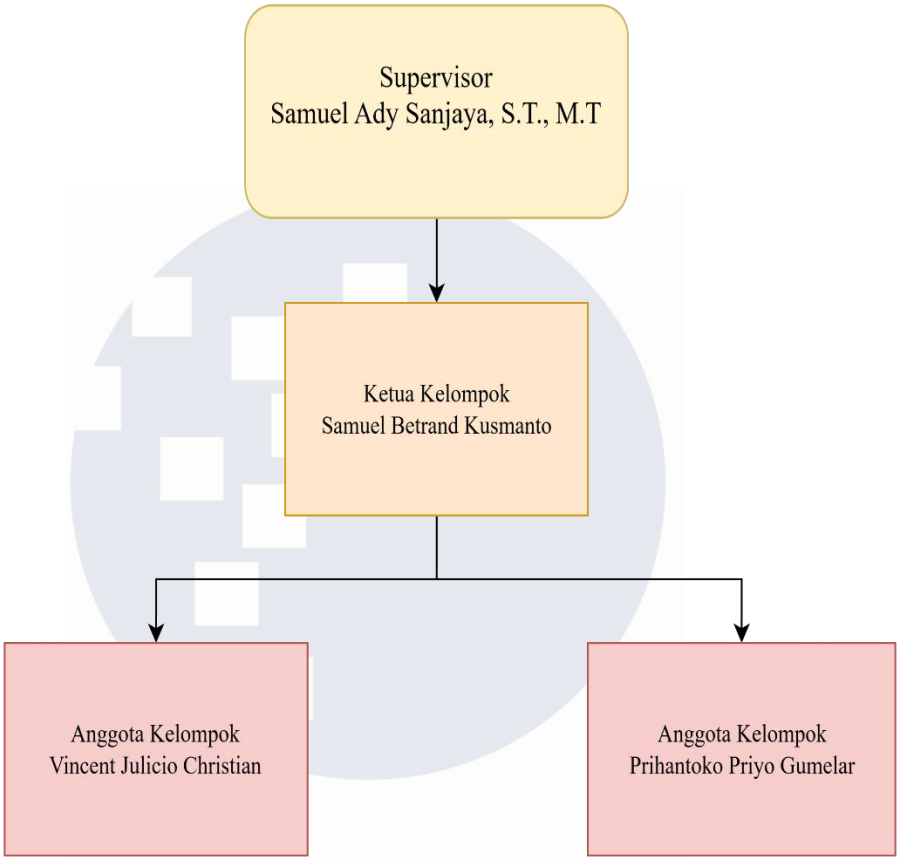
BAB III
PELAKSANAAN PRO-STEP : ROAD TO CHAMPION

3.1 Kedudukan dan Koordinasi

Pelaksanaan program Road to Champion Lomba Data Science LOGIKA UI 2025 disupervisi secara langsung oleh salah satu dosen Universitas Multimedia Nusantara, yaitu Bapak Samuel Ady Sanjaya, S.T., M.T. Beliau berperan sebagai pembimbing yang memastikan seluruh peserta dapat dengan mudah memahami setiap ketentuan, target, serta apa saja yang harus dicapai selama rangkaian tahapan lomba berlangsung. Mulai dari fase persiapan, penyusunan strategi analisis data, hingga tahap final, beliau memberikan arahan mengenai standar yang harus dipenuhi, teknik pengerjaan yang tepat, serta etika saat berkompetisi yang perlu dijunjung tinggi. Supervisi ini bertujuan membantu mengoptimalkan kemampuan teknis dan non-teknis peserta sehingga dapat memperoleh hasil yang benar-benar maksimal dalam kompetisi.

Dalam pelaksanaan program lomba ketua kelompok memiliki tanggung jawab yang sangat penting. Ketua kelompok tidak hanya yang bertugas dalam mengatur koordinasi antara satu anggota dengan anggota yang lain, tetapi juga bertugas untuk memastikan bahwa bobot lomba bisa dibagi secara proporsional, sehingga masing-masing anggota memiliki tanggung jawab yang harus dipenuhi. Ketua kelompok juga bertugas agar pengerjaan sesuai dengan timeline lomba, dan juga menjaga keharmonisan antar tim dan juga anggota, selain itu ketua juga bertugas sebagai penghubung antara tim dengan dosen, sehingga setiap revisi dapat diteruskan kepada anggota. Untuk alur kedudukan akan dibuat menjadi bagan agar lebih mudah dipahami masing-masing jobdesk nya mau dari supervisor, ketua kelompok, dan juga anggota kelompok.



Gambar 3. 1 Bagan Alur Koordinasi

Gambar 3.1 menunjukan bagan yang menjelaskan kedudukan dimana supervisor berada diposisi paling atas, dilanjutkan denga ketua kelompok, kemudian ke anggota kelompok, berikut merupakan penjelasan tugas terstruktur mulai dari supervisor ke ketua kelompok kemudian ke anggota kelompok.

Dalam struktur yang ada, supervisor memiliki tanggung jawab dan juga tahta paling tinggi, karena berperan sebagai pengarah yang memastikan seluruh kegiatan berjalan sesuai standar akademik dan ketentuan lomba. Supervisor memberikan bimbingan menyeluruh mulai dari tahap perencanaan, pengolahan data, pemodelan data, serta memberikan masukan kritis untuk meningkatkan hasil kerja dari mahasiswa. Selain itu, supervisor memastikan setiap prosedur, teknik analisis yang digunakan oleh sesuai,

Pada tingkat berikutnya dalam bagan terdapat ketua kelompok, yang berperan sebagai penghubung utama antara supervisor dan anggota. Ketua memiliki tugas untuk menyampaikan arahan dari supervisor secara jelas kepada

anggota kelompok, dan juga yang mengatur alur kerja internal tim. Tanggung jawab ketua kelompok banyak, beberapa diantaranya berupa penyusunan timeline pengerjaan, pembagian tugas yang proporsional, serta menjaga keharmonisan di dalam tim. Ketua kelompok juga menjadi pengambil keputusan dalam operasional harian tim seperti kapan akan dilakukan diskusi, serta memastikan bahwa setiap anggota memahami perannya dan bekerja secara kolaboratif. Selain itu, ketua kelompok berperan dalam melaporkan perkembangan pengerjaan kepada supervisor, dan meminta masukan untuk model yang telah dibuat.

Sementara itu, anggota kelompok menempati posisi paling bawah dalam bagan namun memiliki peran yang bisa dibilang penting karena anggota juga bantu menjalankan sebagian besar pekerjaan teknis yang berkaitan langsung dengan keperluan lomba. Tugas anggota kelompok diantara lain melaksanakan pekerjaan yang diberikan oleh ketua, seperti mengolah data, membangun model, dan membuat visualisasi. Dalam menjalankan tugasnya, anggota kelompok wajib aktif dalam diskusi, menyampaikan ide maupun solusi, anggota juga harus menjaga komunikasi yang baik demi kelancaran kerja sama antar tim. Anggota juga memiliki tanggung jawab untuk menyelesaikan tugasnya tepat waktu

3.2 Pencatatan Rangkuman Mingguan Proses *PRO-STEP: Road to Champion Program*

terdapat juga rangkuman mingguan yang dilakukan secara sistematis selama proses pengerjaan PRO-STEP Road to Champion. Rangkuman ini berfungsi sebagai dokumentasi kegiatan dari minggu ke minggu,

Tabel 3. 1 Detail Pekerjaan yang Dilakukan PRO-STEP : Road to Champion Program

No.	Minggu	Proyek	Keterangan
1.	Minggu ke 1 Perekrutan anggota kelompok	Membentuk Kelompok	Melakukan Perekrutan anggota kelompok untuk keperluan lomba
2.	Minggu ke 2 sampai minggu ke 4 Perencanaan dan Konsultasi	Review ulang materi	Review materi ulang untuk membantu memahami konsep dasar pengolahan data dan penerapan algoritma dalam menyelesaikan permasalahan.

No.	Minggu	Proyek	Keterangan
3.		Daftar Lomba	Melakukan Pendaftaran untuk mengikuti lomba
4.		Mendapatkan brief Lomba.	Mendapatkan brief lomba dilakukan sebagai acuan utama dalam memahami tujuan, ketentuan, serta batasan teknis yang harus dipenuhi selama pelaksanaan proyek.
5.		Konsultasi dengan dosen pembimbing	Memperoleh arahan, masukan, serta validasi metodologi dan hasil yang dicapai selama pelaksanaan proyek.
6.		pembagian tugas antar tim	melakukan pembagian tugas dan pembagian pekerjaan untuk model apa yang harus dibuat.
7.	Minggu ke 5 Pemahaman dan Preprocessing Data	Pemahaman Dataset Budaya Indonesia	Mengidentifikasi, jumlah kelas, serta karakteristik citra pada dataset yang disediakan oleh pihak lomba.
8.		Membaca Jurnal	Mempelajari jurnal terkait Image Classification, terutama terhadap citra Budaya Indonesia.
9.		Setup Environment	Setup environment local dan instalasi library seperti TensorFlow, Pandas, Scikit-learn.
10		Normalisasi	Mengubah ukuran citra menjadi input standar model.
11		Augmentasi Data	Menerapkan Augmentasi seperti rotasi, flip, dan zoom, untuk memperbanyak variasi data latih guna mencegah overfitting.
12	Minggu ke 6 Pemodelan InceptionV3	Implementasi Arsitektur InceptionV3	Membangun model InceptionV3.
13		Modifikasi Fully Connected Layer	Menambahkan Global Average Pooling dan Dense Layer baru sesuai jumlah kelas Budaya Indonesia.
14		Tuning Hyperparameter	Menguji variasi Learning Rate dan Optimizer pada model

No.	Minggu	Proyek	Keterangan
			InceptionV3.
15		Training InceptionV3	Melakukan pelatihan model dengan epoch untuk mendapatkan hasil yang lebih tinggi.
16		Evaluasi Model InceptionV3	Mencatat akurasi dan hasil training yang telah dilakukan
17	Minggu ke 7 Pemodelan Hybrid EffecientNet-B0 dan ConvNeXt-Tiny	Coding Model EfficientNet-B0	Implementasi arsitektur EfficientNet-B0.
18		Coding Model ConvNeXt Tiny	Implementasi arsitektur ConvNeXt Tiny.
19		Penggabungan Fitur	Membuat Layer Concatenate untuk menggabungkan fitur dari model ConvNeXt-Tiny dan EffecientNet-B0
20		Training Hybrid Model	Melakukan pelatihan model dengan epoch untuk mendapatkan hasil yang lebih tinggi.
21		Pembuatan Confusion Matrix	Membuat visualisasi matriks untuk melihat kelas Budaya Indonesia mana yang sering salah diprediksi.
22		Evaluasi	Menghitung Precision, Recall, F1-Score, dan Akurasi total untuk kedua model.
23		Melakukan Komparasi	Membandingkan hasil peforma paling baik untuk dimasukan kedalam lomba
24		Mengumpulkan hasil lomba	Mengumpulkan hasil lomba sesuai menggunakan model terbaik sesuai dengan arahan yang telah diberi oleh penyelenggara lomba
25	Minggu ke 8 Penyusunan Laporan bagian Pendahuluan.	Mengerjakan laporan bagian Pendahuluan	Menyelesaikan laporan bagian pendahuluan
26	Minggu ke 9 Penyusunan Laporan Bagian Tentang	Mengerjakan laporan bagian tentang lomba dan kompetisi	Menyelesaikan laporan bagian tentang lomba dan kompetisi

No.	Minggu	Proyek	Keterangan
	Lomba dan Kompetisi		
27	Minggu ke 10 penyusunan laporan bab 3 subbab 3.1 dan 3.2	mengisi isi laporan bab 3.1 dan 3.2	Menyelesaikan laporan subbab 3.1 dan 3.2
28	Minggu ke 11 Penyusunan laporan bab 3	Mengerjakan isi laporan bab 3	Menyelesaikan seluruh isi bab 3 mulai dari
29	Minggu ke 12 Penyusunan Laporan bab 4 dan bimbingan	Mengerjakan laporan bab 4 dan melakukan bimbingan	Menyelesaikan isi bab 4 dan melakukan bimbingan agar bisa revisi laporan
30	Minggu ke 13 melakukan revisi	Melakukan revisi laporan	Melakukan revisi secara keseluruhan dari bab 1 sampai bab 4 berdasarkan hasil bimbingan dari pembimbing
31	Minggu ke 14 finalisasi laporan	Melengkapi dokumen yang diperlukan dan abstrak	Melengkapi dokumen yang diperlukan untuk laporan dan mengisi halaman depan laporan, beserta dengan abstrak

Tabel 3.1 menjelaskan rangkaian kegiatan yang dilakukan selama pelaksanaan program PRO-STEP Road to Champion, yang dimulai dari tahap awal hingga tahap penyusunan laporan. Seluruh kegiatan disusun secara bertahap untuk mendukung kelancaran pelaksanaan lomba serta pencapaian hasil yang optimal.

Pada tahap awal, kegiatan diawali dengan pembentukan anggota kelompok yang akan terlibat dalam lomba. Setelah tim terbentuk, kegiatan dilanjutkan dengan perencanaan dan konsultasi, termasuk pembelajaran ulang materi perkuliahan yang relevan untuk memperkuat pemahaman dasar. Tim kemudian menerima briefing lomba dari pihak penyelenggara yang digunakan sebagai acuan utama dalam memahami tujuan, dan aturan lomba. Konsultasi dengan dosen pembimbing juga dilakukan untuk memperoleh arahan serta memastikan yang dilakukan sudah sesuai. Pembagian tugas antar anggota tim dilakukan agar proses pengerjaan dapat berjalan lebih terstruktur.

Memasuki tahap berikutnya, kegiatan difokuskan pada pemahaman data

dan persiapan sebelum pemodelan, kemudian lanjut ke proses pengolahan data yang dilakukan agar data siap digunakan pada tahap selanjutnya.

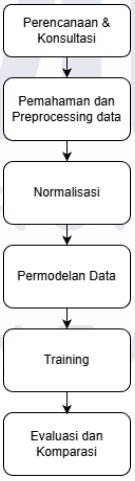
Tahap pemodelan kemudian dilaksanakan dengan mengembangkan beberapa pendekatan model yang relevan. Selanjutnya, dilakukan perbandingan hasil untuk menentukan model dengan kinerja terbaik yang akan digunakan dalam lomba. Hasil akhir kemudian dikumpulkan sesuai dengan ketentuan yang telah ditetapkan oleh penyelenggara.

Pada tahap akhir, kegiatan difokuskan pada penyusunan laporan yang dilakukan secara bertahap.

3.3 Uraian Pelaksanaan Kerja Dalam *PRO-STEP : Road To Champion Program*

Bagian ini berupa penjelasan secara umum mengenai pekerjaan yang dilakukan selama lomba data science LOGIKA UI 2025, dari proses koleksi data hingga proses komparasi model.

3.3.1 Proses Pelaksanaan

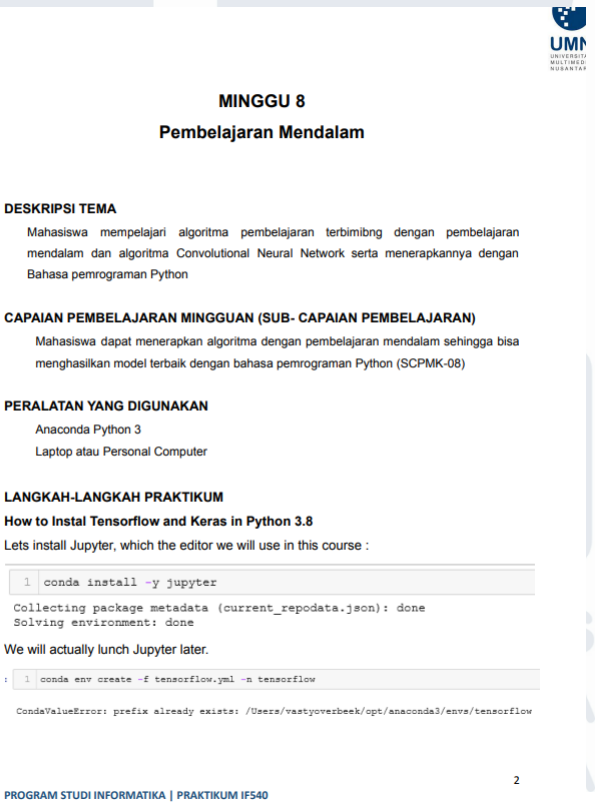


Gambar 3. 2 Alur Proses Pengerjaan

Gambar 3.2 menunjukan alur proses pengerjaan yang dilakukan selama kegiatan

3.3.1.1 Perencanaan dan Konsultasi

Perencanaan dilakukan untuk memastikan agar saat mengikuti lomba, peserta sudah siap dalam mengikuti lomba. Dengan persiapan yang dilakukan secara bertahap, peserta bisa lebih terbiasa menghadapi soal atau tantangan yang muncul. Tahap awal pelaksanaan dimulai dengan proses pembelajaran dari internet, kemudian juga dilakukan review ulang materi-materi yang telah dipelajari saat proses perkuliahan. Pendalaman materi dilakukan dengan mempelajari ulang mata kuliah yang diikut sebelumnya selama beberapa minggu. Pembelajaran ulang materi dapat menjadi langkah untuk membantu menyiapkan dalam kegiatan lomba.



MINGGU 8

Pembelajaran Mendalam

DESKRIPSI TEMA

Mahasiswa mempelajari algoritma pembelajaran terbimibng dengan pembelajaran mendalam dan algoritma Convolutional Neural Network serta menerapkannya dengan Bahasa pemrograman Python

CAPAIAN PEMBELAJARAN MINGGUAN (SUB- CAPAIAN PEMBELAJARAN)

Mahasiswa dapat menerapkan algoritma dengan pembelajaran mendalam sehingga bisa menghasilkan model terbaik dengan bahasa pemrograman Python (SCPMK-08)

PERALATAN YANG DIGUNAKAN

Anaconda Python 3

Laptop atau Personal Computer

LANGKAH-LANGKAH PRAKTIKUM

How to Instal Tensorflow and Keras in Python 3.8

Lets install Jupyter, which the editor we will use in this course :

```
conda install -y jupyter
```

```
Collecting package metadata (current_repodata.json): done
Solving environment: done
```

We will actually lunch Jupyter later.

```
conda env create -f tensorflow.yml -n tensorflow
```

```
CondaValueError: prefix already exists: /Users/vastyoverbeek/opt/anaconda3/envs/tensorflow
```

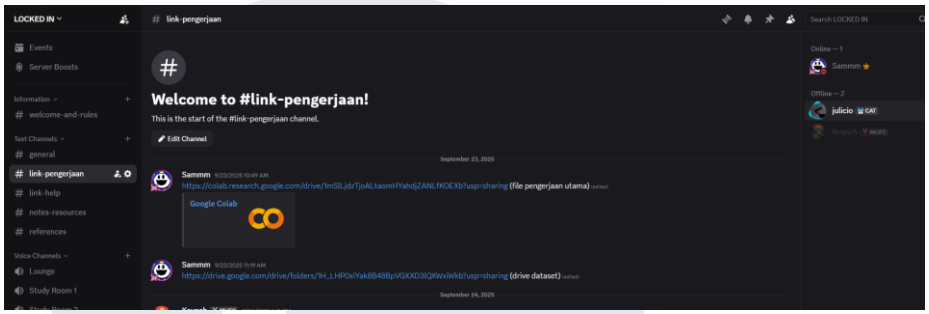
2

PROGRAM STUDI INFORMATIKA | PRAKTIKUM IF540

Gambar 3. 3Modul Mata Kuliah Machine Learning

Gambar 3.3 menunjukan salah satu dari beberapa modul yang diambil dari mata kuliah machine learning untuk membantu selama proses pembelajaran, pada modul tersebut mempelejari mengenai deep learning, mulai dari tahap preproccesing sampai ke tahap akhir. Modul yang diberi memiliki tujuan untuk memberi pengajaran yang terstruktur terhadap algoritma Convolutional Neural Network dan cara penerapan pada python.

Proses review ulang materi akan dilakukan sekaligus dengan pembentukan kelompok lomba. Untuk Kelompok Dataframe, anggota terdiri dari 3 mahasiswa yang berbeda, yaitu Samuel Betrand Kusmanto, Prihantoko Priyo Gumelar, dan Vincent Julicio Christian.



Gambar 3. 4 Grup Diskusi Kelompok

Gambar 3.4 menunjukan grup yang telah dibuat untuk membantu kelancaran komunikasi akan setiap anggota, grup juga akan digunakan untuk membagi progres dari masing-masing anggota.

Kemudian dilanjutkan dengan pencarian dan pendaftaran lomba, pada akhirnya lomba yang diikuti merupakan lomba Data Science yang diselenggarakan oleh LOGIKA UI 2025

Setelah daftar lomba akan dilanjutkan lagi dengan konsultasi dengan dosen internal. Sesi konsultasi berfungsi untuk menyelaraskan rencana kerja tim dengan standar akademik serta mendapatkan masukan strategis terkait arah pengembangan model klasifikasi yang akan dikompetisikan.

Gambar 3. 5 Technical Meeting Online

Gambar 3.5 menunjukan technical meeting yang diikuti, agar peserta dapat memahami aturan-aturan yang ada pada kompetisi.

Gambar 3. 6 QnA Session Technical Meeting

Gambar 3.6 Menunjukan bagian sesi tanya jawab dalam Technical Meeting lomba, dimana tanya jawab dilakukan agar peserta lebih mengerti tentang aturan yang mungkin kurang jelas

Q&A TM DSC LOGIKA UI 2025

File Edit View Insert Format Data Tools Extensions Help

Menus 100% View only

A1

Table1

Pertanyaan	Jawaban
Model pretrained dari seperti keras.applications contohnya Alexnet, ResNet dll dan juga seperti Visual Transformers apakah boleh digunakan?	Diperbolehkan
apakah boleh stacking model?	Diperbolehkan
Apakah diperbolehkan menggunakan dataset eksternal?	Tidak Diperbolehkan
pake pretrained model yang sudah include dalam pytorch bolah gak kak?	Diperbolehkan
apakah model transformer boleh dipakai kak?	
apakah persebaran dataset untuk penilaian private dan public 50:50?	untuk persebaran data set Private 65 Public 35 namun untuk penilaian tetap 50 :50
Metrics apa yang digunakan dalam penilaian?	F1 Macro
submisi yang di google form, yang bagian csvnya itu hasil dari submisi csv di kaggle yang skornya tertinggi kah?	yang kalian pilih untuk di nilai pada private score (umumnya peserta memilih yang tertinggi)
kak, untuk baca data (pandas) nya nanti kita taruh data di google drive terus diconneckin itu boleh, jadi ga baca file csv di local	Diperbolehkan
kak kalau dari dataset aslinya tapi kita crop manual apakah boleh?	Diperbolehkan

Gambar 3. 7 Hasil Sesi QnA

Gambar 3.7 menunjukan hasil sesi tanya jawab yang dijadikan sebuah tabel menggunakan google sheets, dan untuk hasil sesi tanya jawab akan diberi oleh penyelenggara lomba kepada para peserta, agar peserta yang tidak dapat mengikuti Technical Meeting tidak ketinggalan informasi atau kurang paham mengenai aturan lomba.

Dengan adanya technical meeting, tim dapat memahami ketentuan kompetisi secara lebih jelas dan lebih menyeluruh melalui penjelasan langsung serta dokumen brief lomba yang diberikan oleh panitia.

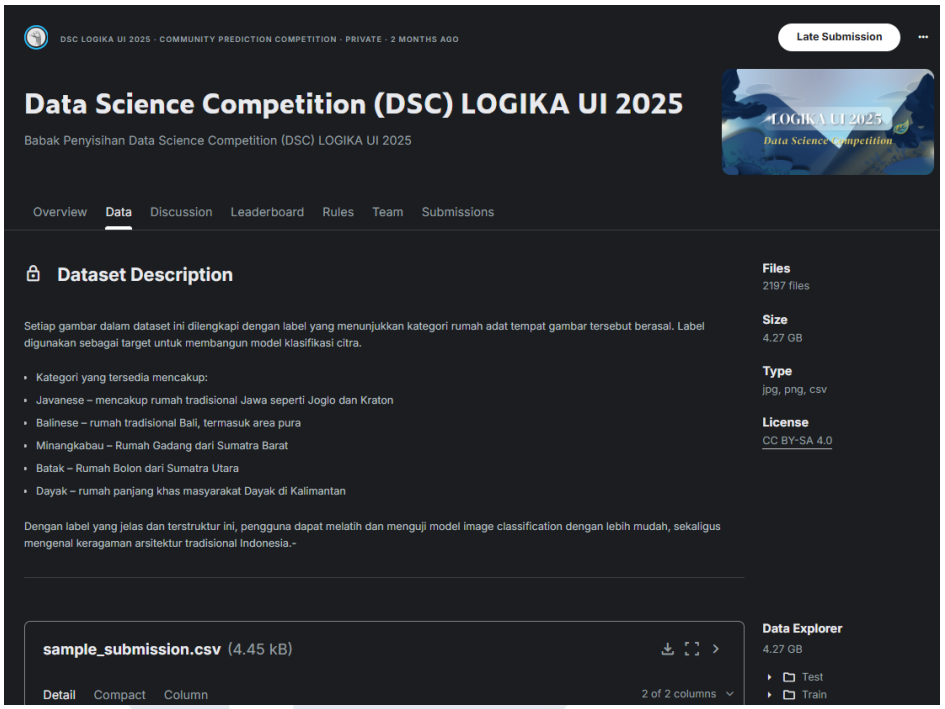
Technical meeting ini membantu peserta untuk mengetahui alur perlombaan, aturan yang berlaku, serta hal-hal teknis yang perlu diperhatikan selama proses pengerjaan kompetisi. Melalui technical meeting yang telah diikuti, tim dapat menentukan batasan masalah, memahami spesifikasi dataset yang digunakan, serta mengetahui kriteria penilaian utama yang menjadi fokus dalam kompetisi. Informasi tersebut kemudian dijadikan sebagai dasar dalam menyusun strategi pengerjaan, termasuk dalam menentukan pendekatan dan arsitektur model yang paling sesuai untuk digunakan dalam perlombaan.

Kemudian tahap selanjutnya adalah melakukan pembagian pekerjaan di dalam tim. Pembagian tugas ini dilakukan secara proporsional. Dengan pembagian tugas yang jelas, proses pengerjaan dapat berjalan lebih terarah dan efisien, serta meningkatkan kerja sama tim dalam mencapai hasil yang optimal.

Pada perlombaan ini, pembagian tugas yang dilakukan berfokus pada pengerjaan model yang berbeda-beda oleh masing-masing anggota tim. Setiap anggota bertanggung jawab untuk mengembangkan dan menguji pendekatan model. Untuk memastikan koordinasi dan komunikasi tetap berjalan dengan baik, pembaruan progres dari masing-masing anggota disampaikan secara berkala melalui grup chat yang telah dibuat. Melalui komunikasi ini, tim dapat saling memberikan informasi, berdiskusi mengenai kendala yang dihadapi, serta menyampaikan perkembangan terbaru, sehingga proses pengerjaan dapat berjalan secara lebih terarah dan efisien.

3.3.1.2 Pemahaman dan Preprocessing Data

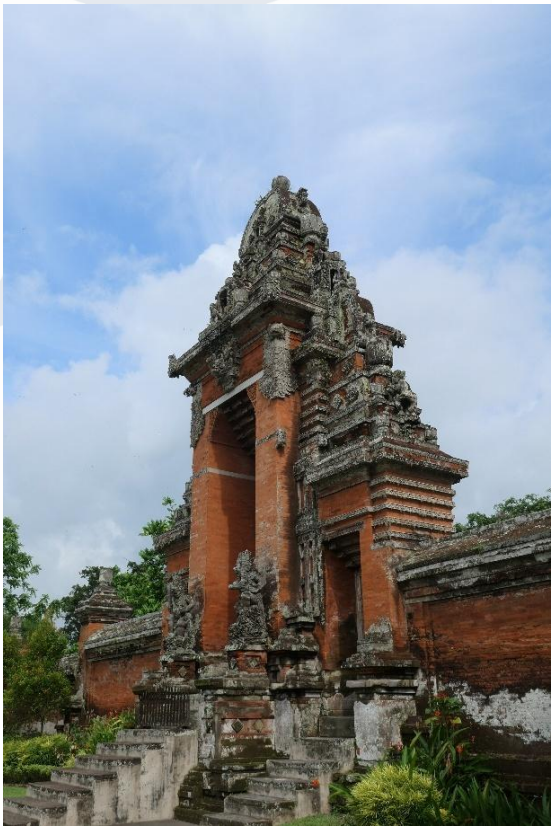
Tahap selanjutnya merupakan tahap pengerjaan, yang diawali dengan proses pengumpulan dataset. Dalam kompetisi LOGIKA UI 2025, pengumpulan data tidak dilakukan secara manual, melainkan dataset yang disediakan oleh panitia lomba. Seluruh data citra diunggah melalui platform Kaggle dan dapat diunduh oleh peserta dengan link https://www.kaggle.com/competitions/dsc-logika-ui-2025/data



Gambar 3. 8 Halaman Kaggle Competition

Gambar 3.8 menunjukan bahwa dataset yang diberi berasal dari platform kaggle, dan dataset yang di distribusi diunggah oleh panitia dari kompetisi LOGIKA UI 2025. File size data berukuran 4.27 GB. Terdapat info tambahan dari kompetisi bahwa jumlah file citra yang ada pada dataset adalah 2197, dan terdapat 5 sub kategori, yakni rumah adat jawa, bali, minangkabau,batak, dan dayak. Pada rumah adat jawa mencakup rumah tradisional seperti joglo, dan keraton, untuk bali terdapat rumah tradisional bali, dan juga area pura, untuk minangkabau terdapat rumah gadang, untuk batak ada rumah bolon, dan pada suku dayak terdapat rumah panjang. Dataset yang terdapat di kaggle sudah dibagi menjadi training dan test. Berikut merupakan gambar-gambar dari setiap folder yang ada di file train dan test. Pada file train berisi file yang sudah dikategorikan sebagai Javanese,Balinese,Minangkabau,Batak, dan Dayak. Akan tetapi isi dari file test acak dari kelima kategori tersebut

Gambar 3. 9 Citra Rumah Adat Bali dari File Train Bali



Gambar 3. 10 Citra Pura Bali dari File Train Bali

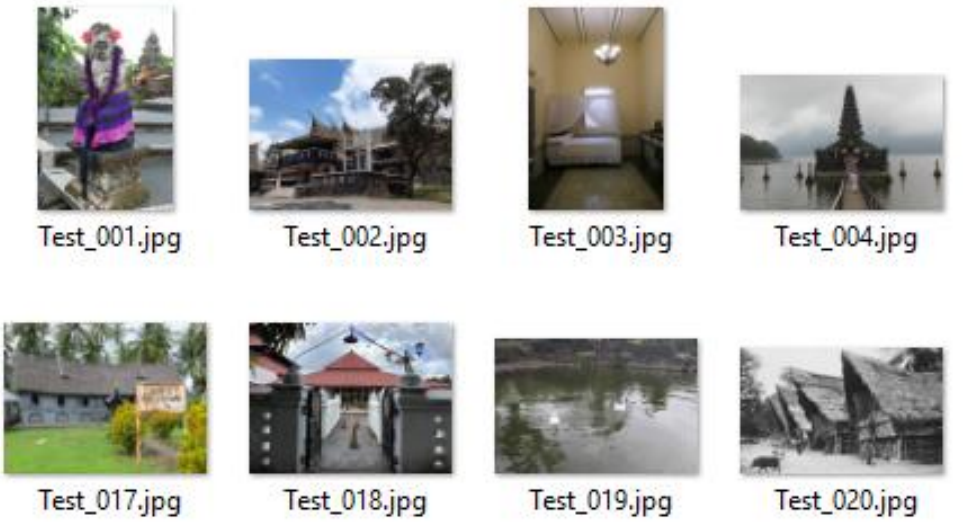
Gambar 3. 11 Citra Rumah Bolon Batak dari File Train Batak

Gambar 3. 12 Citra Rumah Gadang Minangkabau dari File Train Minangkabau

Gambar 3. 13 Citra Rumah Panjang Dayak dari Folder Train Dayak

Gambar 3. 14 Citra Rumah Joglo Jawa, dari Folder Train Javanese

Gambar 3.9, Gambar 3.10, Gambar 3.11, Gambar 3.12, Gambar 3.13, Gambar 3.14 menunjukan contoh citra rumah dalam train Dataset, pada dataset terdapat beberapa jenis Budaya Indonesia yang berbeda, seperti rumah adat Bali, Pura Bali, Rumah Bolon dari suku Batak, Rumah Gadang dari Minangkabau, Rumah Panjang dari suku dayak, dan juga rumah Joglo yang berasal dari Jawa.



Gambar 3. 15 Isi Folder test

Gambar 3.15 menunjukan isi dari folder test, pada folder test terdapat contoh citra Budaya dari berbagai daerah berdasarkan pada hasil klasifikasi.

Tahap selanjutnya merupakan tahap data understanding, jadi data yang sebelumnya peserta diraih dari kaggle mulai dianalisa terlebih dahulu, seperti contoh citra Budaya dari setiap daerah yang diberi, total citra Budaya dari setiap daerah. Hal seperti ini dilakukan agar data agar para peserta dapat mencari keanehan dalam data, seperti jika data yang digunakan ada yang nominal nya lebih sedikit, dan juga ada yang lebih banyak. Dalam Lomba data science LOGIKA UI 2025 ukuran dataset yang digunakan adalah 4,27 GB

```
#jumlah gambar di folder

import os

file_test = os.listdir('C:/Users/Samuel/Anaconda Tasks/LOCKED IN/Test/Test')

train_bali = os.listdir('C:/Users/Samuel/Anaconda Tasks/LOCKED IN/Train/Train/balinese')
train_batak = os.listdir('C:/Users/Samuel/Anaconda Tasks/LOCKED IN/Train/Train/batak')
train_dayak = os.listdir('C:/Users/Samuel/Anaconda Tasks/LOCKED IN/Train/Train/dayak')
train_java = os.listdir('C:/Users/Samuel/Anaconda Tasks/LOCKED IN/Train/Train/javanese')
train_minang = os.listdir('C:/Users/Samuel/Anaconda Tasks/LOCKED IN/Train/Train/minangkabau')

jumlah_test = len(file_test)
print("Gambar Test:", jumlah_test)

jumlah_bali = len(train_bali)
print("Gambar Train Bali:", jumlah_bali)


jumlah_batak = len(train_batak)
print("Gambar Train Batak:", jumlah_batak)

jumlah_dayak = len(train_dayak)
print("Gambar Train Dayak:", jumlah_dayak)

jumlah_java = len(train_java)
print("Gambar Train Java:", jumlah_java)


jumlah_minang = len(train_minang)
print("Gambar Train Minang:", jumlah_minang)
```

Gambar 3. 16 Kode untuk Menghitung Jumlah Gambar

Gambar 3.16 menunjukan kode yang digunakan untuk menghitung jumlah citra yang ada dalam setiap folder dataset. Proses dimulai dengan melakukan pembacaan seluruh file pada folder Test menggunakan fungsi os.listdir(), kemudian hasilnya disimpan ke dalam file_test. Kemudian Langkah tersebut dilakukan pada lima kategori Budaya yang terdapat pada dataset Training, yaitu Balinese, Batak, Dayak, Javanese, dan Minangkabau. Setiap variabel menyimpan daftar nama file gambar pada masing-masing subfolder kelas, contohnya train_bali menyimpan daftar citra yang ada di folder bali.

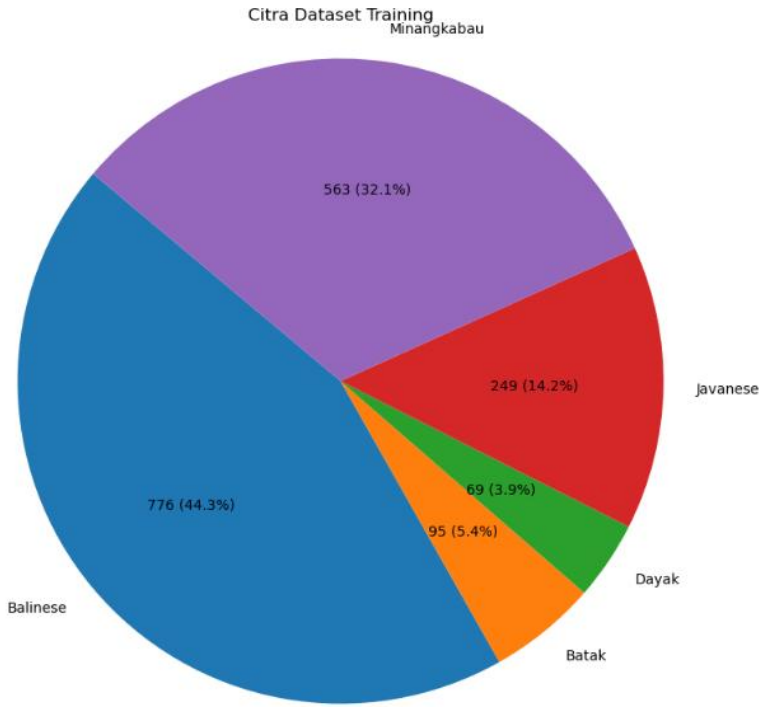
Selanjutnya, kode menghitung jumlah citra dengan menggunakan fungsi len untuk memperoleh total gambar pada masing-masing kategori. Hasil kemudian ditampilkan melalui perintah print, sehingga pengguna dapat mengetahui jumlah gambar untuk setiap kelas training serta total gambar pada folder test. Informasi ini penting untuk tahap

Data Understanding, karena membantu dalam mengevaluasi distribusi data, keseimbangan antar kelas, serta memastikan bahwa dataset telah terbaca dengan benar sebelum masuk ke tahap preprocessing atau pemodelan.

```
Gambar Test: 444
Gambar Train Bali: 776
Gambar Train Batak: 95
Gambar Train Dayak: 69
Gambar Train Java: 249
Gambar Train Minang: 563
```

Gambar 3. 17 Jumlah Citra per Kategori

Gambar 3.17 merupakan hasil output dari kode yang sudah dijalankan, yang dimana ditunjukan total dari masing-masing folder, contohnya Test memiliki jumlah citra 444, Bali memiliki jumlah citra 776, batak memiliki jumlah citra sebanyak 95, dayak memiliki jumlah citra 69, java memiliki jumlah citra 249, dan minang memiliki jumlah citra 563



Gambar 3. 18 Distribusi Budaya pada Dataset Train

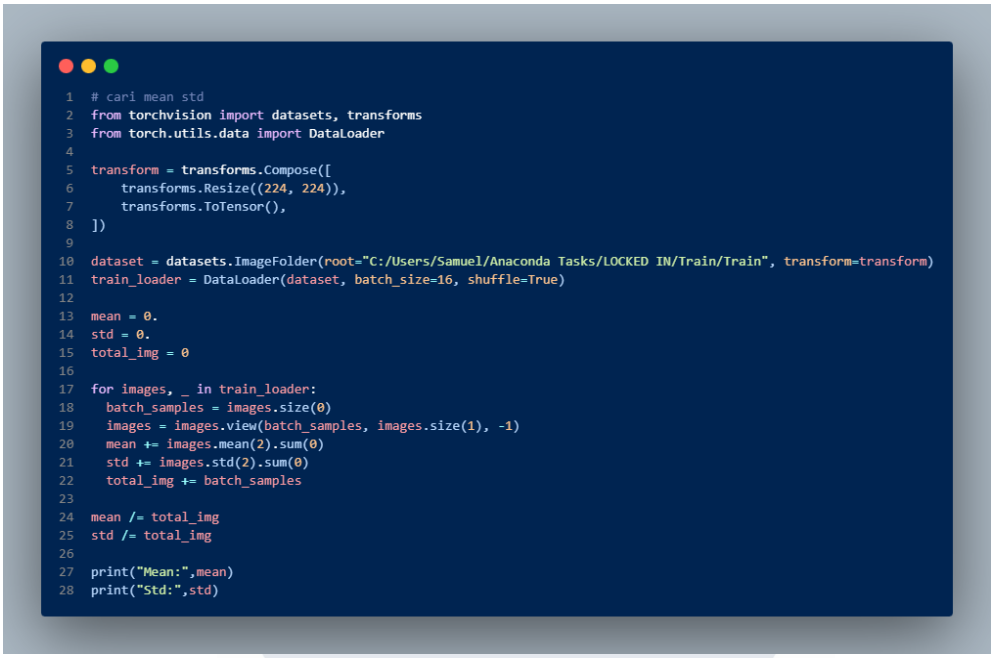
Gambar 3.18 menunjukan visualisasi pie chart dari dataset training yang menunjukan data distribusi citra dari masing masing sub

kategori, dan pada pie chart ditunjukan bahwa jumlah dataset yang diberikan tidak seimbang, terdapat kategori yang hanya memiliki jumlah citra 69 saja yang hanya meliputi 3.9% dari total data, sedangkan jumlah citra yang nominal nya 776 yang mendominasi sekitar 44.3% total data dan jika dilihat pada subkategori yang lain masing-masing berbeda ada beda jumlah citra terlalu signifikan. Sehingga menyimpulkan bahwa dataset yang diberi sangat imbalance, dan sangat diperlukan untuk diperbaiki. Untuk mencegah dataset yang digunakan tidak mengalami gangguan, dan bias perlu dilakukan balancing terhadap data yang imbalance pada tahap selanjutnya, yakni data preparation.

3.3.1.3 Data Normalization

Tahap berikutnya normalisasi data merupakan langkah krusial dalam pra-pemrosesan untuk menyelaraskan format dan distribusi nilai piksel citra sebelum dimasukkan ke dalam arsitektur model. Normalisasi bertujuan untuk mempercepat konvergensi algoritma optimasi gradient descent selama proses pelatihan serta mencegah terjadinya bias numerik yang dapat menghambat performa model dalam mempelajari dataset.

Salah satu dari beberapa permasalahan yang perlu ditangani segera merupakan menyeimbangkan jumlah data antar kelas, setelah itu citra perlu diproses agar memiliki ukuran resolusi yang seragam, biasanya dilakukan dengan teknik cropping atau resizing, sehingga setiap gambar memiliki dimensi piksel yang sama.



```python
# cari mean std
from torchvision import datasets, transforms
from torch.utils.data import DataLoader

transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
])

dataset = datasets.ImageFolder(root="C:/Users/Samuel/Anaconda Tasks/LOCKED IN/Train/Train", transform=transform)
train_loader = DataLoader(dataset, batch_size=16, shuffle=True)

mean = 0.
std = 0.
total_img = 0

for images, _ in train_loader:
  batch_samples = images.size(0)
  images = images.view(batch_samples, images.size(1), -1)
  mean += images.mean(2).sum(0)
  std += images.std(2).sum(0)
  total_img += batch_samples

mean /= total_img
std /= total_img

print("Mean:",mean)
print("Std:",std)
```

Gambar 3. 19 Kode Pencari mean, dan Std

Gambar 3.19 menunjukan kode untuk menghitung nilai mean dan standard deviation dari seluruh citra dalam dataset pelatihan, yang akan digunakan sebagai parameter normalisasi pada tahap preprocessing. Pertama-tama dataset akan dimasukan menggunakan ImageFolder dengan transformasi resize ke ukuran 224×224 piksel dan konversi ke tensor. Kemudian akan dibaca menggunakan dataloader untuk dapat diproses menjadi sebuah batch, kemudian citra akan diratakan untuk mempermudah perhitungan statistik. Nilai mean dan std dihitung untuk setiap channel (RGB) pada seluruh piksel yang ada dalam batch, kemudian hasilnya dijumlahkan. Setelah seluruh batch selesai, total nilai mean dan standard deviation dibagi dengan jumlah citra sehingga diperoleh nilai rata-rata sebenarnya.

```
Mean: tensor([0.4680, 0.4690, 0.4481])
Std: tensor([0.2350, 0.2394, 0.2528])
```

Gambar 3. 20 Hasil Output Kode Pencari Mean, dan Std

Gambar 3.20 menunjukan hasil output dari kode yang sudah dijalankan pada gambar 3.19, Nilai mean sebesar 0.4680, 0.4690, 0.4481 menunjukkan bahwa rata-rata intensitas piksel pada tiga

channel berada di sekitar 0.46–0.47. Ini berarti sebagian besar citra dalam dataset memiliki tingkat kecerahan sedang dan tidak terlalu terang. Sedangkan nilai standard deviation 0.2350, 0.2394, 0.2528 menggambarkan tingkat variasi intensitas piksel pada setiap channel. Variasi yang berada pada rentang 0.23–0.25 menunjukkan bahwa perubahan warna pada citra cukup stabil dan tidak memiliki perbedaan kontras yang ekstrem. Nilai mean dan std ini kemudian dapat digunakan dalam proses normalisasi agar model dapat belajar lebih konsisten dan stabil selama training.

```
train_path = "C:/Users/Samuel/Anaconda Tasks/LOCKED IN/Train/Train"
test_path = "C:/Users/Samuel/Anaconda Tasks/LOCKED IN/Test"

train_transform = transforms.Compose([
    transforms.RandomResizedCrop(224, scale=(0.8, 1.0)),
    transforms.RandomHorizontalFlip(),
    transforms.RandomRotation(25),
    transforms.ColorJitter(brightness=0.3, contrast=0.3, saturation=0.3, hue=0.1),
    transforms.RandomAffine(degrees=0, translate=(0.1, 0.1)),
    transforms.RandomPerspective(distortion_scale=0.2, p=0.4),
    transforms.GaussianBlur(kernel_size=(5, 9), sigma=(0.1, 3)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.4680, 0.4690, 0.4481],
                         std=[0.2350, 0.2394, 0.2528])
])

test_transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.4680, 0.4690, 0.4481],
                         std=[0.2350, 0.2394, 0.2528])
])
```

Gambar 3. 21 Kode Normalisasi Data Feature Fusion

Gambar 3.21 menunjukan kode dari proses augmentasi data dan normalisasi untuk model feature fusion, yang digunakan untuk mempersiapkan dataset sebelum masuk ke dalam proses modeling. Terdapat 2 jenis transformasi yang dilakukan, yaitu untuk folder train, dan test. Tahap ini menjadi sangat penting karena kualitas dan keragaman data yang masuk ke model akan sangat memengaruhi performa akhir dari proses training. Pada bagian train_transform, berbagai teknik augmentasi diterapkan untuk memperkaya variasi citra, seperti pemotongan acak melalui RandomResizedCrop, pembalikan gambar menggunakan RandomHorizontalFlip, rotasi

melalui RandomRotation, serta pengaturan warna dengan ColorJitter. Selain itu, transformasi seperti RandomAffine, RandomPerspective, dan GaussianBlur membantu menciptakan kondisi gambar yang lebih bagus dan asli. Kode diatas juga dijalankan agar model yang dibuat tidak mudah mengalami overfitting karena model dilatih dengan berbagai bentuk variasi gambar. Setelah proses augmentasi selesai, gambar kemudian dikonversi menjadi tensor melalui ToTensor, yang kemudian akan dinormalisasi menggunakan nilai mean dan standar deviasi yang sudah dicari. Untuk test transformasi yang dilakukan tidak sekompleks dengan yang di training, agar evaluasi dilakukan pada data lebih konsisten. Dengan adanya seluruh proses ini modelling data akan menjadi lebih efesien dan tidak terlalu memakan waktu.

```python
# Normalisasi
train_transform = transforms.Compose([
    transforms.RandomResizedCrop(299),
    transforms.RandomHorizontalFlip(),
    transforms.ToTensor(),
    transforms.Normalize(
        mean=[0.4680, 0.4690, 0.4481],
        std=[0.2350, 0.2394, 0.2528]
    )
])

test_transform = transforms.Compose([
    transforms.Resize(320),
    transforms.CenterCrop(299),
    transforms.ToTensor(),
    transforms.Normalize(
        mean=[0.4680, 0.4690, 0.4481],
        std=[0.2350, 0.2394, 0.2528]
    )
])
```

Gambar 3. 22 Kode Normalisasi InceptionV3

Gambar 3.22 menunjukan kode proses augmentasi data dan

normalisasi untuk model InceptionV3, yang digunakan untuk mempersiapkan dataset sebelum masuk ke dalam proses modelling Terdapat 2 jenis transformasi yang dilakukan, yaitu untuk folder train, dan test. Dan normalisasi yang dilakukan pada inceptionV3 lebih sederhana dibandingkan dengan model feature fusion

Kode merupakan rangkaian proses transformasi dan normalisasi untuk model InceptionV3, yang memiliki kebutuhan input berbeda. InceptionV3 secara default menggunakan ukuran input 299 piksel, sehingga seluruh proses augmentasi dan normalisasi dalam train_transform dan test_transform disesuaikan dengan kebutuhan tersebut. Telah dibuat transformation juga agar model tidak mengalami overfitting. Transformasi dimulai dengan RandomResizedCrop(299) yang memotong citra secara acak lalu mengubah ukuran hasil crop menjadi 299 piksel memberikan variasi posisi dan proporsi objek dalam gambar. Kemudian, RandomHorizontalFlip() memberi kemungkinan citra dibalik secara horizontal, sehingga model terbiasa mengenali pola. Setelah augmentasi, gambar dikonversi menjadi tensor melalui ToTensor, lalu dinormalisasi menggunakan nilai mean dan standard deviation yang sudah dicari.

Sementara itu, pada test, proses dibuat sederhana untuk memastikan pengujian tetap konsisten dan tidak dipengaruhi augmentasi acak. Citra pertama-tama di-resize menjadi 320 piksel, kemudian dilakukan CenterCrop(299) agar ukuran output persis sesuai kebutuhan InceptionV3. Langkah selanjutnya sama seperti data training: konversi ke tensor dan normalisasi menggunakan nilai mean dan std yang sama. Dengan transformasi seperti ini.

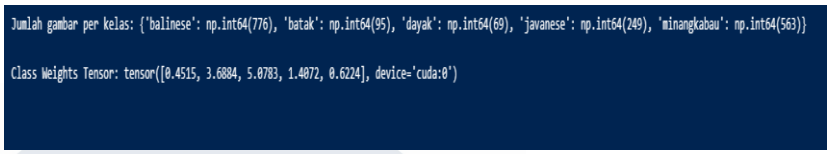
```
#balancing
targets = [label for _, label in train_dataset_full.samples]
unique, counts = np.unique(targets, return_counts=True)

print("Jumlah gambar per kelas:", dict(zip(class_names, counts)))

class_weights = compute_class_weight(
    class_weight='balanced',
    classes=np.arange(len(class_names)),
    y=targets
)
class_weights = torch.tensor(class_weights, dtype=torch.float).to(device)
print("\nClass Weights Tensor:", class_weights)
```

Gambar 3. 23 Kode untuk Balancing Dataset

Gambar 3.23 menunjuka kode tahap terakhir dari data preparation merupakan balancing dataset. Balancing dilakukan untuk memastikan tidak terjadinya overfitting saat sedang permodelan. Kode untuk balancing dimulai dari mengambil seluruh label dari dataset pelatihan menggunakan train_dataset_full.samples. kemudian kode menghitung jumlah citra di setiap kelas untuk menunjukkan seberapa seimbang atau timpangnya distribusi dataset Dengan menggunakan np.unique(), Setelah mengetahui ketidakseimbangan jumlah data, langkah berikutnya adalah menghitung class weights menggunakan fungsi compute_class_weight. class_weight='balanced' memastikan total kelas dihitung secara otomatis berdasarkan proporsi jumlah sampel yang ada pada kelas masing-masing, sehingga kelas yang memiliki nominal terdikit akan mendapatkan data yang lebih banyak daripada kelas yang mempunyai nominal citra yang banyak, dan kelas dengan data melimpah mendapat bobot lebih rendah.

Hasil yang dikonversi kemudian menjadi tensor PyTorch, dan dipindahkan kedalam GPU agar dapat digunakan selama proses pelatihan model. Kemudian tensor akan diberikan ke fungsi loss, dengan adanya ini dapat membantu bila terjadi

kesalahan saat pembelajaran data yang lebih kecil akan diberikan sanksi yang lebih besar.



Gambar 3. 24 Hasil Kode Balancing Dataset

Gambar 3.24 merupakan kode hasil dari yang dijalankan pada gambar 3.23 Balancing dataset, output jumlah gambar per kelas menunjukan bahwa dataset yang diberi memang imbalance, yang dapat membuat model menjadi bisa.

Untuk menghadapi masalah tersebut dilakukanlah perhitungan class weights, yang merupakan bobot khusus untuk setiap kelas berdasarkan proporsi jumlah data. Hasil menunjukan bahwa bali mempunya berat 0.4515, batak mempunyai berat 3.6884, Dayak mempunyai nilai 5.0783, dan Minangkabau memiliki nilai 0.6224.

Nilai yang rendah pada bali dan Minangkabau menunjukan bahwa sampel yang ada untuk kedua kategori tersebut sudah banyak, sehingga tidak perlu membagi sampel kesitu, sedangkan angka yang besar pada batak dan Dayak menunjukan bahwa sampel tidak perlu dibagi banyak, karena memang bobot mereka sudah besar, dengan menggunakan metode ini dataset bisa menjadi lebih balance dan tidak bias, karena semuanya sama rata.

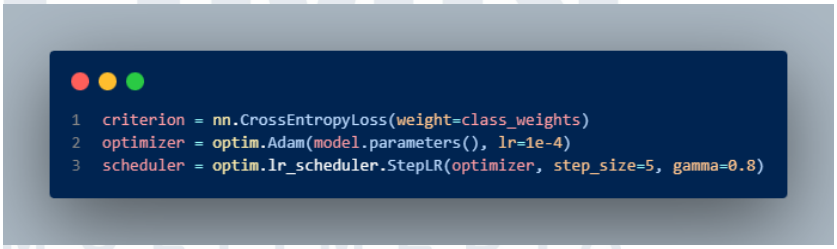
3.3.1.4 Permodelan InceptionV3, dan Hybrid Model

Memasuki tahapan inti pengembangan sistem, langkah pertama yang dilakukan adalah pembangunan model InceptionV3. Pada tahap eksperimen awal ini, proses training dilakukan dengan konfigurasi hyperparameter, dengan durasi pelatihan sebanyak 20 epoch. Dalam data modeling, setiap model diuji dengan tujuan untuk melihat kemampuan masing-masing arsitektur dalam mengekstraksi fitur visual, memahami pola citra, dan

memberikan prediksi yang akurat.

```python
class HybridModel(nn.Module):
    def __init__(self, num_classes):
        super(HybridModel, self).__init__()

        # ConvNeXt Tiny
        convnext = models.convnext_tiny(weights=models.ConvNeXt_Tiny_Weights.IMAGENET1K_V1)
        self.convnext_features = convnext.features  # output 768 dim

        # EfficientNet-B0
        effnet = models.efficientnet_b0(weights=models.EfficientNet_B0_Weights.IMAGENET1K_V1)
        self.effnet_features = effnet.features  # output 1280 dim
        self.effnet_pool = nn.AdaptiveAvgPool2d((1, 1))  # flatten output

        # Fully Connected
        self.fc = nn.Sequential(
            nn.Linear(768 + 1280, 512),  # ✅ total 2048 fitur
            nn.ReLU(),
            nn.Dropout(0.4),
            nn.Linear(512, num_classes)
        )

    def forward(self, x):
        # ConvNeXt branch
        f1 = self.convnext_features(x)
        f1 = f1.mean([-2, -1])  # Global Average Pooling → (batch, 768)

        # EfficientNet branch
        f2 = self.effnet_features(x)
        f2 = self.effnet_pool(f2)
        f2 = torch.flatten(f2, 1)  # (batch, 1280)

        fused = torch.cat((f1, f2), dim=1)  # (batch, 2048)
        out = self.fc(fused)
        return out

model = HybridModel(num_classes=len(class_names)).to(device)
```

Gambar 3. 25 Kode Modeling Feature Fusion ConvNeXt-Tiny dengan EffecientNet-B0



```python
criterion = nn.CrossEntropyLoss(weight=class_weights)
optimizer = optim.Adam(model.parameters(), lr=1e-4)
scheduler = optim.lr_scheduler.StepLR(optimizer, step_size=5, gamma=0.8)
```

Gambar 3. 26 Criterion, Optimizer, dan Scheduler Feature Fusion ConvNeXt-Tiny dengan EffecientNet-B0

Gambar 3.25 dan gambar 3.26 menunjukan proses permodelan dari feature fusion ConvNeXt Tiny dengan EffecientNet-B0 yang keduanya telah dibangun dengan bantuan bobot dari pretrained ImageNet. Tujuannya adalah untuk memaksimalkan kemampuan ekstraksi fitur dengan mengombinasikan dua tipe representasi visual yang saling melengkapi Pada bagian pertama, model memuat arsitektur ConvNeXt Tiny dan hanya mengambil bagian features-nya. ConvNeXt Tiny menghasilkan representasi fitur berdimensi 768 dan

dikenal memiliki kemiripan struktur dengan Vision Transformer. Kemudian pada EffecientNet-B0 hanya diambil bagian feature extractor nya. EfficientNet-B0 menghasilkan fitur berdimensi 1280, dengan karakteristik yang lebih fokus pada detail lokal seperti tekstur kayu, ukiran, pola dinding, dan elemen kecil lainnya. Hasil dari EfficientNet-B0 kemudian diratakan sesuai dengan format vektor fitur. Kedua vektor fitur tersebut masing-masing berukuran 768 dari ConvNeXt dan 1280 dari EfficientNet-B0 digabungkan menghasilkan total 2048 fitur gabungan

Proses tersebut dinamakan Feature Fusion, karena menggabungkan dua arsitekture dengan utujuan meningkatkan kekuatan model untuk mengenal citra. Setelah digabungkan, vektor fitur kemudian dimasukkan ke dalam blok fc yang terdiri dari Linear Layer, aktivasi ReLU, dan Dropout sebesar 0.4 agar tidak terjadi overfitting. Lapisan terakhir adalah Linear Layer yang menghasilkan output sejumlah kelas yang diprediksi. Pada kode tersebut, digunakan Bersama dengan tambahan parameter weight=class_weights agar dataset balanced. Kemudian dilakukan optimisasi menggunakan Adam. Pemilihan Adam bagus untuk model ini, karena feature fusion sangat kompleks. Learning rate 0.0001 dipilih untuk menjaga agar pembaruan bobot tidak terlalu agresif sehingga model dapat beradaptasi.

Pada kode diatas juga terdapat learning rate scheduler StepLR, yang bertahap menurunkan learning rate pada setiapa lima epoch, dengan menurunkan learning rate model dapat lebih focus pada finetuning nantinya.

```python
import torch
import torch.nn as nn
import torchvision.models as models
import torch.optim as optim

model = models.inception_v3(weights=models.Inception_V3_Weights.IMAGENET1K_V1)
# Ganti Fully Connected (FC) sesuai jumlah kelas (5 kelas)
num_features = model.fc.in_features
model.fc = nn.Linear(num_features, 5)
model = model.to(device)

# Loss Function & Optimizer
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.0001)
```

Gambar 3. 27 Modeling InceptionV3

Gambar 3.27 menunjukkan kode proses permodelan untuk InceptionV3 agar dapat digunakan untuk klasifikasi Budaya Indonesia. Model InceptionV3 mendapatkan bobot bantuan dari ImageNet. Setelah model dimuat, langkah berikutnya adalah melakukan penyesuaian pada bagian Fully Connected Layer (FC) karena secara default InceptionV3 memiliki output yang berbeda dari yang dibutuhkan, sehingga dipanggil model.fc.in_features untuk menyisahkan agar sesuai dengan yang dibutuhkan, yaitu 5. Model kemudian dipindahkan ke perangkat komputasi seperti GPU untuk memastikan proses pelatihan dapat berjalan lebih cepat.

Selanjutnya, ditentukan pula Loss Function dan Optimizer yang akan digunakan selama proses pelatihan. CrossEntropyLoss, merupakan standar untuk tugas klasifikasi multi-class karena mampu mengukur seberapa baik model membedakan tiap kategori berdasarkan probabilitas output. Untuk proses optimisasi, digunakan Adam dengan learning rate sebesar 0.0001, yang dikenal sebagai optimizer yang efisien dan stabil pada berbagai kondisi pelatihan, terutama pada dataset yang tidak terlalu besar.

3.3.1.5 Data Training

Tahap selanjutnya merupakan tahap model training, yang dimana data yang sudah dimodelkan akan dilakukan pembelajaran ulang dengan epoch yang sudah ditentukan. Proses training bertujuan agar model bisa mempelajari pola, karakteristik, dan fitur visual dari citra yang terdapat dalam dataset. Satu epoch sama dengan merepresentasikan satu kali proses menyeluruh ketika seluruh data pelatihan dilewatkan ke dalam model, sehingga semakin banyak epoch yang digunakan, semakin banyak pula kesempatan model untuk memperbaiki bobot dan menurunkan nilai loss. Selama tahap training, model akan membandingkan prediksi yang dihasilkan dengan label asli melalui loss function, yang kemudian digunakan untuk memperbarui bobot jaringan dengan menggunakan optimizer sehingga performansi model terus meningkat.

```
#loop
best_val_loss = float("inf")

for epoch in range(1, 21):
    model.train()
    train_loss, correct, total = 0, 0, 0

    for images, labels in train_loader:
        images, labels = images.to(device), labels.to(device)

        optimizer.zero_grad()
        outputs = model(images)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

        train_loss += loss.item() * images.size(0)
        _, predicted = torch.max(outputs, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()

    train_acc = 100 * correct / total
    train_loss /= len(train_loader.dataset)

    print(f"Epoch [{epoch}/20] | Train Loss: {train_loss:.4f} | Train Acc: {train_acc:.2f}%")
```

Gambar 3. 28 Kode Pengulangan Feature Fusion ConvNeXt Tiny dengan EffecientNet B0

Gambar 3.28 menunjukkan kode proses utama dalam melakukan

training loop untuk melatih model selama 20 epoch. setiap epoch diatur dalam sebuah mode training menggunakan model.train(), yang memastikan bahwa seluruh layer yang menggunakan dropout atau batch normalization sesuai saat proses pelatihan. Untuk setiap batch, gradient direset menggunakan optimizer.zero_grad(), kemudian model melakukan forward pass untuk menghasilkan output prediksi citra. Hasil prediksi tersebut dibandingkan dengan label asli menggunakan loss function criterion. Kemudian dilakukan melalui loss.backward(), yang bertujuan untuk memperbarui gradien pada parameter. Terdapat juga optimizer.step() uyang digunakana untuk memperbarui bobot model berdasarkan gradien.

Setelah seluruh batch selesai diproses, akurasi pada epoch akan dihitung dengan membandingkan jumlah prediksi benar terhadap total data yang diproses. Nilai loss juga dirata-ratakan berdasarkan jumlah sampel di dataset pelatihan. Pada akhir setiap epoch, program akan menampilkan hasil berupa training loss dan training accuracy

```
Epoch [1/20] | Train Loss: 1.2234 | Train Acc: 62.56%
Epoch [2/20] | Train Loss: 0.9274 | Train Acc: 70.55%
Epoch [3/20] | Train Loss: 0.6939 | Train Acc: 77.23%
Epoch [4/20] | Train Loss: 0.5435 | Train Acc: 80.82%
Epoch [5/20] | Train Loss: 0.4400 | Train Acc: 83.90%
Epoch [6/20] | Train Loss: 0.3899 | Train Acc: 86.13%
Epoch [7/20] | Train Loss: 0.3133 | Train Acc: 88.36%
Epoch [8/20] | Train Loss: 0.3487 | Train Acc: 88.13%
Epoch [9/20] | Train Loss: 0.2643 | Train Acc: 90.64%
Epoch [10/20] | Train Loss: 0.2185 | Train Acc: 92.29%
Epoch [11/20] | Train Loss: 0.1669 | Train Acc: 94.75%
Epoch [12/20] | Train Loss: 0.1539 | Train Acc: 94.63%
Epoch [13/20] | Train Loss: 0.1565 | Train Acc: 93.72%
Epoch [14/20] | Train Loss: 0.1214 | Train Acc: 95.43%
Epoch [15/20] | Train Loss: 0.1683 | Train Acc: 93.66%
Epoch [16/20] | Train Loss: 0.1080 | Train Acc: 96.18%
Epoch [17/20] | Train Loss: 0.1241 | Train Acc: 96.06%
Epoch [18/20] | Train Loss: 0.0917 | Train Acc: 96.52%
Epoch [19/20] | Train Loss: 0.0813 | Train Acc: 97.09%
Epoch [20/20] | Train Loss: 0.0840 | Train Acc: 96.86%
```

Gambar 3. 29 Hasil Code Epoch Feature Fusion

Gambar 3.29 menunjukan hasil pelatihan selama 20 epoch pada arsitektur Feature Fusion menunjukkan bahwa feature fusion dapat meningkatkan kemampuan model dalam memahami representasi data secara progresif. Pada epoch pertama, nilai training loss masih berada pada tingkat yang tinggi yaitu 1.2234, dengan akurasi sebesar 62.56%. Seiring bertambahnya epoch, terlihat bahwa Pada epoch ke-5, training loss menurun drastis menjadi 0.4400, sementara akurasi meningkat hingga 83.90%. Memasuki epoch ke-10, performa model semakin stabil, dengan training loss mencapai 0.2185 dan akurasi mencapai 92.29%. Pada fase akhir training, yakni epoch 11–20, proses pembelajaran memasuki tahap fine-tuning. Penurunan loss masih terjadi meskipun lebih kecil dibandingkan fase awal, menunjukkan bahwa model mulai mendekati kondisi optimal.

```python
num_epochs = 20

for epoch in range(num_epochs):
    # ---------------------
    # Training
    # ---------------------
    model.train()
    train_loss = 0.0
    train_correct = 0
    train_total = 0

    for images, labels in train_loader:
        images, labels = images.to(device), labels.to(device)

        optimizer.zero_grad()

        # InceptionV3 punya aux_logits saat training
        outputs = model(images)
        if isinstance(outputs, tuple):
            outputs, aux_outputs = outputs
            loss1 = criterion(outputs, labels)
            loss2 = criterion(aux_outputs, labels)
            loss = loss1 + 0.4 * loss2
        else:
            loss = criterion(outputs, labels)

        loss.backward()
        optimizer.step()

        train_loss += loss.item() * images.size(0)
        _, predicted = outputs.max(1)
        train_total += labels.size(0)
        train_correct += predicted.eq(labels).sum().item()

    avg_train_loss = train_loss / len(train_dataset)
    train_acc = 100 * train_correct / train_total

    # ---------------------
    # Validation
    # ---------------------
    model.eval()
    val_loss = 0.0
    val_correct = 0
    val_total = 0

    with torch.no_grad():
        for images, labels in test_loader:
            images, labels = images.to(device), labels.to(device)
            outputs = model(images)

            if isinstance(outputs, tuple):  # saat eval, biasanya tidak ada aux_output
                outputs = outputs[0]

            loss = criterion(outputs, labels)
            val_loss += loss.item() * images.size(0)

            _, predicted = outputs.max(1)
            val_total += labels.size(0)
            val_correct += predicted.eq(labels).sum().item()

    avg_val_loss = val_loss / len(test_dataset)
    val_acc = 100 * val_correct / val_total

    # ---------------------
    # Print Epoch Result
    # ---------------------
    print(f"Epoch [{epoch+1}/{num_epochs}] | "
          f"Train Loss: {avg_train_loss:.4f} | Train Acc: {train_acc:.2f}% | "
          f"Val Loss: {avg_val_loss:.4f} | Val Acc: {val_acc:.2f}%")
```

Gambar 3. 30 Kode Epoch InceptionV3

Gambar 3.30 menunjukan kode pelatihan sebanyak 20 Epoch untuk InceptionV3, Pada tahapan ini proses pelatihan model InceptionV3 dilakukan menggunakan loop sebanyak 20 epoch. Setiap

epoch terdiri dari dua bagian, yaitu proses pelatihan dan proses validasi. Pada proses pelatihan, model diberikan sekumpulan gambar dan label untuk dipelajari. Model kemudian mencoba menebak kelas setiap gambar, lalu menghitung seberapa besar kesalahan dari tebakan tersebut. Kesalahan ini digunakan untuk memperbaiki model agar kemampuannya meningkat di setiap epoch.

Setelah seluruh data melalui proses pelatihan, dihitung nilai akurasi dan loss sebagai gambaran seberapa baik model belajar pada epoch tersebut. Selanjutnya, model diuji pada data validasi. Proses validasi ini bertujuan untuk melihat apakah model dapat mengenali data baru dengan baik dan untuk memastikan bahwa model tidak hanya hafal pada data pelatihan saja. Pada tahap ini model hanya menggunakan keluaran utamanya, tanpa keluaran tambahan. Setelah proses validasi selesai, ditampilkan hasil berupa Train Loss, Train Accuracy, Validation Loss, dan Validation Accuracy untuk setiap epoch. Informasi ini membantu menilai apakah model mengalami peningkatan performa dari waktu ke waktu dan apakah proses pelatihan berjalan dengan baik.

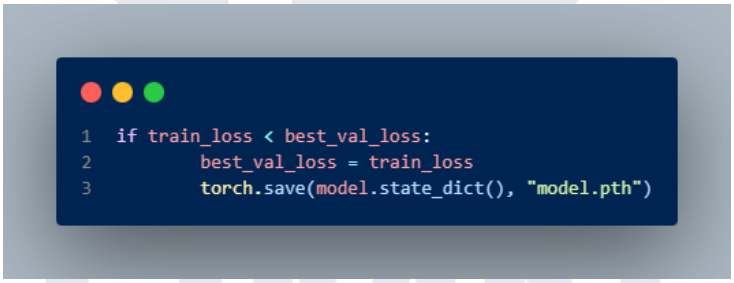
```
Epoch [1/20] | Train Loss: 2.2176 | Train Acc: 60.10% | Val Loss: 1.5241 | Val Acc: 51.35%
Epoch [2/20] | Train Loss: 1.0446 | Train Acc: 75.59% | Val Loss: 1.6380 | Val Acc: 50.90%
Epoch [3/20] | Train Loss: 0.7711 | Train Acc: 80.44% | Val Loss: 1.9177 | Val Acc: 47.52%
Epoch [4/20] | Train Loss: 0.5525 | Train Acc: 86.01% | Val Loss: 2.0636 | Val Acc: 50.68%
Epoch [5/20] | Train Loss: 0.4572 | Train Acc: 89.08% | Val Loss: 2.5041 | Val Acc: 45.50%
Epoch [6/20] | Train Loss: 0.3869 | Train Acc: 90.22% | Val Loss: 1.9492 | Val Acc: 53.38%
Epoch [7/20] | Train Loss: 0.3912 | Train Acc: 90.79% | Val Loss: 2.5894 | Val Acc: 48.87%
Epoch [8/20] | Train Loss: 0.3625 | Train Acc: 90.36% | Val Loss: 2.5205 | Val Acc: 46.62%
Epoch [9/20] | Train Loss: 0.2784 | Train Acc: 92.93% | Val Loss: 2.7759 | Val Acc: 45.50%
Epoch [10/20] | Train Loss: 0.2965 | Train Acc: 91.93% | Val Loss: 2.4804 | Val Acc: 48.20%
Epoch [11/20] | Train Loss: 0.2512 | Train Acc: 93.36% | Val Loss: 3.4392 | Val Acc: 42.57%
Epoch [12/20] | Train Loss: 0.2571 | Train Acc: 93.72% | Val Loss: 3.2214 | Val Acc: 43.24%
Epoch [13/20] | Train Loss: 0.2260 | Train Acc: 93.72% | Val Loss: 2.8484 | Val Acc: 46.62%
Epoch [14/20] | Train Loss: 0.2083 | Train Acc: 95.36% | Val Loss: 3.3728 | Val Acc: 43.47%
Epoch [15/20] | Train Loss: 0.2506 | Train Acc: 93.29% | Val Loss: 3.0270 | Val Acc: 45.27%
Epoch [16/20] | Train Loss: 0.2543 | Train Acc: 93.08% | Val Loss: 3.2050 | Val Acc: 42.57%
Epoch [17/20] | Train Loss: 0.2575 | Train Acc: 93.22% | Val Loss: 3.1153 | Val Acc: 45.95%
Epoch [18/20] | Train Loss: 0.1948 | Train Acc: 95.22% | Val Loss: 2.8583 | Val Acc: 45.95%
Epoch [19/20] | Train Loss: 0.2125 | Train Acc: 93.93% | Val Loss: 2.6824 | Val Acc: 47.97%
Epoch [20/20] | Train Loss: 0.1769 | Train Acc: 95.50% | Val Loss: 2.8744 | Val Acc: 47.30%
```

Gambar 3. 31 Hasil Pelatihan Model InceptionV3

Gambar 3.31 menunjukan hasil pelatihan model InceptionV3 selama 20 epoch menunjukkan bahwa model mampu belajar dengan baik pada data pelatihan, namun mengalami kesulitan untuk menghasilkan performa yang stabil pada data validasi. Terlihat bahwa

nilai Train Loss terus menurun dari epoch pertama hingga akhir, sementara akurasi pelatihan meningkat signifikan dari sekitar 60% menjadi lebih dari 95%. Hal ini menunjukkan bahwa model semakin mahir mengenali pola pada data yang digunakan saat pelatihan. Namun, kondisi ini tidak diikuti oleh peningkatan performa pada data validasi. Val Loss yang dihasilkan fluktuatif, sementara akurasi validasi bergerak di kisaran 42–53% tanpa menunjukkan tren peningkatan yang konsisten.

Perbedaan yang cukup jauh antara akurasi pelatihan dan akurasi validasi menunjukkan bahwa model mengalami overfitting, yaitu model terlalu menghafal data pelatihan sehingga tidak mampu melakukan generalisasi dengan baik pada data baru.

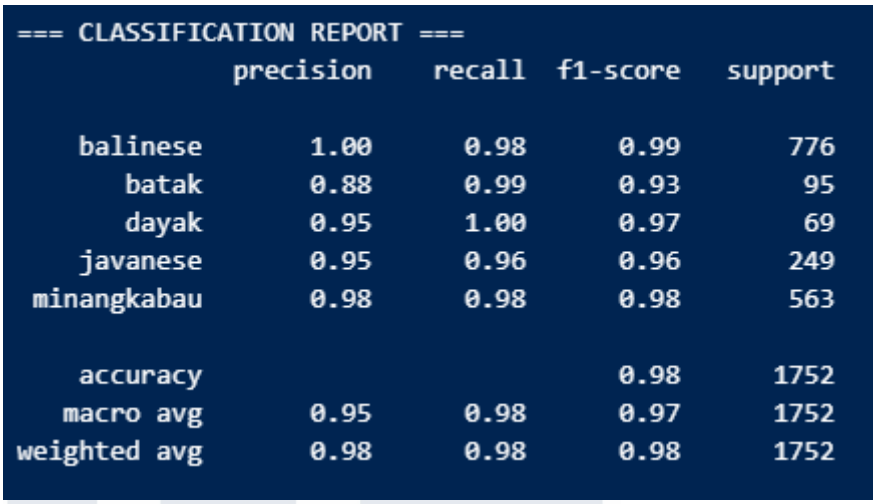


Gambar 3. 32 Kode Menyimpan Model Terbaik Feature Fusion

Gambar 3.32 menunjukan kode untuk menyimpan model terbaik selama proses pelatihan berlangsung

3.3.1.6 Evaluasi dan Komparasi

Selanjutnya model evaluation, merupakan bagian melihat peforma dari model yang telah dibuat. Pada lomba evaluation akan dilakukan melalui klasifikasi report, dan juga confusion matrix



```
=== CLASSIFICATION REPORT ===
              precision    recall  f1-score   support

    balinese       1.00      0.98      0.99       776
       batak       0.88      0.99      0.93        95
       dayak       0.95      1.00      0.97        69
    javanese       0.95      0.96      0.96       249
 minangkabau       0.98      0.98      0.98       563

    accuracy                           0.98      1752
   macro avg       0.95      0.98      0.97      1752
weighted avg       0.98      0.98      0.98      1752
```

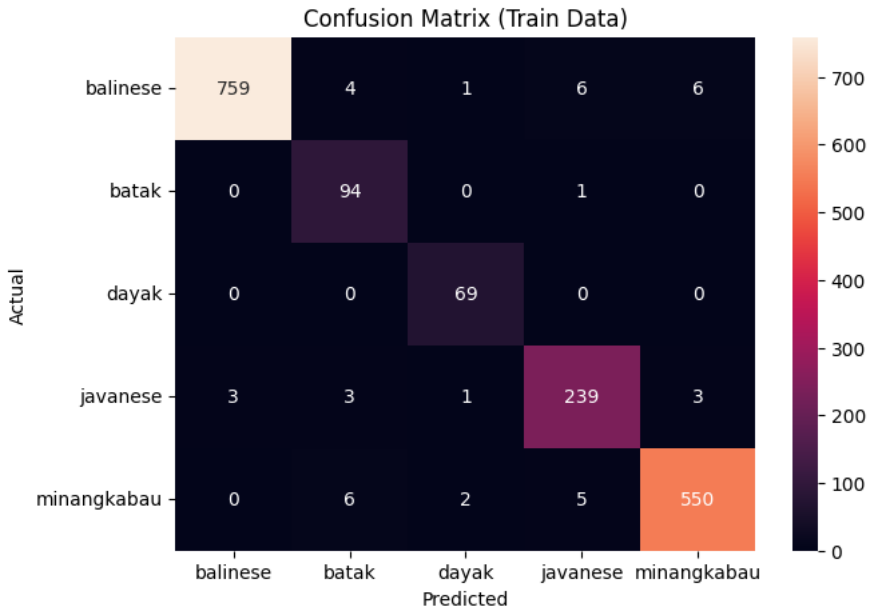
Gambar 3. 33 Classification Report Feature Fusion

Gambar 3.33 menunjukan hasil report classification model feature fusion, menurut hasil report, model feature fusion memiliki performa yang sangat baik dalam mengklasifikasikan seluruh kelas Budaya Indonesia. Secara keseluruhan, model berhasil mencapai akurasi 98%, yang berarti hampir semua prediksi yang dihasilkan benar.

Jika dilihat per kelas, model mampu mengenali citra-citra Budaya Indonesia dengan tingkat ketepatan yang sangat tinggi, di mana nilai precision, recall, dan f1-score semuanya berada di kisaran 95% hingga 99%. Hal ini menunjukkan bahwa model mampu membedakan ciri visual dari masing-masing kelas dengan sangat konsisten.

Untuk kelas batak, performanya sedikit lebih rendah dibanding yang lain, tetapi tetap tinggi dengan f1-score 93%. Ini Rata-rata keseluruhan (macro avg dan weighted avg) juga menunjukkan nilai yang sangat tinggi, mendekati 97–98%, menandakan bahwa performa model stabil dan tidak hanya bagus pada kelas tertentu saja, melainkan

merata di seluruh kelas.



Gambar 3. 34 Confusion Matrix Feature Fusion

Gambar 3.34 menunjukan confusion matrix model feature fusion, berdasarkan hasil Confusion Matrix dapat dilihat bahwa model feature fusion bekerja sangat baik dalam mengenali lima kelas yang ada. Hal ini terlihat dari nilai diagonal yang tinggi, yaitu jumlah prediksi benar untuk setiap kelas. Misalnya, kelas Balinese memiliki 759 prediksi benar, Batak sebanyak 94, Dayak sebanyak 69, Javanese sebanyak 239, dan Minangkabau sebanyak 550. Angka-angka ini menunjukkan bahwa sebagian besar gambar berhasil diklasifikasikan sesuai dengan label aslinya, total yang diprediksi salah hanya 41, hal ini menunjukan bahwa memang model feature fusion yang dibuat sangat bagus.

Classification Report:

	precision	recall	f1-score	support
balinese	0.83	0.92	0.88	142
batak	0.58	0.35	0.44	20
dayak	0.80	0.53	0.64	15
javanese	0.70	0.74	0.72	43
minangkabau	0.83	0.79	0.81	131
accuracy			0.80	351
macro avg	0.75	0.67	0.70	351
weighted avg	0.80	0.80	0.80	351

Gambar 3. 35 Classification Report InceptionV3

Gambar 3.35 merupakan hasil Classification Report dari model inceptionV3, dapat dilihat secara keseluruhan akurasi model berada di angka 80%, yang berarti sebagian besar prediksi sudah sesuai dengan label sebenarnya. Kelas Balinese dan Minangkabau memiliki nilai yang tinggi, terutama pada recall yang menunjukkan model cukup sering menebak kedua kelas ini dengan benar. Hal ini berarti model mampu mengenali ciri khas dari kedua kelas tersebut dengan baik. Kinerja pada kelas Javanese juga cukup stabil, dengan keseimbangan antara prediksi benar dan kesalahan yang masih dalam batas wajar. Namun, model terlihat kesulitan dalam mengenali kelas Batak dan Dayak, yang terlihat dari nilai recall yang rendah—khususnya Batak yang hanya benar dikenali 35% dari total datanya. Ini menunjukkan bahwa banyak gambar dari dua kelas ini yang salah diprediksi menjadi kelas lain.

Secara keseluruhan model InceptionV3 dapat digunakan untuk klasifikasi citra Budaya Indonesia, tetapi hasil nya tidak maksimal, karena sesuai dengan report yang dibuat, masih terdapat kelas dengan nilai f1-score yang rendah, yang menandakan bahwa ada sebuah kesalahan dalam proses permodelan

Tahap terakhir merupakan tahap komparasi model, dimana akan

ditentukan model mana yang akan digunakan untuk mengumpuli hasil lomba.

Tabel 3. 2 Perbandingan Metrik Evaluasi Keseluruhan

Metrik Evaluasi	**InceptionV3**	**Hybrid**	**Perbedaan**
Akurasi (Accuracy)	80%	98%	18%
Macro Avg Precision	0.75	0.95	0.20
Macro Avg Recall	0.67	0.98	0.31
Macro Avg F1-Score	0.7	0.97	0.27
Weighted Avg F1-Score	0.8	0.98	0.18

Tabel 3.2 menunjukkan perbandingan hasil metrik evaluasi antara model InceptionV3 dan Hybrid Model. Berdasarkan tabel, dapat dilihat bahwa Hybrid Model EffecientNet-B0 dengan ConvNext Tiny lebih unggul daripada InceptionV3.

Pada metrik akurasi, Hybrid Model mencapai nilai 98%, sedangkan InceptionV3 memperoleh 80%, sehingga terdapat peningkatan sebesar 18%. Selain itu, nilai Macro Average Precision, Recall, dan F1-Score pada Hybrid Model juga lebih tinggi dibandingkan InceptionV3, yang menunjukkan kemampuan model hybrid dalam melakukan klasifikasi secara lebih merata pada setiap kelas.

Tabel 3. 3 Perbandingan F1-Score Per Kelas (Budaya Indonesia)

Kelas Budaya	F1-Score Inception V3	F1-Score Feature Fusion	Analisis
Balinese	0.88	0.99	Perbedaan signifikan.
Batak	0.44	0.93	Peningkatan Drastis Model Fusion jauh lebih baik mengenali fitur rumah Batak.
Dayak	0.64	0.97	Perbedaan signifikan (+0.33).
Javanese	0.72	0.96	Perbedaan signifikan.
Minangkabau	0.81	0.98	Perbedaan signifikan.

Tabel 3.3 menunjukkan perbandingan F1-Score per kelas Budaya Indonesia antara model InceptionV3 dan Feature Fusion. Secara umum, Feature Fusion memberikan peningkatan performa pada seluruh kelas.

Pada kelas Balinese, nilai F1-Score meningkat dari 0,88 menjadi 0,99. Peningkatan paling signifikan terlihat pada kelas Batak, di mana nilai F1-Score naik drastis dari 0,44 menjadi 0,93, menunjukkan bahwa model fusion jauh lebih baik dalam mengenali ciri rumah

Batak.

Kelas Dayak, Javanese, dan Minangkabau juga mengalami peningkatan yang jelas, dengan selisih nilai yang cukup besar dibandingkan InceptionV3.

Secara keseluruhan, hasil ini menunjukkan bahwa Feature Fusion mampu meningkatkan performa klasifikasi pada setiap kelas Budaya Indonesia, terutama pada kelas yang sebelumnya sulit dikenali oleh model tunggal

Berdasarkan tabel hasil classification report pada tabel 3.2 dan tabel 3.3, Penerapan model Hybrid EffecientNet-B0, dan ConvNeXt-Tiny terbukti memberikan peningkatan performa yang sangat signifikan dibandingkan dengan model InceptionV3. Hal ini ditandai dengan lonjakan akurasi total sebesar 18%, yang meningkat dari 80% pada InceptionV3 menjadi 98% pada model hybrid. Peningkatan ini mengindikasikan bahwa mekanisme penggabungan fitur mampu menangkap karakteristik visual Budaya Indonesia yang kompleks dengan jauh lebih efektif. Keunggulan metode Feature Fusion juga terlihat nyata dalam mengatasi kelemahan InceptionV3 pada kelas-kelas yang sulit diprediksi, di mana F1-Score untuk kelas Batak berhasil melonjak drastis dari 0.44 menjadi 0.93, dan kelas Dayak meningkat dari 0.64 menjadi 0.97. Selain itu, model ini menunjukkan stabilitas dan robustness yang lebih baik, terlihat dari keseimbangan nilai Macro Average Precision (0.95) dan Recall (0.98), yang secara efektif memperbaiki ketimpangan performa pada InceptionV3 yang sebelumnya hanya mencatat Precision 0.75 dan Recall 0.67. Dapat disimpulkan bahwa model feature fusion EfficientNet-B0 dan ConvNeXt-Tiny merupakan model yang paling layak untuk dijadikan hasil akhir dalam perlombaan. Hal ini terlihat dari konsistensi performa model dalam mengenali berbagai kelas dengan tingkat akurasi yang tinggi dan distribusi prediksi yang jauh lebih stabil dibandingkan model InceptionV3.

Pada hasil evaluasi sebelumnya, model feature fusion menunjukkan akurasi yang sangat tinggi pada hampir semua kelas dengan nilai precision, recall, dan f1-score yang mendekati sempurna. Oleh sebab itu akan untuk pengumpulan lomba akan menggunakan feature fusion

3.3.2 Kendala yang Ditemukan

Selama proses pelaksanaan lomba terdapat beberapa kendala yang dialami, beberapa diantaranya berupa:

1. Sulitnya menemukan referensi model dan jurnal yang cocok untuk kegiatan lomba
2. Kesulitan dalam mengerjakan lomba, karena keterbatasan perangkat, dan juga file dataset yang besar, sehingga model yang digunakan terbatas
3. Kurangnya informasi tentang cara pengumpulan hasil lomba di kaggle.
4. Komputasi yang lambat, dan bisa berhenti kapan saja karena menggunakan Google Collab yang terlalu bergantungan pada kecepatan internet

3.3.3 Solusi atas Kendala yang Ditemukan

Tetapi dari semua kendala tersebut, telah dilakukan pencarian solusi atas kendala-kendala berikut berupa:

1. Mencari referensi pada sumber lain, dan tidak hanya berbatas pada jurnal dengan topik yang sama
2. Menggunakan GPU untuk Komputasi karena dengan GPU proses akan lebih cepat, dan juga menggunakan perangkat lain
3. Menggali informasi secara online
4. Pindah alih ke jupyternotebook sehingga

pengerjaan tidak bergantungan dengan internet

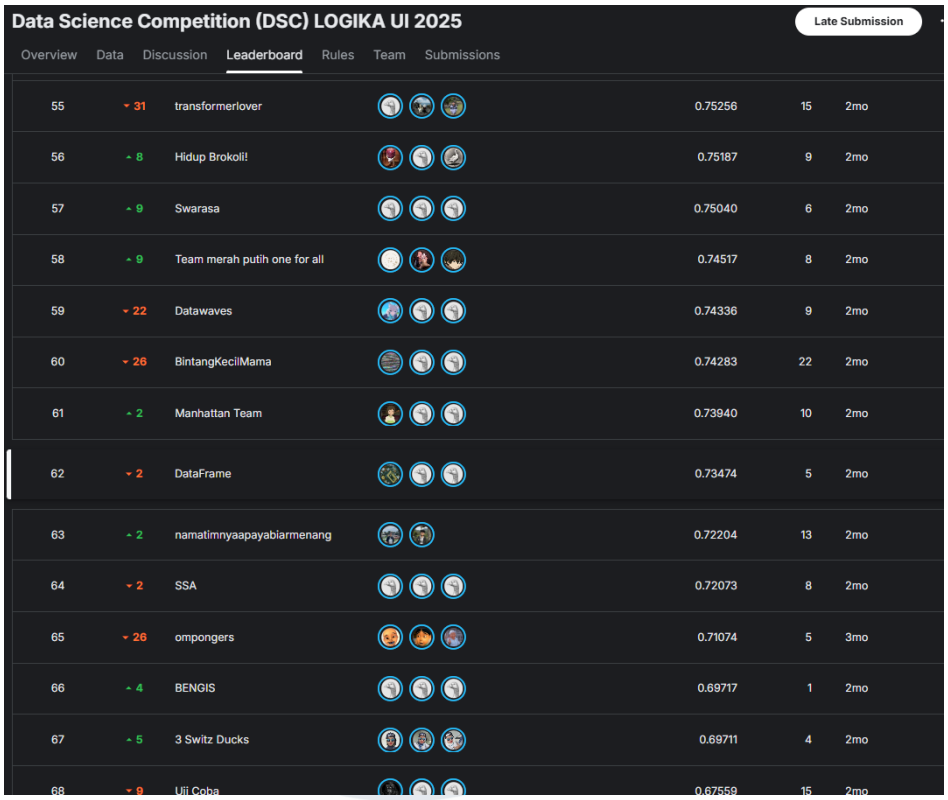
3.4 Hasil Lomba/Kompetisi

Hasil akhir yang dikumpulkan untuk lomba berupa sebuah file CSV yang berisi prediksi model terhadap dataset custom yang telah diuji sebelumnya. File CSV tersebut kemudian diunggah dan disubmit melalui platform Kaggle sesuai dengan ketentuan kompetisi, dan harus dikirim sebelum batas waktu pengumpulan yang telah ditentukan. Dengan demikian, proses submit dapat berjalan sesuai aturan dan hasil dapat dinilai oleh penyelenggara lomba.

id	style
Test_001	balinese
Test_002	minangkabau
Test_003	minangkabau
Test_004	balinese
Test_005	balinese
Test_006	dayak
Test_007	balinese
Test_008	balinese
Test_009	javanese
Test_010	minangkabau

Gambar 3. 36 File Submisi

Gambar 3.36 menunjukan isi dari file testing yang akan dikirim ke Kaggle



Gambar 3. 37 Leaderboard Kelompok Dataframe

Gambar 3.37 merupakan posisi leaderboard kelompok Dataframe, Pada akhir lomba, kelompok dataframe berada di peringkat 62 dari 94 kelompok yang ada, dan tidak lanjut ke babak semifinal dan juga finalis, bisa dilihat bahwa hasil akhir akurasi testing paling tinggi saat melakukan submisi hanya 73,4%

Gambar 3. 38 Sertifikat Kompetisi Lomba LOGIKA UI 2025

Gambar 3.38 merupakan sertifikat yang didapatkan karena sudah mengikuti rangkain acara LOGIKA UI 2025 dari awal sampai akhir, mulai dari 14 juli sampai dengan 23 Novemberd. Sertifikat ini diberi langsung oleh panitia kepada para peserta melalui email.