tidak baku (*slang*) dengan bentuk formalnya. Kedua *file* ini digunakan sebagai dasar untuk melakukan proses substitusi kata.



Gambar GitHub nasalsabila/kamus-alay

Selanjutnya dilakukan pembuatan kamus konversi dari kata slang ke bentuk formal. Kolom pada file leksikon dinormalisasi ke huruf kecil (*lowercase*) dan dihapus karakter kosong di awal atau akhir string menggunakan fungsi *strip()*. Jika terdapat duplikasi pada kolom *slang*, hanya entri pertama yang digunakan agar tidak terjadi konflik dalam pemetaan. Hasilnya adalah sebuah kamus Python (slang2formal) yang berisi pasangan kata tidak baku dan padanan formalnya.



```
met,selamat,1,Met hari netaas kak!? Wish you all the best @tulusm,abreviasi,0,0
netaas,menetas,1,Met hari netaas kak!? Wish you all the best @tulusm,afiksasi,elongasi,0
keberpa,keberapa,0,Birthday yg keberpa kak?,abreviasi,0,0
eeeehhhh,eh,1,Eh ada @ghessawarsana . Eeeehhhh,elongasi,0,0
kata2nyaaa,kata-katanya,0,Kata2nyaaa ?,reduplikasi,elongasi,0
hallo,halo,1,Hallo kakak tulus,elongasi,0,0
kaka,kakak,1,Ngefans banget sama kaka? sukses selalu ka? @tulusm,zeroisasi,0,0
ka,kak,1,Ngefans banget sama kaka? sukses selalu ka? @tulusm,zeroisasi,0,0
daah,dah,1,Senyumnya bikin aku meleleh daah?? @tulusm,elongasi,0,0
aaaaahhhh,ah,1,"Aaaaahhhh @tulusm ,aku suka banget sama pria ini yaa Allah....??",elongasi,0,0
yaa,ya,1,"Aaaaahhhh @tulusm ,aku suka banget sama pria ini yaa Allah....??",elongasi,0,0
smga,semoga,1,"Udo @tulusm happy milad ya..?? smga diberi kesehatan, kebaikan dan sukses slalu. amiin.",abreviasi,0,0
slalu,selalu,1,"Udo @tulusm happy milad ya..?? smga diberi kesehatan, kebaikan dan sukses slalu. amiin.",zeroisasi,0,0
amiin,amin,1,"Udo @tulusm happy milad ya..?? smga diberi kesehatan, kebaikan dan sukses slalu. amiin.",elongasi,0,0
kk,kakak,1,hiii kk ganteng..I realy love you somuch..sukses trus ya kk..,abreviasi,0,0
trus,terus,1,hiii kk ganteng..I realy love you somuch..sukses trus ya kk..,zeroisasi,0,0
kk,kakak,1,hiii kk ganteng..I realy love you somuch..sukses trus ya kk..,abreviasi,0,0
sii,sih,1,Mas tulus kenapa sii.. Orangnya nyenengin bgt. Gemess akuuu .. Love You mas @tulusm . jgn kurus yaa.. Udah gi
.much,zeroisasi,elongasi,0
nyenengin,menyenangkan,1,Mas tulus kenapa sii.. Orangnya nyenengin bgt. Gemess akuuu .. Love You mas @tulusm . jgn kurus
smart .much,afiksasi,0,0
bgt,banget,1,Mas tulus kenapa sii.. Orangnya nyenengin bgt. Gemess akuuu .. Love You mas @tulusm . jgn kurus yaa.. Udah
.much,abreviasi,0,0
gemess,gemas,1,Mas tulus kenapa sii.. Orangnya nyenengin bgt. Gemess akuuu .. Love You mas @tulusm . jgn kurus yaa.. Uda
.much,modifikasi vokal,0,0
akuuu,aku,1,Mas tulus kenapa sii.. Orangnya nyenengin bgt. Gemess akuuu .. Love You mas @tulusm . jgn kurus yaa.. Udah
```

Gambar Daftar padanan tidak baku dari nasalsabila/kamus-alay

Langkah berikutnya adalah membangun pola ekspresi reguler (*regex*) untuk mendeteksi kata-kata yang akan diganti. Daftar kata dalam kamus diurutkan berdasarkan panjang karakter dari yang terpanjang ke terpendek agar sistem dapat memprioritaskan pencocokan frasa panjang terlebih dahulu. Setiap kata di-*escape* agar karakter khusus dapat dibaca dengan benar oleh regex, kemudian digabung menjadi satu pola pencarian yang siap digunakan.

Fungsi normalize_text() kemudian dibuat untuk melakukan substitusi kata dalam teks berdasarkan kamus tersebut. Setiap kata yang cocok dengan pola regex akan digantikan dengan bentuk formalnya menggunakan pemetaan dari kamus *slang2formal*.

Setelah fungsi siap, proses normalisasi diterapkan pada kolom konten_hapus_karakter untuk menghasilkan kolom baru bernama konten_normalized. Proses ini dilakukan secara otomatis untuk setiap baris teks. Setelah normalisasi selesai, teks juga dirapikan dengan menghapus spasi ganda dan memastikan tidak ada karakter kosong di awal maupun akhir kalimat.

	konten_hapus_karakter	konten_normalized
0	garuda indonesia kembali rupslb di tengah isu ...	garuda indonesia kembali rupslb di tengah isu ...
1	garuda gelar rupslb di tengah isu masuknya dir...	garuda gelar rupslb di tengah isu masuknya dir...
2	jakarta ketua komisi v dpr lasarus mengatakan ...	jakarta ketua komisi v dpr lasarus mengatakan ...
3	latar belakang pada pertengahan wacana konsoli...	latar belakang pada pertengahan wacana konsoli...
4	plt menteri badan usaha milik negara bumn seka...	plt menteri badan usaha milik negara bumn seka...

Gambar *Preview dataset* setelah normalisasi ejaan

Langkah terakhir adalah menyimpan hasil normalisasi ke dalam file baru bernama *garudaindonesia_news_normalized.csv* agar dapat digunakan pada tahap analisis selanjutnya. *File* ini berisi versi teks yang sudah bersih dan telah dikoreksi secara ejaan menggunakan leksikon bahasa Indonesia. Kolom konten_normalized inilah yang akan digunakan sebagai data pelatihan dan pengujian model.

A.8 Definisi *dataset*

Dataset yang digunakan:
https://github.com/renaldoaluska/pba2025gasal/blob/main/%23TugasA-1/7-garudaindonesia_news_cleaned_simple.csv

Tabel Definisi *dataset*

No.	Nama Kolom	Deskripsi
1	link	URL atau tautan unik yang merujuk langsung ke sumber artikel berita asli.
2	judul	Judul asli dari artikel berita yang diambil dari sumbernya.
3	konten	Isi atau teks lengkap dari artikel berita yang menjadi objek utama analisis.
4	tanggal	Tanggal publikasi artikel berita, menunjukkan kapan berita tersebut diterbitkan.
5	portal	Nama media atau portal berita yang mempublikasikan artikel.
6	tag	Hasil *tagging* manual berdasarkan judul berita. Penentuan *tag* dilakukan berdasarkan kecocokan ekspresi reguler (*regex*) dari teks (setelah diubah menjadi huruf kecil) dengan daftar kata kunci yang telah ditentukan menggunakan fungsi =ARRAYFORMULA() di Google Spreadsheets. (Lihat bagian A.2) • Tag "Keuangan" diberikan jika teks mengandung salah satu kata berikut: *laba*, *rugi*, *pendapatan*, *utang*, *restrukturisasi*, *pkpu*, *obligasi*, *kuartal*, atau *laporan keuangan*. • Tag "Rute/Operasional" diberikan jika teks mengandung salah satu kata berikut: *rute*, *penerbangan baru*, *buka rute*, *tutup rute*, *frekuensi penerbangan*, *operasional*, *insiden*, *delay*, *batal*, atau *keselamatan*. • Tag "Manajemen" diberikan jika teks mengandung salah satu kata berikut: *direktur*, *komisaris*, *manajemen*, *rups*, *ceo*, *pergantian*, atau *pengurus*. • Tag "Hukum/Regulasi" diberikan jika teks mengandung salah satu kata berikut: *kasus*, *pengadilan*, *kpk*, *dugaan*, *suap*, *hukum*, *sidang*, *sanksi*, *denda*, atau *izin*. • Tag "Lainnya" diberikan jika teks tidak memenuhi kriteria kata kunci dari salah satu kategori di atas.
7	konten_hapus_karakter	Hasil pemrosesan kolom *konten* dengan simbol, angka, atau karakter khusus yang telah dihapus.
8	**konten_normalized**	Hasil pemrosesan kolom *konten_hapus_karakter* dengan penyeragaman istilah-istilah gaul (*slang*). **Kolom ini digunakan untuk *training* dan *testing*.**
9	konten_nostop	Hasil pemrosesan kolom *konten_normalized* dengan *stopwords* (kata yang frekuensinya tinggi namun tidak memiliki makna penting) yang telah dihapus.
10	konten_stem	Hasil pemrosesan kolom *konten_nostop* dengan pengonversian

		setiap katanya ke dalam bentuk kata dasar/akar.
11	**sentiment**	Label hasil analisis polaritas sentimen kalimat: Positive, Neutral, atau Negative hasil klasifikasi GPT-4.1 berdasarkan konteks isi berita (Lihat bagian a.4) **Kolom ini digunakan untuk patokan kelas label sentimen.**
12	tag_new	Kategori topik baru hasil klasifikasi GPT-4.1 berdasarkan konteks isi berita.

A.9 Pembersihan Data dan Label Encoding

Sebelum tokenisasi, data dari file .csv setelah proses-proses pada bagian sebelumnya dibersihkan terlebih dahulu:

- **Pembersihan NaN:** Baris data yang memiliki nilai kosong (NaN) pada kolom konten_normalized atau sentiment akan dibuang (dropna).
- **Label Encoding:** Sentimen yang berbentuk teks ('positive', 'negative') diubah menjadi format angka (integer) yang bisa diproses model. Kode ini menggunakan mapping:
  - 'Positive' -> 1
  - 'Negative' -> 0
- Data yang labelnya tidak ada dalam mapping tersebut (misal 'neutral') akan menjadi NaN dan kemudian dibuang, memastikan hanya data berlabel 0 dan 1 yang dipakai.

A.10 Tokenisasi (*Word-level*)

Proses ini memecah kalimat menjadi unit-unit yang lebih kecil (token).

- **Metode:** Kode ini menggunakan metode tokenisasi *word-level* (tingkat kata) yang sangat sederhana, yaitu str(sentence).split(). Ini berarti setiap kalimat dipecah berdasarkan spasi.
- **Bukan Subword:** Penting dicatat bahwa ini **bukan** tokenisasi *subword* (seperti bpe atau *wordpiece* yang digunakan oleh BERT).
- **Token Khusus [CLS]:** Untuk setiap kalimat, token khusus [CLS] (singkatan dari *classification*) ditambahkan di **awal** kalimat. Token ini nantinya akan digunakan oleh model sebagai representasi agregat dari keseluruhan kalimat untuk tugas klasifikasi.

A.11 Penyusunan Vocabulary dan Numericalisasi

Setelah tokenisasi, kita perlu memetakan setiap kata unik ke sebuah angka (ID).

- **Penyusunan Vocabulary:** Kode ini membangun sebuah Vocabulary (kosakata) dari nol. Ia akan memindai seluruh data latih, dan setiap kata unik yang ditemukan akan diberi ID unik.
- **Token Khusus (Lagi):** Vocabulary ini secara *default* memiliki 3 token khusus:
  - [PAD] (ID: 0): Token yang digunakan untuk "mengganjal" (padding) kalimat yang lebih pendek.
  - [UNK] (ID: 1): Token untuk "unknown". Jika saat validasi atau inferensi ada kata yang tidak ada di vocabulary (kata *out-of-vocabulary*), kata itu akan diganti dengan token ini.
  - [CLS] (ID: 2): Token klasifikasi yang tadi ditambahkan di awal.

- **Numericalisasi:** Setiap kalimat yang sudah ditokenisasi kemudian diubah menjadi urutan angka (IDs) berdasarkan vocabulary yang telah dibuat.

A.12 Penentuan Panjang Maksimum (MAX_LEN)

model transformer memerlukan input dengan panjang yang seragam dalam satu *batch*.

- **Strategi:** Daripada memilih angka MAX_LEN (panjang maksimum) secara manual (misal 512), kode ini mengadopsi strategi statistik.
- **Perhitungan:** Kode ini menghitung panjang semua kalimat dalam dataset, lalu mengambil **persentil ke-95** (np.percentile(lengths, 95)). Artinya, 95% data memiliki panjang di bawah atau sama dengan angka ini. Ini adalah cara yang baik untuk membuang *outlier* (kalimat yang sangat panjang) tanpa kehilangan terlalu banyak informasi.

A.13 Truncation & Padding (Pemotongan & Pengganjalan)

setelah MAX_LEN ditentukan, semua urutan (sequence) harus disesuaikan:

- **Truncation (Pemotongan):** Di dalam SentimentDataset, jika sebuah kalimat (setelah ditambah [CLS]) lebih panjang dari MAX_LEN, kalimat itu akan dipotong (token_ids[:self.max_len]).
- **Padding (Pengganjalan):** Proses ini terjadi di dalam collate_fn saat membuat *batch*. Kode ini menggunakan *dynamic padding*:
    1. Dalam satu *batch*, ia mencari kalimat terpanjang (misal, panjangnya 80).
    2. Semua kalimat dalam *batch* itu yang lebih pendek dari 80 akan diganjal dengan token [PAD] (ID 0) di bagian akhir sampai panjangnya sama-sama 80.
    3. Sebuah padding_mask juga dibuat. Ini adalah tensor boolean yang memberi tahu model (khususnya *attention mechanism*) di mana letak token [PAD] (ditandai True) agar model mengabaikannya.

A.14 Train/Val/Test Split (Pembagian Data)

Dataset yang sudah lengkap kemudian dibagi:

- **Rasio data** dibagi menjadi:
    - **80% data latih** (train_dataset) dan
    - **20% data validasi** (val_dataset) menggunakan random_split.
- **Test Set:**
    - Berdasarkan kode prepare_data, **tidak ada** *test set* (data uji) yang dibuat secara terpisah. Evaluasi hanya dilakukan terhadap data validasi.

B. **Model awal**. Jelaskan desain arsitektur Transformer (*embedding dim*, *positional encoding*, *#heads*, *#layers*, *feed-forward dim*, *dropout*, *classifier head*), inisialisasi, *optimizer*, *loss*;
a) Latih model awal dengan konfigurasi default (mis. lr=1e-4, batch=32, epochs=10) dan laporkan sebagai performance awal.
b) Simpan *loss* & *metric* (*train* & *val*) *per epoch* dan tampilkan grafik *loss/accuracy training* vs *validation*
c) Jelaskan apakah model mengalami *underfit*/baik/*overfit* pada konfigurasi awal ini.

Jawaban Bagian B.a):
Desain arsitektur Transformer (Encoder-only) yang digunakan adalah:

- **embedding dim (d_model):** 64
- **positional encoding:** Sinusoidal (standar)
- **#heads (n_heads):** 4 (dimensi per head 64/4 = 16)
- **#layers (num_layers):** 2 tumpuk Encoder Layer
- **feed-forward dim (ff_dim):** 128
- **dropout:** 0.1
- **classifier head:** 1 Linear layer yang mengambil output token [CLS]

Inisialisasi, Optimizer, dan Loss:

- **Inisialisasi:** Default PyTorch
- **Optimizer:** Adam
- **Loss:** CrossEntropyLoss

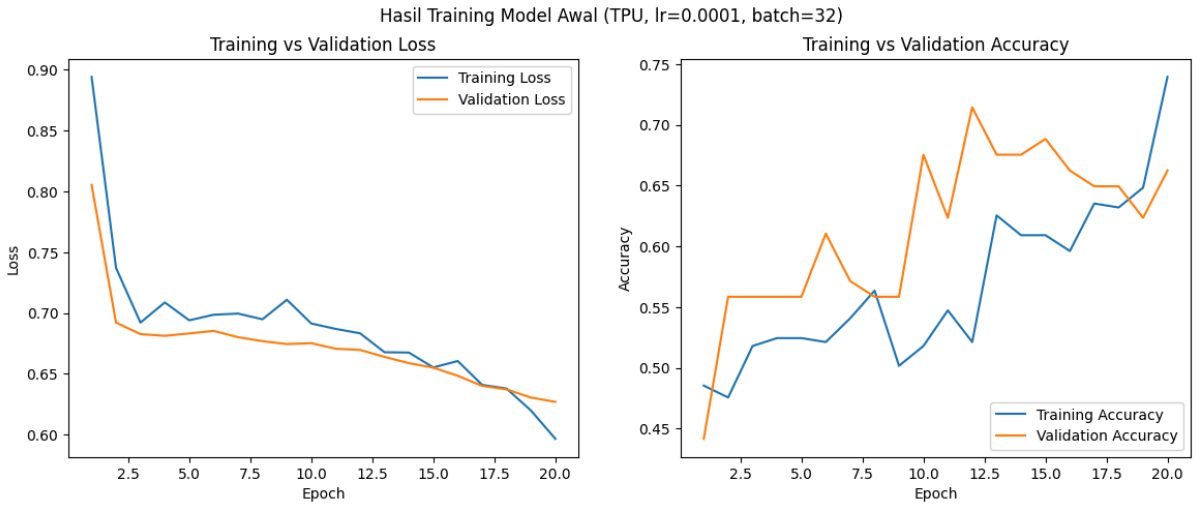
Model awal dilatih dengan konfigurasi:

- **learning rate:** 1e-4
- **batch size:** 32
- **epochs:** 20

Jawaban Bagian B.b):
Performance awal (pada *epoch* terakhir):

- **Final Training Loss:** 0.5965
- **Final Training Accuracy:** 0.7394
- **Final Validation Loss:** 0.6269
- **Final Validation Accuracy:** 0.6623

Berikut adalah grafik *loss/accuracy training* vs *validation*:



Epoch 1/20 | Train Loss: 0.8943 | Train Acc: 0.4853 | Val Loss: 0.8054 | Val Acc: 0.4416
Epoch 2/20 | Train Loss: 0.7371 | Train Acc: 0.4756 | Val Loss: 0.6921 | Val Acc: 0.5584
Epoch 3/20 | Train Loss: 0.6921 | Train Acc: 0.5179 | Val Loss: 0.6826 | Val Acc: 0.5584
Epoch 4/20 | Train Loss: 0.7087 | Train Acc: 0.5244 | Val Loss: 0.6812 | Val Acc: 0.5584
Epoch 5/20 | Train Loss: 0.6940 | Train Acc: 0.5244 | Val Loss: 0.6832 | Val Acc: 0.5584
Epoch 6/20 | Train Loss: 0.6986 | Train Acc: 0.5212 | Val Loss: 0.6853 | Val Acc: 0.6104
Epoch 7/20 | Train Loss: 0.6995 | Train Acc: 0.5407 | Val Loss: 0.6801 | Val Acc: 0.5714
Epoch 8/20 | Train Loss: 0.6948 | Train Acc: 0.5635 | Val Loss: 0.6768 | Val Acc: 0.5584
Epoch 9/20 | Train Loss: 0.7109 | Train Acc: 0.5016 | Val Loss: 0.6744 | Val Acc: 0.5584
Epoch 10/20 | Train Loss: 0.6914 | Train Acc: 0.5179 | Val Loss: 0.6751 | Val Acc: 0.6753
Epoch 11/20 | Train Loss: 0.6869 | Train Acc: 0.5472 | Val Loss: 0.6706 | Val Acc: 0.6234
Epoch 12/20 | Train Loss: 0.6833 | Train Acc: 0.5212 | Val Loss: 0.6696 | Val Acc: 0.7143
Epoch 13/20 | Train Loss: 0.6677 | Train Acc: 0.6254 | Val Loss: 0.6639 | Val Acc: 0.6753
Epoch 14/20 | Train Loss: 0.6674 | Train Acc: 0.6091 | Val Loss: 0.6587 | Val Acc: 0.6753
Epoch 15/20 | Train Loss: 0.6552 | Train Acc: 0.6091 | Val Loss: 0.6550 | Val Acc: 0.6883
Epoch 16/20 | Train Loss: 0.6605 | Train Acc: 0.5961 | Val Loss: 0.6484 | Val Acc: 0.6623
Epoch 17/20 | Train Loss: 0.6409 | Train Acc: 0.6352 | Val Loss: 0.6400 | Val Acc: 0.6494
Epoch 18/20 | Train Loss: 0.6378 | Train Acc: 0.6319 | Val Loss: 0.6371 | Val Acc: 0.6494
Epoch 19/20 | Train Loss: 0.6198 | Train Acc: 0.6482 | Val Loss: 0.6304 | Val Acc: 0.6234
Epoch 20/20 | Train Loss: 0.5965 | Train Acc: 0.7394 | Val Loss: 0.6269 | Val Acc: 0.6623

Jawaban Bagian B.c):

Model ini jelas mengalami ***underfitting***.

Alasan utamanya adalah:

1. **Performa Keseluruhan Sangat Rendah:**
    - **Akurasi Validasi (Val Acc)** berakhir di **0.6623**. Ini adalah performa yang sangat rendah, artinya model gagal memprediksi sepertiga dari data tes.
    - **Akurasi Training (Train Acc)** juga hanya **0.7394**. Ini menunjukkan model bahkan tidak mampu mempelajari (atau "menghafal") data latihnya sendiri dengan baik.
2. **Grafik Tidak Stabil:**
    - Kurva akurasi validasi tidak naik secara konsisten. Ia sempat mencapai **0.7143** (di *epoch* 12) lalu turun lagi ke **0.6234** (di *epoch* 19). Ini bukan tanda "proses belajar sehat", tapi tanda model kebingungan dan tidak dapat menemukan pola yang stabil.
3. **Model Gagal Konvergen:**

- Meskipun *loss* sama-sama turun, nilainya masih sangat tinggi (Val Loss 0.6269). Ini menunjukkan model masih jauh dari solusi optimal.

C. **Eksperimen untuk meningkatkan performa**. Rancang sejumlah rangkaian eksperimen sistematis:
(a) Ubah *epoch* misal sampai 50 dan amati apakah terjadi *overfitting* (buktikan dengan grafik dan metrik)
(b) Lakukan variasi *hyperparameter* arsitektur, minimal terdapat satu kali perubahan untuk setiap parameter: misal adalah jumlah head (mis. 2,4,8), jumlah *encoder layers* (mis. 1,2,4), *embed_dim* (64,128,256), *dropout* (0.1–0.5). Untuk setiap perubahan, catat waktu *training*, dsb.

Berikut ini beberapa eksperimen yang dicoba:

Nama Eksperimen	embed_ dim	ff_dim	n_heads	num_ layers	drop out	epochs
Model Awal (Baseline)	64	128	4	2	0.1	20
C1_Epochs_50	64	128	4	2	0.1	**50**
C2_Heads_2	64	128	**2**	2	0.1	20
C2_Heads_8	64	128	**8**	2	0.1	20
C2_Layers_1	64	128	4	**1**	0.1	20
C2_Layers_4	64	128	4	**4**	0.1	20
C2_Embed_128	**128**	**256**	4	2	0.1	20
C2_Embed_256	**256**	**512**	4	2	0.1	20
C2_Dropout_0.3	64	128	4	2	**0.3**	20
C2_Dropout_0.5	64	128	4	2	**0.5**	20

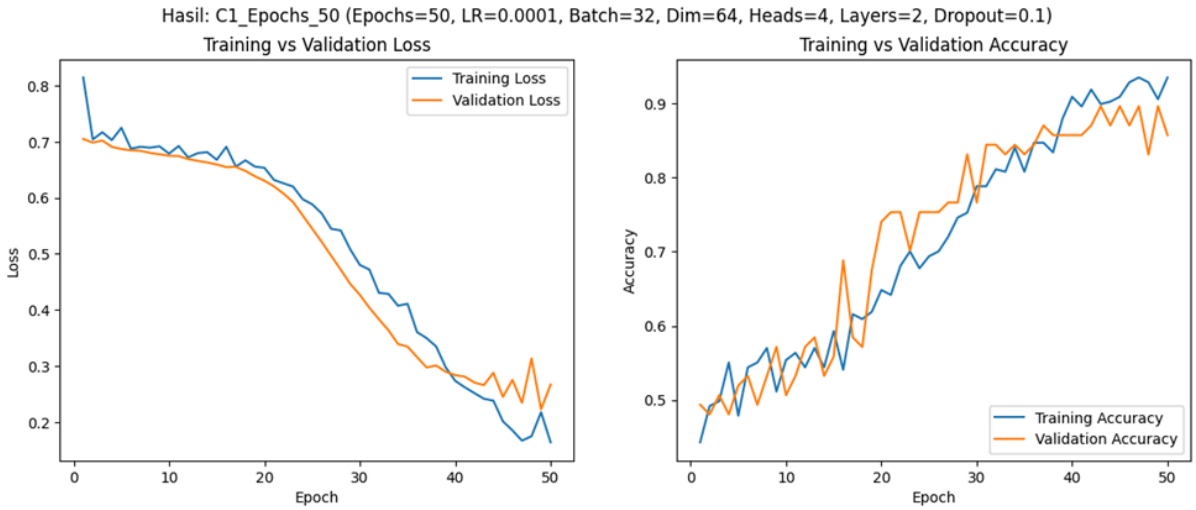
Jawaban Bagian C.a):

1) C1_Epochs_50

Konfigurasi: {'lr': 0.0001, 'batch_size': 32, 'embed_dim': 64, 'n_heads': 4, 'num_layers': 2, 'ff_dim': 128, 'dropout': 0.1, 'epochs': 50, 'weight_decay': 0.0, 'name': 'C1_Epochs_50'}

Hasil Eksperimen C1_Epochs_50:
- **Waktu Training Total**: 307.06 detik

- **Final Train Loss**: 0.1637
- **Final Train Acc**: 0.9349
- **Final Val Loss**: 0.2666
- **Final Val Acc**: 0.8571



Epoch 1/50 | Train Loss: 0.8147 | Train Acc: 0.4430 | Val Loss: 0.7049 | Val Acc: 0.4935
Epoch 2/50 | Train Loss: 0.7042 | Train Acc: 0.4919 | Val Loss: 0.6985 | Val Acc: 0.4805
Epoch 3/50 | Train Loss: 0.7169 | Train Acc: 0.4984 | Val Loss: 0.7023 | Val Acc: 0.5065
Epoch 4/50 | Train Loss: 0.7030 | Train Acc: 0.5505 | Val Loss: 0.6908 | Val Acc: 0.4805
Epoch 5/50 | Train Loss: 0.7250 | Train Acc: 0.4788 | Val Loss: 0.6871 | Val Acc: 0.5195
Epoch 6/50 | Train Loss: 0.6878 | Train Acc: 0.5440 | Val Loss: 0.6848 | Val Acc: 0.5325
Epoch 7/50 | Train Loss: 0.6910 | Train Acc: 0.5505 | Val Loss: 0.6838 | Val Acc: 0.4935
Epoch 8/50 | Train Loss: 0.6894 | Train Acc: 0.5700 | Val Loss: 0.6801 | Val Acc: 0.5325
Epoch 9/50 | Train Loss: 0.6918 | Train Acc: 0.5114 | Val Loss: 0.6776 | Val Acc: 0.5714
Epoch 10/50 | Train Loss: 0.6788 | Train Acc: 0.5537 | Val Loss: 0.6750 | Val Acc: 0.5065
Epoch 11/50 | Train Loss: 0.6924 | Train Acc: 0.5635 | Val Loss: 0.6744 | Val Acc: 0.5325
Epoch 12/50 | Train Loss: 0.6720 | Train Acc: 0.5440 | Val Loss: 0.6690 | Val Acc: 0.5714
Epoch 13/50 | Train Loss: 0.6797 | Train Acc: 0.5700 | Val Loss: 0.6659 | Val Acc: 0.5844
Epoch 14/50 | Train Loss: 0.6814 | Train Acc: 0.5440 | Val Loss: 0.6630 | Val Acc: 0.5325
Epoch 15/50 | Train Loss: 0.6679 | Train Acc: 0.5928 | Val Loss: 0.6594 | Val Acc: 0.5584
Epoch 16/50 | Train Loss: 0.6910 | Train Acc: 0.5407 | Val Loss: 0.6548 | Val Acc: 0.6883
Epoch 17/50 | Train Loss: 0.6556 | Train Acc: 0.6156 | Val Loss: 0.6553 | Val Acc: 0.5844
Epoch 18/50 | Train Loss: 0.6668 | Train Acc: 0.6091 | Val Loss: 0.6479 | Val Acc: 0.5714
Epoch 19/50 | Train Loss: 0.6557 | Train Acc: 0.6189 | Val Loss: 0.6383 | Val Acc: 0.6753
Epoch 20/50 | Train Loss: 0.6535 | Train Acc: 0.6482 | Val Loss: 0.6304 | Val Acc: 0.7403
Epoch 21/50 | Train Loss: 0.6319 | Train Acc: 0.6417 | Val Loss: 0.6204 | Val Acc: 0.7532
Epoch 22/50 | Train Loss: 0.6258 | Train Acc: 0.6808 | Val Loss: 0.6074 | Val Acc: 0.7532
Epoch 23/50 | Train Loss: 0.6201 | Train Acc: 0.7003 | Val Loss: 0.5924 | Val Acc: 0.7013
Epoch 24/50 | Train Loss: 0.5975 | Train Acc: 0.6775 | Val Loss: 0.5686 | Val Acc: 0.7532
Epoch 25/50 | Train Loss: 0.5885 | Train Acc: 0.6938 | Val Loss: 0.5450 | Val Acc: 0.7532
Epoch 26/50 | Train Loss: 0.5722 | Train Acc: 0.7003 | Val Loss: 0.5213 | Val Acc: 0.7532
Epoch 27/50 | Train Loss: 0.5446 | Train Acc: 0.7199 | Val Loss: 0.4963 | Val Acc: 0.7662
Epoch 28/50 | Train Loss: 0.5417 | Train Acc: 0.7459 | Val Loss: 0.4716 | Val Acc: 0.7662
Epoch 29/50 | Train Loss: 0.5079 | Train Acc: 0.7524 | Val Loss: 0.4461 | Val Acc: 0.8312
Epoch 30/50 | Train Loss: 0.4802 | Train Acc: 0.7883 | Val Loss: 0.4268 | Val Acc: 0.7662
Epoch 31/50 | Train Loss: 0.4719 | Train Acc: 0.7883 | Val Loss: 0.4038 | Val Acc: 0.8442

Epoch 32/50 | Train Loss: 0.4304 | Train Acc: 0.8111 | Val Loss: 0.3832 | Val Acc: 0.8442
Epoch 33/50 | Train Loss: 0.4285 | Train Acc: 0.8078 | Val Loss: 0.3636 | Val Acc: 0.8312
Epoch 34/50 | Train Loss: 0.4074 | Train Acc: 0.8404 | Val Loss: 0.3390 | Val Acc: 0.8442
Epoch 35/50 | Train Loss: 0.4108 | Train Acc: 0.8078 | Val Loss: 0.3344 | Val Acc: 0.8312
Epoch 36/50 | Train Loss: 0.3603 | Train Acc: 0.8469 | Val Loss: 0.3157 | Val Acc: 0.8442
Epoch 37/50 | Train Loss: 0.3495 | Train Acc: 0.8469 | Val Loss: 0.2971 | Val Acc: 0.8701
Epoch 38/50 | Train Loss: 0.3347 | Train Acc: 0.8339 | Val Loss: 0.3008 | Val Acc: 0.8571
Epoch 39/50 | Train Loss: 0.2975 | Train Acc: 0.8795 | Val Loss: 0.2895 | Val Acc: 0.8571
Epoch 40/50 | Train Loss: 0.2733 | Train Acc: 0.9088 | Val Loss: 0.2835 | Val Acc: 0.8571
Epoch 41/50 | Train Loss: 0.2619 | Train Acc: 0.8958 | Val Loss: 0.2808 | Val Acc: 0.8571
Epoch 42/50 | Train Loss: 0.2516 | Train Acc: 0.9186 | Val Loss: 0.2704 | Val Acc: 0.8701
Epoch 43/50 | Train Loss: 0.2414 | Train Acc: 0.8990 | Val Loss: 0.2657 | Val Acc: 0.8961
Epoch 44/50 | Train Loss: 0.2380 | Train Acc: 0.9023 | Val Loss: 0.2876 | Val Acc: 0.8701
Epoch 45/50 | Train Loss: 0.2012 | Train Acc: 0.9088 | Val Loss: 0.2447 | Val Acc: 0.8961
Epoch 46/50 | Train Loss: 0.1849 | Train Acc: 0.9283 | Val Loss: 0.2751 | Val Acc: 0.8701
Epoch 47/50 | Train Loss: 0.1666 | Train Acc: 0.9349 | Val Loss: 0.2341 | Val Acc: 0.8961
Epoch 48/50 | Train Loss: 0.1744 | Train Acc: 0.9283 | Val Loss: 0.3131 | Val Acc: 0.8312
Epoch 49/50 | Train Loss: 0.2172 | Train Acc: 0.9055 | Val Loss: 0.2226 | Val Acc: 0.8961
Epoch 50/50 | Train Loss: 0.1637 | Train Acc: 0.9349 | Val Loss: 0.2666 | Val Acc: 0.8571

Ya, hasil eksperimen untuk *epoch* = 50 menunjukkan tanda-tanda ***overfitting***. Alasannya adalah:

1. Kesenjangan (*Gap*) antara *Training* dan Validasi
    - *Final Train Acc*: 0.9349 (sekitar 93.5%)
    - *Final Val Acc*: 0.8571 (sekitar 85.7%)
    - Ada kesenjangan performa yang signifikan (~8%) antara data *training* dan data validasi. Model jadi terlalu "pintar" menghafal *data training*, tapi tidak lagi “pintar” memprediksi data baru (data validasi).
2. Titik Balik (*The Turn*) di Performa Validasi
    - Akurasi Validasi (*Val Acc*):
        - *epoch* 47: mencapai 0.8961
        - *epoch* 48: turun ke 0.8312
        - *epoch* 49: naik lagi ke 0.8961 (ini puncaknya)
        - *epoch* 50: *turun* ke 0.8571
    - *Loss* Validasi (*Val Loss*):
        - *epoch* 47: 0.2341
        - *epoch* 48: naik ke 0.3131
        - *epoch* 49: mencapai 0.2226 (ini titik terendahnya/terbaik)
        - *epoch* 50: *naik* lagi ke 0.2666
    - Setelah *epoch* 49, model tidak lagi membaik di data validasi. Performanya justru mulai memburuk. Sementara itu, performa di *data training* (*train loss*/*acc*) masih terus membaik (*train loss* turun, *train acc* naik).

Model mulai *overfitting* secara jelas setelah *epoch* 49. Model terbaik (*best checkpoint*) sebenarnya ada di *epoch* 49, saat *val_loss* paling rendah (0.2226) dan *val_acc* paling tinggi (0.8961). Di *epoch* 50, modelnya sudah "terlalu jauh" belajar dan performanya di data baru mulai rusak. Untuk eksperimen-eksperimen selanjutnya, bisa pakai *early stopping* yang memonitor *val_loss* atau *val_acc* untuk otomatis berhenti dan menyimpan model di titik terbaik itu (*epoch* 49).

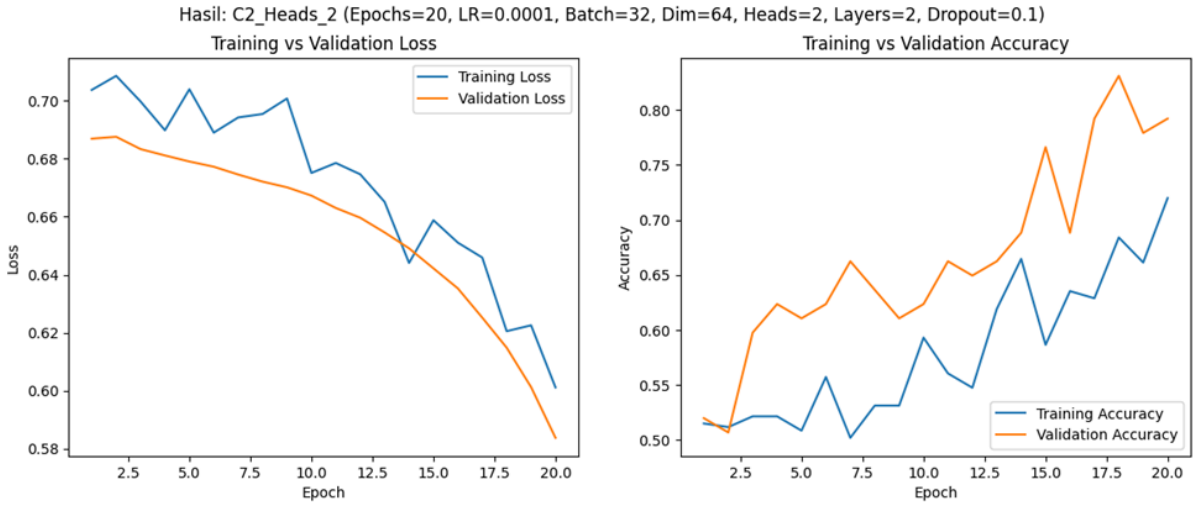
Jawaban Bagian C.b):

2) C2_Heads_2

Konfigurasi: {'lr': 0.0001, 'batch_size': 32, 'embed_dim': 64, 'n_heads': 2, 'num_layers': 2, 'ff_dim': 128, 'dropout': 0.1, 'epochs': 20, 'weight_decay': 0.0, 'name': 'C2_Heads_2'}

Hasil Eksperimen C2_Heads_2:

- **Waktu Training Total**: 422.71 detik
- **Final Train Loss**: 0.6011
- **Final Train Acc**: 0.7199
- **Final Val Loss**: 0.5837
- **Final Val Acc**: 0.7922



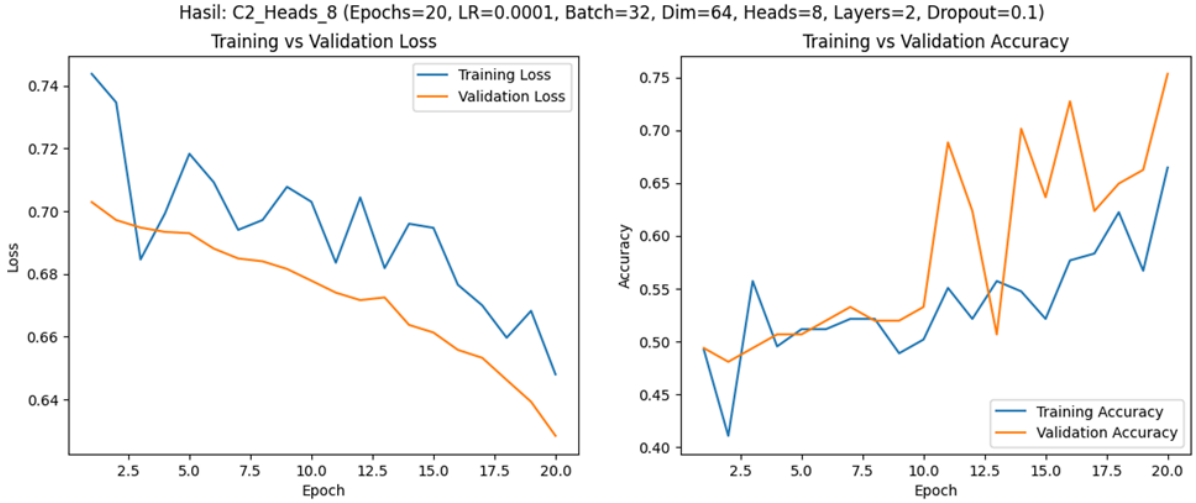
Epoch 1/20 | Train Loss: 0.7038 | Train Acc: 0.5147 | Val Loss: 0.6870 | Val Acc: 0.5195
Epoch 2/20 | Train Loss: 0.7086 | Train Acc: 0.5114 | Val Loss: 0.6876 | Val Acc: 0.5065
Epoch 3/20 | Train Loss: 0.6998 | Train Acc: 0.5212 | Val Loss: 0.6834 | Val Acc: 0.5974
Epoch 4/20 | Train Loss: 0.6898 | Train Acc: 0.5212 | Val Loss: 0.6812 | Val Acc: 0.6234
Epoch 5/20 | Train Loss: 0.7040 | Train Acc: 0.5081 | Val Loss: 0.6791 | Val Acc: 0.6104
Epoch 6/20 | Train Loss: 0.6890 | Train Acc: 0.5570 | Val Loss: 0.6773 | Val Acc: 0.6234
Epoch 7/20 | Train Loss: 0.6943 | Train Acc: 0.5016 | Val Loss: 0.6746 | Val Acc: 0.6623
Epoch 8/20 | Train Loss: 0.6955 | Train Acc: 0.5309 | Val Loss: 0.6721 | Val Acc: 0.6364
Epoch 9/20 | Train Loss: 0.7008 | Train Acc: 0.5309 | Val Loss: 0.6702 | Val Acc: 0.6104
Epoch 10/20 | Train Loss: 0.6751 | Train Acc: 0.5928 | Val Loss: 0.6673 | Val Acc: 0.6234
Epoch 11/20 | Train Loss: 0.6786 | Train Acc: 0.5603 | Val Loss: 0.6630 | Val Acc: 0.6623
Epoch 12/20 | Train Loss: 0.6747 | Train Acc: 0.5472 | Val Loss: 0.6597 | Val Acc: 0.6494
Epoch 13/20 | Train Loss: 0.6651 | Train Acc: 0.6189 | Val Loss: 0.6546 | Val Acc: 0.6623
Epoch 14/20 | Train Loss: 0.6441 | Train Acc: 0.6645 | Val Loss: 0.6491 | Val Acc: 0.6883
Epoch 15/20 | Train Loss: 0.6588 | Train Acc: 0.5863 | Val Loss: 0.6422 | Val Acc: 0.7662
Epoch 16/20 | Train Loss: 0.6511 | Train Acc: 0.6352 | Val Loss: 0.6353 | Val Acc: 0.6883
Epoch 17/20 | Train Loss: 0.6459 | Train Acc: 0.6287 | Val Loss: 0.6252 | Val Acc: 0.7922
Epoch 18/20 | Train Loss: 0.6205 | Train Acc: 0.6840 | Val Loss: 0.6148 | Val Acc: 0.8312
Epoch 19/20 | Train Loss: 0.6225 | Train Acc: 0.6612 | Val Loss: 0.6012 | Val Acc: 0.7792
Epoch 20/20 | Train Loss: 0.6011 | Train Acc: 0.7199 | Val Loss: 0.5837 | Val Acc: 0.7922

3) C2_Heads_8

Konfigurasi: {'lr': 0.0001, 'batch_size': 32, 'embed_dim': 64, 'n_heads': 8, 'num_layers': 2, 'ff_dim': 128, 'dropout': 0.1, 'epochs': 20, 'weight_decay': 0.0, 'name': 'C2_Heads_8'}

Hasil Eksperimen C2_Heads_8:

- **Waktu Training Total**: 406.20 detik
- **Final Train Loss**: 0.6479
- **Final Train Acc**: 0.6645
- **Final Val Loss**: 0.6283
- **Final Val Acc**: 0.7532



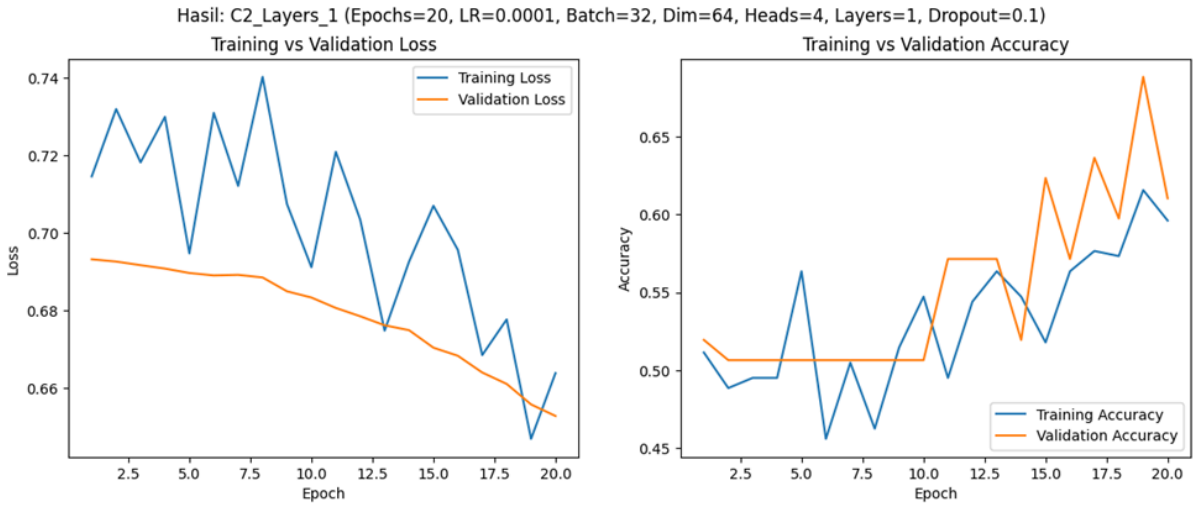
Epoch 1/20 | Train Loss: 0.7437 | Train Acc: 0.4919 | Val Loss: 0.7028 | Val Acc: 0.4935
Epoch 2/20 | Train Loss: 0.7346 | Train Acc: 0.4104 | Val Loss: 0.6971 | Val Acc: 0.4805
Epoch 3/20 | Train Loss: 0.6846 | Train Acc: 0.5570 | Val Loss: 0.6947 | Val Acc: 0.4935
Epoch 4/20 | Train Loss: 0.6992 | Train Acc: 0.4951 | Val Loss: 0.6933 | Val Acc: 0.5065
Epoch 5/20 | Train Loss: 0.7183 | Train Acc: 0.5114 | Val Loss: 0.6929 | Val Acc: 0.5065
Epoch 6/20 | Train Loss: 0.7091 | Train Acc: 0.5114 | Val Loss: 0.6880 | Val Acc: 0.5195
Epoch 7/20 | Train Loss: 0.6940 | Train Acc: 0.5212 | Val Loss: 0.6849 | Val Acc: 0.5325
Epoch 8/20 | Train Loss: 0.6971 | Train Acc: 0.5212 | Val Loss: 0.6840 | Val Acc: 0.5195
Epoch 9/20 | Train Loss: 0.7077 | Train Acc: 0.4886 | Val Loss: 0.6815 | Val Acc: 0.5195
Epoch 10/20 | Train Loss: 0.7029 | Train Acc: 0.5016 | Val Loss: 0.6778 | Val Acc: 0.5325
Epoch 11/20 | Train Loss: 0.6835 | Train Acc: 0.5505 | Val Loss: 0.6740 | Val Acc: 0.6883
Epoch 12/20 | Train Loss: 0.7043 | Train Acc: 0.5212 | Val Loss: 0.6716 | Val Acc: 0.6234
Epoch 13/20 | Train Loss: 0.6818 | Train Acc: 0.5570 | Val Loss: 0.6725 | Val Acc: 0.5065
Epoch 14/20 | Train Loss: 0.6959 | Train Acc: 0.5472 | Val Loss: 0.6637 | Val Acc: 0.7013
Epoch 15/20 | Train Loss: 0.6946 | Train Acc: 0.5212 | Val Loss: 0.6613 | Val Acc: 0.6364
Epoch 16/20 | Train Loss: 0.6765 | Train Acc: 0.5765 | Val Loss: 0.6557 | Val Acc: 0.7273
Epoch 17/20 | Train Loss: 0.6699 | Train Acc: 0.5831 | Val Loss: 0.6532 | Val Acc: 0.6234
Epoch 18/20 | Train Loss: 0.6596 | Train Acc: 0.6221 | Val Loss: 0.6461 | Val Acc: 0.6494
Epoch 19/20 | Train Loss: 0.6682 | Train Acc: 0.5668 | Val Loss: 0.6392 | Val Acc: 0.6623
Epoch 20/20 | Train Loss: 0.6479 | Train Acc: 0.6645 | Val Loss: 0.6283 | Val Acc: 0.7532

4) C2_Layers_1

Konfigurasi: {'lr': 0.0001, 'batch_size': 32, 'embed_dim': 64, 'n_heads': 4, 'num_layers': 1, 'ff_dim': 128, 'dropout': 0.1, 'epochs': 20, 'weight_decay': 0.0, 'name': 'C2_Layers_1'}

Hasil Eksperimen C2_Layers_1:

- **Waktu Training Total**: 327.46 detik
- **Final Train Loss**: 0.6639
- **Final Train Acc**: 0.5961
- **Final Val Loss**: 0.6529
- **Final Val Acc**: 0.6104



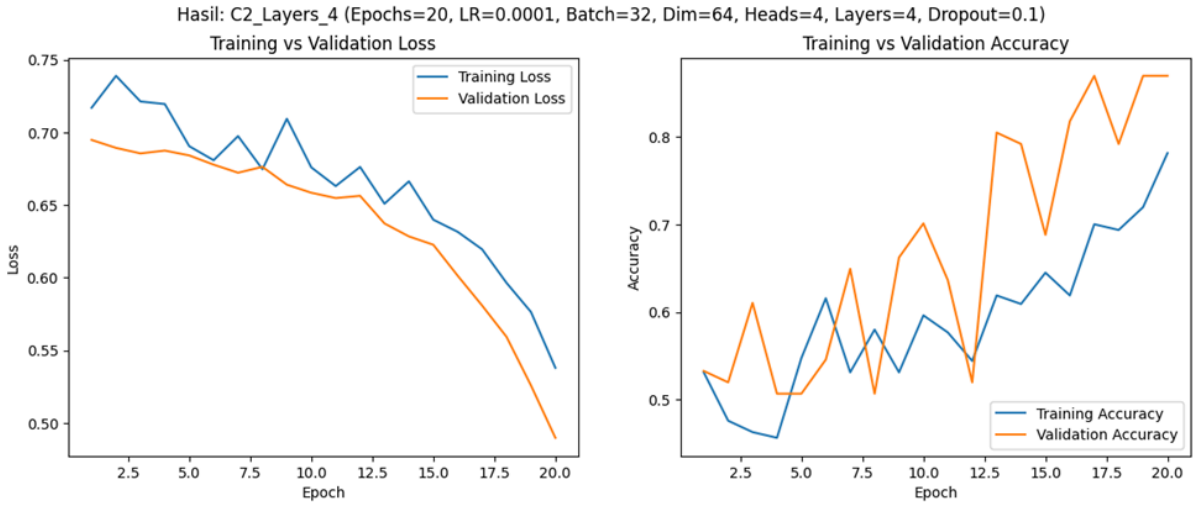
Epoch 1/20 | Train Loss: 0.7146 | Train Acc: 0.5114 | Val Loss: 0.6932 | Val Acc: 0.5195
Epoch 2/20 | Train Loss: 0.7319 | Train Acc: 0.4886 | Val Loss: 0.6926 | Val Acc: 0.5065
Epoch 3/20 | Train Loss: 0.7182 | Train Acc: 0.4951 | Val Loss: 0.6917 | Val Acc: 0.5065
Epoch 4/20 | Train Loss: 0.7299 | Train Acc: 0.4951 | Val Loss: 0.6908 | Val Acc: 0.5065
Epoch 5/20 | Train Loss: 0.6947 | Train Acc: 0.5635 | Val Loss: 0.6897 | Val Acc: 0.5065
Epoch 6/20 | Train Loss: 0.7309 | Train Acc: 0.4560 | Val Loss: 0.6891 | Val Acc: 0.5065
Epoch 7/20 | Train Loss: 0.7121 | Train Acc: 0.5049 | Val Loss: 0.6892 | Val Acc: 0.5065
Epoch 8/20 | Train Loss: 0.7402 | Train Acc: 0.4625 | Val Loss: 0.6885 | Val Acc: 0.5065
Epoch 9/20 | Train Loss: 0.7074 | Train Acc: 0.5147 | Val Loss: 0.6850 | Val Acc: 0.5065
Epoch 10/20 | Train Loss: 0.6912 | Train Acc: 0.5472 | Val Loss: 0.6834 | Val Acc: 0.5065
Epoch 11/20 | Train Loss: 0.7209 | Train Acc: 0.4951 | Val Loss: 0.6807 | Val Acc: 0.5714
Epoch 12/20 | Train Loss: 0.7034 | Train Acc: 0.5440 | Val Loss: 0.6785 | Val Acc: 0.5714
Epoch 13/20 | Train Loss: 0.6749 | Train Acc: 0.5635 | Val Loss: 0.6762 | Val Acc: 0.5714
Epoch 14/20 | Train Loss: 0.6926 | Train Acc: 0.5472 | Val Loss: 0.6750 | Val Acc: 0.5195
Epoch 15/20 | Train Loss: 0.7070 | Train Acc: 0.5179 | Val Loss: 0.6705 | Val Acc: 0.6234
Epoch 16/20 | Train Loss: 0.6956 | Train Acc: 0.5635 | Val Loss: 0.6684 | Val Acc: 0.5714
Epoch 17/20 | Train Loss: 0.6686 | Train Acc: 0.5765 | Val Loss: 0.6641 | Val Acc: 0.6364
Epoch 18/20 | Train Loss: 0.6777 | Train Acc: 0.5733 | Val Loss: 0.6611 | Val Acc: 0.5974
Epoch 19/20 | Train Loss: 0.6470 | Train Acc: 0.6156 | Val Loss: 0.6559 | Val Acc: 0.6883
Epoch 20/20 | Train Loss: 0.6639 | Train Acc: 0.5961 | Val Loss: 0.6529 | Val Acc: 0.6104

5) C2_Layers_4

Konfigurasi: {'lr': 0.0001, 'batch_size': 32, 'embed_dim': 64, 'n_heads': 4, 'num_layers': 4, 'ff_dim': 128, 'dropout': 0.1, 'epochs': 20, 'weight_decay': 0.0, 'name': 'C2_Layers_4'}

Hasil Eksperimen C2_Layers_4:

- **Waktu Training Total**: 611.49 detik
- **Final Train Loss**: 0.5380
- **Final Train Acc**: 0.7818
- **Final Val Loss**: 0.4898
- **Final Val Acc**: 0.8701



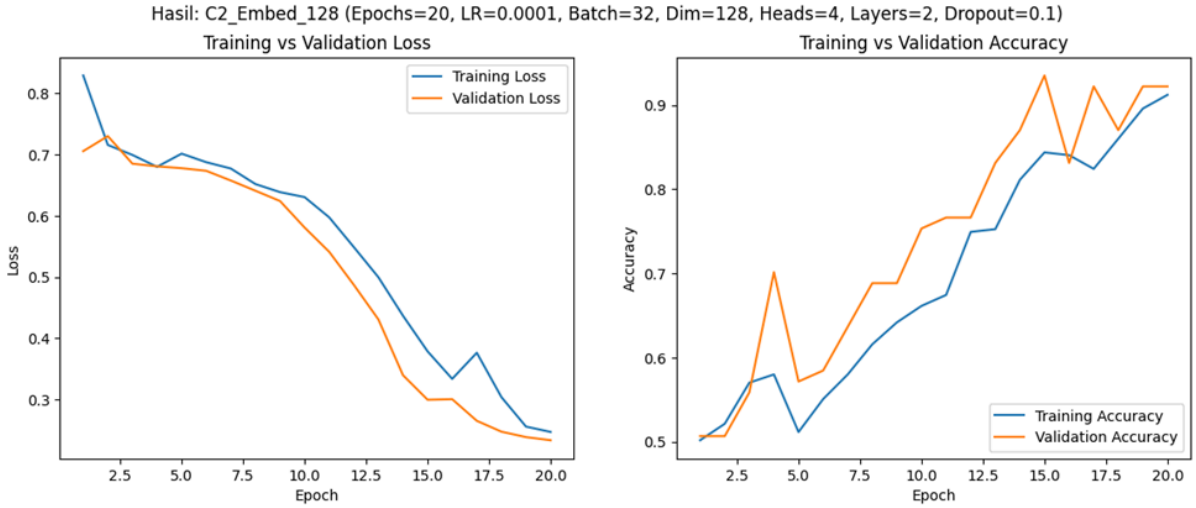
Epoch 1/20 | Train Loss: 0.7170 | Train Acc: 0.5309 | Val Loss: 0.6949 | Val Acc: 0.5325
Epoch 2/20 | Train Loss: 0.7390 | Train Acc: 0.4756 | Val Loss: 0.6894 | Val Acc: 0.5195
Epoch 3/20 | Train Loss: 0.7213 | Train Acc: 0.4625 | Val Loss: 0.6856 | Val Acc: 0.6104
Epoch 4/20 | Train Loss: 0.7196 | Train Acc: 0.4560 | Val Loss: 0.6875 | Val Acc: 0.5065
Epoch 5/20 | Train Loss: 0.6905 | Train Acc: 0.5472 | Val Loss: 0.6842 | Val Acc: 0.5065
Epoch 6/20 | Train Loss: 0.6809 | Train Acc: 0.6156 | Val Loss: 0.6778 | Val Acc: 0.5455
Epoch 7/20 | Train Loss: 0.6975 | Train Acc: 0.5309 | Val Loss: 0.6723 | Val Acc: 0.6494
Epoch 8/20 | Train Loss: 0.6747 | Train Acc: 0.5798 | Val Loss: 0.6763 | Val Acc: 0.5065
Epoch 9/20 | Train Loss: 0.7094 | Train Acc: 0.5309 | Val Loss: 0.6641 | Val Acc: 0.6623
Epoch 10/20 | Train Loss: 0.6760 | Train Acc: 0.5961 | Val Loss: 0.6585 | Val Acc: 0.7013
Epoch 11/20 | Train Loss: 0.6631 | Train Acc: 0.5765 | Val Loss: 0.6549 | Val Acc: 0.6364
Epoch 12/20 | Train Loss: 0.6762 | Train Acc: 0.5440 | Val Loss: 0.6564 | Val Acc: 0.5195
Epoch 13/20 | Train Loss: 0.6510 | Train Acc: 0.6189 | Val Loss: 0.6374 | Val Acc: 0.8052
Epoch 14/20 | Train Loss: 0.6664 | Train Acc: 0.6091 | Val Loss: 0.6284 | Val Acc: 0.7922
Epoch 15/20 | Train Loss: 0.6399 | Train Acc: 0.6450 | Val Loss: 0.6227 | Val Acc: 0.6883
Epoch 16/20 | Train Loss: 0.6316 | Train Acc: 0.6189 | Val Loss: 0.6014 | Val Acc: 0.8182
Epoch 17/20 | Train Loss: 0.6196 | Train Acc: 0.7003 | Val Loss: 0.5809 | Val Acc: 0.8701
Epoch 18/20 | Train Loss: 0.5965 | Train Acc: 0.6938 | Val Loss: 0.5593 | Val Acc: 0.7922
Epoch 19/20 | Train Loss: 0.5765 | Train Acc: 0.7199 | Val Loss: 0.5259 | Val Acc: 0.8701
Epoch 20/20 | Train Loss: 0.5380 | Train Acc: 0.7818 | Val Loss: 0.4898 | Val Acc: 0.8701

6) C2_Embed_128

Konfigurasi: {'lr': 0.0001, 'batch_size': 32, 'embed_dim': 128, 'n_heads': 4, 'num_layers': 2, 'ff_dim': 256, 'dropout': 0.1, 'epochs': 20, 'weight_decay': 0.0, 'name': 'C2_Embed_128'}

Hasil Eksperimen C2_Embed_128:

- **Waktu Training Total**: 642.84 detik
- **Final Train Loss**: 0.2474
- **Final Train Acc**: 0.9121
- **Final Val Loss**: 0.2339
- **Final Val Acc**: 0.9221



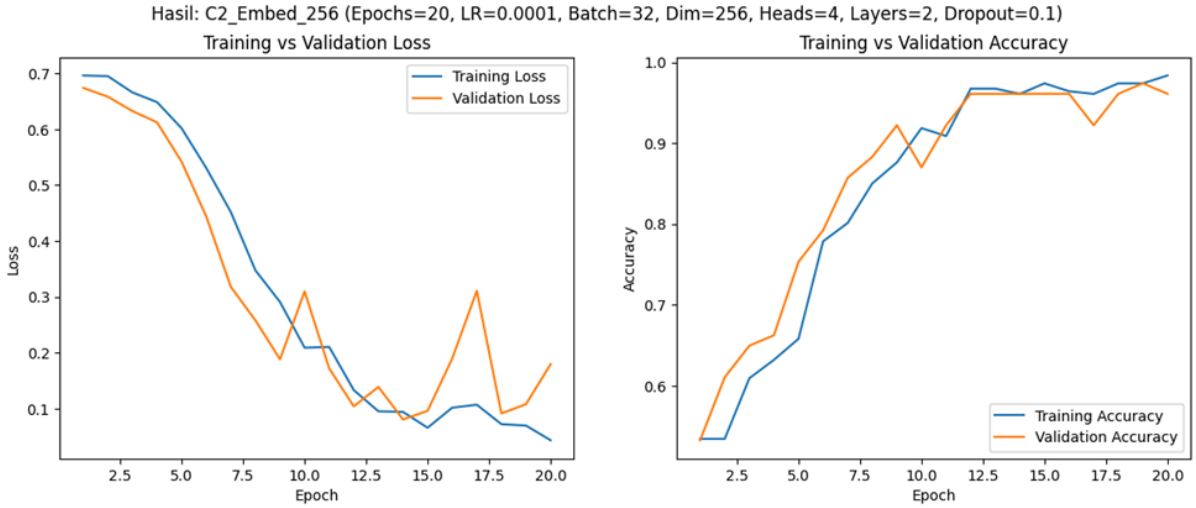
Epoch 1/20 | Train Loss: 0.8289 | Train Acc: 0.5016 | Val Loss: 0.7054 | Val Acc: 0.5065
Epoch 2/20 | Train Loss: 0.7155 | Train Acc: 0.5212 | Val Loss: 0.7300 | Val Acc: 0.5065
Epoch 3/20 | Train Loss: 0.6992 | Train Acc: 0.5700 | Val Loss: 0.6850 | Val Acc: 0.5584
Epoch 4/20 | Train Loss: 0.6799 | Train Acc: 0.5798 | Val Loss: 0.6806 | Val Acc: 0.7013
Epoch 5/20 | Train Loss: 0.7013 | Train Acc: 0.5114 | Val Loss: 0.6776 | Val Acc: 0.5714
Epoch 6/20 | Train Loss: 0.6874 | Train Acc: 0.5505 | Val Loss: 0.6733 | Val Acc: 0.5844
Epoch 7/20 | Train Loss: 0.6771 | Train Acc: 0.5798 | Val Loss: 0.6575 | Val Acc: 0.6364
Epoch 8/20 | Train Loss: 0.6518 | Train Acc: 0.6156 | Val Loss: 0.6408 | Val Acc: 0.6883
Epoch 9/20 | Train Loss: 0.6386 | Train Acc: 0.6417 | Val Loss: 0.6241 | Val Acc: 0.6883
Epoch 10/20 | Train Loss: 0.6304 | Train Acc: 0.6612 | Val Loss: 0.5808 | Val Acc: 0.7532
Epoch 11/20 | Train Loss: 0.5974 | Train Acc: 0.6743 | Val Loss: 0.5412 | Val Acc: 0.7662
Epoch 12/20 | Train Loss: 0.5489 | Train Acc: 0.7492 | Val Loss: 0.4874 | Val Acc: 0.7662
Epoch 13/20 | Train Loss: 0.4997 | Train Acc: 0.7524 | Val Loss: 0.4309 | Val Acc: 0.8312
Epoch 14/20 | Train Loss: 0.4369 | Train Acc: 0.8111 | Val Loss: 0.3399 | Val Acc: 0.8701
Epoch 15/20 | Train Loss: 0.3788 | Train Acc: 0.8436 | Val Loss: 0.2999 | Val Acc: 0.9351
Epoch 16/20 | Train Loss: 0.3340 | Train Acc: 0.8404 | Val Loss: 0.3008 | Val Acc: 0.8312
Epoch 17/20 | Train Loss: 0.3765 | Train Acc: 0.8241 | Val Loss: 0.2656 | Val Acc: 0.9221
Epoch 18/20 | Train Loss: 0.3042 | Train Acc: 0.8599 | Val Loss: 0.2478 | Val Acc: 0.8701
Epoch 19/20 | Train Loss: 0.2561 | Train Acc: 0.8958 | Val Loss: 0.2389 | Val Acc: 0.9221
Epoch 20/20 | Train Loss: 0.2474 | Train Acc: 0.9121 | Val Loss: 0.2339 | Val Acc: 0.9221

7) C2_Embed_256

Konfigurasi: {'lr': 0.0001, 'batch_size': 32, 'embed_dim': 256, 'n_heads': 4, 'num_layers': 2, 'ff_dim': 512, 'dropout': 0.1, 'epochs': 20, 'weight_decay': 0.0, 'name': 'C2_Embed_256'}

Hasil Eksperimen C2_Embed_256:

- **Waktu Training Total**: 528.87 detik
- **Final Train Loss**: 0.0434
- **Final Train Acc**: 0.9837
- **Final Val Loss**: 0.1794
- **Final Val Acc**: 0.9610



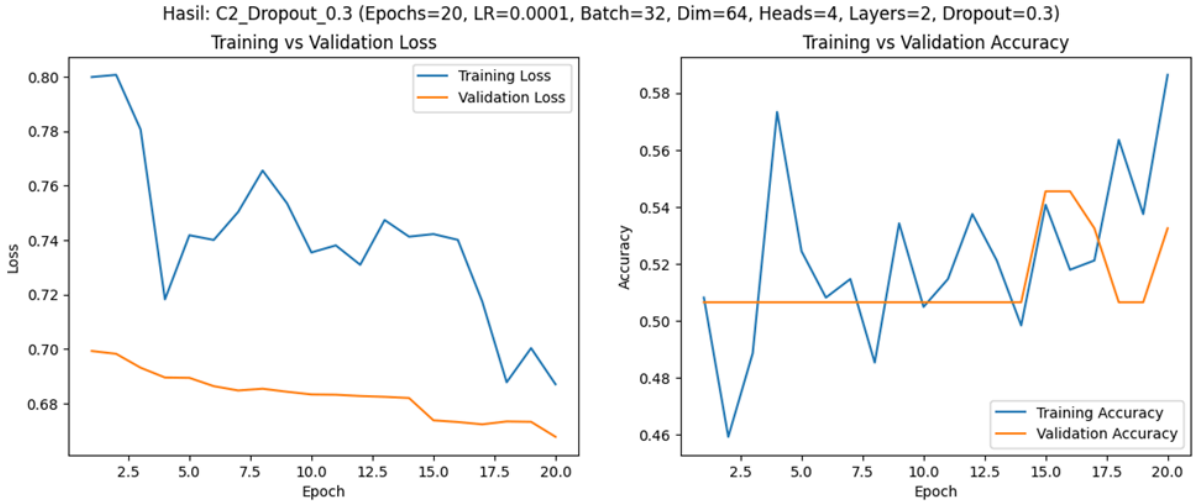
Epoch 1/20 | Train Loss: 0.6962 | Train Acc: 0.5342 | Val Loss: 0.6739 | Val Acc: 0.5325
Epoch 2/20 | Train Loss: 0.6950 | Train Acc: 0.5342 | Val Loss: 0.6582 | Val Acc: 0.6104
Epoch 3/20 | Train Loss: 0.6658 | Train Acc: 0.6091 | Val Loss: 0.6325 | Val Acc: 0.6494
Epoch 4/20 | Train Loss: 0.6485 | Train Acc: 0.6319 | Val Loss: 0.6123 | Val Acc: 0.6623
Epoch 5/20 | Train Loss: 0.6016 | Train Acc: 0.6580 | Val Loss: 0.5418 | Val Acc: 0.7532
Epoch 6/20 | Train Loss: 0.5307 | Train Acc: 0.7785 | Val Loss: 0.4439 | Val Acc: 0.7922
Epoch 7/20 | Train Loss: 0.4518 | Train Acc: 0.8013 | Val Loss: 0.3180 | Val Acc: 0.8571
Epoch 8/20 | Train Loss: 0.3474 | Train Acc: 0.8502 | Val Loss: 0.2579 | Val Acc: 0.8831
Epoch 9/20 | Train Loss: 0.2908 | Train Acc: 0.8762 | Val Loss: 0.1882 | Val Acc: 0.9221
Epoch 10/20 | Train Loss: 0.2090 | Train Acc: 0.9186 | Val Loss: 0.3097 | Val Acc: 0.8701
Epoch 11/20 | Train Loss: 0.2103 | Train Acc: 0.9088 | Val Loss: 0.1719 | Val Acc: 0.9221
Epoch 12/20 | Train Loss: 0.1330 | Train Acc: 0.9674 | Val Loss: 0.1040 | Val Acc: 0.9610
Epoch 13/20 | Train Loss: 0.0952 | Train Acc: 0.9674 | Val Loss: 0.1388 | Val Acc: 0.9610
Epoch 14/20 | Train Loss: 0.0941 | Train Acc: 0.9609 | Val Loss: 0.0806 | Val Acc: 0.9610
Epoch 15/20 | Train Loss: 0.0657 | Train Acc: 0.9739 | Val Loss: 0.0960 | Val Acc: 0.9610
Epoch 16/20 | Train Loss: 0.1016 | Train Acc: 0.9642 | Val Loss: 0.1895 | Val Acc: 0.9610
Epoch 17/20 | Train Loss: 0.1070 | Train Acc: 0.9609 | Val Loss: 0.3108 | Val Acc: 0.9221
Epoch 18/20 | Train Loss: 0.0721 | Train Acc: 0.9739 | Val Loss: 0.0913 | Val Acc: 0.9610
Epoch 19/20 | Train Loss: 0.0696 | Train Acc: 0.9739 | Val Loss: 0.1079 | Val Acc: 0.9740
Epoch 20/20 | Train Loss: 0.0434 | Train Acc: 0.9837 | Val Loss: 0.1794 | Val Acc: 0.9610

8) C2_Dropout_0.3

Konfigurasi: {'lr': 0.0001, 'batch_size': 32, 'embed_dim': 64, 'n_heads': 4, 'num_layers': 2, 'ff_dim': 128, 'dropout': 0.3, 'epochs': 20, 'weight_decay': 0.0, 'name': 'C2_Dropout_0.3'}

Hasil Eksperimen C2_Dropout_0.3:

- **Waktu Training Total**: 118.37 detik
- **Final Train Loss**: 0.6870
- **Final Train Acc**: 0.5863
- **Final Val Loss**: 0.6677
- **Final Val Acc**: 0.5325



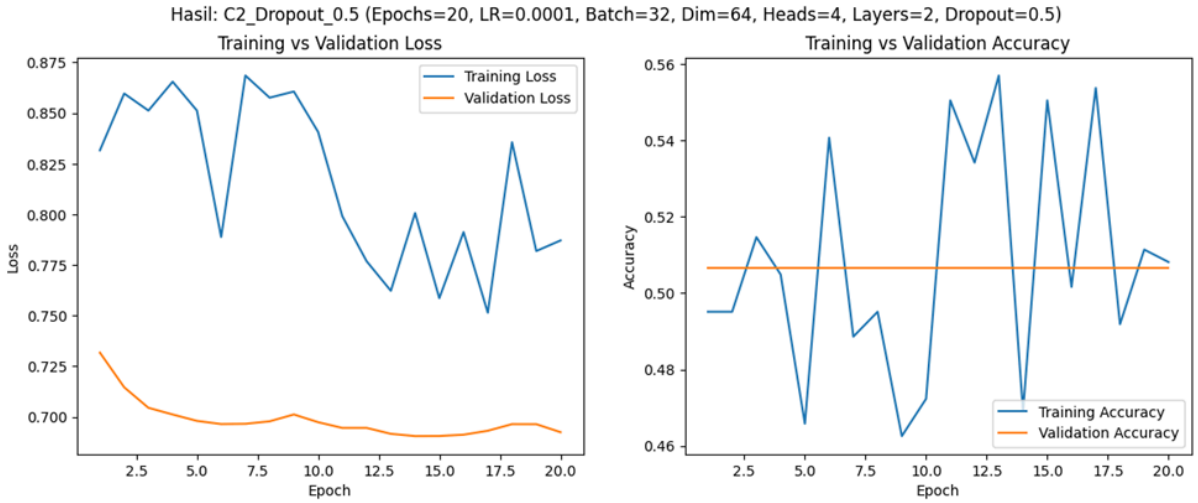
Epoch 1/20 | Train Loss: 0.7998 | Train Acc: 0.5081 | Val Loss: 0.6992 | Val Acc: 0.5065
Epoch 2/20 | Train Loss: 0.8006 | Train Acc: 0.4593 | Val Loss: 0.6982 | Val Acc: 0.5065
Epoch 3/20 | Train Loss: 0.7806 | Train Acc: 0.4886 | Val Loss: 0.6931 | Val Acc: 0.5065
Epoch 4/20 | Train Loss: 0.7182 | Train Acc: 0.5733 | Val Loss: 0.6895 | Val Acc: 0.5065
Epoch 5/20 | Train Loss: 0.7417 | Train Acc: 0.5244 | Val Loss: 0.6894 | Val Acc: 0.5065
Epoch 6/20 | Train Loss: 0.7400 | Train Acc: 0.5081 | Val Loss: 0.6863 | Val Acc: 0.5065
Epoch 7/20 | Train Loss: 0.7503 | Train Acc: 0.5147 | Val Loss: 0.6847 | Val Acc: 0.5065
Epoch 8/20 | Train Loss: 0.7655 | Train Acc: 0.4853 | Val Loss: 0.6854 | Val Acc: 0.5065
Epoch 9/20 | Train Loss: 0.7535 | Train Acc: 0.5342 | Val Loss: 0.6843 | Val Acc: 0.5065
Epoch 10/20 | Train Loss: 0.7354 | Train Acc: 0.5049 | Val Loss: 0.6833 | Val Acc: 0.5065
Epoch 11/20 | Train Loss: 0.7380 | Train Acc: 0.5147 | Val Loss: 0.6832 | Val Acc: 0.5065
Epoch 12/20 | Train Loss: 0.7309 | Train Acc: 0.5375 | Val Loss: 0.6827 | Val Acc: 0.5065
Epoch 13/20 | Train Loss: 0.7473 | Train Acc: 0.5212 | Val Loss: 0.6824 | Val Acc: 0.5065
Epoch 14/20 | Train Loss: 0.7412 | Train Acc: 0.4984 | Val Loss: 0.6820 | Val Acc: 0.5065
Epoch 15/20 | Train Loss: 0.7422 | Train Acc: 0.5407 | Val Loss: 0.6738 | Val Acc: 0.5455
Epoch 16/20 | Train Loss: 0.7400 | Train Acc: 0.5179 | Val Loss: 0.6731 | Val Acc: 0.5455
Epoch 17/20 | Train Loss: 0.7174 | Train Acc: 0.5212 | Val Loss: 0.6723 | Val Acc: 0.5325
Epoch 18/20 | Train Loss: 0.6877 | Train Acc: 0.5635 | Val Loss: 0.6734 | Val Acc: 0.5065
Epoch 19/20 | Train Loss: 0.7003 | Train Acc: 0.5375 | Val Loss: 0.6732 | Val Acc: 0.5065
Epoch 20/20 | Train Loss: 0.6870 | Train Acc: 0.5863 | Val Loss: 0.6677 | Val Acc: 0.5325

9) C2_Dropout_0.5

Konfigurasi: {'lr': 0.0001, 'batch_size': 32, 'embed_dim': 64, 'n_heads': 4, 'num_layers': 2, 'ff_dim': 128, 'dropout': 0.5, 'epochs': 20, 'weight_decay': 0.0, 'name': 'C2_Dropout_0.5'}

Hasil Eksperimen C2_Dropout_0.5:

- **Waktu Training Total**: 150.39 detik
- **Final Train Loss**: 0.7871
- **Final Train Acc**: 0.5081
- **Final Val Loss**: 0.6925
- **Final Val Acc**: 0.5065



Epoch 1/20 | Train Loss: 0.8316 | Train Acc: 0.4951 | Val Loss: 0.7317 | Val Acc: 0.5065
Epoch 2/20 | Train Loss: 0.8596 | Train Acc: 0.4951 | Val Loss: 0.7146 | Val Acc: 0.5065
Epoch 3/20 | Train Loss: 0.8512 | Train Acc: 0.5147 | Val Loss: 0.7045 | Val Acc: 0.5065
Epoch 4/20 | Train Loss: 0.8655 | Train Acc: 0.5049 | Val Loss: 0.7012 | Val Acc: 0.5065
Epoch 5/20 | Train Loss: 0.8512 | Train Acc: 0.4658 | Val Loss: 0.6980 | Val Acc: 0.5065
Epoch 6/20 | Train Loss: 0.7888 | Train Acc: 0.5407 | Val Loss: 0.6965 | Val Acc: 0.5065
Epoch 7/20 | Train Loss: 0.8685 | Train Acc: 0.4886 | Val Loss: 0.6966 | Val Acc: 0.5065
Epoch 8/20 | Train Loss: 0.8576 | Train Acc: 0.4951 | Val Loss: 0.6978 | Val Acc: 0.5065
Epoch 9/20 | Train Loss: 0.8606 | Train Acc: 0.4625 | Val Loss: 0.7012 | Val Acc: 0.5065
Epoch 10/20 | Train Loss: 0.8406 | Train Acc: 0.4723 | Val Loss: 0.6974 | Val Acc: 0.5065
Epoch 11/20 | Train Loss: 0.7990 | Train Acc: 0.5505 | Val Loss: 0.6946 | Val Acc: 0.5065
Epoch 12/20 | Train Loss: 0.7769 | Train Acc: 0.5342 | Val Loss: 0.6946 | Val Acc: 0.5065
Epoch 13/20 | Train Loss: 0.7623 | Train Acc: 0.5570 | Val Loss: 0.6917 | Val Acc: 0.5065
Epoch 14/20 | Train Loss: 0.8006 | Train Acc: 0.4691 | Val Loss: 0.6905 | Val Acc: 0.5065
Epoch 15/20 | Train Loss: 0.7586 | Train Acc: 0.5505 | Val Loss: 0.6906 | Val Acc: 0.5065
Epoch 16/20 | Train Loss: 0.7912 | Train Acc: 0.5016 | Val Loss: 0.6912 | Val Acc: 0.5065
Epoch 17/20 | Train Loss: 0.7514 | Train Acc: 0.5537 | Val Loss: 0.6932 | Val Acc: 0.5065
Epoch 18/20 | Train Loss: 0.8356 | Train Acc: 0.4919 | Val Loss: 0.6964 | Val Acc: 0.5065
Epoch 19/20 | Train Loss: 0.7819 | Train Acc: 0.5114 | Val Loss: 0.6964 | Val Acc: 0.5065
Epoch 20/20 | Train Loss: 0.7871 | Train Acc: 0.5081 | Val Loss: 0.6925 | Val Acc: 0.5065

RINGKASAN EKSPERIMEN

Epoch	Nama Eksperimen	Waktu (detik)	Val Acc Akhir
20	***Baseline***	Tidak dihitung	0.6623
50	**C1_Epochs_50**	307.06	0.8571
20	**C2_Heads_2**	422.71	0.7922
	C2_Heads_8	406.20	0.7532
	C2_Layers_1	327.46	0.6104
	C2_Layers_4	611.49	0.8701
	C2_Embed_128	642.84	0.9221
	C2_Embed_256	528.87	0.9610
	C2_Dropout_0.3	118.37	0.5325
	C2_Dropout_0.5	150.39	0.5065

D. **Analisis *overfitting***. Dari sejumlah grafik yang telah dibuat, jelaskan apakah ukuran *metrics* menurun, tunjukkan langkah mitigasi yang dicoba dan efeknya terhadap upaya menanggulangi *overfitting* tersebut.

Jawaban Bagian D:

Nama Eksperimen	*Overfitting* / Tidak?	Alasan
Baseline	**Tidak (*Underfit*)**	Performa sangat rendah (Val Acc berakhir di 0.6623), artinya model gagal memprediksi sepertiga dari data tes. Akurasi Training (Train Acc) juga hanya 0.7394, model tidak mampu mempelajari (atau "menghafal") data latihnya sendiri dengan baik.
C1_Epochs_50	**Ya (Jelas)**	Ada *gap* signifikan (Train Acc 0.9349 vs Val Acc 0.8571). Val Loss juga naik drastis di *epoch* 48 (dari 0.23 ke 0.31) setelah lama turun, tanda jelas model sudah melewati titik optimalnya.

C2_Heads_2	**Tidak (*Underfit*)**	Performa masih rendah (Val Acc 0.7922) dan *gap* tidak ada (Val Acc justru lebih tinggi dari Train Acc). Model belum belajar optimal.
C2_Heads_8	**Tidak (*Underfit*)**	Sama seperti C2_Heads_2, performa rendah (Val Acc 0.7532) dan Val Acc lebih tinggi dari Train Acc. Model gagal belajar.
C2_Layers_1	**Tidak (Sangat *Underfit*)**	Performa sangat buruk (Val Acc 0.6104). Model dengan 1 layer tidak cukup kompleks untuk mempelajari data.
C2_Layers_4	**Tidak (Sehat)**	Performa bagus (**Val Acc 0.8701**) dan tidak ada *overfitting*. **Val Acc (0.8701)** justru lebih tinggi dari Train Acc (0.7818), menunjukkan generalisasi yang baik (atau data validasi yang lebih mudah).
C2_Embed_128	**Tidak (Sangat Sehat)**	Model terbaik dalam hal generalisasi. *Gap* hampir tidak ada dan performa tinggi (**Train Acc 0.9121 vs Val Acc 0.9221**). Val Loss juga lebih rendah dari Train Loss.
C2_Embed_256	**Ya (Mulai *Overfit*)**	Performa sangat tinggi (**Val Acc 0.9610),** tapi *gap* mulai muncul (**Train Acc 0.9837**). Val Loss mencapai titik terendah di Epoch 14 (0.0806) lalu cenderung naik lagi di akhir (0.1794), tanda *early overfitting*.

C2_Dropout_0.3	**Tidak (Sangat *Underfit*)**	Performa hancur (Val Acc 0.5325). Nilai *dropout* terlalu tinggi sehingga model tidak mampu belajar (kasus *underfitting* akibat regularisasi berlebih).
C2_Dropout_0.5	**Tidak (Sangat *Underfit*)**	Performa terburuk (Val Acc 0.5065), setara tebakan acak. *Dropout* 0.5 terlalu ekstrem dan "membunuh" proses pelatihan.

E. ***Baseline Zero-Shot Multilanguage* (*optional*)**. Terapkan model *zero-shot* (mis. Hugging Face *pipeline* dengan model seperti `joeddav/xlm-roberta-large-xnli` atau `facebook/bart-largemnli`) untuk tugas *sentiment* (tanpa melakukan *fine-tune*). Bandingkan performanya dengan model *from-the-scratch*.

Jawaban Bagian E:

Output:

```
--------------------------------------------------
HASIL EKSPERIMEN E: ZERO-SHOT (facebook/bart-large-mnli)
Model: facebook/bart-large-mnli
Total Sampel Validasi: 77
Prediksi Benar: 42
Akurasi Zero-Shot: 54.55%
--------------------------------------------------
--------------------------------------------------
HASIL EKSPERIMEN E: ZERO-SHOT (joeddav/xlm-roberta-large-xnli)
Model: joeddav/xlm-roberta-large-xnli
Total Sampel Validasi: 77
Prediksi Benar: 45
Akurasi Zero-Shot: 58.44%
--------------------------------------------------
```

Tabel ini mencakup semua model, mulai dari yang dilatih *from-the-scratch* (baseline dan eksperimen) hingga model *zero-shot*.

model	tipe model	akurasi validasi
c2_embed_256	**from-the-scratch**	**96.10%**
c2_embed_128	from-the-scratch	92.21%

c2_layers_4	from-the-scratch	87.01%
c1_epochs_50	from-the-scratch	85.71%
c2_heads_2	from-the-scratch	79.22%
c2_heads_8	from-the-scratch	75.32%
model awal (baseline)	from-the-scratch	66.23%
c2_layers_1	from-the-scratch	61.04%
joeddav/xlm-roberta-large-xnli	zero-shot (pre-trained)	58.44%
facebook/bart-large-mnli	zero-shot (pre-trained)	54.55%
c2_dropout_0.3	from-the-scratch	53.25%
c2_dropout_0.5	from-the-scratch	50.65%

Secara keseluruhan, eksperimen *from-the-scratch* dengan meningkatkan kapasitas model (terutama **embed_dim ke 256**) memberikan peningkatan performa paling drastis, mencapai akurasi **96.10%**. Performa ini jauh melampaui model *zero-shot* (maksimal 58.44%) yang kinerjanya bahkan lebih buruk daripada *baseline* awal (66.23%). Di sisi lain, regularisasi yang terlalu agresif (seperti dropout 0.5) terbukti "membunuh" proses pelatihan dan menghasilkan performa terburuk, setara dengan tebakan acak (50.65%).

F. **Sertakan *script*** yang digunakan.

Jawaban:

Ada di berkas Tugas_2_5026221144_Part1.pdf dan Tugas_2_5026221144_Part2.pdf (atau *file* aslinya berupa .ipynb) yang berisi *script* beserta *output*-nya (tetapi kadang *preview*-nya tidak muncul, jadi saya *convert* ke .pdf agar aman), atau di *link* berikut ini:
https://github.com/renaldoaluska/pba2025gasal/blob/main/%23Tugas2/Snippet-Output_Tugas_2_5026221144_Part1.pdf
https://github.com/renaldoaluska/pba2025gasal/blob/main/%23Tugas2/Snippet-Output_Tugas_2_5026221144_Part2.pdf