

DETEKSI PENYAKIT PADA TANAMAN PADI MENGUNAKAN EXTRAME GRADIENT BOOST (XGBoost) DAN VISION TRANSFORMER (ViT) BERDASARKAN CITRA DAUN.

Oleh :

Nama : Safiqur Rohman


NPM : 21081010109



Latar Belakang



Padi memiliki peran penting dalam ketahanan pangan di Indonesia karena merupakan bahan makanan pokok. Namun, serangan penyakit pada padi seperti hawar daun dan bercak daun seringkali membuat hasil panen tidak maksimal. Banyak petani kesulitan dalam mengidentifikasi penyakit dengan cepat, sehingga pengendalian yang dilakukan seringkali terlambat. Untuk mengatasi masalah ini, teknologi pengolahan citra digital yang dipadukan dengan kecerdasan buatan (AI) dapat menjadi solusi yang efektif. Dengan menggunakan metode seperti Xxtreme Gradient Boosting (XGBoost) dan Vision Transformer (ViT), teknologi ini mampu mendeteksi penyakit dengan cepat dan akurat. Diharapkan, sistem ini dapat membantu petani mengendalikan penyakit secara lebih efisien, mengurangi ketergantungan pada pestisida, serta mendukung keberlanjutan sektor pertanian.



Penelitian Terdahulu

Penelitian yang dilakukan oleh Amanda Caecillia Milano, Achmad Yusid, dan Rima Tri Wahyuningrum pada (2023) dengan judul “KLASIFIKASI PENYAKIT DAUN PADI MENGGUNAKAN MODEL DEEP LEARNING EFFICIENTNET-B6 ” mengusulkan sistem klasifikasi pada tanaman Padi menggunakan Convolutional Neural Network (CNN) berbasis arsitektur EfficientNet-B6. Berdasarkan pengujian, sistem ini berhasil mencapai akurasi sebesar 77.05%. Hal ini menunjukkan potensi sistem tersebut dalam membantu petani mendeteksi penyakit pada tanaman padi sehingga dapat meningkatkan kualitas produksi secara signifikan.



Research Gap



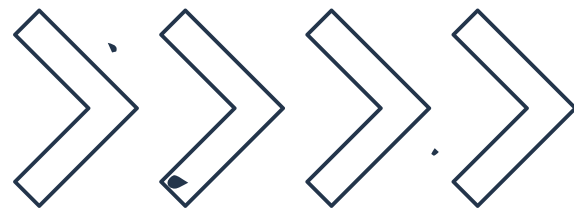
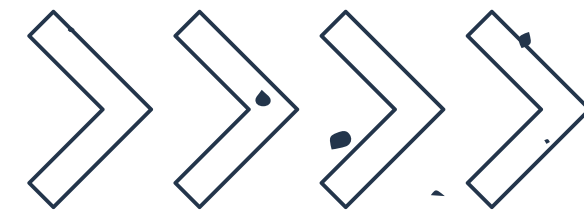
Penelitian sebelumnya menggunakan arsitektur CNN, seperti EfficientNet-B6, namun terbatas pada pendekatan berbasis CNN tanpa eksplorasi metode ensemble seperti XGBoost. Studi ini mengisi celah dengan menerapkan XGBoost untuk meningkatkan akurasi dan menangani data tidak seimbang secara lebih efektif.

Penelitian sebelumnya menggunakan CNN untuk ekstraksi fitur citra, tetapi kurang mampu menangkap hubungan global antar piksel. Penelitian ini mengatasi celah tersebut dengan Vision Transformer (ViT), yang lebih unggul dalam memahami konteks global gambar secara mendalam.

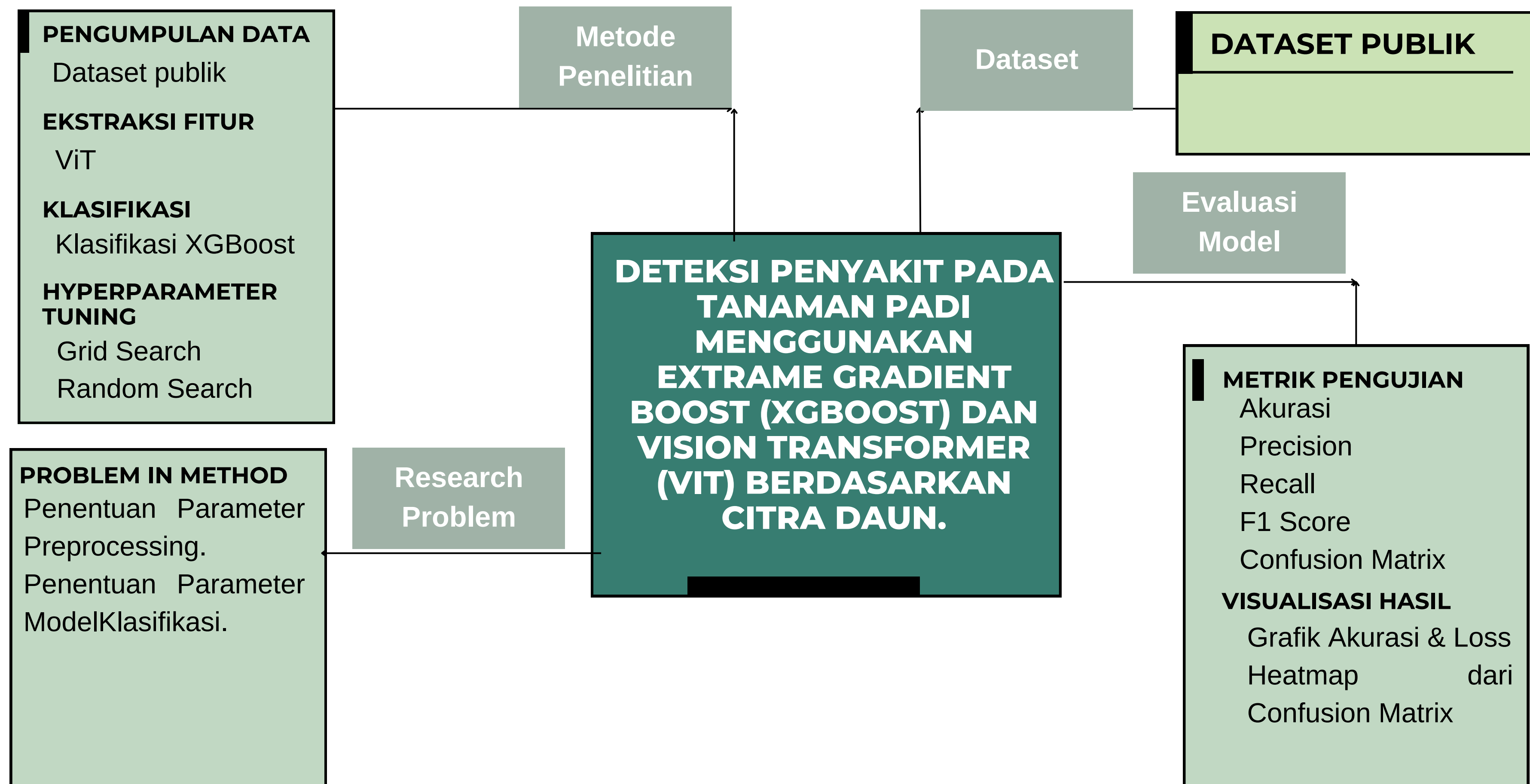


Rumusan Masalah

Bagaimana implementasi metode XGBoost dapat meningkatkan akurasi dalam deteksi dan klasifikasi penyakit pada tanaman padi



Bagaimana penerapan ekstraksi fitur menggunakan Vision Transformer (ViT) dapat meningkatkan pemahaman hubungan global antar piksel dalam citra daun padi



»»»» Vision Transformer (ViT) ✨


Vision Transformer (ViT) adalah model deep learning yang menggunakan arsitektur transformer untuk pengolahan citra. Berbeda dengan CNN, ViT memecah gambar menjadi potongan kecil (patch) dan memprosesnya secara paralel menggunakan mekanisme perhatian (self-attention), memungkinkan model menangkap hubungan global antar elemen gambar. ViT terbukti efektif, terutama pada dataset besar, dan sering mengalahkan CNN dalam pengenalan gambar.





XGBoost

Extreme Gradient Boosting (XGBoost) adalah algoritma machine learning berbasis pohon keputusan yang menggunakan teknik boosting untuk meningkatkan akurasi. Setiap pohon baru memperbaiki kesalahan pohon sebelumnya, dengan fitur regularisasi untuk menghindari overfitting. XGBoost dikenal cepat, efisien, dan sering digunakan dalam kompetisi data science karena kemampuannya menghasilkan model yang akurat.



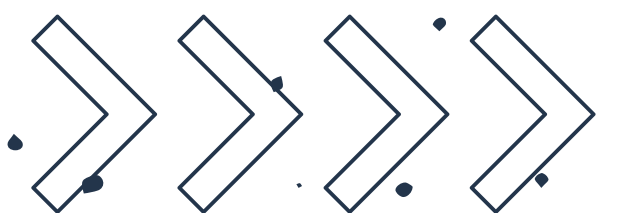
Progres

```
▶ # Cek apakah folder 'train' ada di path yang diinginkan
train_folder = '/content/rice_leaf_dataset/Rice_Leaf_Disease/Rice_Leaf_Disease/train'
test_folder = '/content/rice_leaf_dataset/Rice_Leaf_Disease/Rice_Leaf_Disease/test'

print(f"Train folder exists: {os.path.exists(train_folder)}")
print(f"Test folder exists: {os.path.exists(test_folder)}")

# Jika folder tidak ada, tampilkan daftar folder yang ada
print('Folder yang ada:')
print(os.listdir('/content/rice_leaf_dataset'))
```

```
➞ Train folder exists: True
Test folder exists: True
Folder yang ada:
['Rice_Leaf_AUG', 'Rice_Leaf_Disease']
```



Progres

```
def load_and_resize_images_from_folders(train_folder, test_folder, target_size=(224, 224)):
    images = []
    labels = []

    # Fungsi untuk memuat gambar dan label dari satu folder (train atau test)
    def load_images_from_folder(folder):
        for class_label in os.listdir(folder):
            class_path = os.path.join(folder, class_label)
            if os.path.isdir(class_path):
                for filename in os.listdir(class_path):
                    img_path = os.path.join(class_path, filename)
                    img = cv2.imread(img_path)
                    if img is not None:
                        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
                        img = cv2.resize(img, target_size)
                        images.append(img)
                        labels.append(class_label)

    # Memuat gambar test dan train
    load_images_from_folder(train_folder)
    load_images_from_folder(test_folder)

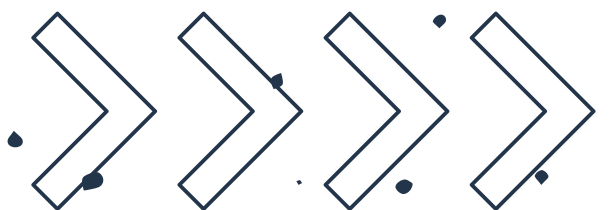
    return np.array(images), np.array(labels)

train_folder = '/content/rice_leaf_dataset/Rice_Leaf_Disease/Rice_Leaf_Disease/train'
test_folder = '/content/rice_leaf_dataset/Rice_Leaf_Disease/Rice_Leaf_Disease/test'

images, labels = load_and_resize_images_from_folders(train_folder, test_folder)

#tampilkan hasilnya
print(f'Jumlah gambar: {len(images)}')
print(f'Jumlah label: {len(labels)}')
```

```
➦ Jumlah gambar: 18445
  Jumlah label: 18445
```



Progres

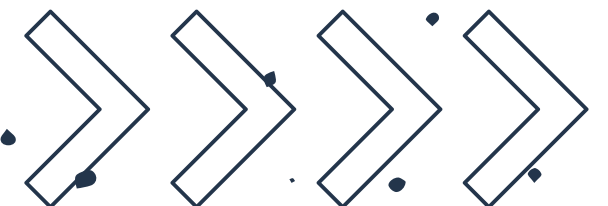
```
▶ from sklearn.model_selection import train_test_split

# Tentukan path folder untuk data train dan test
TRAIN_PATH = '/content/rice_leaf_dataset/Rice_Leaf_Disease/Rice_Leaf_Disease/train'
TEST_PATH = '/content/rice_leaf_dataset/Rice_Leaf_Disease/Rice_Leaf_Disease/test'

# Memuat dan mengubah ukuran gambar untuk data train dan test
X_train, y_train = load_and_resize_images_from_folders(TRAIN_PATH, TEST_PATH)
X_test, y_test = load_and_resize_images_from_folder(TEST_PATH)

# Membagi data pelatihan menjadi data pelatihan dan validasi (80% train, 20% val)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.3, random_state=42)

# Lihat ukuran data setelah pemisahan
print(f"Train data: {X_train.shape}, Validation data: {X_val.shape}, Test data: {X_test.shape}")
```



Progres

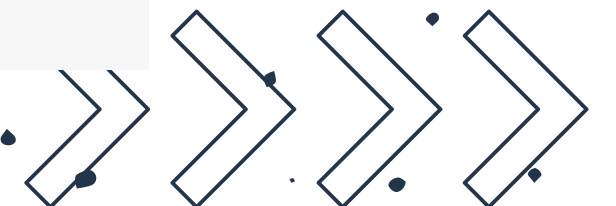
```
▶ # Membuat daftar kelas yang unik
unique_classes = list(set(y_train))
random_classes = random.sample(unique_classes, 10)

# Menyusun gambar berdasarkan kelas acak yang dipilih
selected_images = []
selected_labels = []

for class_label in random_classes:
    # Menemukan semua gambar dengan label kelas tertentu
    class_indices = [i for i, label in enumerate(y_train) if label == class_label]
    # Memilih gambar acak dari kelas tersebut
    random_index = random.choice(class_indices)
    selected_images.append(X_train[random_index])
    selected_labels.append(y_train[random_index])

# Plot gambar-gambar terpilih
plt.figure(figsize=(15, 8))
for i, (img, label) in enumerate(zip(selected_images, selected_labels), 1):
    plt.subplot(2, 5, i) # Layout 2 baris dan 5 kolom
    plt.imshow(img)
    plt.title(f'class: {label}')
    plt.axis('off')

plt.show()
```





Terima Kasih

