

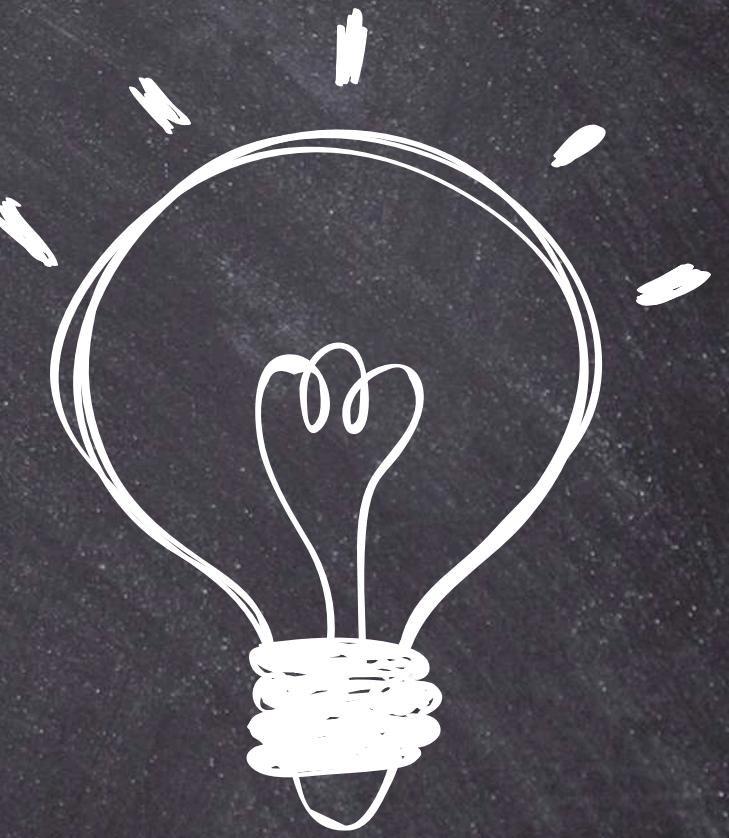
**OPTIMALISASI DETEKSI PENYAKIT DAUN
JAGUNG MENGGUNAKAN
PREPROCESSING DAN TEKNIK
AUGMENTASI PADA CNN**



INTRODUCTION

→ MUHAMMAD IQBAL FIKRI ROBANI AMIN

21081010334



PENELITIAN TERDAHULU

PENELITIAN TERDAHULU MENUNJUKKAN BAHWA CONVOLUTIONAL NEURAL NETWORK (CNN) EFEKTIF DALAM MENDETEKSI PENYAKIT DAUN JAGUNG DENGAN AKURASI TINGGI. MISALNYA, ÇETİNER (2023) MENGGUNAKAN CNN UNTUK MENDETEKSI BLIGHT, COMMON RUST, DAN GREY LEAF SPOT. MALIK ET AL. (2022) JUGA MENYOROTI PENTINGNYA CNN DALAM ANALISIS CITRA DAUN.



RESEARCH GAP

Penelitian sebelumnya menunjukkan efektivitas CNN dalam mendeteksi penyakit tanaman, namun terdapat beberapa **kekurangan**. Teknik augmentasi data sering kurang optimal, dataset yang digunakan terbatas dan kurang representatif, serta minimnya fokus pada penyakit daun jagung di Indonesia. Selain itu, banyak model yang digunakan memiliki kompleksitas tinggi sehingga sulit diimplementasikan pada perangkat terbatas, dan evaluasi kinerja sering kali tidak mencakup metrik yang lebih komprehensif. Penelitian ini bertujuan untuk **mengatasi celah** tersebut dengan mengembangkan model CNN yang lebih efisien, akurat, dan siap diterapkan dalam konteks nyata.

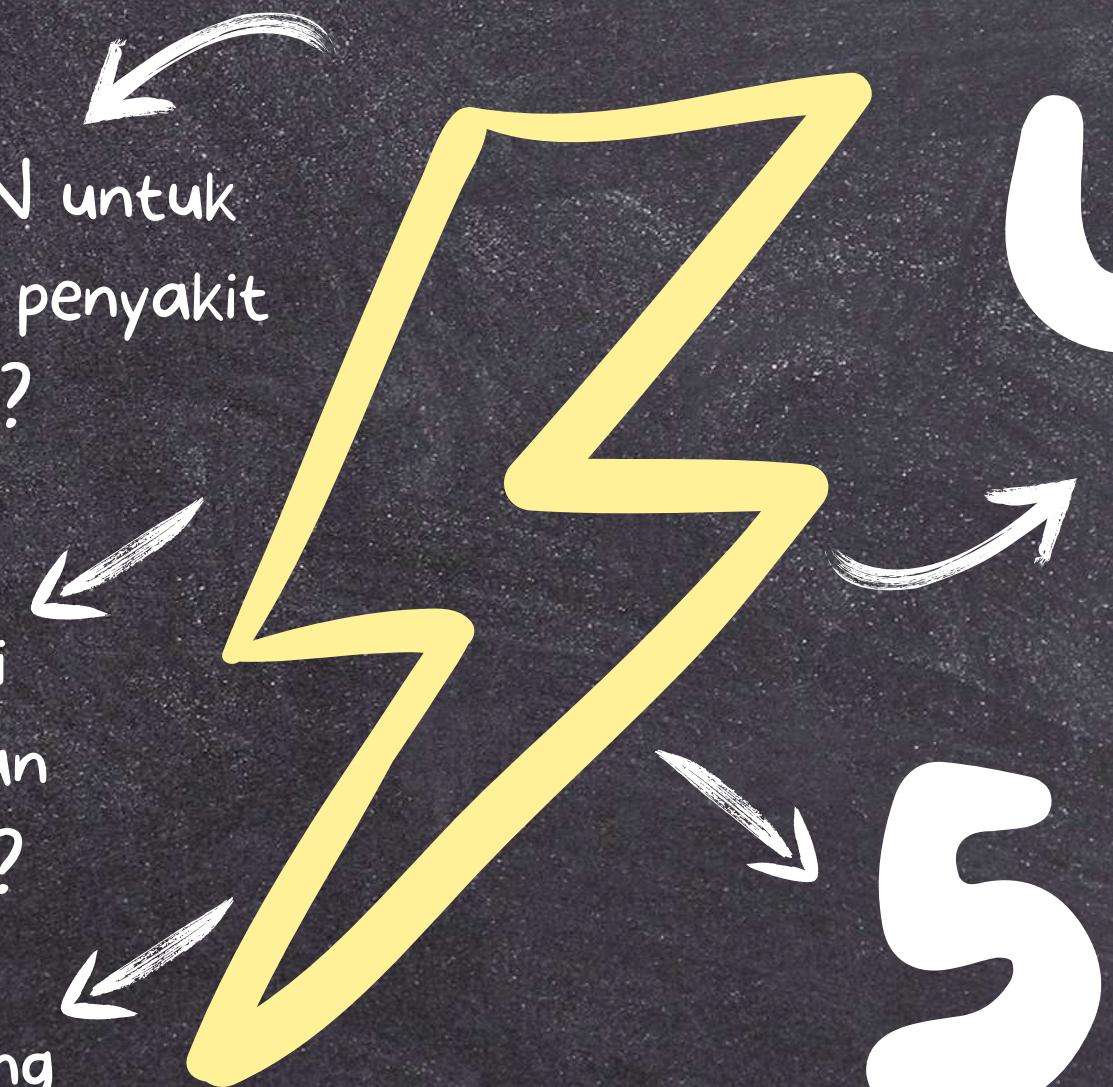
2

Bagaimana teknik augmentasi data meningkatkan keragaman dataset dan performa model?

3

Apa langkah preprocessing data yang optimal untuk akurasi model CNN?

Bagaimana merancang model CNN untuk mendeteksi dan mengklasifikasikan penyakit daun jagung dengan akurasi tinggi?



5

Apa kendala utama dalam pengembangan model dan solusinya?

Bagaimana mengevaluasi kinerja model menggunakan metrik akurasi, presisi, recall, dan F1-score?

PERUMUSAN MASALAH

MIND MAP

JAGUNG

CNN

PREPROCESSING

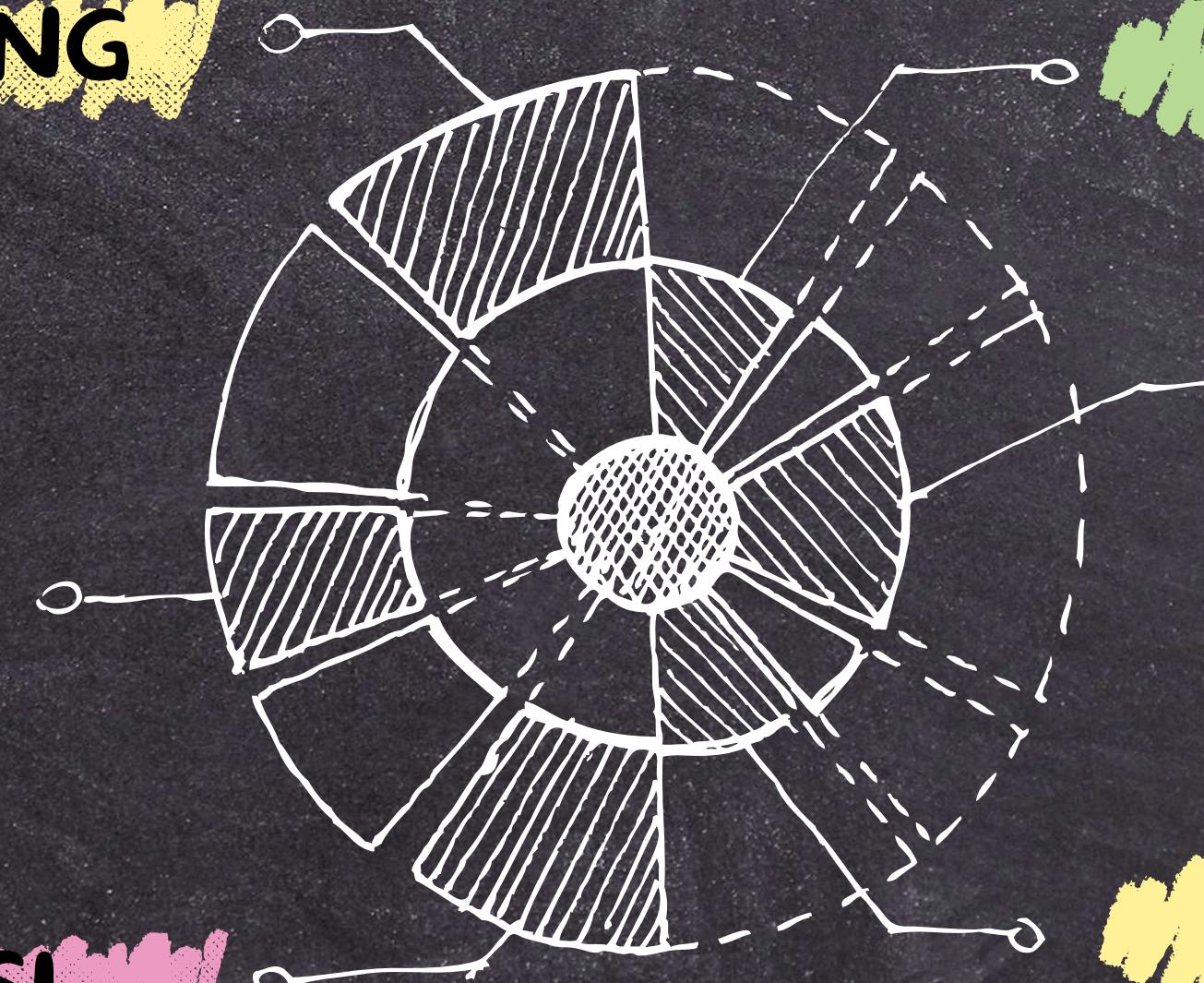
TENSORFLOW

AUGMENTASI

EVALUASI

Riset IT

Riset IT



METODE

Penelitian ini menggunakan pendekatan deep learning dengan model CNN yang dioptimalkan melalui beberapa tahapan penting. Proses preprocessing data meliputi langkah resizing untuk menyamakan dimensi gambar dan rescaling untuk normalisasi nilai piksel ke rentang $[0, 1]$, sehingga kualitas data input dapat ditingkatkan. Selain itu, teknik augmentasi data diterapkan untuk meningkatkan keragaman dataset dengan menggunakan metode seperti Random Flip (membalik gambar secara acak) dan Random Rotation (memutar gambar secara acak). Kombinasi preprocessing dan augmentasi ini bertujuan untuk meningkatkan performa model dalam mengenali pola pada citra daun jagung.

METRIK PENGUJIAN

- akurasi
- presisi
- recall
- F1 -

Score

METRICS

AKURASI

$$\text{Akurasi} = \frac{TP + TN}{TP + TN + FP + FN}$$

PRESISI

$$\text{Presisi} = \frac{TP}{TP + FP}$$

RECALL

$$\text{Recall} = \frac{TP}{TP + FN}$$

F1 - SCORE

$$\text{F1-Score} = 2 \times \frac{\text{Presisi} \times \text{Recall}}{\text{Presisi} + \text{Recall}}$$



PROGRESS

```
# Model definition
model = tf.keras.Sequential([
    # Input layer
    tf.keras.layers.Input(shape=(IMAGE_SIZE, IMAGE_SIZE, 3), name='input'),

    # Preprocessing layers
    Resizing(IMAGE_SIZE, IMAGE_SIZE, name='resizing'),
    Rescaling(scale=1./255, name='rescaling'),

    # Augmentation layers
    RandomFlip(mode="horizontal_and_vertical", name='random_flip'),
    RandomRotation(factor=0.2, name='random_rotation'),

    # Convolutional layers
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu', name='conv2d_1'),
    tf.keras.layers.MaxPooling2D((2, 2), name='max_pooling_2d_1'),

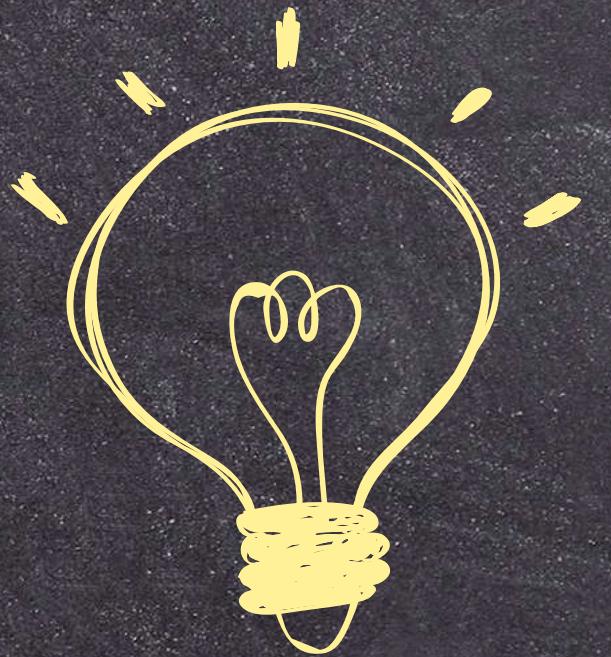
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu', name='conv2d_2'),
    tf.keras.layers.MaxPooling2D((2, 2), name='max_pooling_2d_2'),

    # Fully connected layers
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(64, activation='relu', name='dense_1'),
    tf.keras.layers.Dropout(0.5, name='dropout'),
    tf.keras.layers.Dense(NUM_CLASSES, activation='softmax', name='output')
], name='agrocare')

# Compile model
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=LEARNING_RATE),
    loss=tf.keras.losses.CategoricalCrossentropy(),
    metrics=['accuracy']
)
```

PROSES TRAINING DATA

```
Epoch 25/32
738/738 [=====] - 38s 51ms/step - loss: 0.2241 - accuracy: 0.9141
- val_loss: 0.2282 - val_accuracy: 0.9168
Epoch 26/32
738/738 [=====] - 38s 51ms/step - loss: 0.2227 - accuracy: 0.9159
- val_loss: 0.3110 - val_accuracy: 0.8901
Epoch 27/32
738/738 [=====] - 38s 51ms/step - loss: 0.2183 - accuracy: 0.9159
- val_loss: 0.3307 - val_accuracy: 0.8898
Epoch 28/32
738/738 [=====] - 38s 51ms/step - loss: 0.2129 - accuracy: 0.9193
- val_loss: 0.3492 - val_accuracy: 0.8856
Epoch 29/32
738/738 [=====] - 38s 51ms/step - loss: 0.2133 - accuracy: 0.9184
- val_loss: 0.3836 - val_accuracy: 0.8671
Epoch 30/32
738/738 [=====] - 37s 51ms/step - loss: 0.2136 - accuracy: 0.9173
- val_loss: 0.2611 - val_accuracy: 0.9086
Epoch 31/32
738/738 [=====] - 37s 51ms/step - loss: 0.2120 - accuracy: 0.9197
- val_loss: 0.3209 - val_accuracy: 0.8789
Epoch 32/32
738/738 [=====] - 38s 51ms/step - loss: 0.2061 - accuracy: 0.9229
- val_loss: 0.3974 - val_accuracy: 0.8603
```



TERIMAKASIH