Analisa Model Convolutional Neural Networks Lanjutan Terhadap Model Klasifikasi Pakaian

Theovanno Tjahyamulia[#1], Hendra Bunyamin[*2]

[#]*Teknik Informatika , Universitas Kristen Maranatha*
Jl. Surya Sumantri No.65, Sukawarna, Kec. Sukajadi, Kota Bandung, Indonesia
[1]1972048@maranatha.ac.id

[*] *Teknik Informatika , Universitas Kristen Maranatha*
Jl. Surya Sumantri No.65, Sukawarna, Kec. Sukajadi, Kota Bandung, Indonesia
[2]hendra.bunyamin@maranatha.ac.id

Abstract — **Penelitian ini dilakukan dengan tujuan untuk mempelajari bagaimana cara untuk melakukan pelatihan dan pemilihan model yang baik. Pada penelitian ini dilakunan pelatihan terhadap model- model Convolutional Neural Networks Lanjutan, dengan tujuan untuk melakukan perbandingan kinerja model dan menentukan model mana yang tepat untuk digunakan untuk dalam kasus klasifikasi pakaian. Penelitian ini akan dilakukan terhadap beberapa model Convolutional Neural Networks Lanjutan yang superpower atau memiliki tingkat akurasi yang tinggi diantaranya adalah DenseNet, EfficientNetB6, EfficientNetB7, dan Xception. Dalam penelitian ini akan dilakukan dengan menggunakan dataset yang sama sebagai patokan untuk menilai kinerja tiap model. Berikut adalah hal-hal yang akan dilakukan dalam penelitian ini : Preprocessing dataset, Hyperparameter Tuning, Training Final Model, dan Penilaian F1_scores terhadap model yang akan diteliti. Tahapan Preprocesing dataset dilakukan untuk menghindari bias data, Hyperparameter Tuning guna memaksimalkan kinerja model, dan penilaian F1_scores terhadap final model dari setiap modelnya untuk melakukan perbandingan terhadap kinerja maksimum dari tiap model. Penilaian model yang akan dilakukan akan meliputi analisis terhadap model, durasi dan penilaian F1_Scores.**

Keywords— **Dataset, F1_Scores, Hyperparameter Tuning, Kinerja Model, Model .**

I. PENDAHULUAN

Di era modern ini aktivitas jual beli tidak hanya dapat dilakukan secara langsung, dengan adanya sarana media online seperti Tokopedia dan Shopee kegiatan jual beli dapat dilakukan dengan lebih cepat dan efisien. Dengan adanya media tersebut aktivitas jual beli dapat dilakukan kapanpun, dimanapun dan oleh siapapun yang memiliki akses Internet dan gawai ataupun perangkat elektronik lainnya. Salah satu kebutuhan yang sering dicari diantaranya adalah pakaian. Pakaian merupakan suatu komoditas umum dan penting dalam kehidupan manusia, pakaian juga dapat mencerminkan karakteristik seseorang berdasarkan pakaian yang dipakai seperti umur, jenis kelamin, status sosial maupun gaya hidup. Akan tetapi dalam proses pencarian produk tersebut masih terdapat berbagai kekurangan ketika produk yang ditemukan tidak sesuai dengan kategori produk yang ingin dicari.

Hal ini disebabkan salah satunya oleh kurangnya pemahaman konsep kategori pakaian oleh penjual sehingga penjual dapat saja memasukkan kategori yang salah dalam melakukan pengkategorian data barang yang hendak dijual yang berdampak kepada hasil pencarian calon pembeli, yaitu barang yang dicari dapat saja tidak tampil dalam data barang yang dicari ataupun data yang ditampilkan tidak sesuai dengan benda yang dicari oleh calon pembeli. Dalam rangka mengatasi masalah tersebut, maka dilakukan penelitian untuk melatih model machine learning yang dapat mengelompokkan pakaian berdasarkan kategori yang ada saat penjual memasukan data barang yang hendak dijual serta dapat merekomendasikan pakaian yang sejenis bagi para calon pembeli pakaian.

Berdasarkan latar belakang di atas, maka dapat dirumuskan masalah sebagai berikut. Pertama, Bagaimana cara melatih model machine learning agar model dapat melakukan pengkategorian pakaian dengan tingkat akurasi yang baik. Kedua, Apakah model machine learning yang tepat untuk digunakan dalam melakukan klasifikasi pakaian yang akurat?

Tujuan pembahasan dari penelitian ini adalah untuk mencari dan melatih model machine learning yang dapat melakukan pengklasifikasian pakain yang baik dan akurat.

II. KAJIAN TEORI

A. *Machine Learning*

Machine Learning (ML) adalah proses menggunakan model data matematika untuk membantu komputer belajar tanpa instruksi langsung [1]. Machine learning merupakan salah satu cabang dari Artificial Intelligence (AI) yang akan membuka pintu menuju potensi yang tak terbatas. Machine learning dapat belajar dan membuat prediksi tanpa panduan langsung, dapat mengungkapkan kemampuan luar biasa dari komputer untuk menyerap informasi dan menemukan pola tersembunyi dengan memanfaatkan data yang disediakan untuk membuat prediksi yang akurat, mengambil tindakan yang tepat, dan terus meningkatkan kinerja sistem dari hari ke hari.

Gambar 1 Alur Kerja Machine Learning

Traditional programming melibatkan penulisan aturan-aturan yang dinyatakan dalam bahasa pemrograman, yang bertindak berdasarkan data yang diberikan dan memberikan jawaban berdasarkan aturan yang telah diberikan sebelumnya . Hal ini berlaku hampir di mana-mana bahwa sesuatu dapat diprogram dengan kode. Dalam berbagai situasi, metode ini sangat berguna untuk mengontrol bagaimana sesuatu berperilaku menggunakan aturan dari kode yang ditulis sebelumnya.

Gambar 2 Alur Kerja Traditional Programing

Kemampuan beradaptasi pembelajaran mesin menjadikannya pilihan yang bagus dalam skenario dimana data selalu berubah, sifat permintaan atau tugas selalu berubah, atau pengkodean solusi secara efektif tidak mungkin [1].

B. *Supervised Learning*

Supervised learning, juga dikenal sebagai pembelajaran mesin yang dipandu adalah subkategori machine learning. Proses pembelajaran atau sering dikatakan training dimulai pada saat data input dimasukkan ke dalam model. Model akan menyesuaikan bobotnya hingga kesalahan prediksi model menjadi sangat kecil. Supervised learning banyak digunakan untuk membantu organisasi dalam memecahkan berbagai masalah; contohnya: mengklasifikasikan email spam ke dalam folder terpisah dari kotak masuk email.

Dengan supervised learning dapat dipelajari cara mengajarkan komputer untuk memahami dan merespons data dengan sangat cerdas. Dengan bimbingan yang tepat, komputer dapat mengenali pola, memprediksi hasil, dan mengambil keputusan berdasarkan contoh-contoh yang telah kita berikan.

Supervised learning menggunakan dataset yang digunakan model untuk belajar sehingga model dapat menghasilkan output yang diinginkan. Dataset dibagi menjadi dua, yaitu: dataset pelatihan (train data) dan dataset uji (test data). Dataset pelatihan ini mencakup input dan output yang benar, yang memungkinkan model untuk belajar dari waktu ke waktu. Akurasi prediksi algoritma machine learning diukur melalui fungsi kerugian (loss function), yang menyesuaikan model sehingga kesalahan model cukup diminimalkan [5].

Supervised learning dapat dipisahkan menjadi dua jenis masalah, yaitu klasifikasi (classification) dan regresi (regression):

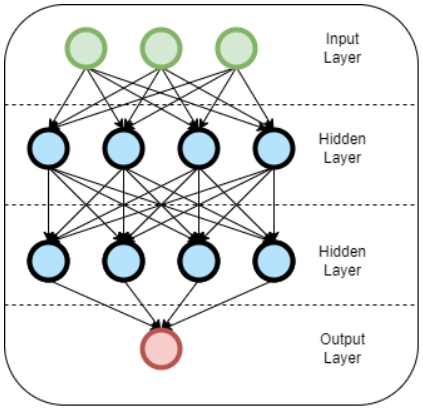
- Classification menggunakan algoritma untuk secara akurat menentukan data uji ke dalam kategori tertentu. Ini mengenali entitas tertentu dalam kumpulan data dan mencoba menarik beberapa kesimpulan berupa pola-pola tentang bagaimana entitas tersebut harus diberi label atau didefinisikan. Algoritma klasifikasi yang umum adalah linear classifiers, support vector machines (SVM), decision trees, k-nearest neighbor, and random forest.
- Regression digunakan untuk memahami hubungan antara variabel terikat dan variabel bebas. Hal ini biasanya digunakan untuk membuat proyeksi, seperti untuk pendapatan penjualan untuk bisnis tertentu. Linear regression, logistical regression, and polynomial regression adalah algoritma regresi yang populer.

Berbagai algoritma dan teknik komputasi yang digunakan dalam supervised learning. Di bawah ini adalah penjelasan singkat dari beberapa metode pembelajaran yang paling umum digunakan menyelesaikan masalah klasifikasi, biasanya

dihitung melalui penggunaan program seperti R atau Python:

1) Artificial Neural Networks (ANNs)

ANNs dapat digunakan untuk klasifikasi dan regresi. Hal ini dimanfaatkan untuk algoritma deep learning, neural networks memproses data pelatihan dengan meniru interkonektivitas otak manusia melalui lapisan node. Setiap node terdiri dari input, bobot, bias (atau ambang batas), dan output. Jika nilai output itu melebihi ambang batas yang diberikan, node akan diaktifkan dan node akan meneruskan data ke layer berikutnya di dalam network. Neural networks mempelajari fungsi pemetaan ini melalui pembelajaran yang diawasi dan menyesuaikan berdasarkan loss function melalui proses gradient descent. Ketika loss function berada pada atau mendekati nol, akurasi model diyakini menghasilkan jawaban yang benar [3].



Gambar 3 Contoh Struktur Sederhana Arsitektur ANNs

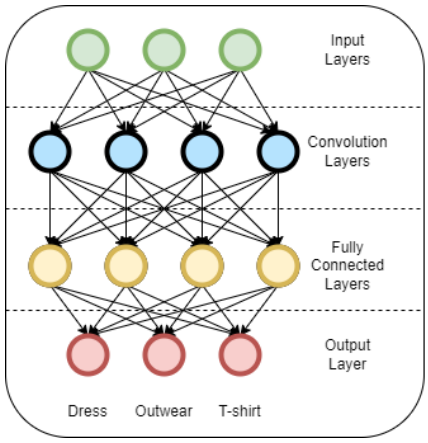
Koneksi antara neuron dalam setiap layer memiliki bobot yang ditentukan selama proses pembelajaran jaringan saraf. Proses pembelajaran melibatkan iterasi melalui serangkaian dataset pelatihan, di mana bobot diperbarui berdasarkan perbedaan antara output yang dihasilkan oleh jaringan dan output yang diharapkan.

Proses pembelajaran ini bertujuan untuk mengoptimalkan bobot agar jaringan dapat menghasilkan prediksi yang akurat dan relevan dengan masukan yang diberikan.

Jaringan saraf buatan mampu belajar secara mandiri untuk mengenali pola atau hubungan dalam data, dan kemudian dapat digunakan untuk membuat prediksi atau mengambil keputusan berdasarkan pola-pola tersebut.

2) CNN

Convolutional Neural Networks atau CNN, adalah jenis khusus dari neural networks dalam memproses data yang memiliki topologi seperti grid. Contoh grid 1D adalah data deret waktu yang mengambil sampel pada interval waktu yang teratur dan contoh grid 2D adalah data gambar. CNN telah sangat sukses diterapkan dalam berbagai aplikasi praktis. Nama "Convolutional Neural Networks" menunjukkan bahwa jaringan menggunakan operasi matematika yang disebut convolution. Convolution adalah jenis khusus dari operasi linier. CNN hanyalah neural networks yang menggunakan convolution sebagai pengganti perkalian matriks umum pada setidaknya salah satu hidden layer [3].

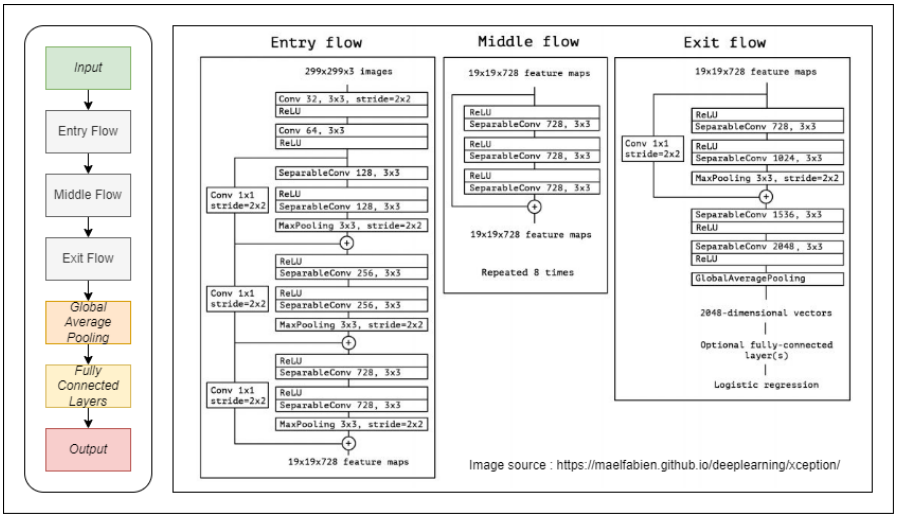


Gambar 4 Contoh Struktur Sederhana Arsitektur CNN

Selama proses pelatihan, bobot di setiap layers disesuaikan menggunakan metode backpropagation untuk mengoptimalkan prediksi jaringan. CNN terkenal karena kemampuannya dalam memproses data spasial seperti gambar dengan cara yang efektif dan efisien, dengan mempertahankan informasi spasial dan mengekstraksi fitur-fitur yang relevan. CNN telah berhasil digunakan dalam berbagai tugas Computer Vision, seperti pengenalan objek, segmentasi gambar, deteksi wajah, dan banyak lagi.

3) Xception

Xception adalah singkatan dari "extreme inception" yang berarti prinsip-prinsip Inception digunakan pada setiap hidden layer. Di Inception, convolution 1x1 digunakan untuk melakukan kompresi pada input asli, dan dari masing-masing input space tersebut berbagai jenis filter pada setiap depth space digunakan. Xception hanya membalikkan langkah ini. Pada inception, convolution 1x1 digunakan di awal sedangkan pada xception, convolution 1x1 digunakan di akhir. Ada satu lagi yang menjadi perbedaan antara Inception dan Xception, yaitu: ada atau tidaknya non-linearitas setelah operasi pertama. Dalam model Inception, kedua operasi diikuti oleh non-linearitas ReLU, namun Xception tidak menggunakan non-linearitas apa pun [2].



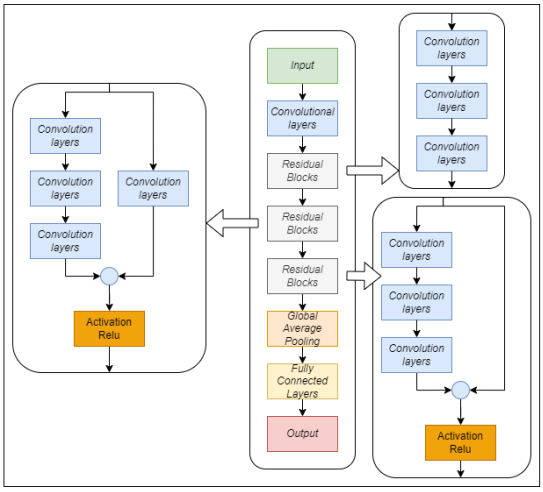
Gambar 5 Struktur Arsitektur Xception (source)

Selama proses pelatihan, bobot di setiap layer disesuaikan menggunakan metode backpropagation untuk mengoptimalkan prediksi jaringan. Xception telah terbukti efektif dalam tugas-tugas komputer visi, terutama dalam klasifikasi gambar pada dataset ImageNet.

4) ResNet

Residual Neural Network atau yang lebih dikenal dengan ResNet dikembangkan pada tahun 2015 oleh grup dari tim Microsoft Research1. Tim tersebut memperkenalkan arsitektur modul residual baru dengan skip connection. Network ini juga menampilkan batch normalization yang berat untuk setiap hidden layer. Teknik ini memungkinkan untuk melatih neural network yang sangat dalam dengan 50, 101, dan 152 weight layers sambil tetap memiliki kompleksitas yang lebih rendah daripada jaringan yang lebih kecil seperti VGGNet (19 lapisan). ResNet mampu mencapai tingkat kesalahan 5 teratas sebesar 3,57% dalam kompetisi ILSVRC 2015, yang mengalahkan kinerja semua ConvNet sebelumnya [3].

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep Residual Learning for Image Recognition," 2015, http://arxiv.org/abs/1512.03385.



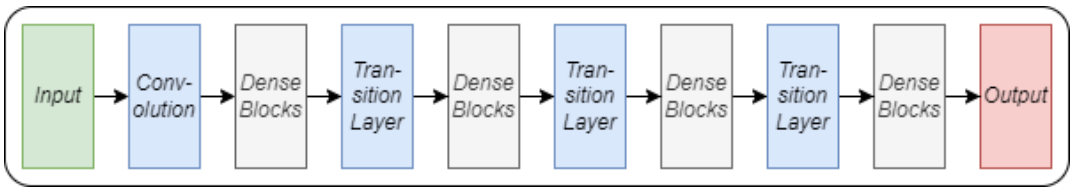
Gambar 6 Struktur Arsitektur ResNet

Selama proses pelatihan, bobot di setiap lapisan disesuaikan menggunakan metode backpropagation untuk mengoptimalkan prediksi jaringan. ResNet telah terbukti sangat efektif dalam berbagai tugas Computer Vision, termasuk klasifikasi gambar, deteksi objek, dan segmentasi.

5) DenseNet

Arsitektur DenseNet meninjau kembali konsep skip connection dengan ide yang baru. Artikel DenseNet , oleh Gao Huang et al2. menyarankan untuk memberi input convolution layer dengan semua output dari layer sebelumnya, dengan cara membuat sebanyak mungkin skip connections yang diperlukan. Selanjutnya, data digabungkan sepanjang depth axis (channels) bukannya ditambahkan, seperti pada arsitektur ResNet. Hasil eksperimen menyatakan bahwa penggabungan data ini berfungsi mirip dengan pertambahan yang terdapat pada arsitektur ResNet.

Dense blocks adalah blok dasar dari arsitektur DenseNet. Dalam dense block, convolutions dikelompokkan secara berpasangan, dengan masing-masing pasangan menerima convolutions sebagai inputnya adalah hasil output dari semua pasangan convolution sebelumnya. Dalam dense block, data digabungkan berdasarkan channel. Semua convolution diproses kembali dengan menggunakan batch normalization. Penggabungan berdasarkan channel hanya berfungsi jika tinggi dan lebar dimensi data adalah sama. Oleh karena itu, semua convolution dalam dense block memiliki stride 1 dan tidak mengubah dimensi dari data yang diproses. Pooling layer akan disisipkan di antara dense block ke dalam neural network sederhana dengan banyak layer dan hidden neurons untuk mengurangi resolusi peta fitur dan membuat buat informasi lokasi kurang akurat. [7]



Gambar 7 Struktur Arsitektur DenseNet201

Dense Blocks dan Transition Layers dapat berulang beberapa kali dalam arsitektur DenseNet-201, memberikan representasi fitur yang semakin dalam dan kompleks. Selama proses pelatihan, bobot di setiap lapisan disesuaikan menggunakan metode backpropagation untuk mengoptimalkan prediksi jaringan.DenseNet-201 telah terbukti sangat efektif dalam tugas-tugas Computer Vision, terutama ketika digunakan pada dataset yang besar dan kompleks.

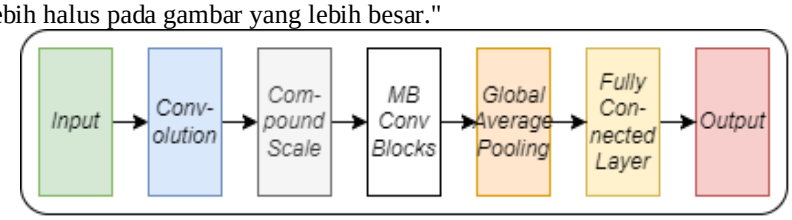
6) EfficientNet

Arsitektur ini dikembangkan dengan menggunakan ruang pencarian yang sama persis dengan algoritma pencarian arsitektur jaringan MnasNet, tetapi tujuan pengoptimalannya disesuaikan untuk meningkatkan akurasi prediksi daripada latensi inferensi seluler. Inverted residual bottlenecks dari MobileNetV2 merupakan blok bangunan dasar dari EfficientNet [3]. Arsitektur convolution dari EfficientNet memiliki tiga cara utama penskalaan, yaitu: Menggunakan lebih banyak layer,

[2] https://arxiv.org/abs/1608.06993

Menggunakan lebih banyak channel di setiap layer ,Menggunakan gambar input yang beresolusi lebih tinggi.

Artikel EfficientNet menunjukkan bahwa ketiga sumbu penskalaan ini tidak independen, yaitu: “Jika gambar input lebih besar, maka jaringan membutuhkan lebih banyak layer untuk meningkatkan bidang reseptif dan lebih banyak channel untuk menangkap pola yang lebih halus pada gambar yang lebih besar."



Gambar 8 Struktur Arsitektur EfficientNet

Selama proses pelatihan, bobot di setiap lapisan disesuaikan menggunakan metode backpropagation untuk mengoptimalkan prediksi jaringan. Arsitektur EfficientNet telah terbukti memberikan performa yang baik dengan jumlah parameter yang lebih kecil, membuatnya menjadi pilihan yang efisien dalam berbagai tugas Computer Vision.

III. Analisis Dan Rancangan Sistem

A. Metodologi Penelitian

Untuk membandingkan kinerja dari model-model yang akan digunakan maka akan digunakan dataset sebagai titik ukur untuk tiap modelnya. Model yang dilatih akan menggunakan parameter yang sama sehingga tidak terjadi bias pada hasil pelatihan model. Dataset yang akan digunakan adalah dataset pakaian3.

B. Dataset

Dataset merupakan sekumpulan data baik text, gambar maupun video dengan format yang sama. Misalnya dataset gambar hanya terdiri atas banyak gambar. Dataset terbagi menjadi 3 bagian yaitu train, validation, dan test. Bagian dataset yang akan digunakan untuk melatih model adalah dataset train dan validation. Sedangkan pengujian akan menggunakan dataset test.

Dalam pelatihan model diperlukan banyak data untuk meningkatkan akurasi dari prediksi model dalam menangani hal tersebut pelatihan model dapat dilakukan dengan 2 cara, yaitu:

1) Train-Validation-Test

Biasa kita jumpai 2 jenis dataset ,yaitu :dataset utuh dan dataset yang sudah dibagi menjadi 3 bagian yaitu dataset train, validation , dan test. Biasanya pemrosesan dataset train-validation-test dilakukan pada dataset yang belum diproses. Untuk mempermudah pelatihan model maka dataset akan dibagi menjadi 3 bagian, dataset dibagi berdasarkan persentase data sesuai kebutuhan. Biasanya dataset train akan lebih besar daripada dataset validation dan test.

2) K-fold CV – Test

K-fold cross-validation adalah penafsiran kinerja model yang jauh lebih baik, bahkan lebih baik daripada split Train-Validation-Test [4]. Cara kerja dari K-fold CV adalah sebagai berikut:

- Akan dibagi sebanyak k fold dengan ukuran yang sama dari dataset dengan k bernilai 3, 5, atau 10.
- Untuk setiap "fold" cross-validation, setiap fold akan dianggap sebagai k-1 bagian sebagai set pelatihan, dan bagian yang tersisa akan digunakan sebagai set pengujian.
- Untuk fold yang tersisa, untuk setiap bagian k-1 yang berbeda dipertimbangkan untuk menjadi dataset pelatihan dan bagian yang berbeda akan menjadi dataset pelatihan pada bagian pelatihan yang berbeda.
- Perhitungan metrik akan diperlakukan untuk setiap k-fold cross-validation.
- Skor akhir adalah hasil rata-rata skor dari setiap pelatihan.

C. Pelatihan dan Analisis Model

Model yang akan dibandingkan adalah model yang dilatih menggunakan teknik DenseNet [4], EfficientNetB6 [5],

[3] https://github.com/alexeygrigorev/clothing-dataset
[4] https://www.kaggle.com/code/theovannotjahyamulia/densenet201-base
[5] https://www.kaggle.com/code/theovannotjahyamulia/efficientnetb6-base

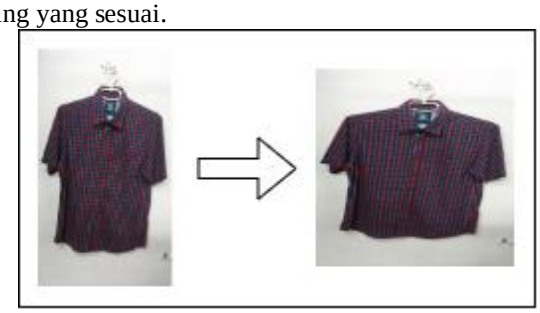
EfficientNetB7[6], dan Xception[7]. Berikut link kode program yang dapat membantu visualisasi dalam proses analisis model dapat dilihat pada footnote.

1) Preprocessing Dataset

Preprocessing melibatkan proses pengubahan data mentah menjadi format yang lebih cocok untuk proses analisis atau dimasukkan ke dalam model. Langkah-langkah preprocessing dapat bervariasi tergantung pada tugas spesifik dan karakteristik data, tetapi berikut adalah beberapa teknik preprocessing yang umum:

- Pembersihan Data: Langkah ini melibatkan penanganan nilai yang hilang, outlier, dan ketidakkonsistenan dalam data. Nilai yang hilang dapat diperhitungkan atau dihapus, outlier dapat dideteksi dan diperlakukan atau dihapus, dan ketidakkonsistenan dapat diselesaikan.
- Integrasi Data: Jika dimiliki beberapa kumpulan data dari sumber yang berbeda, integrasi data melibatkan penggabungannya menjadi satu kumpulan data. Proses ini termasuk menyelesaikan ketidakkonsistenan dalam nama variabel, format data, dan menggabungkan kumpulan data berdasarkan atribut umum.
- Transformasi Data: Terkadang, data mungkin perlu diubah untuk memenuhi asumsi model statistik tertentu atau meningkatkan kinerja algoritme pembelajaran mesin. Transformasi umum meliputi normalisasi (penskalaan fitur numerik ke rentang umum), transformasi log, dan transformasi daya.
- Pemilihan Fitur: Dalam beberapa kasus, Anda mungkin memiliki banyak fitur, tetapi tidak semuanya berkontribusi secara signifikan terhadap analisis atau performa model. Teknik pemilihan fitur membantu mengidentifikasi fitur yang paling relevan untuk tugas yang sedang dikerjakan, mengurangi dimensi dan meningkatkan efisiensi.
- Pengodean Fitur: Variabel kategori biasanya dikodekan ke dalam representasi numerik sebelum digunakan dalam algoritme pembelajaran mesin. Teknik umum termasuk pengkodean satu-panas, pengkodean label, dan pengkodean ordinal.
- Penskalaan Fitur: Saat bekerja dengan fitur numerik, seringkali perlu untuk menskalakannya ke rentang umum untuk menghindari bias fitur atau algoritme tertentu yang sensitif terhadap skala data. Teknik penskalaan yang umum termasuk standardisasi (mengurangi rata-rata dan membaginya dengan standar deviasi) dan penskalaan min-maks (menskalakan ke rentang tertentu, biasanya 0 hingga 1).
- Pemisahan Data: Dataset biasanya dibagi menjadi set pelatihan, validasi, dan pengujian. Set pelatihan digunakan untuk melatih model, set validasi digunakan untuk menyetel hyperparameter dan mengevaluasi model yang berbeda, dan set pengujian digunakan untuk menilai performa akhir dari model yang dilatih.

Ini adalah beberapa langkah yang terlibat dalam preprocessing data. Teknik dan langkah spesifik dapat bervariasi tergantung pada data dan persyaratan analisis atau tugas model machine learning. Penting untuk memahami karakteristik data dan memilih teknik preprocessing yang sesuai.



Gambar 9 Contoh Prepocessing Ukuran Gambar

2) Pengembangan dan Pelatihan Model

Model pembelajaran mesin mengubah data input menjadi output yang berarti, sebuah proses yang "dipelajari" dari paparan contoh input dan output yang diketahui.

3) Analisis Model

Semua output diperlukan selama pelatihan, tetapi hanya output pada langkah terakhir yang berguna untuk melakukan prediksi dan evaluasi. Penilaian evaluasi model akan dibandingkan dengan evaluation metric (F1_score, Precision, dan

[6] https://www.kaggle.com/code/theovannotjahyamulia/efficientnetb7-base
[7] https://www.kaggle.com/code/theovannotjahyamulia/xception-base

Recall).

F1_score adalah metrik evaluasi yang mengukur keseimbangan antara presisi (precision) dan recall dalam suatu klasifikasi. F1_score menggabungkan kedua metrik ini menjadi satu angka yang mencerminkan kinerja keseluruhan dari model klasifikasi. F1_score didefinisikan sebagai rata-rata harmonik dari precision dan recall.

Precision adalah perbandingan jumlah True Positive (TP) dengan jumlah TP ditambah False Positive (FP). Dalam konteks klasifikasi, True Positive adalah jumlah instance yang diklasifikasikan dengan benar (positif) oleh model, sedangkan False Positive adalah jumlah instance yang sebenarnya salah (negatif) tetapi secara keliru diklasifikasikan sebagai positif oleh model.

Recall juga dikenal sebagai sensitivitas atau True Positive Rate (TPR), adalah metrik evaluasi yang mengukur sejauh mana model klasifikasi dapat mengenali dan menemukan semua instance positif yang sebenarnya. Recall didefinisikan sebagai perbandingan antara jumlah True Positive dengan jumlah True Positive dan False Negative (FN).

Nilai Precision didapatkan dengan rumus berikut:

$$Precision = \frac{TP}{TP + FP}$$

TP adalah nilai True Positive dan FP adalah nilai False Positive.

Nilai Recall didapatkan dengan rumus berikut:

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

TP adalah nilai True Positive dan FN adalah nilai False Negative.

Nilai F1_Score didapatkan dengan rumus berikut:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

D. Setting Hyperparameter

Dalam pengembangan model selalu melibatkan penyetelan konfigurasi parameter: misalnya, memilih jumlah layer, ukuran layer, nilai epoch, learning rate, nilai k pada k-fold cross-validation, batch_size, ukuran dari data yang akan dianalisis contohnya pada gambar dengan ukuran 100x100px(disebut hyperparameter model, untuk membedakannya dari parameter, yang merupakan bobot dari parameter jaringan). Penyetelan ini dilakukan dengan menggunakan umpan balik sinyal kinerja model pada data validasi. Pada intinya penyetelan ini adalah bentuk dari pembelajar mesin dari pencarian konfigurasi yang baik di beberapa ruang parameter. Konfigurasi yang baik dapat kita lihat dengan dilihat berdasarkan nilai accuracy pada model yang dilatih ataupun nilai F1_Scores pada hasil prediksi yang dilakukan oleh model. Penyetel konfigurasi model berdasarkan kinerjanya pada set validasi dapat dengan cepat mengakibatkan overfitting ke set validasi, meskipun model tidak pernah dilatih secara langsung.

IV. IMPLEMENTASI

A. Pelatihan Model

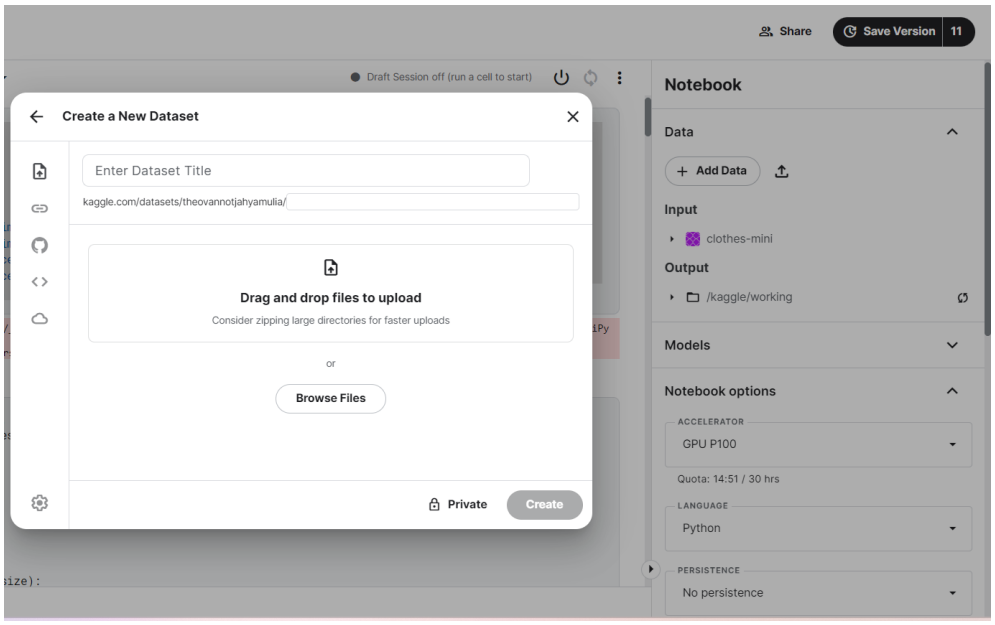
Beberapa tahapan yang akan dilakukan dalam pelatihan model adalah sebagai berikut :

- Mengunggah dataset yang akan digunakan dalam pelatihan model.
- Melakukan Import library yang akan digunakan dalam pelatihan model.
- Preprocessing dataset.
- Melakukan setting hyperparameter pada model.
- Melakukan pelatihan pada model.

Berikut adalah contoh dari tahap-tahap yang dilakukan dalam melakukan pelatihan model dengan menggunakan model Xception. Untuk kode program lebih lengkap dapat dilihat pada EfficientNetB6[8], EfficientNetB7[9], DenseNet201[10], Xception[11]

[8] https://www.kaggle.com/code/theovannotjahyamulia/efficientnetb6-hpt-best-model
[9] https://www.kaggle.com/code/theovannotjahyamulia/efficientnetb7-hpt-best-model
[10] https://www.kaggle.com/theovannotjahyamulia/densenet201-hpt
[11] https://www.kaggle.com/theovannotjahyamulia/xception-hpt

1) Mengunggah Dataset



Gambar 10 Cara Mengunggah Dataset

Pengunggahan dataset dapat dilakukan dengan menekan "Add Data", jika dataset pernah diunggah sebelumnya ataupun ingin menggunakan dataset yang tersedia pada kaggle maka atau pun dengan menekan tombol dengan logo upload untuk mengunggah dataset baru.

2) Import Library

Berikut adalah library yang digunakan dalam melatihan model CNN Xception.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications.xception import Xception
from tensorflow.keras.applications.xception import preprocess_input
from sklearn.metrics import F1_score, precision_score, recall_score, confusion_matrix
```

Kode Program 1 Import Libary

3) Preprocessing Dataset

Berikut adalah kode yang digunakan dalam melakukan preprocessing agar dataset dapat digunakan untuk melatih model.

```
def create_dataset(path, target_size, batch_size):
    gen = ImageDataGenerator(preprocessing_function=preprocess_input)
    ds = gen.flow_from_directory(
        path,
        seed=1,
        target_size=target_size,
        batch_size=batch_size,
    )
    return ds
def get_datasets(target_size, batch_size):
```

```
train_ds = create_dataset(train, target_size, batch_size)
test_ds = create_dataset(test, target_size, batch_size)
val_ds = create_dataset(val, target_size, batch_size)
return train_ds, val_ds, test_ds
```

Kode Program 2 Function Preprocessing Dataset

Pada function di atas berfungsi untuk melakukan preprocessing dataset yang terdapat pada lokasi path sehingga gambar pada dataset berukuran sebesar target_size dan memiliki sebanyak batch_size data untuk setiap batch pelatihan dataset dan untuk mengambil semua dataset yang telah di proses.

4) Hyperparameter Tuning

Pada tahap setting hyperparameter akan dicari nilai F1_score terbaik dari parameter berikut:

```
target_sizes = [(300, 300), (400, 400), (500, 500)]
batch_sizes = [32, 48, 64]
learning_rates = [0.01, 0.001, 0.0001]
```

Kode Program 3 Parameter Hyperparameter Tuning

Berikut adalah kode function dalam pembuatan model dasar dari model yang ingin dilatih.

```
def create_base_model(input_shape):
    base_model=Xception(
        weights = "imagenet",
        input_shape = input_shape,
        include_top = False
    )
    base_model.trainable = False
    return base_model
```

Kode Program 4 Function Pembuatan Base Model

Berikut adalah kode function untuk membuat model sesuai dengan parameter yang digunakan dalam melakukan hyperparameter tuning.

```
def create_model(base_model, input_shape, learning_rate):
    inputs = keras.Input(shape=input_shape)
    base = base_model(inputs, training=False)
    vector = keras.layers.GlobalAveragePooling2D()(base)
    outputs = keras.layers.Dense(10)(vector)
    model = keras.Model(inputs, outputs)
    model.compile(
        optimizer=keras.optimizers.Adam(learning_rate),
        loss=keras.losses.CategoricalCrossentropy(from_logits=True),
        metrics=["accuracy"],
    )
    return model
```

Kode Program 5 Function Pembuatan Model Yang Akan Digunakan

Berikut adalah kode program yang digunakan untuk membuat model sesuai dengan parameter yang digunakan dalam melakukan hyperparameter tuning.

```
Datasets=[]
Models=[]
Parameters=[]
for target_size in target_sizes:
    for batch_size in batch_sizes:
        for learning_rate in learning_rates:
            input_shape = target_size+(3,)
            dataset = get_datasets(target_size, batch_size)
            model = create_model(create_base_model(input_shape), input_shape,
                                 learning_rate)
            Parameters.append({'image_shape':target_size, 'batch_size':batch_size,
```

```
                                'learning_rate':learning_rate})
            Datasets.append(dataset)
            Models.append(model)
```

Kode Program 6 Kode Hyperparameter Tuning

Berikut adalah kode program yang digunakan untuk melatih model guna menghitung nilai F1_Scores.

```
Historys=[]
for i in range (0,len(Datasets),1):
  history = Models[i].fit(Datasets[i][0], epochs=2, validation_data=Datasets[i][1])
  Historys.append(history)
```

Kode Program 7 Kode Penyimpanan Data Hasil Model Yang Telah Dilatih

Berikut adalah function yang digunakan untuk menghitung nilai F1_Scores dari tiap model yang telah dilatih dan mendapatkan nilai parameter dari model dengan nilai F1_Scores terbaik.

```
def get_score(path):
  scores=[]
  for i in range(0,len(Models),1):
    gen = ImageDataGenerator(preprocessing_function=preprocess_input)
    ds = gen.flow_from_directory(
      path,
      seed=1,
      target_size=Parameters[i]['image_shape'],
      batch_size=Parameters[i]['batch_size'],
      shuffle=False,
    )
    pred = Models[i].predict(ds)
    prediction = np.argmax(pred, axis=-1)
    correct_labels = ds.labels
    score = F1_score(correct_labels, prediction, average='macro')
    scores.append({'F1_score':score})
  return scores

def get_best_model(scores):
  best = 0
  for i in range(0,len(scores),1):
    print("F1 scores:",scores[i],"with parameter",Parameters[i])
    if(scores[i]['F1_score']>scores[best]["F1_score"]):
      best = i
  print("F1 scores:",scores[best],"with best parameter",Parameters[best])
  return best
```

Kode Program 8 Function Yang Digunakan Untuk Mencari Model Denga F1 Score Terbaik

5) Pelatihan Best Model dari tiap model CNN

Berikut adalah table perbandingan dari hasil perhitungan nilai F1_Scores untuk setiap model dengan parameternya.

TABEL I
PERBANDINGAN DARI HASIL PERHITUNGAN NILAI F1_SCORES BESERTA HASIL HYPERPARAMETER TUNING

F1_Scores				Parameter		
EfficientNetB6	EfficientNetB7	DenseNet201	Xception	image_shape	batch_size	learning_rate
85.59%	84.53%	83.55%	84.61%	(300, 300)	32	0.01
86.54%	82.44%	85.71%	82.93%	(300, 300)	32	0.001
62.51%	65.79%	36.44%	47.08%	(300, 300)	32	0.0001
86.00%	84.59%	88.56%	84.30%	(300, 300)	48	0.01
83.86%	83.02%	84.73%	82.31%	(300, 300)	48	0.001
54.30%	61.90%	25.51%	33.98%	(300, 300)	48	0.0001

87.28%	87.87%	88.74%	87.29%	(300, 300)	64	0.01
85.72%	81.82%	81.79%	81.39%	(300, 300)	64	0.001
49.15%	53.02%	21.34%	32.62%	(300, 300)	64	0.0001
83.89%	84.25%	88.28%	87.74%	(400, 400)	32	0.01
88.43%	85.36%	80.35%	86.07%	(400, 400)	32	0.001
60.52%	67.19%	22.36%	39.61%	(400, 400)	32	0.0001
86.26%	87.21%	87.84%	90.19%	(400, 400)	48	0.01
87.17%	84.27%	77.67%	82.05%	(400, 400)	48	0.001
54.32%	56.06%	15.65%	33.20%	(400, 400)	48	0.0001
89.68%	85.29%	90.73%	84.93%	(400, 400)	64	0.01
86.25%	83.15%	76.26%	83.91%	(400, 400)	64	0.001
42.90%	54.96%	10.35%	27.81%	(400, 400)	64	0.0001
88.61%	87.35%	80.80%	88.88%	(500, 500)	32	0.01
84.85%	85.01%	74.51%	82.79%	(500, 500)	32	0.001
57.04%	63.40%	22.99%	36.62%	(500, 500)	32	0.0001
90.13%	87.77%	86.21%	88.17%	(500, 500)	48	0.01
84.76%	85.60%	70.70%	83.79%	(500, 500)	48	0.001
51.07%	62.02%	12.03%	27.59%	(500, 500)	48	0.0001
88.83%	88.51%	81.78%	86.23%	(500, 500)	64	0.01
85.76%	84.88%	66.17%	78.82%	(500, 500)	64	0.001
43.62%	51.30%	11.27%	28.76%	(500, 500)	64	0.0001

Pada pelatihan best model digunakan epoch sebesar 5 sementara itu proses hyperparameter tuning akan dilakukan dengan 2 epoch. Berikut adalah hasil dari best model dari model CNN yang telah dilatih.

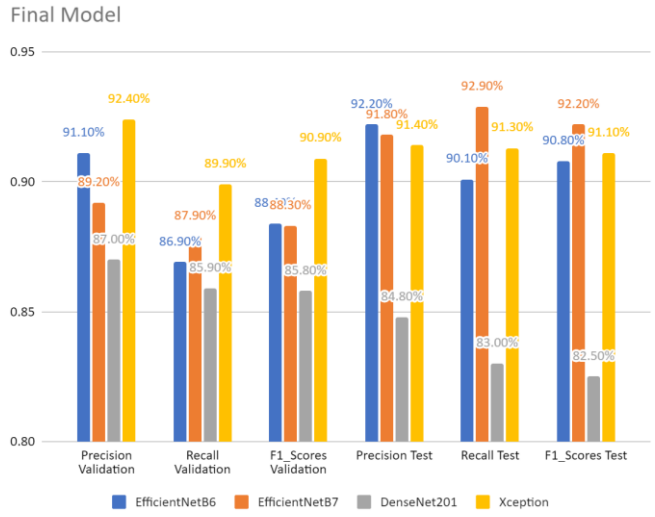


Chart 1 Hasil Pelatihan Model Dengan Best Hyperparameter Masing-Masing

e-ISSN : 2443-2229

V. PENGUJIAN

A. Pengujian Terhadap Model

1) Jumlah Parameter Tiap Model

Berikut adalah pie chart perbandingan dari jumlah parameter yang diperlukan untuk melatih model-model CNN .

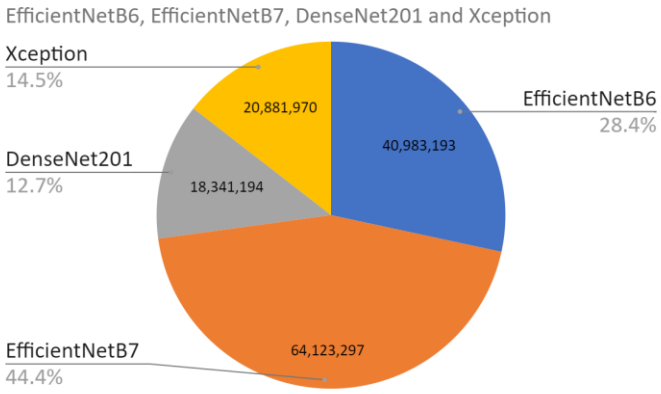


Chart 2 Perbanding Jumlah Parameter Tiap Model

2) Durasi Hyperparameter Tuning dan Pelatihan Model

Berikut adalah pie chart yang berisi perbandingan durasi hyperparameter tuning model dengan satuan nilai detik.

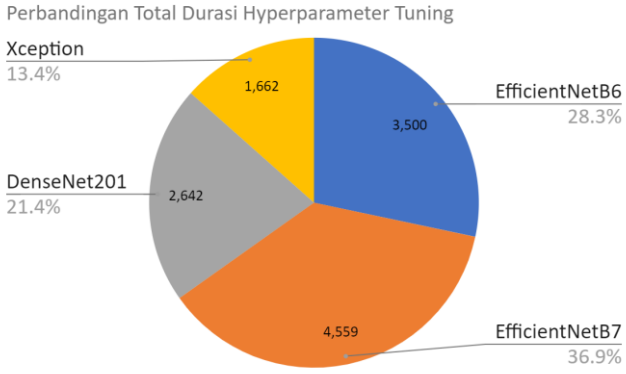


Chart 3 Perbandingan Durasi Hyperparameter Tuning Model

Berikut adalah pie chart yang berisi durasi pelatihan model dengan satuan nilai detik.

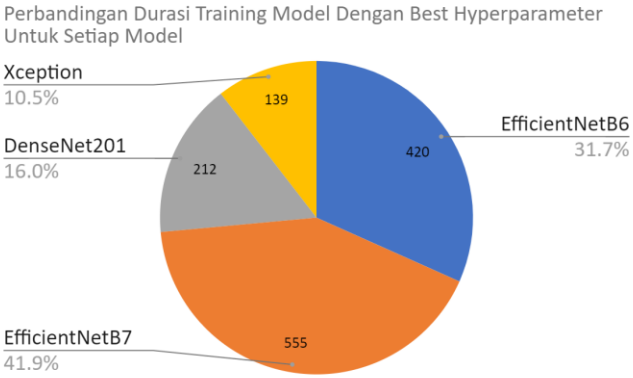


Chart 4 Perbandingan Durasi Pelatihan Final Model

A. Akurasi Prediksi

Berikut adalah chart yang berisi akurasi prediksi model.

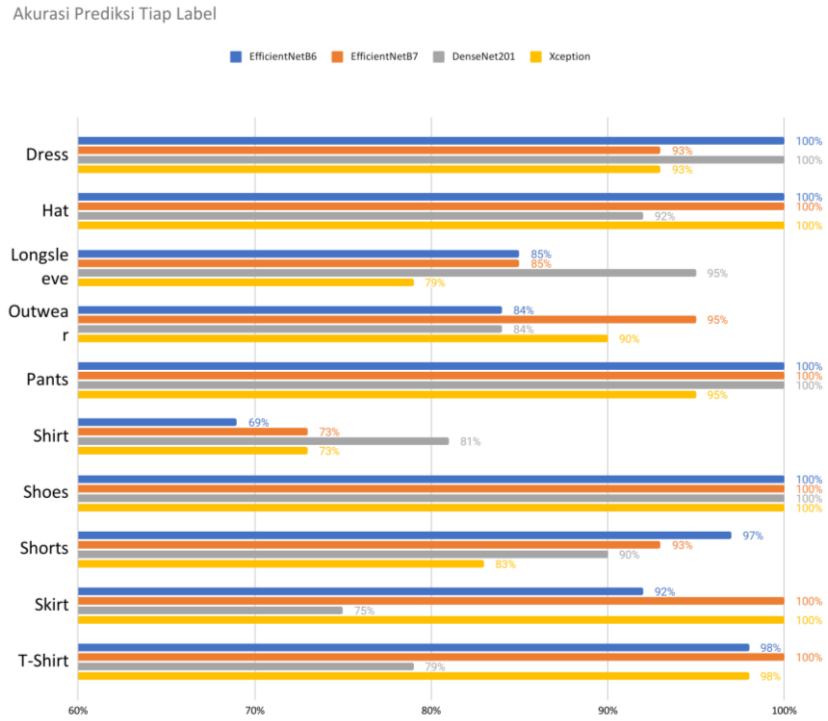


Chart 5 Perbandingan Akurasi Prediksi Dari Tiap Final Model

B. Percobaan Terhadap Gambar di luar Kategori yang Dilatih

Berikut adalah hasil prediksi gambar di luar kategori yang dilatih.

TABEL III
PERCOBAAN PREDIKSI PADA DATA YANG TIDAK DILATIH

Label	EfficientNetB6	EfficientNetB7	DenseNet201	Xception
Apple	dress	shoes	hat	hat
Apple	dress	hat	hat	hat
Car	outwear	shorts	hat	dress
Car	outwear	skirt	outwear	dress
Cat	pants	hat	shoes	hat
Cat	outwear	shorts	dress	shoes
Fountain	t-shirt	dress	shirt	hat
Fountain	outwear	dress	shirt	hat
Phone	outwear	t-shirt	shoes	shoes
Phone	outwear	t-shirt	hat	t-shirt

VI. SIMPULAN DAN SARAN

A. Simpulan

Berikut adalah table simpulan dari hasil penelitian ini berdasarkan hasil pengujian terhadap model-model yang telah dilatih.

TABEL IIIII
PERBANDINGAN KINERJA TIAP MODEL

Precision	92.20%	91.80%	84.80%	91.40%
Recall	90.10%	92.90%	83.00%	91.30%
F1_Scores	90.80%	92.20%	82.50%	91.10%
Total parameter	40,983,193	64,123,297	18,341,194	20,881,970
Total Durasi Training (per satuan second)	420	555	212	139
Total Durasi HPT (persatuan second)	3500	4559	2642	1662

Berdasarkan kesimpulan diatas, maka model CNN Xception dapat menjadi opsi utama dalam melakukan pelatihan model yang cepat dan akurat. Model tersebut memerlukan parameter yang cukup sedikit dan tidak memakan waktu dengan hasil yang hampir setara dengan model CNN EfficientNet yang memerlukan waktu yang relatif lebih lama.

B. Saran

Saran dapat secara garis besar berisi dua hal yaitu saran untuk pengembangan di masa yang akan datang atau saran untuk pelaksanaan penelitian dengan lebih baik di masa yang akan datang dapat dilakukan seperti: Melakukan Hyperparameter Tuning dengan lebih banyak parameter, dan atau melakukan perbandingan dengan model yang lebih powerfull dibandingkan model pembanding sebelumnya, ataupun model-model baru yang akan datang.

UCAPAN TERIMA KASIH

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas segala berkat dan rahmat karunia-Nya, sehingga dapat diselesaikan jurnal dengan judul "Analisa Model Convolutional Neural Networks Lanjutan Terhadap Model Klasifikasi Pakaian". Dalam pengerjaan jurnal ini penulis ingin mengucapkan terimakasih kepada Bapak Hendra Bunyamin, S.Si., M.T. sebagai dosen pembimbing atas kesediaan waktu dan pikiran sehingga jurnal ini dapat selesai dengan tepat waktu, ucapan terimakasih juga disampaikan kepada:

1. Bapak Ir. Teddy Marcus Zakaria, M.T Selaku Dekan Fakultas Teknologi Informasi Universitas Kristen Maranatha.
2. Ibu Julianti Kasih, S.E., M. Kom Selaku Ketua Prodi Teknik Informatika Fakultas Teknologi Informasi Universitas Kristen Maranatha.
3. Bapak Hendra Bunyamin, S.Si., M.T. Selaku Koordinator STA/TA.
4. Keluarga penulis yang selalu memberi dukugan dan doa kepada Penulis.
5. Teman-teman penulis yang tidak dapat disebutkan satu per satu.

Pembuatan laporan ini tidak luput dari kesalahan dan kekurangan karena keterbatasan ilmu serta kemampuan. Diharap dengan adanya kritik dan saran, laporan ini dapat disempurnakan. Laporan ini dibuat dengan harapan dapat bermanfaat dalam bidang keilmuan sebagai panduan maupun referensi untuk laporan kedepannya. Tuhan Yesus Memberkati.

DAFTAR PUSTAKA

[1] L. Moroney, AI and Machine Learning for Coders, Sebastopol: O'Reilly Media, 2020.
[2] [Online]. Available: https://iq.opengenus.org/xception-model/.
[3] M. ELGENDY, Deep Learning For Visions System, Shelter Island, NY: Manning Publications Co., 2020.
[4] S. Ozdemir, Principles of Data Science, Packt Publishing, 2016.
[5] [Online]. Available: https://azure.microsoft.com/en-in/solutions/ai/artificial-intelligence-vs-machine-learning/#introduction.
[6] "IBM," IBM, [Online]. Available: https://www.ibm.com/cloud/learn/supervised-learning.
[7] V. Lakshmanan , M. Görner dan R. Gillard, Practical Machine Learning for Computer Vision, Sebastopol: O'Reilly Media, Inc, 2021.
[8] A. Géron, Hands-On Machine Learning SECOND EDITION, Sebastopol: O'Reilly Media, Inc., 2019.
[9] A. Grigorev, Machine Learning, Shelter Island, NY: Manning Publications Co., 2021.
[10] F. CHOLLET, Deep Learning with Python Second Edition, Shelter Island, NY: Manning Publications Co, 2021