
brain_gym
Kelompok 7

Anggota Kelompok 7 brain_gym

H071221066

Zabryna Andiny

H071221031

Aan Syawaluddin

H071221101

Khalizatul Jannah

Latar Belakang

Brain gym adalah serangkaian gerakan sederhana yang meningkatkan fungsi kognitif, koordinasi motorik, dan kesehatan mental. Aktivitas ini mengaktifkan kedua belahan otak untuk mendukung konsentrasi, daya ingat, dan kemampuan berpikir kritis.

Teknologi deep learning dapat menganalisis gerakan brain gym melalui pose estimation, namun penelitian masih terbatas akibat kurangnya dataset standar.

Masalah yang dihadapi

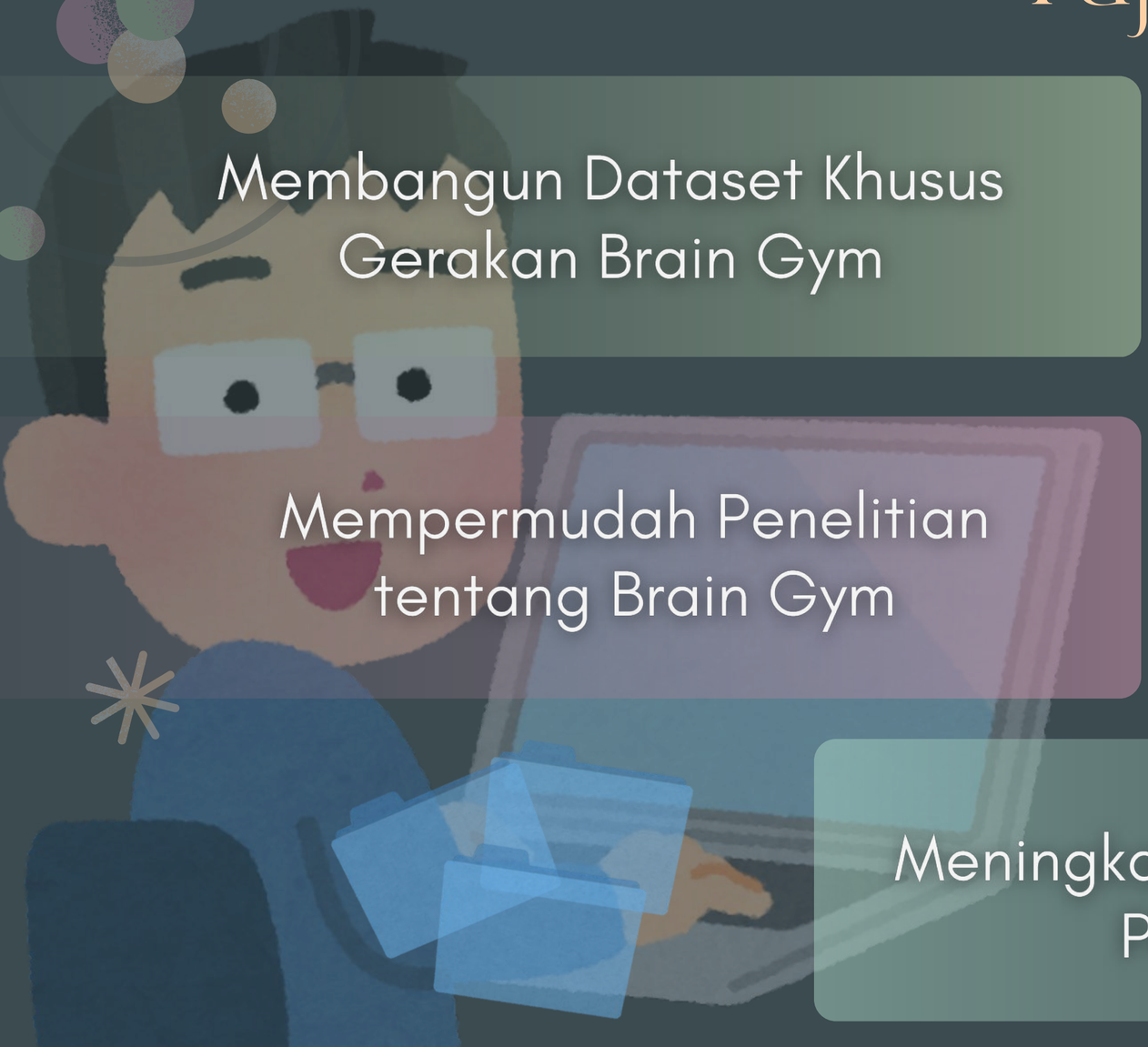
Ketiadaan Dataset Terstandar

Keterbatasan Aksesibilitas Data

Kurangnya Kompatibilitas dengan Teknologi Modern

Sulitnya Personalisasi Latihan

Tujuan



Membangun Dataset Khusus Gerakan Brain Gym

Mendukung Pengembangan Model Teknologi

Mempermudah Penelitian tentang Brain Gym

Mendukung Pengembangan Aplikasi Kesehatan

Meningkatkan Aksesibilitas Teknologi bagi Peneliti dan Pengembang

Literatur Review

Brain Gym atau Senam Otak adalah serangkaian 26 gerakan yang dirancang untuk meningkatkan fungsi kognitif, seperti konsentrasi, memori, koordinasi motorik, serta kesejahteraan psikologis. Program ini berfokus pada sinkronisasi otak kanan dan kiri, peningkatan konsentrasi, serta pengendalian emosi, logika, peningkatan fungsi kognitif dan pengurangan stres pada lansia (Azizah et al., 2017) serta pengurangan kecemasan dan depresi pada mahasiswa (Siroya et al., 2021).

Model deteksi gerakan dengan teknologi seperti Mediapipe dan LSTM (Kumar et al., 2024) menunjukkan potensi integrasi AI dalam terapi gerakan. Tantangannya adalah pengembangan dataset yang dapat menangkap variasi lingkungan untuk meningkatkan akurasi dan efisiensi terapi berbasis gerakan.

Dataset dan Material

Dataset Brain Gym dikembangkan dengan merekam enam gerakan tangan spesifik (default, back slap, little finger, peace, pistol, thumb side) menggunakan kamera. Perekaman dilakukan selama 2 detik per gerakan, dengan variasi sudut, kecepatan, dan jarak tangan dalam lingkungan terkendali untuk memastikan kualitas data. Video dianotasi dan dilabeli menggunakan RoboFlow, yang juga membagi data untuk pelatihan, validasi, dan pengujian, mendukung akurasi model analisis gerakan tangan.

Kelas Dataset	Jumlah Dataset (Video)	Durasi
default	50 video	2 detik/video
peace	20 video	2 detik/video
pistol	26 video	2 detik/video
thumb_side	40 video	2 detik/video
little_finger	20 video	2 detik/video
back_slap	31 video	2 detik/video

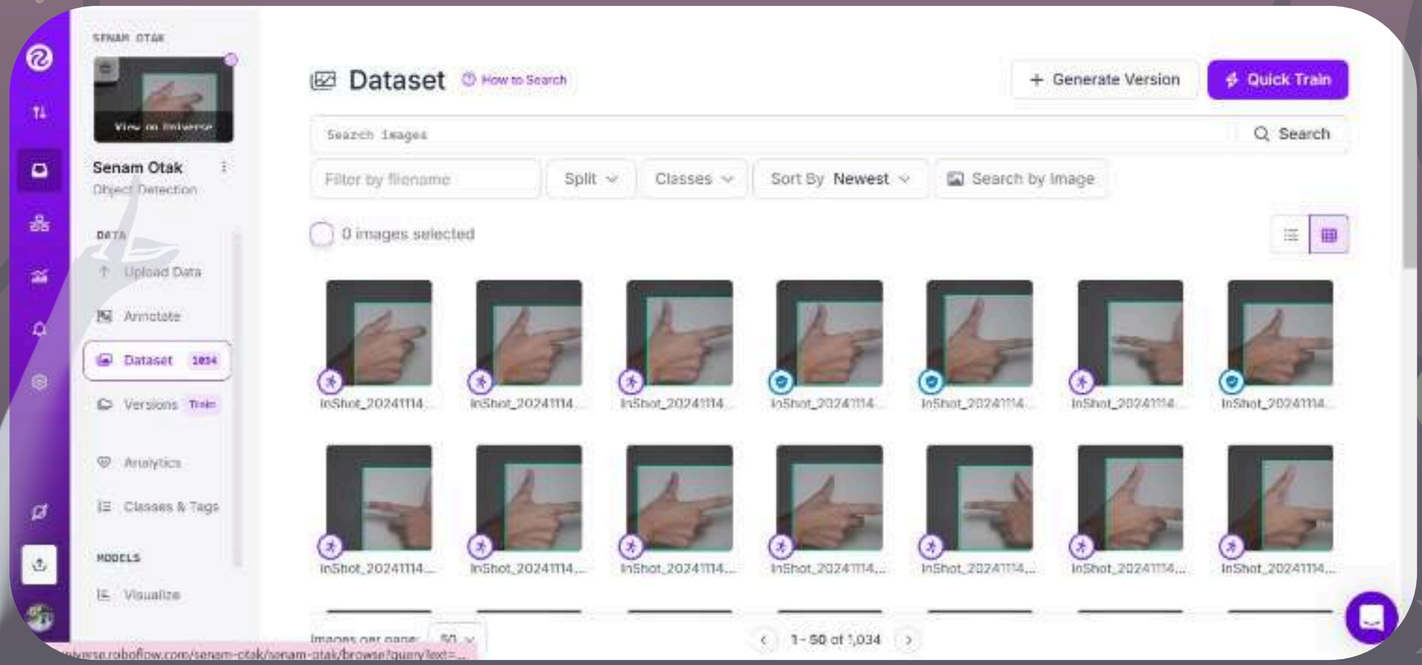
Langkah-langkah Pra-proses

Ekstraksi Frame

Anotasi dengan RoboFlow

Ekspor Dataset

Langkah-langkah Pra-proses



Kelas Dataset	Jumlah Dataset (Video)	Jumlah Anotasi
default	50 video	313 gambar
peace	20 video	302 gambar
pistol	26 video	314 gambar
thumb_side	40 video	300 gambar
little_finger	20 video	298 gambar
back_slap	31 video	301 gambar
Total	**187 video**	**1.828 gambar**

Fitur, Label dan Tools

Dataset ini terdiri dari gambar frame berdimensi 224x224 piksel dari video gerakan tangan. Fitur berupa visual setiap gerakan, sedangkan label mencakup enam kategori: default, back slap, little finger, peace, pistol, dan thumb side. Label digunakan untuk melatih model CNN agar mampu mengenali gerakan dengan akurat.

Camera Tools

Google Collab

RoboFlow

TensorFlow/Keras

TensorFlow

Matplotlib & Seaborn

Persiapan Pembuatan Model

Import Package

```
[ ] import os
    import json
    import random
    import numpy as np
    import matplotlib.pyplot as plt
    from roboflow import Roboflow
    from collections import Counter
    from matplotlib import patches
    from tensorflow.keras.preprocessing import image
    from tensorflow.keras.applications.vgg16 import preprocess_input
    from tensorflow.keras.applications import VGG16
    from tensorflow.keras.models import Model
    from tensorflow.keras.preprocessing.image import load_img, img_to_array
    from tensorflow.keras.layers import LSTM, Dense, TimeDistributed, Flatten, Dropout
    from pycocotools.coco import COCO
    from tensorflow.keras.utils import to_categorical
```

Mengunduh Dataset

```
[ ] # Mengunduh dataset menggunakan Roboflow API
    rf = Roboflow(api_key="O2yMGJAWCYipgASp95jo")
    project = rf.workspace("senam-otak").project("senam-otak")
    version = project.version(3)
    dataset = version.download("coco")
```

```
loading Roboflow workspace...
loading Roboflow project...
Downloading Dataset Version Zip in Senam-Otak-3 to coco:: 100%|██████████| 30700/30700 [00:03<00:00, 9700.43it/s]

Extracting Dataset Version Zip to Senam-Otak-3 in coco:: 100%|██████████| 1858/1858 [00:00<00:00, 2603.48it/s]
```

Direktori Dataset dan Path File Anotasi COCO

```
[ ] # Direktori dataset
    dataset_dir = "/content/Senam-Otak-3"

    # Path ke file anotasi COCO untuk train, valid, dan test
    train_annotation_file = os.path.join(dataset_dir, 'train/_annotations.coco.json')
    valid_annotation_file = os.path.join(dataset_dir, 'valid/_annotations.coco.json')
    test_annotation_file = os.path.join(dataset_dir, 'test/_annotations.coco.json')
```

Exploratory Data Analysis (EDA)

Menampilkan kategori dalam anotasi COCO

```
[ ]  coco_annotation_file = os.path.join(dataset_dir, 'train/_annotations.coco.json')

    if os.path.exists(coco_annotation_file):
        coco = COCO(coco_annotation_file)

        # Memuat kategori
        categories = coco.loadCats(coco.getCatIds())
        print("Kategori dalam anotasi COCO:")
        for category in categories:
            print(f"ID: {category['id']}, Nama: {category['name']}")
    else:
        print(f"File anotasi tidak ditemukan: {coco_annotation_file}")
```

```
loading annotations into memory...
Done (t=0.01s)
creating index...
index created!
Kategori dalam anotasi COCO:
ID: 0, Nama: exercise
ID: 1, Nama: back_slap
ID: 2, Nama: default
ID: 3, Nama: little_finger
ID: 4, Nama: peace
ID: 5, Nama: pistol
ID: 6, Nama: thumb_side
```

Menampilkan kategori dalam anotasi COCO

```
[ ]  def count_images_in_coco(json_file):
        with open(json_file, 'r') as f:
            data = json.load(f)

        images = data.get('images', [])
        return len(images)

    total_images = 0
    for annotation_file in [train_annotation_file, valid_annotation_file, test_annotation_file]:
        if os.path.exists(annotation_file):
            num_images = count_images_in_coco(annotation_file)
            print(f"Jumlah gambar di {annotation_file}: {num_images}")
            total_images += num_images
        else:
            print(f"File anotasi tidak ditemukan: {annotation_file}")

    print(f'Jumlah gambar dalam dataset COCO (train, valid, test): {total_images}')
```

```
Jumlah gambar di /content/Senam-Otak-3/train/_annotations.coco.json: 1300
Jumlah gambar di /content/Senam-Otak-3/valid/_annotations.coco.json: 369
Jumlah gambar di /content/Senam-Otak-3/test/_annotations.coco.json: 181
Jumlah gambar dalam dataset COCO (train, valid, test): 1850
```

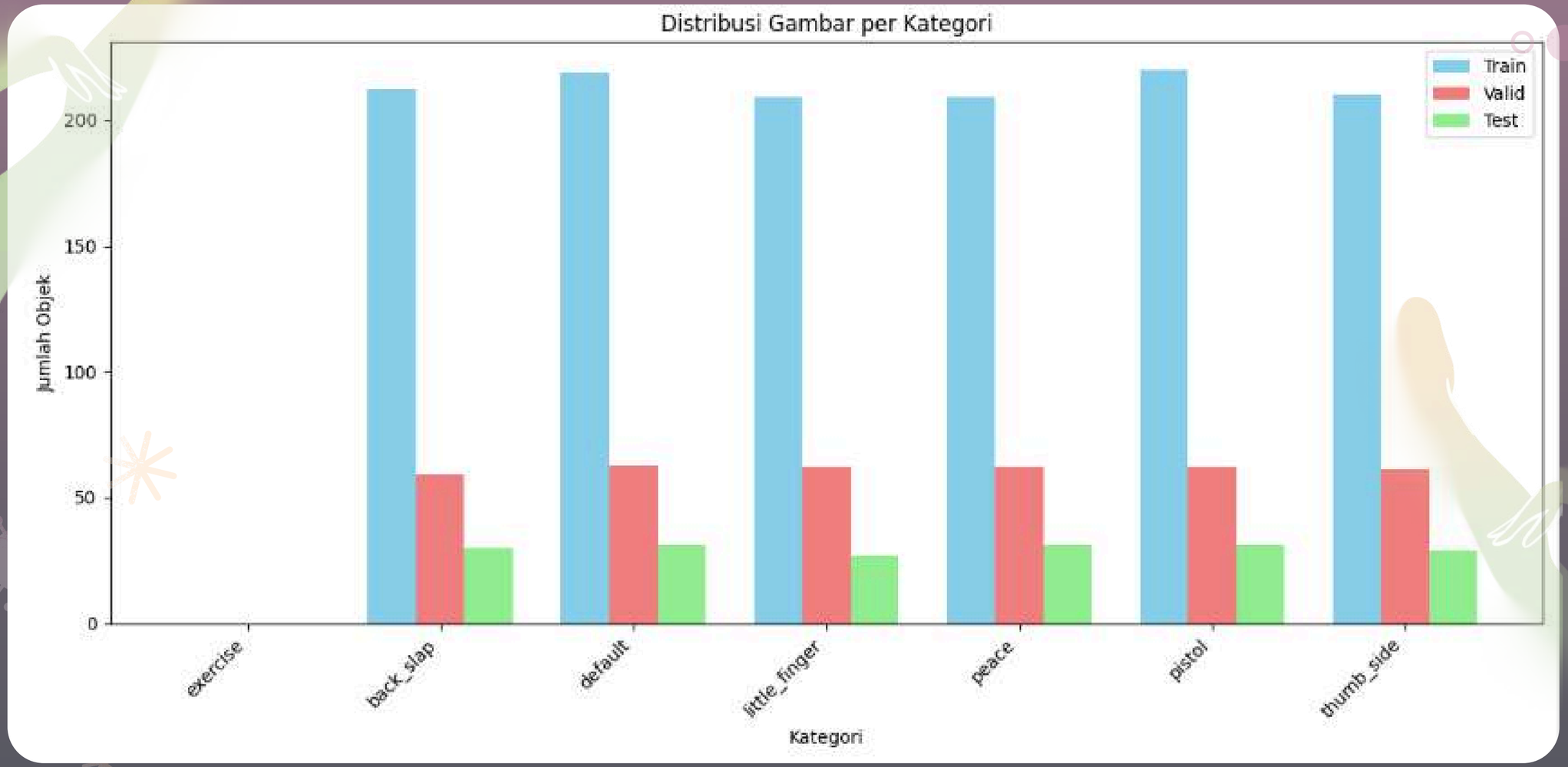
Exploratory Data Analysis (EDA)

Menampilkan contoh gambar beserta anotasi

```python
def display_sample_images(image_paths, coco, num_samples=6):
    plt.figure(figsize=(15, 5))
    displayed_labels = set()
    displayed_count = 0

    random.shuffle(image_paths)

    for img_path in image_paths:
        if displayed_count >= num_samples:
            break

        img = plt.imread(img_path)
        img_filename = os.path.basename(img_path)

        # Mendapatkan ID gambar berdasarkan nama file
        img_ids = [img['id'] for img in coco.dataset['images'] if img['file_name'] == img_filename]

        if img_ids:
            img_id = img_ids[0]
            annotation_ids = coco.getAnnIds(imgIds=img_id)

            if annotation_ids:
                annotations = coco.loadAnns(annotation_ids)
                labels = {coco.loadCats(ann['category_id'])[0]['name'] for ann in annotations}

                if not labels - displayed_labels:
                    continue

                displayed_labels.update(labels)

                plt.subplot(1, num_samples, displayed_count + 1)
                plt.imshow(img)
                plt.axis('off')
                plt.title(f"Label: {', '.join(labels)}")
                displayed_count += 1
        else:
            continue

    plt.show()

# Memuat anotasi COCO
coco = COCO(coco_annotation_file)

image_dir = os.path.join(dataset_dir, 'train')
image_paths = [os.path.join(image_dir, fname) for fname in os.listdir(image_dir) if fname.endswith('.jpg')]

display_sample_images(image_paths, coco)
```

```
loading annotations into memory...
Done (t=0.01s)
creating index...
index created!
```

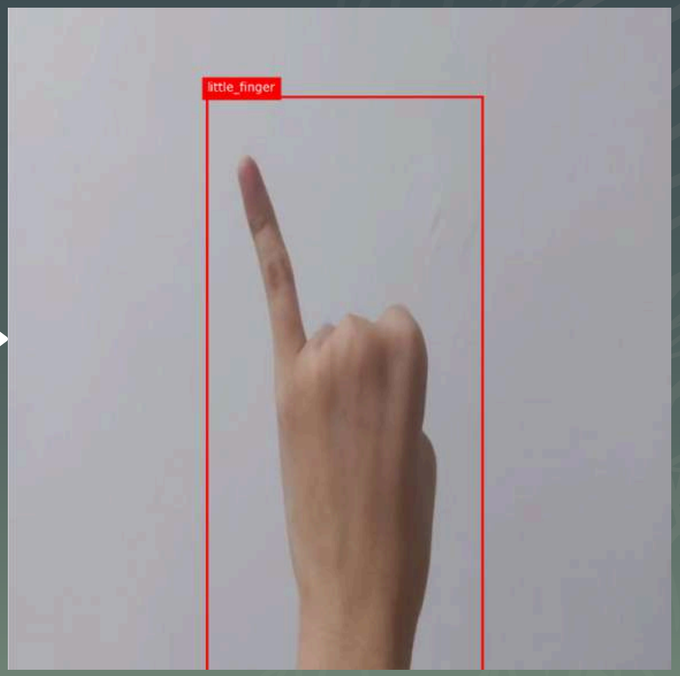
Exploratory Data Analysis (EDA)



Exploratory Data Analysis (EDA)

Visualisasi Contoh Anotasi

```python
def visualize_annotations(image_id, coco):
    img_info = coco.loadImgs(image_id)[0]
    img_path = os.path.join(image_dir, img_info['file_name'])
    img = plt.imread(img_path)

    fig, ax = plt.subplots(1, figsize=(10, 10))
    ax.imshow(img)

    # Memuat anotasi
    ann_ids = coco.getAnnIds(imgIds=img_info['id'])
    anns = coco.loadAnns(ann_ids)

    for ann in anns:
        bbox = ann['bbox']
        category_name = coco.loadCats(ann['category_id'])[0]['name']
        rect = patches.Rectangle((bbox[0], bbox[1]), bbox[2], bbox[3],
                                 linewidth=2, edgecolor='r', facecolor='none')
        ax.add_patch(rect)
        plt.text(bbox[0], bbox[1] - 5, category_name, color='white',
                 backgroundcolor='red', fontsize=10)
    plt.axis('off')
    plt.show()

image_ids = coco.getImgIds()

random_image_id = random.choice(image_ids)

visualize_annotations(random_image_id, coco)
```

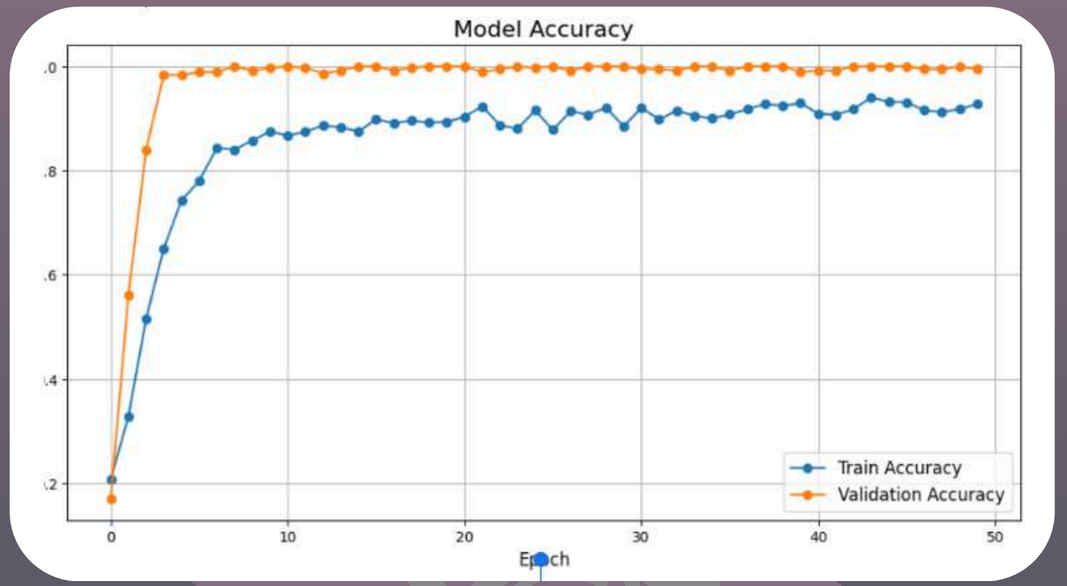


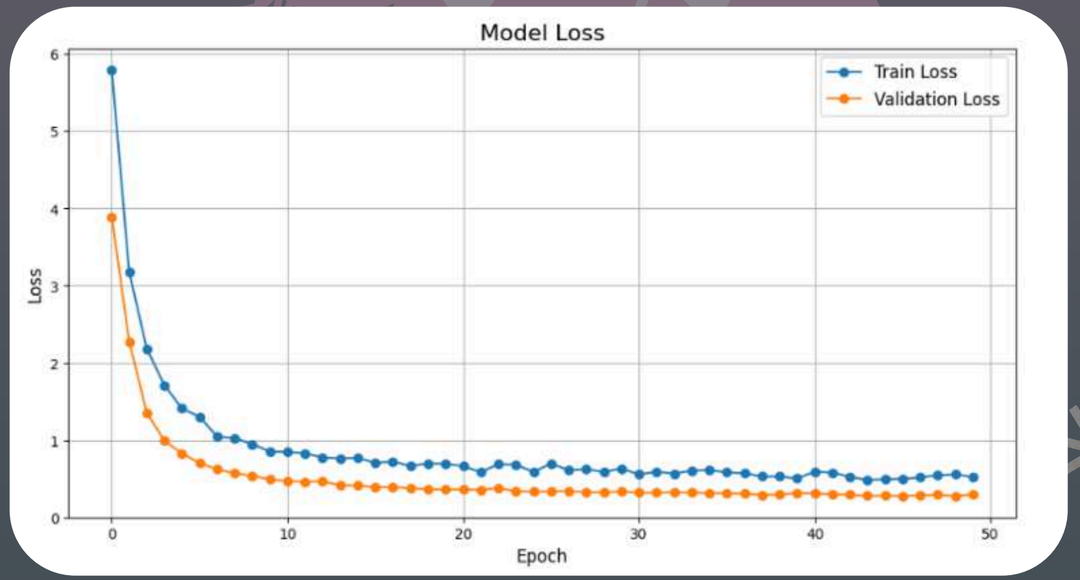
Proses Pembuatan Model

```python
# Model CNN-LSTM
def create_model(input_shape, num_classes):
    cnn_base = tf.keras.applications.VGG16(include_top=False, weights="imagenet", input_shape=(*input_shape, 3))
    cnn_base.trainable = False

    model = models.Sequential([
        cnn_base,
        layers.GlobalAveragePooling2D(),
        layers.Dropout(0.5),
        layers.RepeatVector(10),
        layers.LSTM(128, return_sequences=False, kernel_regularizer=l2(0.01)),
        layers.Dense(128, activation="relu", kernel_regularizer=l2(0.01)),
        layers.Dropout(0.5),
        layers.Dense(num_classes, activation="softmax", kernel_regularizer=l2(0.01))
    ])

    return model
```

```python
# Compile model
model = create_model(input_shape, num_classes)
model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"])


# Training model
history = model.fit(
    train_generator,
    validation_data=(x_valid, y_valid),
    epochs=50,
)
```

Visualisasi Grafik Accuracy dan Loss





Optimisasi

```
[ ] # 2. Optimisasi
    model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"])
    reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=3, min_lr=1e-6, verbose=1)

    history = model.fit(
        x_train, y_train,
        validation_data=(x_valid, y_valid),
        epochs=10,
        batch_size=32,
        callbacks=[reduce_lr]
    )
```

```
Epoch 1/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 11s 218ms/step - accuracy: 0.9596 - loss: 0.3916 - val_accuracy: 0.9946 - val_loss: 0.2720 - learning_rate: 0.0010
Epoch 2/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 9s 197ms/step - accuracy: 0.9648 - loss: 0.3958 - val_accuracy: 1.0000 - val_loss: 0.2498 - learning_rate: 0.0010
Epoch 3/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 10s 196ms/step - accuracy: 0.9704 - loss: 0.3478 - val_accuracy: 1.0000 - val_loss: 0.2449 - learning_rate: 0.0010
Epoch 4/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 10s 197ms/step - accuracy: 0.9760 - loss: 0.3330 - val_accuracy: 0.9810 - val_loss: 0.2923 - learning_rate: 0.0010
Epoch 5/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 10s 196ms/step - accuracy: 0.9604 - loss: 0.3769 - val_accuracy: 0.9973 - val_loss: 0.2464 - learning_rate: 0.0010
Epoch 6/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 0s 153ms/step - accuracy: 0.9761 - loss: 0.3133
Epoch 6: ReduceLROnPlateau reducing learning rate to 0.0005000000237487257.
40/40 ━━━━━━━━━━━━━━━━━━━━ 11s 220ms/step - accuracy: 0.9757 - loss: 0.3142 - val_accuracy: 0.9919 - val_loss: 0.2613 - learning_rate: 0.0010
Epoch 7/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 9s 198ms/step - accuracy: 0.9710 - loss: 0.3318 - val_accuracy: 1.0000 - val_loss: 0.2296 - learning_rate: 5.0000e-04
Epoch 8/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 10s 199ms/step - accuracy: 0.9914 - loss: 0.2707 - val_accuracy: 1.0000 - val_loss: 0.2181 - learning_rate: 5.0000e-04
Epoch 9/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 10s 197ms/step - accuracy: 0.9777 - loss: 0.3032 - val_accuracy: 0.9973 - val_loss: 0.2186 - learning_rate: 5.0000e-04
Epoch 10/10
40/40 ━━━━━━━━━━━━━━━━━━━━ 10s 198ms/step - accuracy: 0.9782 - loss: 0.2865 - val_accuracy: 1.0000 - val_loss: 0.2121 - learning_rate: 5.0000e-04
```

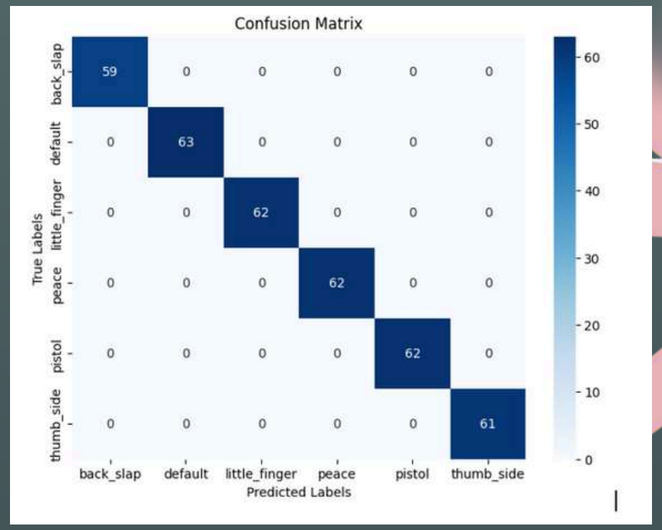
Hasil Optimisasi

Tabel Precision, Recall, F1-Score, and Support for Each Label

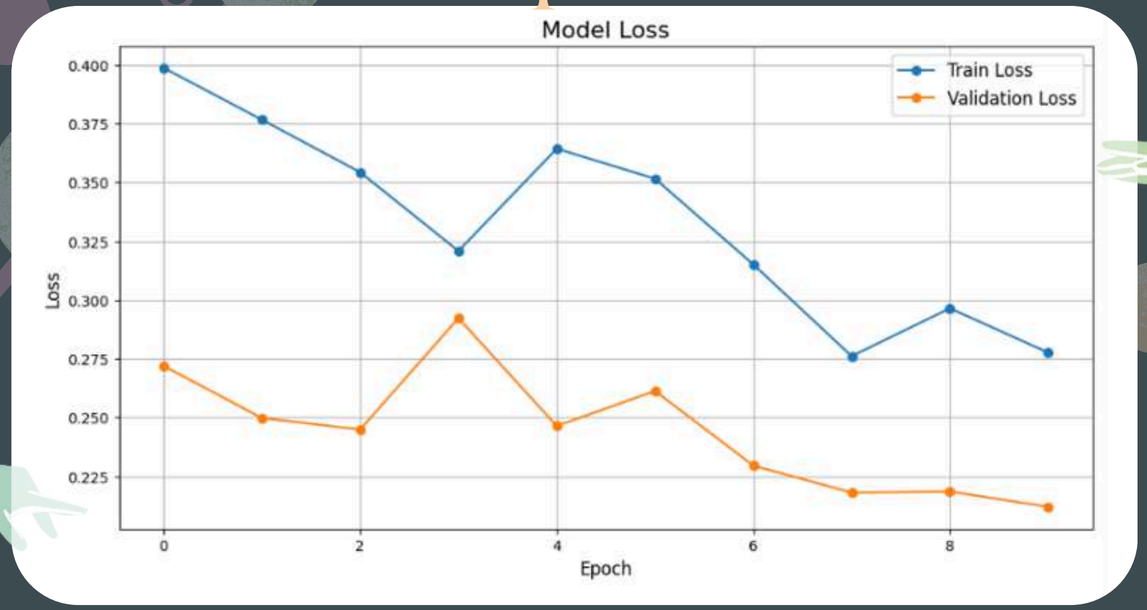
Label	Precision	Recall	F1-score	Support
1. back_slap	1.00	1.00	1.00	59
2. default	1.00	1.00	1.00	63
3. little_finger	1.00	1.00	1.00	62
4. peace	1.00	1.00	1.00	62
5. pistol	1.00	1.00	1.00	62
6. thumb_side	1.00	1.00	1.00	61
Accuracy			**1.00**	**369**
Macro Avg	1.00	1.00	1.00	369
Weighted Avg	1.00	1.00	1.00	369

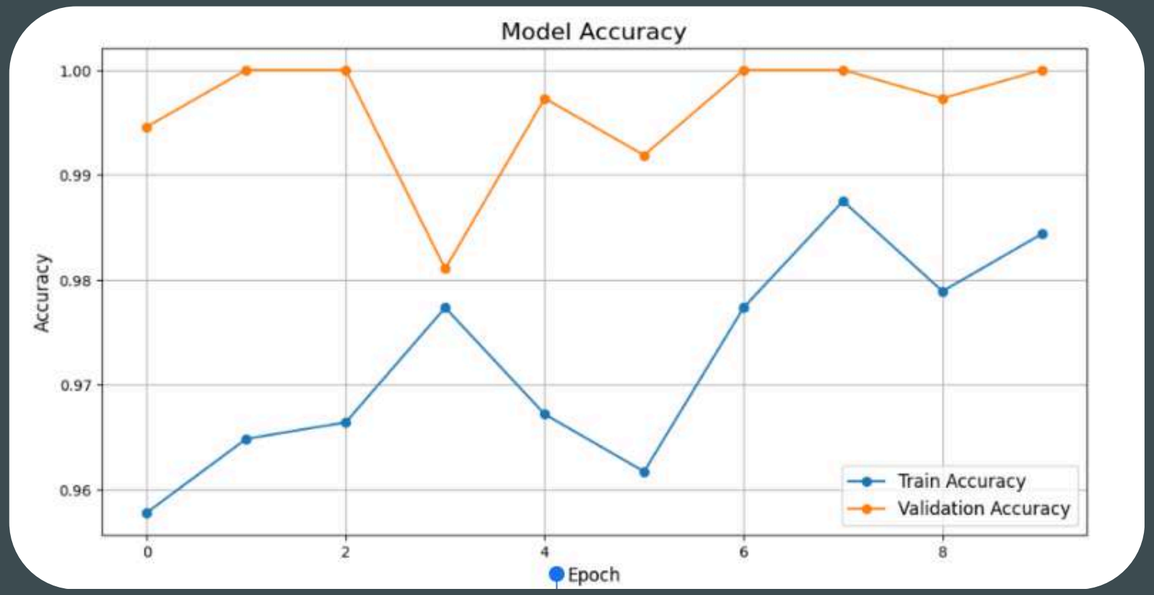
Tabel Hasil Evaluasi Model

Metric	Value
Test Loss	0.2445
Test Accuracy	0.9962



Hasil Optimisasi





Kesimpulan

Penelitian ini berhasil mengembangkan dataset senam otak (Brain Gym) mencakup enam gerakan tangan spesifik yang dianalisis menggunakan model CNN dan LSTM. Model menunjukkan performa tinggi dengan precision, recall, dan F1-score sempurna, serta validation accuracy 99.62% dan validation loss 0.2445, mengindikasikan kemampuan mengenali pola dan generalisasi data baru.

Namun, potensi overfitting terlihat dari validation accuracy yang lebih tinggi dibandingkan train accuracy. Pengujian dengan dataset yang lebih besar, teknik augmentasi data, atau transfer learning diperlukan untuk meningkatkan kemampuan model dalam menangani variasi gerakan.

Penelitian ini menyediakan dataset terstruktur dan menunjukkan potensi pembelajaran mesin untuk aplikasi pendidikan, terapi fisik, dan kesehatan mental, meski pengembangan lebih lanjut diperlukan untuk memastikan penerapan praktis di dunia nyata.

Referensi

Smart trainer: Combining video analysis and deep learning for efficient and accurate gym exercise classification and form correction. (2024). *Yogesh Kumar; Pratik Saria; Vikas Bhandari; Dinesh Kumar Vishwakarma*.

Personalized Daily Hand Movement Training Methods and Effects: A Case Study. ` (n.d.). MDPI. https://www.mdpi.com/2076-3417/14/12/5297

View of The Improvement of Cognitive Function and Decrease the Level of Stress in the Elderly with Brain Gym. (n.d.). INTERNATIONAL JOURNAL OF NURSING AND MIDWIFERY SCIENCE (IJNMS). https://www.jknusantara.com/index.php/ijnms/article/view/33/4

Brain gym exercises: an approach in improving the psychological perception in graduate students. (2021). JMPAS. https://jmpas.com/admin/assets/article_issue/1637781336JMPAS_SEPTEMBER-OCTOBER_2021.pdf

Thank You

Kelompok 7 - Brain Gym