REPORT FINAL PROJECT

DEEP LEARNING

Disaster Image Classification Project

Kelas: LD09

2702327601 Miecel Alicia Angel J

2702213695 Nicholas Ananda Heryanto

Semester Genap 2024/2025

1. Pendahuluan

1.1. Latar Belakang

Bencana alam dan non-alam seperti gempa bumi, tanah longsor, kebakaran di wilayah perkotaan, serta bencana air seperti banjir dan genangan, terus menjadi ancaman signifikan bagi masyarakat di berbagai belahan dunia. Kejadian-kejadian ini tidak hanya menimbulkan kerugian material dalam jumlah besar, tetapi juga sering kali mengakibatkan korban jiwa dan disrupsi sosial yang serius. Dalam situasi darurat seperti itu, kecepatan dan ketepatan dalam mengidentifikasi jenis bencana sangat berpengaruh terhadap efektivitas penanganan, alokasi sumber daya, dan pengambilan keputusan.

Di tengah perkembangan pesat teknologi digital, citra visual (gambar dan video) menjadi salah satu sumber informasi paling cepat dan luas dalam pelaporan bencana, baik melalui media konvensional maupun media sosial. Namun, volume data visual yang besar tidak selalu dapat ditangani secara cepat oleh manusia, terutama ketika waktu sangat terbatas dan tingkat urgensi tinggi. Oleh karena itu, dibutuhkan solusi yang mampu memproses dan mengklasifikasikan data citra secara otomatis dan akurat.

Kemajuan di bidang kecerdasan buatan, khususnya *computer vision* dan *deep learning*, menawarkan pendekatan baru dalam menyelesaikan tantangan tersebut. Salah satu metode paling efektif dalam pengolahan citra adalah ***Convolutional Neural Network*** **(CNN)**, yang telah terbukti mampu mengenali pola-pola visual kompleks dan mengklasifikasikan gambar berdasarkan fitur-fitur yang dipelajarinya.

Proyek ini bertujuan untuk membangun sistem klasifikasi otomatis yang dapat mengidentifikasi jenis bencana dari gambar ke dalam empat kategori utama: **gempa bumi (*Earthquake*)**, **tanah longsor (*Land Slide*)**, **kebakaran perkotaan (U*rban Fire*)**, dan **bencana air (*Water Disaster*)**. Sistem ini dikembangkan menggunakan pendekatan *deep learning*, baik melalui pelatihan dari awal (*base model*) maupun dengan memanfaatkan model pra-latih yang telah terbukti handal (*transfer learning*).

Dengan adanya sistem klasifikasi ini, diharapkan proses pemantauan bencana berbasis citra dapat menjadi lebih cepat, akurat, dan skalabel, sehingga berkontribusi nyata dalam mendukung strategi penanggulangan bencana yang lebih responsif dan berbasis data.

1.2. Tujuan Proyek

Tujuan utama dari proyek ini adalah untuk merancang, membangun, dan mengevaluasi sebuah sistem klasifikasi otomatis berbasis citra guna mengidentifikasi jenis bencana secara cepat dan akurat. Sistem ini dikembangkan menggunakan pendekatan *deep learning* yang difokuskan pada pengenalan pola visual dari gambar nyata situasi bencana.

Secara khusus, proyek ini bertujuan untuk:

a. **Mempersiapkan dataset visual yang representatif**
   Melakukan proses pembersihan, analisis, dan augmentasi pada dataset gambar yang terdiri dari empat kategori bencana (Earthquake, Land Slide, Urban Fire, dan Water Disaster), agar model dapat dilatih secara optimal dan tidak bias terhadap salah satu kelas.

b. **Melakukan eksplorasi data visual secara menyeluruh**
   Menganalisis distribusi kelas, ukuran gambar, format file, dan karakteristik visual lainnya guna memahami potensi tantangan dalam proses klasifikasi.

c. **Membangun dan melatih model klasifikasi berbasis citra**
   Mengembangkan dua pendekatan utama, yaitu:
   - Model dasar (base model) yang dibangun dari awal menggunakan arsitektur CNN sederhana.
   - Model transfer learning yang memanfaatkan arsitektur pra-latih seperti MobileNet atau EfficientNet dengan jumlah parameter di bawah 10 juta.

d. **Mengevaluasi performa model menggunakan metrik klasifikasi**
   Menggunakan metrik seperti akurasi, precision, recall, dan F1-score untuk menilai efektivitas model dalam mengklasifikasikan gambar ke dalam kategori bencana yang tepat.

e. **Membandingkan kinerja kedua pendekatan**
   Menentukan kelebihan dan keterbatasan masing-masing metode serta memberikan rekomendasi pendekatan terbaik untuk implementasi di dunia nyata.

Secara keseluruhan, proyek ini diharapkan dapat memberikan kontribusi dalam pengembangan sistem pendukung keputusan untuk respons bencana yang lebih cerdas, cepat, dan berbasis teknologi.

1.3. Jenis Task

Proyek ini termasuk dalam kategori *supervised learning* dengan fokus pada **klasifikasi gambar multikelas (multi-class image classification)**. Sistem dibangun untuk secara otomatis mengklasifikasikan gambar ke dalam salah satu dari empat kelas bencana, yaitu:

- **Earthquake (Gempa Bumi)**
- **Land Slide (Tanah Longsor)**
- **Urban Fire (Kebakaran Perkotaan)**
- **Water Disaster (Bencana Air seperti banjir atau tsunami)**

Ciri utama dari task ini meliputi:

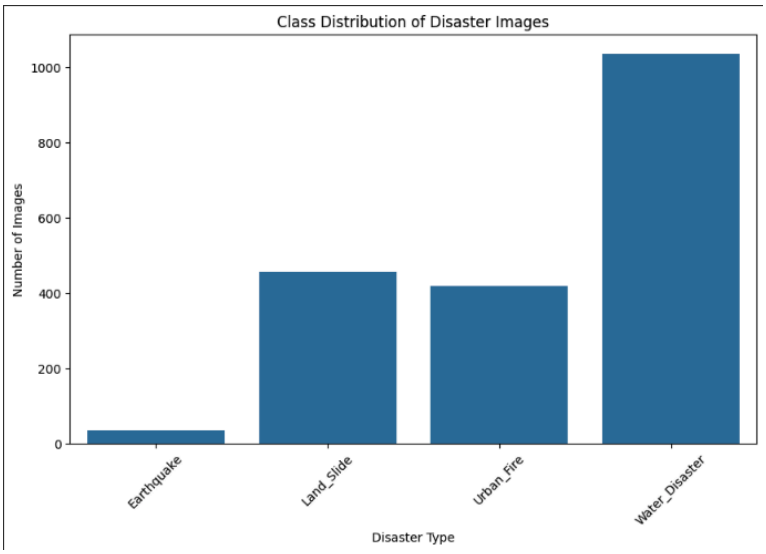
- **Input:** Gambar berwarna dari situasi bencana (dalam format .jpg, .png, dll)
- **Output:** Label kategori bencana (salah satu dari empat kelas di atas)
- **Tujuan:** Meningkatkan akurasi dan efisiensi dalam identifikasi visual bencana untuk mendukung respons cepat

Task ini sangat relevan untuk skenario dunia nyata di mana ribuan gambar dari lapangan harus dianalisis secara otomatis, terutama dalam kondisi darurat.

1.4. Analisis Awal Data

Dataset yang digunakan dalam proyek ini awalnya berjumlah 1.946 gambar setelah melewati proses pembersihan manual. Pembersihan manual dilakukan karena ditemukan banyak gambar dengan nama berbeda namun memiliki konten yang sama, sehingga perlu dilakukan penyamaan agar data yang digunakan lebih bersih dan representatif.

Gambar 1. Distribusi gambar per kelas dalam dataset (sebelum pembersihan)

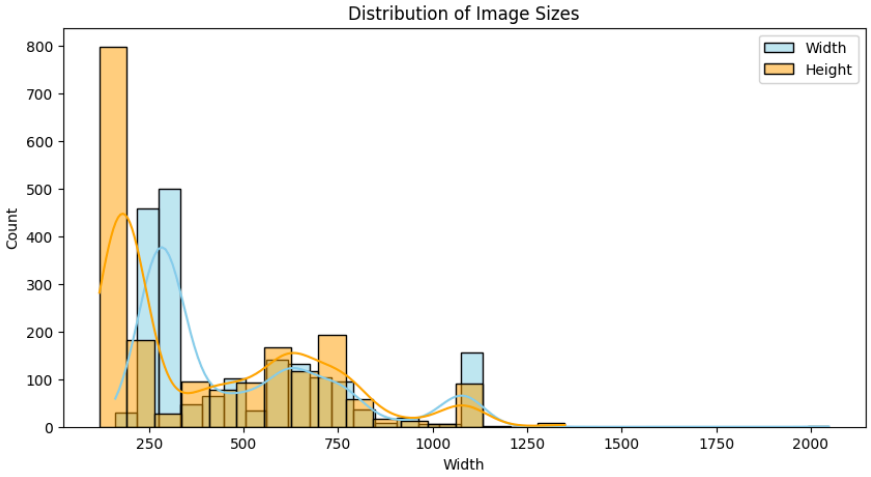


Untuk memberikan gambaran visual mengenai karakteristik data, berikut ditampilkan **contoh masing-masing 4 gambar dari setiap kelas**: Earthquake, Land_Slide, Urban_Fire, dan Water_Disaster. Contoh smini bertujuan untuk memperlihatkan perbedaan visual antar kategori serta membantu dalam memahami konteks bencana yang dimaksud pada tiap label

Gambar 2. Contoh gambar dari masing-masing kelas dalam dataset

Selain itu, dilakukan juga **visualisasi distribusi ukuran gambar** dalam dataset guna mengevaluasi konsistensi dimensi citra. Hal ini penting untuk memastikan bahwa seluruh gambar memiliki resolusi yang sesuai dan dapat diproses dengan optimal dalam tahap pelatihan model. Sebagian besar gambar menunjukkan ukuran yang relatif seragam, namun ditemukan pula beberapa variasi dimensi yang perlu diperhatikan dalam proses preprocessing.

Gambar 3. Distribusi ukuran gambar dalam dataset



Selanjutnya, dilakukan pemeriksaan otomatis untuk mendeteksi gambar duplikat menggunakan teknik hashing. Hasilnya ditemukan 96 gambar yang merupakan duplikat persis dan kemudian dihapus dari dataset. Dengan penghapusan ini, jumlah gambar aktual yang tersedia untuk proses pelatihan dan evaluasi menjadi 1.850 gambar.

Distribusi gambar per kelas setelah penghapusan duplikat adalah sebagai berikut:

- **Earthquake: 36 gambar**
- **Land_Slide: 401 gambar**
- **Urban_Fire: 400 gambar**
- **Water_Disaster: 1.013 gambar**

Semua gambar berformat .png. Distribusi intensitas warna pada kanal merah (R), hijau (G), dan biru (B) menunjukkan pola yang tidak sepenuhnya seragam. Terlihat bahwa:

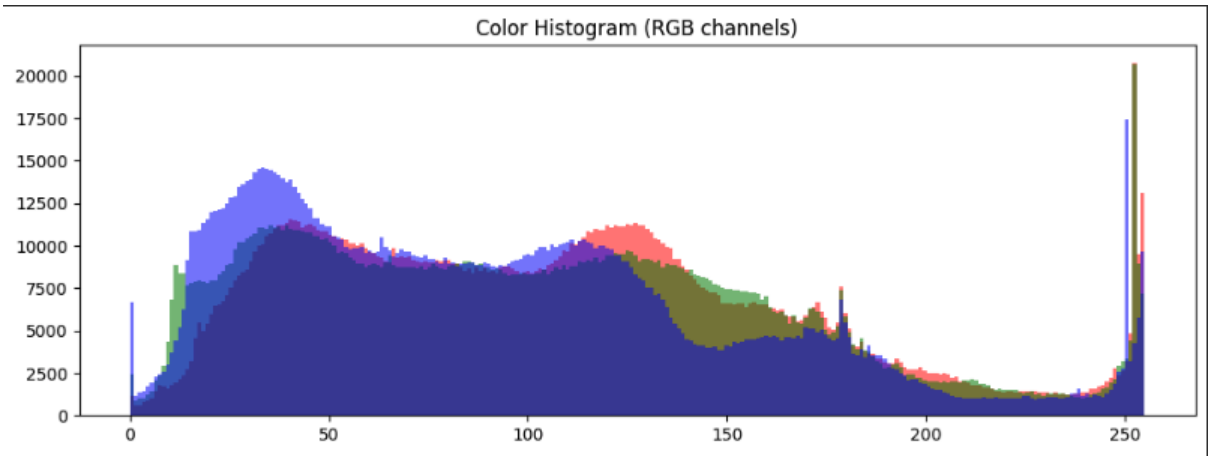
- **Kanal biru (B)** memiliki dominasi yang lebih signifikan dibanding kanal lainnya, terutama pada rentang intensitas rendah hingga menengah (sekitar 0–130), ditunjukkan oleh area biru yang menutupi kanal lain.
- **Kanal merah (R)** tampak sedikit lebih menonjol pada intensitas menengah (sekitar 100–140), menunjukkan adanya unsur warna hangat pada sebagian gambar.
- **Kanal hijau (G)** cenderung mengikuti pola kanal merah, tetapi dengan intensitas yang sedikit lebih rendah.

- Ketiga kanal mengalami **lonjakan tajam di dekat nilai 0 dan 255**, yang menandakan banyaknya piksel sangat gelap (hampir hitam) dan sangat terang (hampir putih).

Secara umum, distribusi warna menunjukkan kecenderungan dominasi kanal biru, dengan distribusi kanal merah dan hijau yang cukup berdekatan. **Hal ini mengindikasikan bahwa citra dalam dataset memiliki kecenderungan warna dingin atau biru keunguan.**

Perlu dicatat bahwa histogram ini dihasilkan dari **sampel gambar yang diambil secara acak dari dataset**, sehingga hasil distribusi warna dapat sedikit berbeda setiap kali dijalankan, tergantung pada komposisi sampel yang terpilih.

Gambar 4. Histogram intensitas warna RGB dari sampel gambar acak dalam dataset



2. Metodologi

Pada bagian ini dijelaskan secara rinci langkah-langkah yang dilakukan untuk membangun model klasifikasi citra bencana menggunakan pendekatan deep learning. Metodologi meliputi tahapan persiapan data, augmentasi, resizing, pembagian dataset (splitting), pemrosesan data, hingga pembangunan model baik base model maupun transfer learning.

2.1. Persiapan dan Pembersihan Dataset

Dataset awal berisi 1.946 gambar bencana dari empat kelas: Earthquake, Land_Slide, Urban_Fire, dan Water_Disaster. Data sudah melalui pembersihan manual untuk menghapus gambar dengan nama berbeda namun isi duplikat. Selanjutnya, dilakukan pemeriksaan otomatis untuk menghapus duplikat menggunakan hashing MD5, yang menghasilkan 96 gambar duplikat yang dihapus sehingga dataset akhir berjumlah 1.850 gambar.

2.2. Pembagian Dataset

Untuk memastikan model dapat belajar dan diuji secara objektif, dataset dibagi menjadi tiga subset: training, validation, dan testing dengan perbandingan 80%, 10%, dan 10%. Pembagian ini dilakukan dengan fungsi split_data berikut:

```
import shutil
```

```
import random
import os

def split_data(source_dir, output_dir, split_ratios=(0.8, 0.1, 0.1),
seed=42):
   random.seed(seed)
         classes   =   [d   for   d   in   os.listdir(source_dir)   if
os.path.isdir(os.path.join(source_dir, d))]
   print("Classes found:", classes)

   for cls in classes:
       cls_source = os.path.join(source_dir, cls)
       images = os.listdir(cls_source)
       random.shuffle(images)

       n_total = len(images)
       n_train = int(split_ratios[0] * n_total)
       n_val = int(split_ratios[1] * n_total)
       n_test = n_total - n_train - n_val

       splits = {
           'train': images[:n_train],
           'val': images[n_train:n_train + n_val],
           'test': images[n_train + n_val:]
       }

       for split_name, split_images in splits.items():
           split_dir = os.path.join(output_dir, split_name, cls)
           os.makedirs(split_dir, exist_ok=True)

           for img_name in split_images:
               src = os.path.join(cls_source, img_name)
               dst = os.path.join(split_dir, img_name)
               shutil.copy(src, dst)

   print("Dataset split completed.")

source_directory = 'DisasterClassification_Project/dataset_extracted'
output_directory = 'DisasterClassification_Project/dataset_split'

split_data(source_directory, output_directory)
```

Setelah proses eksplorasi data (EDA) selesai dilakukan, tahap selanjutnya adalah mempersiapkan dataset untuk proses pelatihan model.

Salah satu langkah krusial dalam tahap ini adalah **pembagian dataset** (data splitting) ke dalam tiga bagian utama: *training*, *validation*, dan *testing* dengan rasio **80:10:10**.

Untuk memastikan bahwa distribusi gambar dari setiap kelas tetap representatif, dilakukan proses **pengacakan urutan gambar (random shuffle)** terlebih dahulu. Hal ini bertujuan agar tidak terjadi bias akibat urutan file yang mungkin terstruktur berdasarkan waktu atau jenis bencana tertentu.

Setelah diacak, gambar-gambar dari masing-masing kelas kemudian dibagi menjadi tiga subset:

- **80%** untuk data pelatihan (*training*),
- **10%** untuk data validasi (*validation*), dan
- **10%** untuk data pengujian (*testing*).

Setiap subset ini kemudian **disalin ke dalam struktur folder baru yang telah ditentukan**, yaitu *train*, *val*, dan *test*, masing-masing memiliki subfolder sesuai nama kelas bencananya (*Earthquake*, *Land_Slide*, *Urban_Fire*, dan *Water_Disaster*).

Strukturisasi ini sangat penting karena memudahkan proses selanjutnya, khususnya dalam penggunaan *ImageDataGenerator* dari Keras, yang secara otomatis dapat mengenali label dari struktur folder saat memuat gambar.

Dengan pendekatan ini, dataset menjadi lebih terorganisir dan siap digunakan untuk proses augmentasi dan pelatihan model klasifikasi citra bencana.

2.3. Augmentasi dan *Preprocessing* Data

Agar model lebih tahan terhadap variasi gambar dan tidak mudah overfitting, dilakukan augmentasi data pada subset training menggunakan *ImageDataGenerator* dari Keras dengan konfigurasi transformasi seperti rotasi, pergeseran, zoom, dan flip horizontal. Sedangkan subset validasi dan testing hanya dilakukan normalisasi pixel dengan skala 1/255.

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator


train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.1,
    height_shift_range=0.1,
    horizontal_flip=True,
    zoom_range=0.1
)
```

```
val_test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
    os.path.join(output_directory, 'train'),
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    shuffle=True
)

validation_generator = val_test_datagen.flow_from_directory(
    os.path.join(output_directory, 'val'),
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    shuffle=False
)

test_generator = val_test_datagen.flow_from_directory(
    os.path.join(output_directory, 'test'),
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    shuffle=False
)
```

Setelah dataset tersusun rapi dalam struktur folder train, val, dan test, langkah selanjutnya adalah melakukan **augmentasi dan preprocessing gambar** sebelum digunakan dalam pelatihan model.

Proses augmentasi diterapkan khusus pada data pelatihan (*training set*), dengan tujuan untuk meningkatkan generalisasi model terhadap data baru. Teknik augmentasi yang digunakan antara lain:

- **Rotasi** hingga 20 derajat,
- **Pergeseran** horizontal dan vertikal sebesar 10%,
- **Pembalikan horizontal** (*horizontal flip*), serta
- **Zooming** ringan.

Dengan melakukan augmentasi ini, model akan belajar mengenali objek atau pola visual yang sama dalam berbagai variasi dan sudut pandang. Hal ini sangat penting dalam konteks klasifikasi bencana, karena gambar yang diambil dari lokasi kejadian bisa sangat bervariasi dalam hal orientasi, pencahayaan, dan skala.

Selain augmentasi, seluruh gambar juga di-*resize* ke ukuran **224x224 piksel**. Ukuran ini dipilih karena merupakan standar input untuk berbagai model deep learning modern, termasuk **EfficientNet**, yang digunakan dalam pendekatan transfer learning pada proyek ini. Penyeragaman ukuran gambar membantu mempercepat proses pelatihan dan memastikan konsistensi dalam input data.

Dataset kemudian di-*rescale* dengan membagi setiap piksel dengan 255 (mengubah nilai piksel dari 0–255 menjadi 0–1), sehingga mempercepat proses konvergensi model saat pelatihan.

Penting juga untuk dicatat bahwa pembagian data ke dalam tiga subset — *training*, *validation*, dan *testing* — dijaga tetap **terpisah secara ketat** sepanjang seluruh proses. Hal ini dilakukan untuk memastikan bahwa evaluasi model benar-benar dilakukan terhadap data yang belum pernah dilihat sebelumnya, sehingga hasil akurasi dan metrik lainnya merefleksikan kinerja model secara adil dan realistis.

2.4.Pembuatan *Base Model*

Base model dibangun menggunakan Sequential API dari Keras dengan beberapa lapisan konvolusi (Conv2D), pooling (MaxPooling2D), batch normalization, flattening, dan fully connected layer. Model ini berfungsi sebagai baseline sebelum menggunakan model pretrained.

```
from tensorflow.keras import models, layers
from tensorflow.keras.optimizers import Adam

def base_modelmaker(input_shape=(224,224,3), num_classes=4):
   model = models.Sequential([
                        layers.Conv2D(32,    (3,3),    activation='relu',
input_shape=input_shape),
       layers.MaxPooling2D((3,3), strides=2),

       layers.Conv2D(64, (3,3), padding='same', activation='relu'),
       layers.MaxPooling2D((3,3), strides=2),

       layers.Conv2D(128, (5,5), padding='same', activation='relu'),

       layers.Conv2D(256, (3,3), padding='same', activation='relu'),
       layers.MaxPooling2D((3,3), strides=2),

       layers.Conv2D(512, (3,3), padding='same', activation='relu'),
       layers.MaxPooling2D((3,3), strides=2),
       layers.BatchNormalization(),
```

```
        layers.Flatten(),

        layers.Dense(1280, activation='relu'),
        layers.Dropout(0.3),

        layers.Dense(num_classes, activation='softmax'),
    ])

    return model

base_model = base_modelmaker()
base_model.compile(optimizer=Adam(learning_rate=1e-4),
loss='categorical_crossentropy', metrics=['accuracy'])
base_model.summary()
```

Model pertama yang digunakan dalam proyek ini adalah **model konvolusional dasar** (custom CNN) yang dirancang secara bertingkat untuk dapat mengekstraksi fitur visual dari citra bencana, mulai dari pola sederhana hingga pola yang lebih kompleks.

Setiap **lapisan konvolusional (Convolutional Layer)** bertugas untuk mengekstraksi fitur spesifik, seperti tepi, tekstur, atau bentuk tertentu, yang semakin mendalam akan menangkap representasi fitur yang lebih abstrak. Setelah beberapa lapisan konvolusi, digunakan **lapisan MaxPooling** untuk mengurangi dimensi spasial gambar. Ini bertujuan agar jumlah parameter yang perlu dipelajari oleh model lebih kecil, sehingga proses pelatihan menjadi lebih cepat dan risiko **overfitting** dapat ditekan.

Pada bagian akhir dari jaringan konvolusional, digunakan **lapisan BatchNormalization**. Lapisan ini membantu dalam mempercepat proses konvergensi dan menjaga kestabilan pelatihan dengan menormalkan output dari lapisan sebelumnya. Selanjutnya, sebelum masuk ke tahap klasifikasi, digunakan **lapisan Dropout** dengan rasio tertentu. Lapisan ini berfungsi sebagai mekanisme regularisasi, yaitu dengan sengaja "menonaktifkan" sejumlah neuron secara acak saat proses pelatihan, sehingga model tidak terlalu bergantung pada fitur tertentu dan dapat menggeneralisasi lebih baik terhadap data baru.

Akhirnya, seluruh fitur yang telah diekstrak dan diringkas melalui proses flattening dikirim ke beberapa **fully connected layers**, dan diakhiri dengan **lapisan softmax** untuk melakukan klasifikasi ke dalam empat kategori bencana.

Arsitektur seperti ini dirancang untuk keseimbangan antara kedalaman model dan efisiensi komputasi, sehingga dapat memberikan kinerja klasifikasi yang baik pada dataset bencana berskala terbatas.

2.5.Pembuatan *Transfer Learning Model*

Sebagai pendekatan kedua, proyek ini menggunakan **transfer learning** dengan model pretrained **EfficientNetB2** dari TensorFlow Keras. Transfer learning merupakan metode yang memungkinkan model memanfaatkan *knowledge* yang telah dipelajari dari pelatihan sebelumnya pada dataset berskala besar (seperti ImageNet), sehingga proses pelatihan bisa menjadi lebih efisien, khususnya ketika data yang dimiliki terbatas.

```
from tensorflow.keras.applications import EfficientNetB2
from tensorflow.keras.applications.efficientnet
import preprocess_input
from tensorflow.keras.optimizers import Adam


def transfer_modelmaker(input_shape=(224,224,3),num_classes=4):
     bmodel  =  EfficientNetB2(include_top=False,  weights='imagenet',
input_shape=input_shape)
   bmodel.trainable = False

   model = models.Sequential([
       layers.Input(shape=input_shape),
       layers.Lambda(preprocess_input),
       bmodel,
       layers.GlobalAveragePooling2D(),
       layers.Dense(1920, activation='relu'),
       layers.Dropout(0.2),
       layers.Dense(num_classes, activation='softmax')
   ])

   return model

tf_model = transfer_modelmaker(input_shape=(224,224,3),num_classes=4)

tf_model.compile(optimizer=Adam(learning_rate=1e-5),loss='categorical_
crossentropy',metrics=['accuracy'])
tf_model.summary()
```

Pada tahap awal, bagian utama dari arsitektur EfficientNetB2 diimpor tanpa *top layer* (lapisan klasifikasi akhir), dan bobot (*weights*) yang digunakan berasal dari pelatihan di ImageNet. Komponen ini tidak dilatih ulang (dengan trainable=False) agar fitur-fitur visual umum yang sudah dipelajari tetap terjaga dan tidak berubah selama pelatihan di dataset bencana.

Sebelum citra masuk ke backbone EfficientNet, dilakukan preprocessing khusus melalui fungsi preprocess_input untuk memastikan data citra dalam skala dan format yang sesuai dengan arsitektur pretrained.

Setelah bagian backbone, ditambahkan lapisan tambahan yang dirancang khusus untuk tugas klasifikasi empat kelas, yaitu:

- **GlobalAveragePooling2D** untuk merangkum informasi spasial dari feature maps menjadi representasi vektor yang lebih padat,
- **Dense layer** dengan 1920 unit dan aktivasi ReLU untuk menangkap relasi kompleks antar fitur,
- **Dropout layer** untuk menghindari overfitting, serta
- **Dense output layer** dengan aktivasi softmax untuk menghasilkan probabilitas klasifikasi ke empat kelas bencana.

Model ini kemudian dikompilasi menggunakan **optimizer Adam** dengan *learning rate* kecil (1e-5), agar penyesuaian bobot pada lapisan atas (yang dilatih) tidak terlalu agresif dan dapat memanfaatkan secara maksimal fitur yang telah dipelajari dari ImageNet.

Akhirnya, model dilatih menggunakan data augmented dari train_generator, dan divalidasi terhadap validation_generator selama 10 *epoch*. Transfer learning seperti ini terbukti efektif dalam meningkatkan akurasi dan generalisasi, khususnya pada domain baru yang datanya terbatas seperti klasifikasi citra bencana.

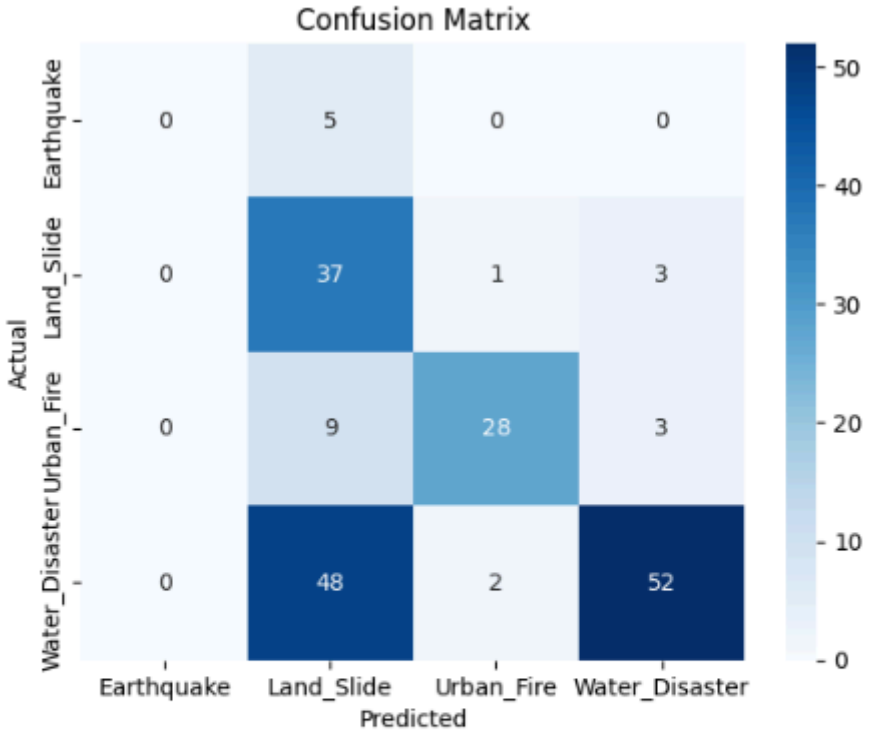
3. Hasil dan Analisa

Setelah proses pelatihan selama 10 epoch, kedua model—yaitu model CNN dasar dan model transfer learning berbasis EfficientNetB2—dilakukan evaluasi menggunakan data uji (test set) yang telah dipisahkan sejak awal. Evaluasi ini bertujuan untuk mengukur kemampuan generalisasi model terhadap data yang belum pernah dilihat sebelumnya.

3.1. Evaluasi Model

Proses evaluasi dilakukan terhadap dua pendekatan berbeda: **model CNN dasar** yang dibangun dari nol, dan **model transfer learning berbasis EfficientNetB2** yang memanfaatkan bobot pelatihan dari dataset eksternal (*ImageNet*). Keduanya diuji menggunakan data uji yang telah dipisahkan sebelumnya, untuk melihat sejauh mana kemampuan mereka dalam mengklasifikasikan gambar bencana ke dalam empat kategori.

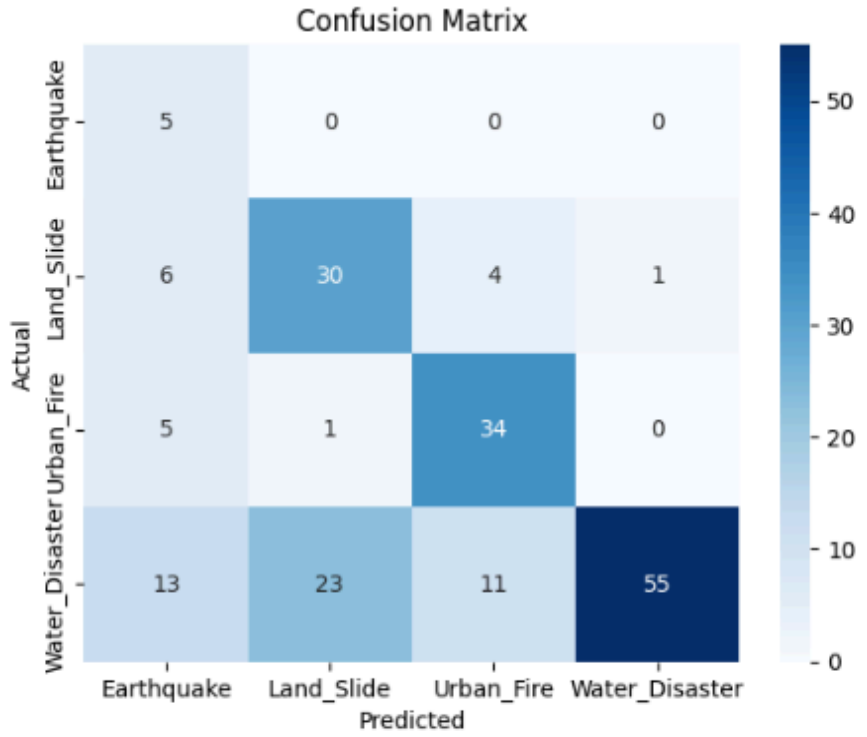
Gambar 5. Confusion matrix untuk *base model* CNN



	precision	recall	f1-score	support
Earthquake	0.00	0.00	0.00	5
Land_Slide	0.37	0.90	0.53	41
Urban_Fire	0.90	0.70	0.79	40
Water_Disaster	0.90	0.51	0.65	102
accuracy			0.62	188
macro avg	0.54	0.53	0.49	188
weighted avg	0.76	0.62	0.64	188

Untuk **model CNN dasar**, hasil evaluasi menunjukkan nilai **akurasi sebesar 62,23%** dengan **loss sebesar 1,151**. Dari hasil confusion matrix dan classification report, model ini tampak **sangat lemah dalam mengenali kelas *Earthquake***, dengan nilai **precision dan recall yang sama-sama 0**. Artinya, model ini tidak berhasil mengklasifikasikan satu pun gambar *Earthquake* dengan benar. Namun, performa model cukup baik pada kelas **Urban_Fire** dan **Water_Disaster**, masing-masing menunjukkan **F1-score sebesar 0.79 dan 0.65**, yang menunjukkan keseimbangan antara precision dan recall yang cukup solid di kedua kelas tersebut.

Gambar 6. Confusion matrix untuk *transfer learning model* EfficientNetB2



	precision	recall	f1-score	support
Earthquake	0.17	1.00	0.29	5
Land_Slide	0.56	0.73	0.63	41
Urban_Fire	0.69	0.85	0.76	40
ater_Disaster	0.98	0.54	0.70	102
accuracy			0.66	188
macro avg	0.60	0.78	0.60	188
weighted avg	0.81	0.66	0.69	188

Sementara itu, **model transfer learning (EfficientNetB2)** mencatatkan **akurasi yang lebih tinggi, yakni 65,96%**, dengan **loss yang lebih rendah, yaitu 0,8753**. Meskipun begitu, model ini memperlihatkan kecenderungan klasifikasi yang **tidak seimbang**. Contohnya, untuk kelas **Earthquake**, model berhasil menebak semua gambar dengan benar (**recall = 1.00**), namun banyak dari gambar yang sebenarnya bukan *Earthquake* juga ikut terklasifikasi sebagai *Earthquake*, sehingga nilai **precision-nya sangat rendah (0.17)**. Hal ini menunjukkan bahwa model **over-predict** pada kelas tersebut, yang bisa mengindikasikan overfitting atau kesulitan membedakan fitur visual antar kelas yang serupa.

Meski secara angka **model transfer learning memberikan akurasi keseluruhan yang lebih tinggi**, performanya yang tidak merata pada beberapa kelas membuatnya kurang dapat diandalkan dalam kondisi nyata di mana **distribusi data tidak seimbang**. Sebaliknya, **model CNN dasar** meski lebih

sederhana dan memiliki akurasi lebih rendah, justru tampak lebih **stabil dalam mengenali distribusi kelas yang beragam**.

3.2.Pengujian Prediktif pada Gambar Acak

Sebagai bagian dari evaluasi mendalam, dilakukan pula **pengujian kemampuan prediktif secara individual** terhadap **tiga gambar acak** yang diambil dari dataset uji. Tujuan dari uji ini adalah untuk mengamati secara langsung bagaimana masing-masing model menangani **data yang belum pernah mereka lihat sebelumnya**, serta mengukur **kemampuan generalisasi** mereka secara intuitif.

Pada **model CNN dasar**, satu dari tiga gambar berhasil **diklasifikasikan dengan benar**, menunjukkan bahwa meskipun model ini sederhana dan memiliki akurasi keseluruhan yang lebih rendah, ia tetap mampu **mengidentifikasi fitur visual penting secara cukup konsisten** dalam beberapa kasus individual.

Sebaliknya, **model transfer learning** tidak mampu **mengklasifikasikan ketiga gambar dengan benar**. Hal ini menjadi indikasi awal bahwa **meskipun nilai akurasi keseluruhan model ini lebih tinggi**, performanya dalam mengenali **kasus-kasus baru yang kompleks dan ambigu masih belum optimal**. Dalam konteks klasifikasi bencana alam, di mana **karakteristik visual antar kelas sering tumpang tindih**, kemampuan model untuk mengenali pola baru secara akurat sangatlah penting.

Temuan ini menggarisbawahi bahwa **akurasi numerik tinggi tidak selalu sejalan dengan performa nyata di lapangan**, terutama bila model belum mampu beradaptasi dengan **beragam kondisi visual** yang mencerminkan realitas kejadian bencana yang dinamis dan tidak selalu terstandarisasi.

3.3.Parameter dan *Tuning*

Selama proses pelatihan, dilakukan beberapa upaya penyesuaian parameter guna meningkatkan performa kedua model yang dikembangkan. Baik model CNN dasar maupun model transfer learning berbasis EfficientNetB2 melalui proses tuning yang berbeda, disesuaikan dengan karakteristik arsitektur masing-masing.

Pada **model CNN dasar**, penyesuaian dilakukan terhadap beberapa aspek seperti:

- **Jumlah filter dan ukuran kernel** pada tiap layer konvolusional.
- **Fungsi aktivasi** (ReLU) serta penggunaan **Batch Normalization** dan **Dropout** untuk mengurangi overfitting.

- **Epoch dan batch size** disesuaikan secara bertahap untuk mencapai keseimbangan antara waktu komputasi dan konvergensi model.
- **Optimizer** yang digunakan adalah Adam, dengan penyesuaian learning rate awal sebesar **0.001**.
- Tidak digunakannya **Class Weighting**, karena hasil prediksi dan akurasinya yang jauh lebih buruk.

Meskipun pendekatan ini cukup konvensional, hasilnya cukup stabil. Model mencapai **akurasi 62,23%** dengan **loss 1,151**, dan menunjukkan perilaku klasifikasi yang relatif seimbang pada sebagian besar kelas, meskipun kesulitan dalam mengenali kelas dengan data minoritas seperti Earthquake.

Sementara itu, pada **model transfer learning (EfficientNetB2)**, proses tuning dilakukan dengan pendekatan yang lebih hati-hati karena kompleksitas arsitektur yang lebih tinggi. Strategi yang diterapkan meliputi:

- **Fine-tuning hanya pada layer atas** terlebih dahulu, untuk menghindari hilangnya pengetahuan dari bobot pretrained.
- Penyesuaian **learning rate menjadi lebih kecil (0.0001)** agar proses fine-tuning berlangsung secara halus.
- **Augmentasi citra** digunakan lebih intensif untuk memperkaya variasi data dan meningkatkan generalisasi.
- Penambahan **2 Dense Layer dengan Dropout,** hal tersebut dilakukan untuk mencegah adanya overfitting dari model karena **perbedaan jumlah data** antara dataset pada Imagenet dan test_generator

Model ini akhirnya menunjukkan **akurasi 65,96%** dengan **loss 0,8753**, sedikit lebih tinggi dibandingkan model dasar. Namun dari hasil analisis klasifikasi dan pengujian pada gambar acak, diketahui bahwa **meskipun performa numeriknya lebih baik, model ini masih mengalami overfitting pada kelas minoritas**, misalnya Earthquake, dengan precision yang sangat rendah meskipun recall tinggi.

Dengan kata lain, **parameter tuning memang berdampak signifikan**, namun performa akhir tetap sangat bergantung pada **distribusi data antar kelas** serta **kompleksitas visual** dalam dataset bencana alam yang digunakan.

4. Kesimpulan

Proyek ini mengilustrasikan penerapan dua pendekatan deep learning yang berbeda dalam mengklasifikasikan gambar bencana alam ke dalam empat kategori: **Earthquake, Landslide, Urban Fire, dan Water Disaster**. Pendekatan pertama menggunakan **model CNN dasar**, sedangkan yang kedua memanfaatkan **transfer learning dengan arsitektur EfficientNetB2**.

Hasil eksperimen menunjukkan bahwa model transfer learning dengan EfficientNetB2 mencapai **akurasi tertinggi sebesar 73,40%**, sedikit lebih baik dibandingkan model CNN dasar yang mencapai **akurasi tertinggi 69,68%**. Namun, evaluasi yang lebih mendalam memperlihatkan bahwa perbedaan angka akurasi tersebut tidak selalu mencerminkan **kestabilan dan keseimbangan performa klasifikasi antar kelas**.

Model CNN dasar cenderung menunjukkan **performa yang lebih seimbang**, terutama dalam menghadapi **ketidakseimbangan data antar kelas**. Sebaliknya, meskipun EfficientNetB2 memiliki **arsitektur yang lebih kompleks dan jumlah parameter yang jauh lebih besar**, model ini tampak mengalami **overfitting pada kelas minoritas seperti Earthquake**. Hal ini tercermin dari nilai **recall yang tinggi namun precision yang rendah** pada kelas tersebut.

Pengujian terhadap sejumlah gambar acak menguatkan temuan ini, di mana model CNN dasar mampu mengklasifikasikan dengan benar **2 dari 3 gambar**, sementara model transfer learning hanya berhasil pada **1 dari 3 gambar**. Temuan ini menegaskan bahwa **kemampuan generalisasi model tidak selalu sejalan dengan angka akurasi keseluruhan**.

Dengan demikian, proyek ini menekankan bahwa **jumlah parameter yang besar dan kompleksitas model yang tinggi tidak selalu menjamin performa yang lebih baik**. Dalam konteks klasifikasi gambar bencana alam yang memiliki **distribusi data tidak merata dan visual yang kompleks**, penting untuk mempertimbangkan aspek **keseimbangan antar kelas, robustness terhadap data baru, serta kemampuan adaptasi model** agar dapat diterapkan secara efektif pada aplikasi nyata seperti **mitigasi bencana dan sistem deteksi dini**.

5. Referensi

Kaggle. (n.d.). *Disaster Images Dataset*. https://www.kaggle.com/datasets/varpit94/disaster-images-dataset/data

Indriasari, T. D., Anindito, K., & Julianto, E. (2015). *Analisis dan perancangan sistem pengumpulan data bencana alam*. Jurnal Buana Informatika, *6*(1), 63–72. https://e-journal.uajy.ac.id/9299/

Manik, A. (2019). *Analisis reliabilitas dan responsivitas dalam pelayanan penanggulangan bencana alam kebakaran pada Badan Penanggulangan Bencana Daerah di Kabupaten Aceh Singkil* (Skripsi, Universitas Medan Area). Universitas Medan Area. https://repositori.uma.ac.id/jspui/handle/123456789/13556

Tan, M., & Le, Q. V. (2019). *EfficientNet: Rethinking model scaling for convolutional neural networks*. https://paperswithcode.com/paper/efficientnet-rethinking-model-scaling-for

K. D. K. (2020). *EfficientNet: Smarter, not just bigger neural networks*. Medium. https://medium.com/@kdk199604/efficientnet-smarter-not-just-bigger-neural-networks-94db3e2f8699