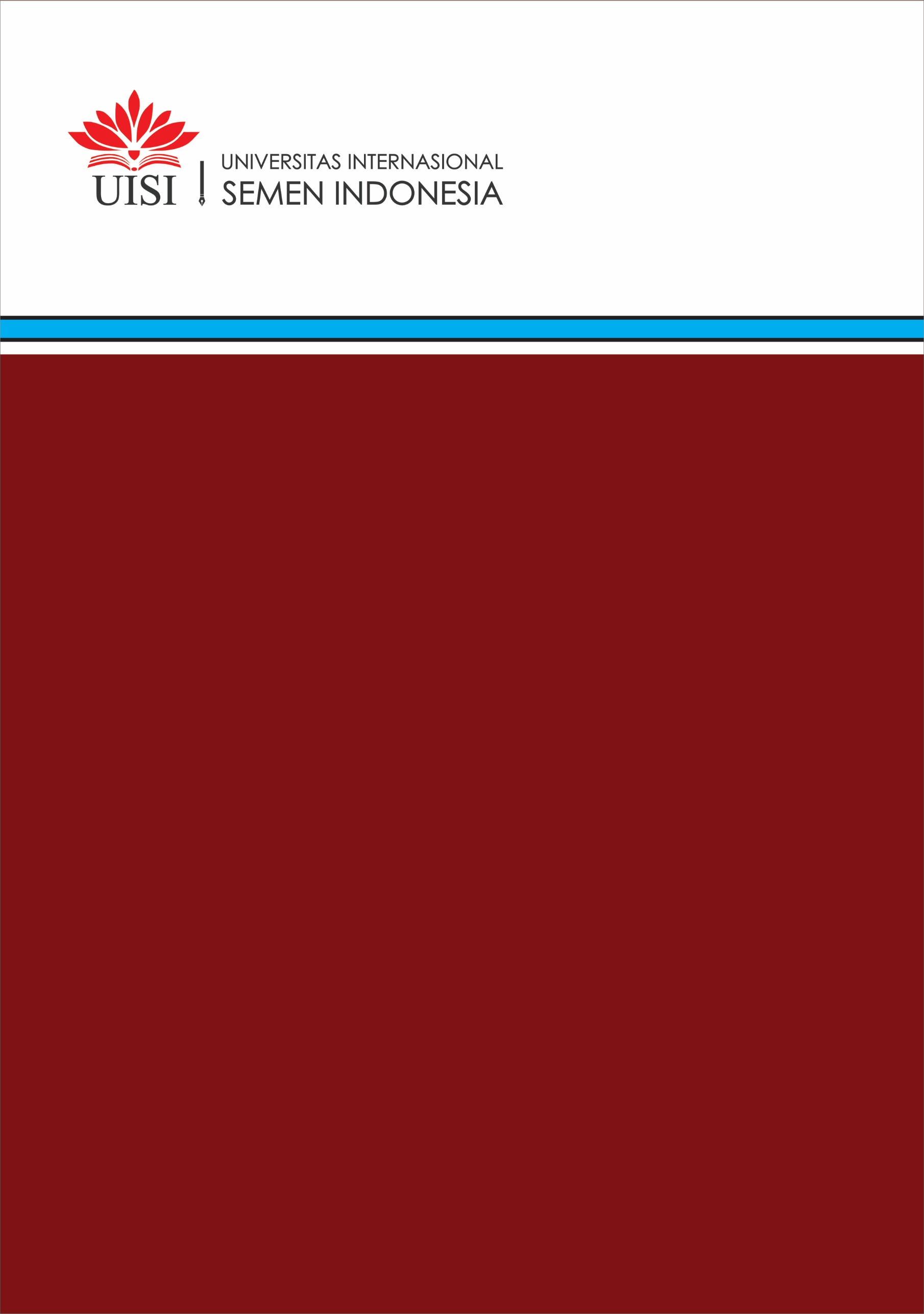
**INDO**



**KLASIFIKASI CITRA *BRAND FASHION* BERDASARKAN KONDISI BARANG ORISINAL ATAU TIRUAN MENGGUNAKAN METODE CNN**

**SKRIPSI - IF18TA46**

**Oleh:**

**MOHAMAD FAJAR ZULKARNAIN**

**NIM: 3011810029**

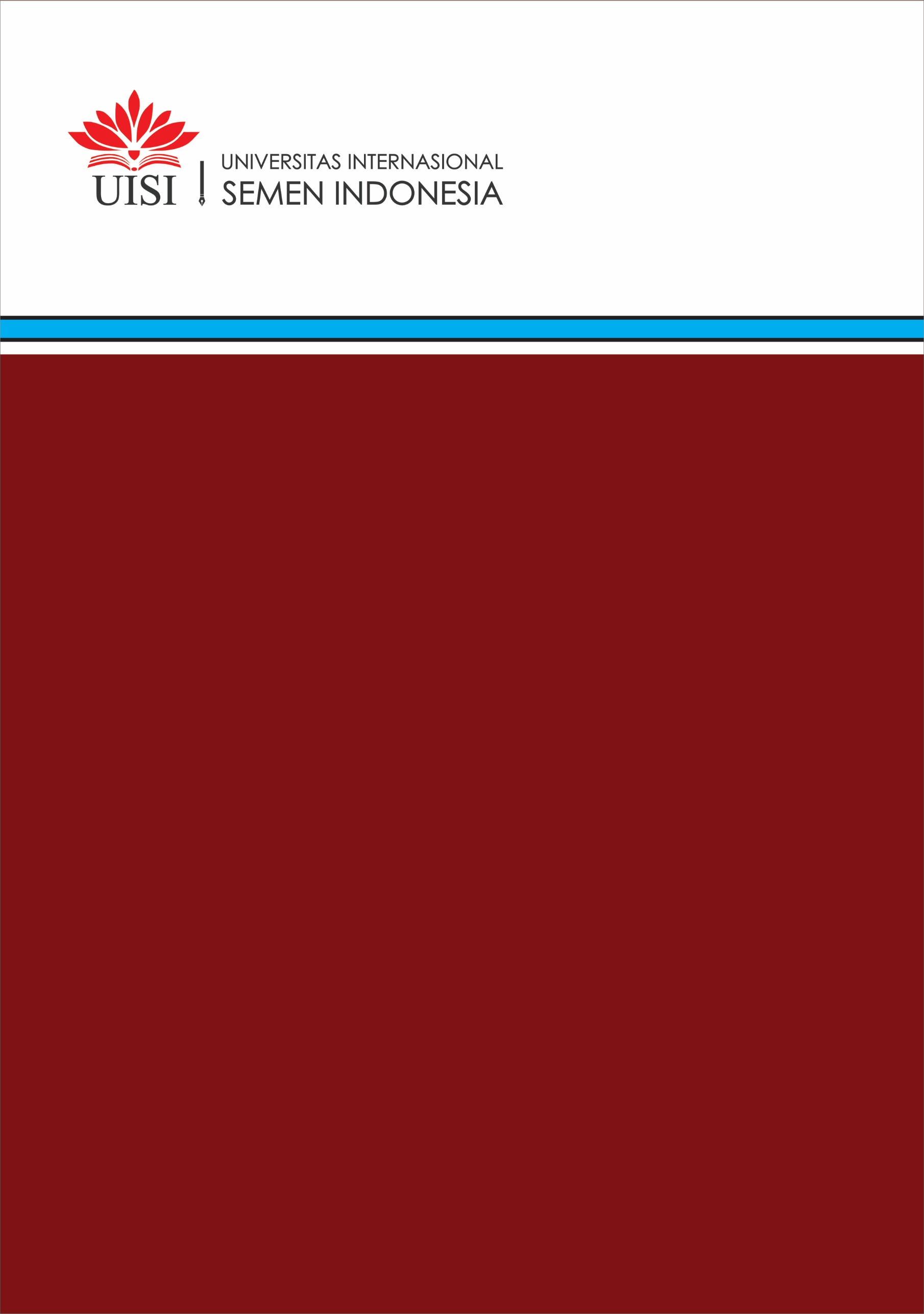
**DOSEN PEMBIMBING**

**YOHANES INDRA RISKAJAYA, S.Kom., M.Kom.**

**DEPARTEMEN INFORMATIKA**

**UNIVERSITAS INTERNASIONAL SEMEN INDONESIA**

**TAHUN 2022**



**KLASIFIKASI CITRA BRAND FASHION BERDASARKAN KONDISI BARANG ORISINAL ATAU TIRUAN MENGGUNAKAN METODE CNN**

**SKRIPSI - IF18TA46**

**Oleh:**

**MOHAMAD FAJAR ZULKARNAIN**

**NIM: 3011810029**

**DOSEN PEMBIMBING**

**YOHANES INDRA RISKAJAYA, S.Kom., M.Kom.**

**DEPARTEMEN INFORMATIKA**

**UNIVERSITAS INTERNASIONAL SEMEN INDONESIA**

**TAHUN 2022**

# HALAMAN PENGESAHAN

# HALAMAN PERNYATAAN ORISINALITAS

**Skripsi ini adalah hasil karya saya sendiri, dan semua sumber baik yang dikutip maupun dirujuk telah saya nyatakan dengan benar.**

Nama : Mohamad Fajar Zulkarnain

NIM : 3011810029

Engineering drawing

Description automatically generated with medium confidenceTanda Tangan :

Tanggal : 10 Agustus 2022

# HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI

**TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS**

Sebagai sivitas akademik Universitas Internasional Semen Indonesia, saya yang bertanda tangan di bawah ini:

Nama : Mohamad Fajar Zulkarnain

NIM : 3011810029

Departemen : Informatika

Jenis karya : Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Internasional Semen Indonesia **Hak Bebas Royalti Noneksklusif (Non-exclusive Royalty- Free Right)** atas karya ilmiah saya yang berjudul:

**“KLASIFIKASI CITRA *BRAN FASHION* BERDASARKAN KONDISI BARANG ORISINAL ATAU TIRUAN METODE CNN”**

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Internasional Semen Indonesia berhak menyimpan, mengalihmedia/format- kan, mengelola dalam bentuk pangkalan data (database), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Tuban

Engineering drawing

Description automatically generated with medium confidencePada tanggal : 10 Agustus 2022

Yang menyatakan

(Mohamad Fajar Zulkarnain)

**KLASIFIKASI CITRA BRAND FASHION BERDASARKAN KONDISI BARANG ORISINAL ATAU TIRUAN MENGGUNAKAN METODE CNN**

|  |  |
| --- | --- |
| Nama | : Mohamad Fajar Zulkarnain |
| NIM | : 3011810029 |
| Pembimbing | : Yohanes Indra Riskajaya, S.Kom.,M.Kom. |

# ABSTRAK

Perkembangan teknologi masa kini sangatlah pesat membuat masyarakat Indonesia lebih terbuka pada pengetahuan global dalam berbagai bidang salah satunya bidang *fashion*. Banyak masyarakat yang kesulitan untuk membedakan *Fashion* itu *Orisinal* atau *Tiruan*. Oleh karena itu dibuatlah sebuah program aplikasi mobile yang mampu mengklasifikasikan *Fashion* itu *Orisinal* atau *Tiruan.* Menggunakan salah satu *Brand Fashion* yaitu Nike. Produk yang digunakan hanya tiga yaitu kaos, sepatu, dan topi. Metode yang digunakan adalah *Convolutional Neural Network* atau CNN digunakan untuk data terlabel berbentuk visual atau citra yang dapat mengklasifikasi citra objek berdasarkan karakteristiknya. Proses pembuatan model klasifikasi menggunakan beberapa arsitektur diantaranya yaitu VGG16, MobileNetV2, ResNet50V2, Xception, dan DenseNet121. MobileNetV2 mendapatkan hasil *Accuracy* tertinggi yaitu *Accuracy 53%.*

**Kata Kunci:** *Convolutional Neural Network*, *Fashion*, Klasifikasi, *Orisinal* atau *Tiruan*

# CLASSIFICATION OF FASHION BRAND IMAGE BASED ON THE CONDITION OF ORIGINAL OR IMPACT USING THE CNN METHOD

|  |  |
| --- | --- |
| *Name* | : Mohamad Fajar Zulkarnain |
| *Student Identity Number* | : 3011810029 |
| *Supervisor* | : Yohanes Indra Riskajaya, S.Kom., M.Kom. |

# *ABSTRACT*

*The development of today's technology makes Indonesian people more open to global knowledge in various fields, one of which is the field of fashion. Many people have difficulty distinguishing Fashion is Original or Counterfeit. Therefore, a mobile application program was created that is able to classify Fashion as Original or Counterfeit. Using one of the Fashion Brands, namely Nike. The products used are only three, namely t-shirts, shoes, and hats. The method used is Convolutional Neural Network or CNN is used for labeled data in the form of visuals or images that can classify objects based on their characteristics. The process of making a classification model uses several architectures including VGG16, MobileNetV2, ResNet50V2, Xception, and DenseNet121. MobileNetV2 gets the highest Accuracy result, namely Accuracy of 53%.*

***Keywords:*** *Classification, Convolutional Neural Network, Fashion, Original* or *Fake*

# KATA PENGANTAR

Puji Syukur kehadirat Allah SWT atas kehendak-Nya maka penelitian dan penulisan skripsi dengan judul “Klasifikasi *Brand Fashion* Orisinal atau Tiruan Menggunakan Convolutional Neural Network” ini dapat diselesaikan dengan baik. Penulisan skripsi ini dilakukan sebagai syarat untuk mencapai gelar Sarjana Informatika Fakultas Teknologi Industri Kreatif Universitas Internasional Semen Indonesia. Penulis menyampaikan terima kasih kepada pihak yang telah membantu dan membimbing dalam penyusunan skripsi. Penulis ingin berterima kasih kepada:

1. Allah SWT yang selalu memberikan nikmat serta karunia-Nya kepada penulis sehingga penulis dapat menyelesaikan skripsi dengan baik.
2. Kedua orang tua penulis, Acmad Basuki dan Zetni Hidayati, yang selalu memberikan kasih sayang, doa, nasehat, serta kesabaran yang luar biasa dalam setiap langkah hidup penulis, yang merupakan anugrah terbesar dalam hidup. Penulis berharap dapat menjadi anak yang dapat dibanggakan.
3. Adik penulis tercinta, Sinta Amalia, terima kasih atas doa dan segala dukungannya.
4. Saudara penulis, khususnya Muhammad Abdullah Hafidh, Ekhtya dc, terima kasih atas segala dukungannya.
5. Dosen pembimbing yaitu Bapak Yohanes Indra Riskajaya, S.Kom., M.Kom., yang telah menyediakan waktu, tenaga dan pikiran dalam membimbing dan memberikan masukan kepada penulis.
6. Doni Setio Pambudi, S.Kom., M.Kom., dan Ngatini, S.Si., M.Si. selaku dosen penguji yang telah membimbing dan mendampingi serta memberikan berbagai masukan dalam penyusunan skripsi ini.
7. Seluruh dosen pengajar Departemen Informatika yang telah memberikan ilmu pengetahuan yang tak ternilai selama penulis menempuh pendidikan di Universitas Internasional Semen Indonesia.
8. Salah satu sabahat penulis, Nurus Sa’adah., S.E. terima kasih atas doa dan segala dukungannya.
9. Seluruh teman-teman penulis di Departemen Informatika, khususnya Akhmad Fuad Arif, Dicky Agung Pratama, Muhammad Eriansyah S.P. Terima kasih atas dukungannya dalam proses skripsi ini.
10. Semua pihak yang tidak dapat disebutkan satu per satu, yang telah dengan tulus ikhlas memberikan doa dan motivasi sehingga dapat terselesaikannya skripsi ini.

Akhir kata, saya berharap Tuhan Yang Maha Esa berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga skripsi ini membawa manfaat bagi pengembangan ilmu.

Gresik, 10 Agustus 2022

Penulis

# DAFTAR ISI

[HALAMAN PENGESAHAN i](#_Toc112094315)

[HALAMAN PERNYATAAN ORISINALITAS ii](#_Toc112094316)

[HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI iii](#_Toc112094317)

[ABSTRAK iv](#_Toc112094318)

[*ABSTRACT* v](#_Toc112094319)

[KATA PENGANTAR vi](#_Toc112094320)

[DAFTAR ISI viii](#_Toc112094321)

[BAB 1 PENDAHULUAN 1](#_Toc112094322)

[1.1 Latar Belakang 1](#_Toc112094323)

[1.2 Rumusan Masalah 2](#_Toc112094324)

[1.3 Tujuan 2](#_Toc112094325)

[1.4 Batasan Masalah 2](#_Toc112094326)

[1.5 Manfaat Penelitian 2](#_Toc112094327)

[BAB II TINJAUAN PUSTAKA 3](#_Toc112094328)

[2.1 Penelitian Terdahulu 3](#_Toc112094329)

[2.2 Brand Nike 4](#_Toc112094330)

[2.3 Kualitas Barang Fashion 4](#_Toc112094331)

[2.4 Convolutional Neural Network (CNN) 5](#_Toc112094332)

[2.4.1 *Convolution Layer* 5](#_Toc112094333)

[2.4.2 Pooling 6](#_Toc112094334)

[2.4.3 *Fully-connected Layer* 7](#_Toc112094335)

[2.5 *Confusion Matrix* 7](#_Toc112094336)

[2.6 Arsitektur CNN (Convolutional Neural Network) 8](#_Toc112094337)

[2.6.1 MobileNetV2 8](#_Toc112094338)

[2.6.2 VGG16 9](#_Toc112094339)

[2.6.3 ResNet50V2 9](#_Toc112094340)

[2.6.4 DenseNet121 9](#_Toc112094341)

[2.6.5 Xception 9](#_Toc112094342)

[BAB III METODOLOGI PENELITIAN 10](#_Toc112094343)

[3.1 Studi Pustaka 10](#_Toc112094344)

[3.2 Pengumpulan Data 10](#_Toc112094345)

[3.3 Pengelolaan Data Awal / Pre-Processing Data 14](#_Toc112094346)

[3.4 Pembagian Dataset 15](#_Toc112094347)

[3.5 Pembuatan Model 15](#_Toc112094348)

[3.6 Pengujian Model 19](#_Toc112094349)

[3.7 Pembuatan *Prototype* 20](#_Toc112094350)

[3.8 *Testing Prototype* 20](#_Toc112094351)

[3.9 Dokumentasi 21](#_Toc112094352)

[BAB IV HASIL DAN PEMBAHASAN 22](#_Toc112094353)

[4.1 Pembuatan Model menggunakan CNN 22](#_Toc112094354)

[4.1.1 Arsitektur VGG16 22](#_Toc112094355)

[4.1.2 Arsitektur MobileNetV2 24](#_Toc112094356)

[4.1.3 Arsitektur ResNet50V2 26](#_Toc112094357)

[4.1.4 Arsitektur DenseNet121 28](#_Toc112094358)

[4.1.5 Arsitektur Xception 31](#_Toc112094359)

[4.2 Hasil Pengujian 33](#_Toc112094360)

[4.2.1 Arsitektur VGG16 33](#_Toc112094361)

[4.2.2 Arsitektur MobileNetV2 35](#_Toc112094362)

[4.2.3 Arsitektur ResNet50V2 36](#_Toc112094363)

[4.2.4 Arsitektur DenseNet121 38](#_Toc112094364)

[4.2.5 Arsitektur Xception 39](#_Toc112094365)

[4.2.6 Perbandingan Hasil Arsitektur CNN 41](#_Toc112094366)

[4.3 Hasil Implementasi Aplikasi 42](#_Toc112094367)

[BAB V PENUTUP 45](#_Toc112094368)

[5.1 Kesimpulan 45](#_Toc112094369)

[5.2 Saran 45](#_Toc112094370)

[DAFTAR PUSTAKA 46](#_Toc112094371)

[LAMPIRAN 47](#_Toc112094372)

[BIODATA PENULIS 53](#_Toc112094373)

# DAFTAR KODE

[Kode 3. 1 Penghubung dari *Google Collab* dengan *Google Drive* 16](#_Toc112135234)

[Kode 3. 2 Augmentasi Data Citra 17](#_Toc112135235)

[Kode 3. 3 Model Arsitektur 18](#_Toc112135236)

[Kode 3. 4 *Callback Early Stopping* 18](#_Toc112135237)

[Kode 3. 5 Pelatihan Fit Model 18](#_Toc112135238)

[Kode 3. 6 Konversi *Tensorflow Lite* 20](#_Toc112135239)

# DAFTAR GAMBAR

[Gambar 2. 1 Contoh Proses Convolution Neural Network 5](#_Toc112096775)

[Gambar 2. 2 Langkah Menghitung CNN 6](#_Toc112096776)

[Gambar 2. 3 Contoh *Max-Pooling* 7](#_Toc112096777)

[Gambar 2. 4 Pengukuran *Confusion Matrix* 8](#_Toc112096778)

[Gambar 3. 1 Diagram Alur 10](#_Toc112096787)

[Gambar 4. 1 Grafik *Training accuracy vs validation* VGG16 23](#_Toc112096794)

[Gambar 4. 2 Grafik *validasi loss* VGG16 24](#_Toc112096795)

[Gambar 4. 3 Grafik *Training accuracy vs validation* MobileNetV2 25](#_Toc112096796)

[Gambar 4. 4 Grafik *validasi loss* MobileNetV2 26](#_Toc112096797)

[Gambar 4. 5 Grafik *Training accuracy vs validation* ResNet50V2 28](#_Toc112096798)

[Gambar 4. 6 Grafik *validasi loss* ResNe50V2 28](#_Toc112096799)

[Gambar 4. 7 Grafik *Training accuracy vs validation* DenseNet121 30](#_Toc112096800)

[Gambar 4. 8 Grafik *validasi loss* DenseNet121 31](#_Toc112096801)

[Gambar 4. 9 Grafik *Training accuracy vs validation Xception* 32](#_Toc112096802)

[Gambar 4. 10 *Validasi loss Xception* 33](#_Toc112096803)

[Gambar 4. 11 Hasil Akurasi Pelatihan Model 42](#_Toc112096804)

[Gambar 4. 12 Tampilan Awal Splash Screen 42](#_Toc112096805)

[Gambar 4. 13 Tampilan Pilihan Media 43](#_Toc112096806)

[Gambar 4. 14 Tampilan Klasifikasi 43](#_Toc112096807)

[Gambar 4. 15 Tampilan Deteksi dengan Kamera 44](#_Toc112096808)

# DAFTAR TABEL

[Tabel 3. 1 Citra Orisinal dan Tiruan 11](#_Toc112134013)

[Tabel 3. 2 Cropping Gambar 15](#_Toc112134014)

[Tabel 4. 1 *Confusion Matrix* VGG16 34](#_Toc112134021)

[Tabel 4. 2 Hasil *Confusion Matriks* VGG16 34](#_Toc112134022)

[Tabel 4. 3 Hasil *Accuracy* VGG16 35](#_Toc112134023)

[Tabel 4. 4 *Confusion Matrix* MobileNetV2 35](#_Toc112134024)

[Tabel 4. 5 Hasil *Confusion Matriks* MobileNetV2 36](#_Toc112134025)

[Tabel 4. 6 Hasil *Accuracy* MobileNetV2 36](#_Toc112134026)

[Tabel 4. 7 *Confusion Matrix* ResNet50V2 37](#_Toc112134027)

[Tabel 4. 8 Hasil *Confusion Matriks* ResNet50V2 37](#_Toc112134028)

[Tabel 4. 9 Hasil *Accuracy* ResNet50V2 38](#_Toc112134029)

[Tabel 4. 10 *Confusion Matrix* DenseNet121 38](#_Toc112134030)

[Tabel 4. 11 Hasil *Confusion Matriks* DenseNet121 39](#_Toc112134031)

[Tabel 4. 12 Hasil *Accuracy* DenseNet121 39](#_Toc112134032)

[Tabel 4. 13 *Confusion Matrix* Xception 40](#_Toc112134033)

[Tabel 4. 14 Hasil *Confusion Matriks* Xception 40](#_Toc112134034)

[Tabel 4. 15 Hasil *Accuracy* Xception 41](#_Toc112134035)

[Tabel 4. 16 Perbandingan Hasil Arsitektur 41](#_Toc112134036)

# BAB I PENDAHULUAN

## 1.1 Latar Belakang

*Fashion* menjadi hal terpenting dalam kehidupan sehari-hari baik dalam hal penampilan maupun gaya keseharian yang bertujuan untuk meningkatkan kepercayaan diri dan sebagai alat komunikasi untuk menyampaikan identitas pribadi atau penilaian seseorang. Di dalam masyarakat, persoalan gaya merupakan hal yang segalanya. Setiap orang diminta untuk bisa memainkan dan mengontrol peranan mereka sendiri dan semua manusia adalah performer. Gaya pakaian, gayarambut, dan aksesoris-aksesoris lainnya merupakan bagian dari pertunjukan identitas dan kepribadian diri. Pakaian tidak hanya sebagai penutup dan penunjangpenampilan, melaikan dapat mencerminkan kepribadian seseorang.

Perkembangan teknologi masa kini sangatlah pesat membuat kepribadian seseorang lebih terbuka pada pengetahuan global dalam berbagai bidang salah satunya bidang *fashion,* hal ini didukung oleh berbagai sisi sisi positif, baik desainer yang semakin potensial, tingkat perekonomian yang baik, hingga area ritel semakin berkembang pesat.

Nilai positif dari perkembangan teknologi dibidang *fashion* contohnya perkembangan jual beli produk *fashion* melalui media sosial, yang pada zaman dahulu penjual atau perusahaan hanya dapat menjual produknya melalui store ataupun toko di daerah masing-masing dan pembeli hanya dapat membeli produk *fashion* yang mereka inginkan di toko tersebut, akan tetapi pada zaman sekarang ini sering kita temui banyak orang yang menjual produknya melalui platform media sosial dan pembeli dapat membeli produk yang mereka inginkan dimanapun lokasi toko. Meskipun perkembangan teknologi zaman semakin signifikan, masih ada sisi negatif dari perkembangan zaman sekarang ini, yaitu banyak pembeli yang belum mengetahui atau sedikit pengetahuan tentang perbedaan fashion bermerek dengan brand orisinal atau asli dengan fashion bermerek yang tiruan atau palsu di platform media sosial seperti Instagram, tiktok, ataupun online shop lainnya.

Berdasarkan permasalahan tersebut akan dilakukan implementasi metode *deep learning* yaitu CNN atau *Convolution Neural Network*, yang dapat digunakan untuk mengklasifikasi citra objek berdasarkan karakteristiknya. Dimana dalam metode CNN atau *Convolution Neural Network* dapat melakukan analisis berdasarkan karakteristik tertentu dari objek yang akan diklasifikasi pada pengalaman di masa sebelumnya (Eka Putra, 2016).

## 1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, adapun rumusan masalah yang terdapat pada penelitian ini, yaitu :

* + 1. Bagaimana cara menerapkan perbedaan produk *brand fashion* yang orisinalatau tiruandengan menggunakan metode CNN ?
    2. Bagaimana cara membuat *prototype* android untuk jenis *brand fashion* nike ?

## 1.3 Tujuan

Berdasarkan uraian di atas, penelitian ini bertujuan sebagai :

* + 1. Melakukan klasifikasi untuk mengetahui perbedaan produk *fashion* dengan memakai *brand orisinal* dan *tiruan* menggunakan metode CNN
    2. Membantu konsumen mengetahui *brand fashion* Nike tiruan atau orisinal dengan menggunakan prototype aplikasi android untuk jenis *brand fashion* nike.

## 1.4 Batasan Masalah

Adapun Batasan dari penelitian ini yaitu:

* + 1. Produk yang digunakan meliputi sepatu, topi, dan t-shirt.
    2. Brand yang digunakan hanya dari *brand* Nike
    3. Area citra yang diambil untuk produk sepatu, *t-shirt* dan topi dari logo.

## 1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan pengetahuan dan kemudahan konsumen untuk membedakan antara *brand* produk orisinalatau *brand* produk tiruan*.*

# BAB II TINJAUAN PUSTAKA

## 2.1 Penelitian Terdahulu

Dalam ini penulis meninjau beberapa klasifikasi menggunakan metode CNN. Penelitian telah dilakukan dengan judul “Klasifikasi Pakaian Berdasarkan Gambar Menggunakan Metode YOLOv3 dan CNN” (Wujaya & Santoso, 2021). Penelitian ini membahas tentang mengklasifikasikan pakaian dalam gambar atau menentukan jenis suatu pakaian. Menggunakan model arsitektur ResNet dengan beberapa tahapan seperti *learning rate, dropout, epoch, dense layer, freeze layer,* dan *data augmentation.* Dengan tingkat akurasi rata-rata yang diperoleh adalah 84,44%.

Penelitian sebelumnya dengan menggunakan metode CNN tetapi menggunakan objek berbeda yang berjudul “Identifikasi Foto Fashion Dengan Menggunakan Convolutional Neural Network (CNN)” (Arther Sandag et al., 2021). Penelitian ini menggunakan CNN, Tensorflow, dan untuk pengujian dataset menggunakan Fashion MNIST. Terdapat dua hasil yang berbeda, hasil pengujian pertama terjadi overfitting, menghasilkan akurasi sebesar 91%. Dan pengujian kedua terjadi penambahan Dropout layers, sehingga menghasilkan akurasi lebih baik sebesar 93%.

Penelitian dengan menggunakan metode lain yang berjudul “Klasifikasi Motif Batik Berbasis Kemiripan Ciri Dengan Wavelet Transform dan Fuzzy Neural Network” (Rangkuti, 2014). Penelitian ini berbasis pada kemiripan ciri dengan menggabungkan metode Wavelet Transform dan Fuzzy Neural Network (FNN), *Fuzzykasi* dilakukan dilakukan dengan lima kategori yaitu Sangat Rendah (SR), Rendah (R), Sedang (S), Tinggi (T) dan Sangat Tinggi (ST). Hasil dari klasifikasi FNN dengan akurasi 86-92%

Penelitian sebelumnya yang terkait dengan metode CNN adalah penelitian yang berjudul “Implementasi Metode Convolution Neural Network Menggunakan Arsitektur LeNet-5 untuk Pengenalan Doodle” (Alwanda et al., 2020). Penelitian ini memakai metode CNN dengan arsitektur LeNet-5 untuk pengenalan jenisdoodle dengan 5 objek gambar yaitu baju, celana, kursi, kupu-kupu dan sepeda. Hasil pengujian pertama, kedua, keempat objek sepeda menunjukkan dengan *accuracy* 93% - 98%, *recall* 86% - 93%, dan precision 81% - 93%. Untuk objek baju dengan nilai *accuracy* 94%, *recall* 86%, dan *precision* 83%.

## 2.2 Brand Nike

Brand atau merek adalah nama, istilah, tanda, lambang, atau desain atau kombinasi dari semua yang memperlihatkan identitas produk atau jasa dari satu penjual atau sekelompok penjual dan membedakan produk itu dari produk pesaing. Penelitian ini memakai brand atau merek untuk barang yang diklasifikasikan yaitu brand dari Nike.

Perusahaan Nike didirikan pada 25 Januari 1964 di Oregon oleh Philip Knight dan Billy Bowerman. Phil Knight adalah pelari jarak menengah yang berasal dari Portland yang dilatih di bawah pelatih atletik Bill Bowerman. Sementara Bowerman sedang mencari cara untuk meningkatkan kinerja muridnya dan mencoba memperbaiki sepatu mereka di waktu luangnya. Mereka telah mencoba banyak kombinasi yang berbeda-beda tetapi mereka tidak menemukan keberhasilan. Philip dan Billy, menggunakan nama Nike di perusahaan sepatunya demi sebuah tekad kemenangan.

Nama Nike diambil dari mitologi dewa Yunani. Nike adalah nama dari Dewi Nike yang merupakan simbol kemenangan atau keberhasilan. Nike adalah salah satu merek sepatu, peralatan olahraga, dan pakaian terbesar dan paling terkenal di dunia. (www.kompas.com).

## 2.3 Kualitas Barang Fashion

Untuk melakukan klasifikasi data terdapat beberapa jenis, antara lain:

1. *Orisinal* atau Asli

Orisinal adalah barang resmi dari pihak pembuatnya. Barang ini murni di produksi, di seleksi, oleh sang produsen sehingga kualitas barang benar-benar terjaga. Membeli produk Orisinal juga merupakan kebanggaan tersendiri bagi orang yang membelinya. Produk *tiruan* atau palsu seringkali menyamai penampilan produk asli. Membuat konsumen menjadikan produk alternatif untuk dimiliki.

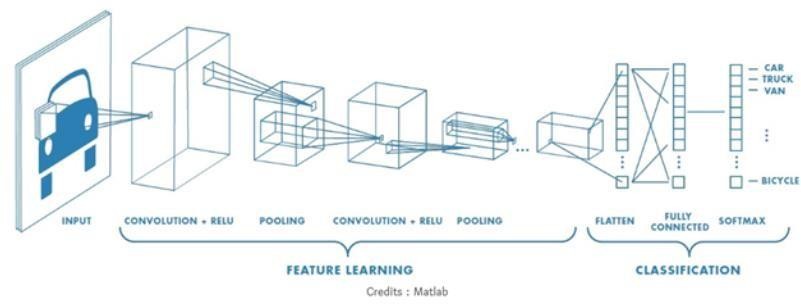
1. *Tiruan*

Tiruan adalah barang tiruan yang menyalin merek atau logo, memiliki ejaan yang salah, yang lebih modern modelnya akan dimiripkan seperti produk asli atau orisinalnya, sehingga sulit untuk membedakan antara yang asli atau palsu.

## 2.4 Convolutional Neural Network (CNN)

Algoritma *Convolutional Neural Network* atau CNN merupakan algoritma *deep learning* yang digunakan untuk data terlabel berbentuk visual atau citra. Menggunakan metode supervised learning. Supervised learning adalah data yang dilatih dan terdapat variabel yang ditargetkan sehingga tujuan dari metode ini adalah mengelompokan suatu data ke data yang sudah ada. CNN terdiri dari tiga jenis lapisan atau blok penyusun, yaitu lapisan konvolusional, lapisan sambungan, dan lapisan yang sepenuhnya terhubung. Lapisan konvolusi terdiri dari sekumpulan operasi matematika, yang merupakan operasi linier khusus. Output dari konvolusi yang diterapkan ke input tunggal akan tetap sebagai piksel tunggal, dan output dari konvolusi yang diterapkan ke seluruh gambar 3 dimensi akan menjadi gambar 2 dimensi.

Proses CNN terdiri dari rangkaian *Feature Learning dan Classification.*



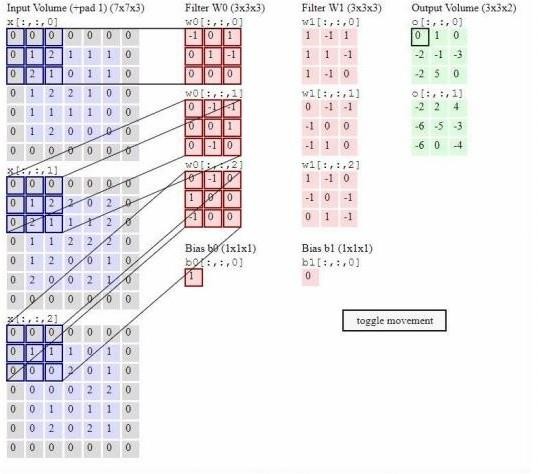
Gambar 2. Contoh Proses Convolution Neural Network

(Sumber: Sofia, 2019)

Lapisan *Feature Learning* berguna untuk mengubah input menjadi *feature* berdasarkan ciri dari input yang berbentuk angka dalam vector. Sedangkan lapisanklasifikasi bertujuan untuk mengklasifikasikan tiap neuron yang diektrasi pada fitur sebelumnya.

### 2.4.1 *Convolution Layer*

*Convolutional* adalah operasi matematika yang menggabungkan dua set informasi. Lapisan *Convolutional Layer* terdiri dari neuron yang tersusun membentuk sebuah filter atau menghasilkan *feature map* dengan panjang dan tinggi (*pixels*). Contoh, layer pertama pada *feature extraction layer* biasanya adalah *convolutional layers* dengan ukuran 5x5x3. Panjang 5 *pixels*, tinggi 5 *pixels* dan tebal/jumlah 3 buah sesuai dengan *channel* dari *image*. Ketiga filter digeser ke seluruh bagian gambar. Untuk ilustrasi *Convolutional* dapat dilihat sebagai berikut

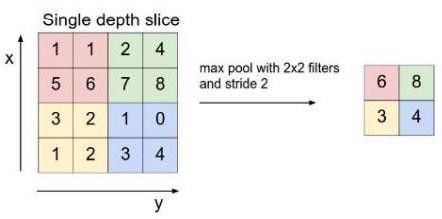


Gambar 2. Langkah Menghitung CNN

(Sumber: Sofia, 2019)

### 2.4.2 *Pooling*

*Pooling Layer* berada setelah *convolutional layer*, *pooling layer* terdiri dari filter dengan ukuran dan stride bergeser secara bergantian pada area *feature map.* Terdapat dua macam *pooling layer*, yaitu *average pooling* dan *max pooling*. *Average pooling* mengambil nilai rata-rata, sedangkan untuk *max-pooling*mengambil nilai maksimal. Bentuk *max-pooling* menggunakan filter dengan ukuran 2x2 diaplikasikan sebanyak dua Langkah dan beroperasi di setiap inputnya. Contoh gambar *max-pooling* dapat dilihat sebagai berikut :



Gambar 2. Contoh *Max-Pooling*

(Sumber: Medium Nadhifa Sofia, 2019)

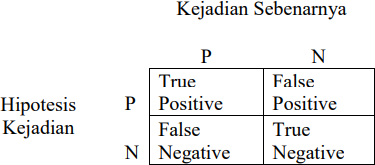
Dari gambar di atas menunjukan output dari *max-pooling* adalah sebuah matriks dengan dimensi yang lebih kecil dibandingkan dengan citra awal. Jika dilihat dari gambar diatas operasi *max-pooling* menggunakan ukuran filter 2x2. Citra awal masukan dari proses tersebut berukuran 4x4, masing-masing 4 angka pada *output* tersebut diambil maksimal dan dilanjutkan membuat *output* baru menjadi ukuran 2x2.

### 2.4.3 *Fully-connected Layer*

Lapisan yang menghubungkan semua neuron dari lapisan sebelumnya. Lapisan ini digunakan pada MLP (*Multi Layer Perceptron*) bertujuan untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan. Perbedaan lapisan *Fully-Connected* dengan lapisan *Convolution Layer* adalah neuron di lapisan konvolusi terhubung hanya ke daerah tertentu pada input, sementara lapisan *Fully-Connected* memiliki neuron yang secara keseluruhan terhubung.

## 2.5 *Confusion Matrix*

Perhitungan faktor-faktor penentuan baik atau tidaknya performa suatu model klasifikasi dapat dilihat dari parameter pengukuran performa, yaitu tingkat akurasi, *recall*, dan presisi. Gambar 2.5 adalah salah satu pengukuran *Confusion- matrix.*



Gambar 2. Pengukuran *Confusion Matrix*

Berdasarkan gambar di atas terdapat beberapa nilai didalam matriks yaitu “*True Positive*” (TP), “*True Negative*” (TN), “*False Positive*” (FP), dan “*False Negative*” (FN), seluruh kemungkinan kejadian sebenarnya positif (P) dan seluruh kemungkinan kejadian sebenarnya negatif (N).

(2.1)

Untuk menghitung tingkat presisi prediksi kejadian dapat digunakan persamaan:

(2.2)

Presisi menggambarkan ketepatan suatu model prediksi. Selain presisi dan akurasi, untuk dapat melihat lebih detail lagi kinerja suatau sistem, dapat dilihat suatu kelas yaitu *recall*. *Recall* dihitung dengan persamaan sebagai berikut:

(2.3)

Dan untuk rumus perhitungan F1-Score yaitu:

(2.4)

## 2.6 Arsitektur CNN (Convolutional Neural Network)

### 2.6.1 MobileNetV2

Model CNN MobileNet, merupakan salah satu arsitektur yang dapat digunakan untuk computing resource berlebih. Seperti namanya, Mobile, para peneliti dari Google membuat arsitektur CNN yang dapat digunakan untuk ponsel. MobileNet merilis versi kedua pada April 2017. Seperti MobilenetV1, MobileNetV2 masih menggunakan depthwise dan pointwise convolution. MobileNetV2 memiliki kelebihan untuk mengatasi kebutuhan *resource* komputasi dan nilai akurasi yang tinggi (Budiman et al., 2021). MobileNetV2 menambahkan dua fitur baru yaitu:

1) *linear bottleneck.*

2) *shortcut connections* antar *bottlenecks*.

### 2.6.2 VGG16

VGGNet merupakan arsitektur CNN yang dirancang oleh Karen Simonyan dan Andrew Zisserman dari Visual Geometry Group. VGG16 menggunakan ukuran 3×3 sebagai ukuran kernelnya dan tiga lapisan terakhir adalah lapisan fully connected (FC) (Hasan et al., 2021).

### 2.6.3 ResNet50V2

ResNet-50V2 adalah salah satu varian ResNet yang memiliki 50 layer. Jika pada varian ResNet sebelumnya dilakukan skip connection sebanyak 2 layer, maka ResNet-50 melewati 3 layer dan terdapat 1x1 convolution layer (Wujaya & Santoso, 2021).

### 2.6.4 DenseNet121

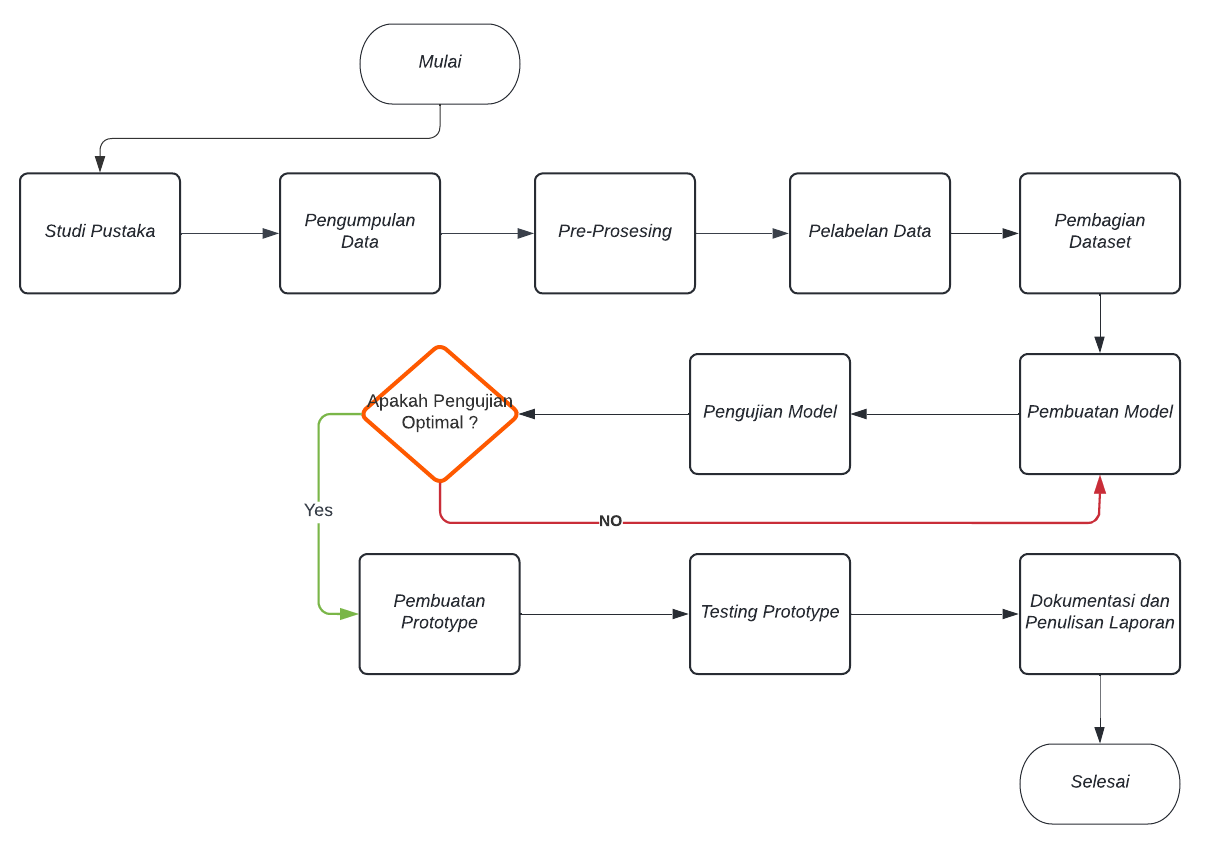
DenseNet121 merupakan model arsitektur dengan dense block dimana pada block tersebut setiap *layer* terhubung dengan semua *layer* secara langsung. *Layer* mengambil *input* dari *output* semua *layer* sebelumnya dan memberikan *output* untuk semua *layer* setelahnya (Pardede & Putra, 2020).

### 2.6.5 Xception

Xception adalah singkatan dari Xtreme of Inception. Seperti namanya arsitektur ini terinspirasi dari Inceptionv3. Xception sedikit lebih unggul daripada inceptionv3 untuk dataset imagenet, dan secara signifikan mengungguli InceptionV3 pada kasus klasifikasi citra dengan dataset sebanyak 350 juta citra dengan 17.000 class (Wahid, 2020).

# BAB III METODOLOGI PENELITIAN

Tahap ini merupakan tahapan yang akan dilakukan untuk menyelesaikan penelitian. Sumber data yang diperoleh pada penelitian ini dengan cara mengambil gambar secara manual menggunakan kamera *smartphone* pada *store* atau toko dan mengambil gambar dari internet. Berikut ini adalah tahapan perancangan pada penelitian yang dapat digambarkan melalui diagram alur pada Gambar 3.1



Gambar 3. Diagram Alur

## 3.1 Studi Pustaka

Studi Pustaka merupakan tahapan yang dilakukan bertujuan untuk penyusunan dasar teori, pendalaman mengenai klasifikasi citra dengan metode CNN, dengan cara mengumpulkan dan membaca buku, jurnal, artikel ilmiah serta sumber yang mendukung penelitian ini.

## 3.2 Pengumpulan Data

Penelitian ini membutuhkan data citra atau gambar sebagai proses klasifikasi. Dalam penelitian ini penulis menggunakan data yang diperoleh dari *Official Store* resmi yang berada di mall Surabaya dan Store lain di Gresik, Surabaya, Bojonegoro dan Tuban yang terdapat brand nike dan pengambilan data dari *website official* [www.Nike.com](http://www.Nike.com) atau akun *Store Instagram* yang menjual produk dari Nike, adapun nama akun Instagramnya ialah nk\_jogja.id, sepatu\_nike\_bandung, sepatusneakersindonesia, rsp.store, sportscornershop, premium\_cap, topi\_branded, rubbersouljkt. Data diambil sebanyak 1622. Proses pengambilan data yang dilakukan menggunakan kamera *smartphone* Realme 6 secara langsung. Data yang digunakan untuk penelitian ini meliputi citra dari logosepatu, logo topi, dan logo t-shirt. Dataset yang digunakan pada setiap *class* memakai 200 data train, 50 data untuk setiap *class* *validation* dan 20 data untuk setiap *class test* untuk jenis produk *tiruan* atau *orisinal.*

Tabel 3. Citra Orisinal dan Tiruan

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Kelas** | **Citra Orisinal dan Tiruan** |
| 1 | Sepatu Orisinal | Sumber: (Dokumentasi Pribadi) |
| 2 | Sepatu *Tiruan* | Sumber: (Dokumentasi Pribadi) |
| 3 | Topi Orisinal | G:\skripsi fajar\bismillahbagus\test\topioriginal\topioriginal259.jpgSumber: (Dokumentasi Pribadi) |
| 4 | Topi Tiruan | G:\skripsi fajar\bismillahbagus\test\topifake\topifake265.jpgSumber : (Dokumentasi Pribadi) |
| 5 | Kaos Orisinal | G:\skripsi fajar\bismillahbagus\test\kaosoriginal\kaosoriginal256.jpgSumber: (Dokumentasi Pribadi) |
| 6 | Kaos Tiruan | G:\skripsi fajar\bismillahbagus\test\kaosfake\kaosfake253.jpgSumber : (Dokumentasi Pribadi) |

Terdapat perbedaan ciri-ciri dari barang Orisinal dan Tiruan, salah satunya yaitu pada bagian logo sepatu, Nike orisinal akan terlihat rapi untuk penempatan logo karena telah melewati proses *quality control.* Produk yang tiruan memiliki logo yang tidak rapi di beberapa bagian sisinya.

## 3.3 Pengelolaan Data Awal / Pre-*Processing Data*

Agar memperoleh hasil akurasi yang maksimal akan dilakukan tahap pre-processing. Pada penelitian ini menggunakan dua pre-processing yang akan dilakukan, yaitu :

* + 1. *Cropping*

Melakukan *Cropping* digunakan untuk membuat gambar lebih ke area citra dibagian logo.

Tabel 3. Cropping Gambar

|  |  |
| --- | --- |
| **Sebelum *Cropping*** | **Sesudah *Cropping*** |
| Sumber : (Dokumentasi Pribadi) | Sumber : (Dokumentasi Pribadi) |

* + 1. *Resize*

*Resize* dilakukan dengan *size* 300 x 300 untuk mendapatkan hasil citra dengan ukuran yang lebih kecil dari citra aslinya. karena citra aslinya memakai kamera 64 *megapixel* yang digunakan untuk data *training* dan data *testing*.

## 3.4 Pembagian Dataset

Tahap dataset membagi seluruh data yang digunakan dalam proses *training*, *validation*, dan *test*. Persentase data yang digunakan pada *training* set adalah 70% dengan jumlah dataset 200 per *class*, pada *validation* 20% dengan 50 dataset per *class*, dan pada *test set* 10% dengan 20 dataset per *class*. Terdapat 3 folder, yaitu folder *training*, folder *validation*, dan folder *test*.

Dalam setiap folder dibagi menjadi 6 kelas atau 6 folder, yaitu folder topi orisinal, folder topi tiruan, folder sepatu orisinal, folder sepatu tiruan, folder kaos orisinal, dan folder test.

## 3.5 Pembuatan Model

Pada tahap pembuatan model dilakukan proses pelatihan data yang akan dilatihan dengan beberapa *epoch* iterasi dan training data menggunakan *library tensorflow*, nanti akan dicoba beberapa arsitektur dari CNN, diantaranya adalah MobileNet dan VGGNet, ResNet dan Xception. Langkah awal yaitu dengan menghubungkan *Google Collab* dengan *Google Drive*. Langkah ini dilakukan untuk mendapatkan data yang telah diunggah pada *Google Drive.*

|  |
| --- |
| from google.colab import drive  drive.mount(‘/content/drive’) |

Kode 3. Penghubung dari *Google Collab* dengan *Google Drive*

Setelah terhubung, selanjutnya yaitu menjalankan data yang telah diunggah atau melakukan augmentasi data untuk mendapatkan informasi dan akurasi. Proses augmentasi merupakan proses mengubah gambar atau *memodifikasi* gambar.

|  |
| --- |
| IMAGE\_SIZE = 300  BATCH\_SIZE = 10  train\_dir = os.path.join(‘/content/drive/MyDrive/skripsi/bismillahbagus/train’)  datagen = tf.keras.preprocessing.image.ImageDataGenerator(  horizontal\_flip=True,  brightness\_range=[0.2, 0.8],  vertical\_flip=True,  rescale=1/255,  )  train\_generator = datagen.flow\_from\_directory(  train\_dir,  target\_size = (IMAGE\_SIZE, IMAGE\_SIZE),  batch\_size=BATCH\_SIZE,  )  val\_dir = os.path.join(‘/content/drive/MyDrive/skripsi/bismillahbagus/validation’)  val\_generator = datagen.flow\_from\_directory(  val\_dir,  target\_size =(IMAGE\_SIZE, IMAGE\_SIZE),  batch\_size = BATCH\_SIZE,  )  test\_dir =os.path.join(‘/content/Drive/MyDrive/skripsi/bismillahbagus/test’)  test\_generator = datagen.flow\_from\_directory(  test\_dir,  target\_size =(IMAGE\_SIZE, IMAGE\_SIZE),  batch\_size =BATCH\_SIZE,  class\_mode=’categorical’,  shuffle=False) |

Kode 3. Augmentasi Data Citra

Langkah selanjutnya ialah melakukan pembuatan model dengan mencoba menggunakan Arsitektur VGG16, MobileNetV2, ResNet50, Xception, dan DenseNet121. Dengan code base\_model = tf.keras.applications.VGG16 dan jika mengganti arsitekturnya maka .VGG16 bisa diganti dengan arsitektur lainnya yaitu MobileNetV2, ResNet50, Xception dan DenseNet121.

|  |
| --- |
| IMG\_SHAPE = (300, 300, 3)  from tensorflow.keras.models import Sequential  from tensorflow.python.keras.layers import Dense, Flatten, Dropout  model=Sequential()  base\_model= t.keras.applications.MobileNetV2(input\_shape=IMG\_SHAPE,  include\_top=False,  weights=’imagenet’)  for layer in base\_model.layers:  layer.trainable=False  model.add(base\_model)  model.add(Flatten())  model.add(Dense(1024, activation=’relu’))  model.add(Dense(512, activation=’relu’))  model.add(Dropout(0.2))  model.add(Dense(256, activation=’relu’))  model.add(Dropout(0.2))  model.add(Dense(128, activation=’relu’))  model.add(Dense(64, activation=’relu’))  model.add(Dropout(0.2))  model.add(Dense(32, activation=’relu’))  model.add(Dense(16, activation=’relu’))  model.add(Dropout(0.2))  model.add(Dense(8, activation=’relu’))  model.add(Dense(6, activation=’sofmax’))  model.compile(“adam”,  loss=”categorical\_crossentropy”,  metrics=[“acc”])  model.summary() |

Kode 3. Model Arsitektur

Untuk mengurangi terjadinya *overfitting* dengan menggunakan *callback early stopping* dengan patience = 30. Proses *training* berhenti ketika terdapat 30 proses yang buruk dengan menggunakan parameter “acc” untuk monitor kinerja pengujian.

|  |
| --- |
| import tensorflow.keras as kr  aerly\_stop=kr.callbacks.EarlyStopping(monitor=’acc’,min\_delta-1e3,patience=30,  verbose=1, mode=’max’, restore\_best\_weights-True) |

Kode 3. *Callback Early Stopping*

Untuk melakukan pelatihan model digunakan *fit model* dengan menggunakan *epochs*=100.

|  |
| --- |
| history = model.fit(train\_generator),  epochs=100,  validation\_data=val\_generator,  callbacks=[early\_stop] |

Kode 3. Pelatihan Fit Model

Setelah selesai melakukan serangkain proses pelatihan model akan didapatkan hasil output dari proses model.fit dengan *epochs*=100 dan proses berhenti pada *epochs*=89 untuk menghindari terjadinya *overfitting*.

Epoch 1/100

120/120 [==============================] - 450s 4s/step - loss: 2.7020 - acc: 0.2042 - val\_loss: 1.4767 - val\_acc: 0.2467

Epoch 2/100

120/120 [==============================] - 17s 141ms/step - loss: 1.5595 - acc: 0.2875 - val\_loss: 1.4188 - val\_acc: 0.3400

Epoch 3/100

120/120 [==============================] - 16s 131ms/step - loss: 1.4598 - acc: 0.3433 - val\_loss: 1.7904 - val\_acc: 0.5233

Epoch 4/100

120/120 [==============================] - 16s 133ms/step - loss: 1.4004 - acc: 0.4050 - val\_loss: 1.3203 - val\_acc: 0.4633

Epoch 5/100

120/120 [==============================] - 16s 132ms/step - loss: 1.2842 - acc: 0.4833 - val\_loss: 1.1460 - val\_acc: 0.5933

………………………………………………………

Epoch 85/100

120/120 [==============================] - 15s 124ms/step - loss: 0.2790 - acc: 0.9692 - val\_loss: 2.1464 - val\_acc: 0.7533

Epoch 86/100

120/120 [==============================] - 16s 133ms/step - loss: 0.1053 - acc: 0.9692 - val\_loss: 1.5958 - val\_acc: 0.7967

Epoch 87/100

120/120 [==============================] - 15s 124ms/step - loss: 0.0918 - acc: 0.9733 - val\_loss: 2.1723 - val\_acc: 0.7567

Epoch 88/100

120/120 [==============================] - 15s 124ms/step - loss: 0.0894 - acc: 0.9742 - val\_loss: 2.2115 - val\_acc: 0.7800

Epoch 89/100

120/120 [==============================] - ETA: 0s - loss: 0.1633 - acc: 0.9617Restoring model weights from the end of the best epoch: 59.

120/120 [==============================] - 15s 127ms/step - loss: 0.1633 - acc: 0.9617 - val\_loss: 2.4001 - val\_acc: 0.7900

Epoch 89: early stopping.

## 3.6 Pengujian Model

Setelah dilakukan *training* model atau pelatihan model tahapan selanjutnya menguji model CNN dengan data *test.* Proses *training* bertujuan melatihan algoritma CNN dalam mengenali dataset. Proses *testing* bertujuan menguji model yang dibentuk pada proses *training.* Pengujian meliputi akurasi, recall, presisi. Dan f1-measure. Pengujian ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan sistem merupakan bagian dari presisi, akurasi adalah tingkat kedekatan antara nilai prediksi dengan nilai actual, recall adalah tingkat keberhasilan sistem dalam menemukan sebuah informasi. dan untuk F1-measure merupakan perhitungan evaluasi yang mengkombinasikan recall dan presisi.

## 3.7 Pembuatan *Prototype*

Pada tahap ini penulis membuat implementasi prototype perangkat lunak melalui *coding* dengan *generate* dari Google yang bernama Google *Collaboratory* berbasis *cloud*, dari bahasa pemrograman Phtyon akan diimplementasikan menggunakan TensorFlow Lite dan *software* Android Studio.

|  |
| --- |
| saved\_model\_dir = 'save/model'  tf.saved\_model.save(model, saved\_model\_dir)  converter = tf.lite.TFLiteConverter.from\_saved\_model(saved\_model\_dir)  tflite\_model = converter.convert()  with open('Klasifikasi\_3\_Jenis\_Beras.tflite', 'wb') as f:    f.write(tflite\_model) |

Kode 3. Konversi *Tensorflow Lite*

## 3.8 *Testing Prototype*

Setelah pembuatan *prototype*, uji coba atau *testing* dilakukan untuk mengetahui data yang digunakan dapat bekerja dengan akurasi keberhasilan pencapaian prototype yang telah dibuat. Pengujian dari segi fungsional dengan Metode *Blackbox Testing*, untuk mengetahui apakah fungsi, masukan dan keluaran dari aplikasi sesuai dengan yang dibutuhkan.

## 3.9 Dokumentasi

Penulis membuat laporan dari keseluruhan hasil penelitian. Laporan yang memuat dari pendahuluan, tinjauan pustaka, metodologi penelitian, pembahasan dan saran untuk penulis.

# BAB IV HASIL DAN PEMBAHASAN

Pada bab ini menjelaskan proses klasifikasi *brand fashion* orisinal atau tiruan menggunakan metode Convolutional Neural Network (CNN). Proses pembuatan model menggunakan Google Colab sedangkan pembuatan aplikasi mobile menggunakan android studio dan library tensorflow lite.

## 4.1 Pembuatan Model menggunakan CNN

Pembuatan model dilakukan dengan mencoba lima arsitektur, yaitu VGG16, MobileNetV2, ResNet50, ResNet152V2, Xception. Pengujian model dengan menggunakan beberapa layer dense dan dijalankan menggunakan epoch 100. Model di jalankan dengan menggunakan *categorical\_crossentropy* dan *confusion matriks* untuk mengukur model dalam melakukan prediksi, dan menggunakan *Callbacks* untuk mengurangi *overfitting* tanpa mengurasi akurasi pada model yang diuji.

### 4.1.1 Arsitektur VGG16

Hasil pengujian untuk arsitektur VGG16 dilakukan melalui kode pada bab 3.5 dengan menggunakan 100 *epochs* dan *callback early stopping*.

Epoch 1/100

120/120 [==============================] - 782s 6s/step - loss: 1.9301 - acc: 0.1675 - val\_loss: 1.7984 - val\_acc: 0.1667

Epoch 2/100

120/120 [==============================] - 18s 150ms/step - loss: 1.8020 - acc: 0.1558 - val\_loss: 1.7920 - val\_acc: 0.1667

Epoch 3/100

120/120 [==============================] - 19s 154ms/step - loss: 1.7980 - acc: 0.1683 - val\_loss: 1.7768 - val\_acc: 0.1833

Epoch 4/100

120/120 [==============================] - 18s 153ms/step - loss: 1.7504 - acc: 0.2033 - val\_loss: 1.6256 - val\_acc: 0.3033

Epoch 5/100

120/120 [==============================] - 19s 154ms/step - loss: 1.5776 - acc: 0.2983 - val\_loss: 1.5971 - val\_acc: 0.3133

………………………………..

Epoch 75/100

120/120 [==============================] - 19s 160ms/step - loss: 0.6099 - acc: 0.6617 - val\_loss: 2.8096 - val\_acc: 0.6000

Epoch 76/100

120/120 [==============================] - 19s 160ms/step - loss: 0.8590 - acc: 0.6133 - val\_loss: 0.7879 - val\_acc: 0.6100

Epoch 77/100

120/120 [==============================] - 19s 160ms/step - loss: 0.6339 - acc: 0.6408 - val\_loss: 1.5322 - val\_acc: 0.6133

Epoch 78/100

120/120 [==============================] - ETA: 0s - loss: 0.6451 - acc: 0.6558Restoring model weights from the end of the best epoch: 68.

120/120 [==============================] - 19s 161ms/step - loss: 0.6451 - acc: 0.6558 - val\_loss: 2.0881 - val\_acc: 0.5833

Epoch 78: early stopping

Setelah *epochs* selesai dijalankan akan didapatkan Grafik Training accuracy vs validation dan validasi *loss.*

Chart, line chart

Description automatically generated

Gambar 4. Grafik *Training accuracy vs validation* VGG16

Chart, histogram

Description automatically generated

Gambar 4. Grafik *validasi loss* VGG16

### 4.1.2 Arsitektur MobileNetV2

Hasil pengujian untuk arsitektur MobileNetV2 dengan *epochs*=100 dan didapatkan Grafik Training accuracy vs validation dan validasi *loss.*

Epoch 1/100

120/120 [==============================] - 450s 4s/step - loss: 2.7020 - acc: 0.2042 - val\_loss: 1.4767 - val\_acc: 0.2467

Epoch 2/100

120/120 [==============================] - 17s 141ms/step - loss: 1.5595 - acc: 0.2875 - val\_loss: 1.4188 - val\_acc: 0.3400

Epoch 3/100

120/120 [==============================] - 16s 131ms/step - loss: 1.4598 - acc: 0.3433 - val\_loss: 1.7904 - val\_acc: 0.5233

Epoch 4/100

120/120 [==============================] - 16s 133ms/step - loss: 1.4004 - acc: 0.4050 - val\_loss: 1.3203 - val\_acc: 0.4633

Epoch 5/100

120/120 [==============================] - 16s 132ms/step - loss: 1.2842 - acc: 0.4833 - val\_loss: 1.1460 - val\_acc: 0.5933

………………………………………………………….

Epoch 85/100

120/120 [==============================] - 15s 124ms/step - loss: 0.2790 - acc: 0.9692 - val\_loss: 2.1464 - val\_acc: 0.7533

Epoch 86/100

120/120 [==============================] - 16s 133ms/step - loss: 0.1053 - acc: 0.9692 - val\_loss: 1.5958 - val\_acc: 0.7967

Epoch 87/100

120/120 [==============================] - 15s 124ms/step - loss: 0.0918 - acc: 0.9733 - val\_loss: 2.1723 - val\_acc: 0.7567

Epoch 88/100

120/120 [==============================] - 15s 124ms/step - loss: 0.0894 - acc: 0.9742 - val\_loss: 2.2115 - val\_acc: 0.7800

Epoch 89/100

120/120 [==============================] - ETA: 0s - loss: 0.1633 - acc: 0.9617Restoring model weights from the end of the best epoch: 59.

120/120 [==============================] - 15s 127ms/step - loss: 0.1633 - acc: 0.9617 - val\_loss: 2.4001 - val\_acc: 0.7900

Epoch 89: early stopping

**A picture containing chart

Description automatically generated**

Gambar 4. Grafik *Training accuracy vs validation* MobileNetV2

Chart, histogram

Description automatically generated

Gambar 4. Grafik *validasi loss* MobileNetV2

### 4.1.3 Arsitektur ResNet50V2

Hasil pengujian untuk arsitektur RenNet50V2 dengan *epochs*=100 didapatkan Grafik Training accuracy vs validation dan validasi *loss.*

Epoch 1/100

120/120 [==============================] - 756s 6s/step - loss: 2.6707 - acc: 0.1817 - val\_loss: 1.7926 - val\_acc: 0.1667

Epoch 2/100

120/120 [==============================] - 523s 4s/step - loss: 1.8152 - acc: 0.2050 - val\_loss: 1.6810 - val\_acc: 0.2600

Epoch 3/100

120/120 [==============================] - 533s 4s/step - loss: 1.7157 - acc: 0.2600 - val\_loss: 1.5157 - val\_acc: 0.3467

Epoch 4/100

120/120 [==============================] - 523s 4s/step - loss: 1.5619 - acc: 0.3075 - val\_loss: 1.3758 - val\_acc: 0.3567

Epoch 5/100

120/120 [==============================] - 530s 4s/step - loss: 1.4649 - acc: 0.3358 - val\_loss: 1.3497 - val\_acc: 0.4800

……………………………………………………

Epoch 15/100

120/120 [==============================] - 509s 4s/step - loss: 1.3416 - acc: 0.3942 - val\_loss: 1.1006 - val\_acc: 0.4833

Epoch 16/100

120/120 [==============================] - 505s 4s/step - loss: 1.2537 - acc: 0.4275 - val\_loss: 1.1291 - val\_acc: 0.4800

Epoch 17/100

120/120 [==============================] - 513s 4s/step - loss: 1.3897 - acc: 0.4133 - val\_loss: 1.8042 - val\_acc: 0.2167

Epoch 18/100

120/120 [==============================] - 510s 4s/step - loss: 1.7900 - acc: 0.2600 - val\_loss: 1.6618 - val\_acc: 0.2667

Epoch 19/100

120/120 [==============================] - 514s 4s/step - loss: 1.5498 - acc: 0.3100 - val\_loss: 1.5784 - val\_acc: 0.3133

Epoch 20/100

120/120 [==============================] - ETA: 0s - loss: 1.5848 - acc: 0.3033Restoring model weights from the end of the best epoch: 10.

120/120 [==============================] - 499s 4s/step - loss: 1.5848 - acc: 0.3033 - val\_loss: 1.6623 - val\_acc: 0.2600

Epoch 20: early stopping

Chart, line chart

Description automatically generated

Gambar 4. Grafik *Training accuracy vs validation* ResNet50V2

Chart, line chart, histogram

Description automatically generated

Gambar 4. Grafik *validasi loss* ResNe50V2

### 4.1.4 Arsitektur DenseNet121

Hasil pengujian untuk arsitektur DenseNet121 dengan *epochs*=100 didapatkan Grafik *Training accuracy vs validation* dan validasi *loss.*

Epoch 1/100

120/120 [==============================] - 345s 3s/step - loss: 2.4745 - acc: 0.2692 - val\_loss: 1.3578 - val\_acc: 0.3100

Epoch 2/100

120/120 [==============================] - 18s 151ms/step - loss: 1.4311 - acc: 0.3375 - val\_loss: 1.1576 - val\_acc: 0.4067

Epoch 3/100

120/120 [==============================] - 18s 148ms/step - loss: 1.1838 - acc: 0.4325 - val\_loss: 1.0422 - val\_acc: 0.5400

Epoch 4/100

120/120 [==============================] - 18s 149ms/step - loss: 0.9615 - acc: 0.5083 - val\_loss: 0.7589 - val\_acc: 0.6100

Epoch 5/100

120/120 [==============================] - 19s 157ms/step - loss: 0.8024 - acc: 0.5650 - val\_loss: 0.7412 - val\_acc: 0.6967

*…………………………………………………………………..*

Epoch 30/100

120/120 [==============================] - 17s 144ms/step - loss: 0.1184 - acc: 0.9550 - val\_loss: 2.3159 - val\_acc: 0.7500

Epoch 31/100

120/120 [==============================] - 18s 146ms/step - loss: 0.2864 - acc: 0.9108 - val\_loss: 1.7016 - val\_acc: 0.7100

Epoch 32/100

120/120 [==============================] - 18s 150ms/step - loss: 0.1499 - acc: 0.9542 - val\_loss: 2.2616 - val\_acc: 0.7167

Epoch 33/100

120/120 [==============================] - 17s 145ms/step - loss: 0.1720 - acc: 0.9375 - val\_loss: 1.8180 - val\_acc: 0.7933

Epoch 34/100

120/120 [==============================] - 17s 145ms/step - loss: 0.2232 - acc: 0.9442 - val\_loss: 1.7708 - val\_acc: 0.7500

Epoch 35/100

120/120 [==============================] - 17s 144ms/step - loss: 0.1800 - acc: 0.9583 - val\_loss: 2.0490 - val\_acc: 0.7367

Epoch 36/100

120/120 [==============================] - 17s 144ms/step - loss: 0.1962 - acc: 0.9550 - val\_loss: 1.6158 - val\_acc: 0.7733

Epoch 37/100

120/120 [==============================] - ETA: 0s - loss: 0.1763 - acc: 0.9483Restoring model weights from the end of the best epoch: 7.

120/120 [==============================] - 18s 153ms/step - loss: 0.1763 - acc: 0.9483 - val\_loss: 1.5506 - val\_acc: 0.7667

Epoch 37: early stopping

Chart, line chart

Description automatically generated

Gambar 4. Grafik *Training accuracy vs validation* DenseNet121

Chart, histogram

Description automatically generated

Gambar 4. Grafik *validasi loss* DenseNet121

### 4.1.5 Arsitektur Xception

Hasil pengujian untuk arsitektur Xception dengan *epochs*=100 didapatkan Grafik Training accuracy vs validation dan validasi *loss.*

Epoch 1/100

120/120 [==============================] - 339s 3s/step - loss: 2.4163 - acc: 0.2133 - val\_loss: 1.5689 - val\_acc: 0.2133

Epoch 2/100

120/120 [==============================] - 18s 148ms/step - loss: 1.5445 - acc: 0.2950 - val\_loss: 1.1891 - val\_acc: 0.4500

Epoch 3/100

120/120 [==============================] - 18s 150ms/step - loss: 1.2210 - acc: 0.4167 - val\_loss: 0.8188 - val\_acc: 0.5233

Epoch 4/100

120/120 [==============================] - 18s 149ms/step - loss: 0.9613 - acc: 0.4867 - val\_loss: 0.7863 - val\_acc: 0.4933

Epoch 5/100

120/120 [==============================] - 19s 158ms/step - loss: 0.8431 - acc: 0.4908 - val\_loss: 0.7247 - val\_acc: 0.5833

…………………………………………………………..

Epoch 50/100

120/120 [==============================] - 18s 149ms/step - loss: 0.0816 - acc: 0.9825 - val\_loss: 1.9252 - val\_acc: 0.7533

Epoch 51/100

120/120 [==============================] - 18s 149ms/step - loss: 0.0549 - acc: 0.9900 - val\_loss: 1.4563 - val\_acc: 0.7967

Epoch 52/100

120/120 [==============================] - 19s 155ms/step - loss: 0.0746 - acc: 0.9842 - val\_loss: 1.6898 - val\_acc: 0.7667

Epoch 53/100

120/120 [==============================] - 18s 149ms/step - loss: 0.2407 - acc: 0.9417 - val\_loss: 1.4677 - val\_acc: 0.7400

Epoch 54/100

120/120 [==============================] - ETA: 0s - loss: 0.0733 - acc: 0.9875Restoring model weights from the end of the best epoch: 44.

120/120 [==============================] - 18s 151ms/step - loss: 0.0733 - acc: 0.9875 - val\_loss: 1.7014 - val\_acc: 0.7667

Epoch 54: early stopping

Chart, line chart

Description automatically generated

Gambar 4. Grafik *Training accuracy vs validation Xception*

Chart, line chart, histogram

Description automatically generated

Gambar 4. *Validasi loss Xception*

## 4.2 Hasil Pengujian

Hasil yang diperoleh dari pengujian model dilakukan dengan mencoba lima arsitektur, yaitu VGG16, MobileNetV2, ResNet50, DenseNet121, Xception.

### 4.2.1 Arsitektur VGG16

Hasil pengujian menggunakan arsitektur VGG16 didapatkan True Positif atau berhasil mendeteksi 59 citra *brand* orisinal atau tiruan. Setelah didapatkan hasil True Positive, True Negative, False Positive, dan False Negative. Selanjutnya diperoleh nilai *Precision*, *Recall*, f1-score dan accuracy. Pada tabel 4.3 didapatkan dengan nilai *accuracy* 0,49%. Untuk menghitung nilai dari *precession* atau presisi dapat dilakukan dengan cara yakni :

Precission = (TP) / (TP+FP) atau Precession = 18 / (18+18) = 18 / 36 = 0,5

Recall = (TP) / (TP + FN) atau Recall = (18) / (18+34) = 18 / 52 = 0,35

F1 Score = 2 \* (Recall\*Precission) / (Recall + Precission) atau 2\*(0,35 \* 0,5) / (0,35 + 0,5) = 2 \* 0,175 / 0,85 = 2 \* 0,20 = 0,4

Tabel 4. *Confusion Matrix* VGG16

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Kaos Tiruan** | **Kaos Orisinal** | **Sepatu Tiruan** | **Sepatu Orisinal** | **Topi Tiruan** | **Topi Orisinal** |
| **Kaos Tiruan** | 18 | 1 | 0 | 0 | 0 | 1 |
| **Kaos Orisinal** | 18 | 0 | 1 | 0 | 0 | 1 |
| **Sepatu Tiruan** | 0 | 0 | 18 | 1 | 0 | 1 |
| **Sepatu Orisinal** | 0 | 0 | 18 | 1 | 0 | 1 |
| **Topi Tiruan** | 0 | 0 | 1 | 0 | 11 | 1 |
| **Topi Orisinal** | 0 | 0 | 0 | 0 | 9 | 11 |

Tabel 4. Hasil *Confusion Matriks* VGG16

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **True Positive** | **False Positive** | **True Negative** | **False Negative** |
| **Kaos Tiruan** | 18 | 18 | 2 | 34 |
| **Kaos Orisinal** | 0 | 0 | 20 | 50 |
| **Sepatu Tiruan** | 18 | 19 | 2 | 31 |
| **Sepatu Orisinal** | 1 | 0 | 19 | 49 |
| **Topi**  **Tiruan** | 11 | 9 | 2 | 49 |
| **Topi Orisinal** | 11 | 9 | 0 | 39 |
| **Total** | **59** | **55** | **45** | **252** |

Tabel 4. Hasil *Accuracy* VGG16

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F1-Score** | **Accuracy** |
| **Kaos Tiruan** | 0.5000 | 0.9000 | 0.6429 |  |
| **Kaos Orisinal** | 0.0000 | 0.0000 | 0.0000 |  |
| **Sepatu Tiruan** | 0.4737 | 0.9000 | 0.6207 |  |
| **Sepatu Orisinal** | 0.5000 | 0.0500 | 0.0909 | **0.4917** |
| **Topi Tiruan** | 0.5500 | 0.5500 | 0.5500 |  |
| **Topi Orisinal** | 0.4783 | 0.5500 | 0.5116 |  |
| **Average** | 0.4170 | 0.4917 | 0.4027 |  |

### 4.2.2 Arsitektur MobileNetV2

Hasil pengujian menggunakan arsitektur MobileNetV2 didapatkan True Positif atau berhasil mendeteksi 77 citra *brand* orisinal atau tiruan. Diperoleh nilai *accuracy* sebesar 0,53%.

Tabel 4. *Confusion Matrix* MobileNetV2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Kaos Tiruan** | **Kaos Orisinal** | **Sepatu Tiruan** | **Sepatu Orisinal** | **Topi Tiruan** | **Topi Orisinal** |
| **Kaos Tiruan** | 17 | 3 | 0 | 0 | 0 | 0 |
| **Kaos Orisinal** | 15 | 5 | 0 | 0 | 0 | 0 |
| **Sepatu Tiruan** | 0 | 0 | 16 | 4 | 0 | 0 |
| **Sepatu Orisinal** | 0 | 0 | 19 | 1 | 0 | 0 |
| **Topi Tiruan** | 1 | 0 | 2 | 0 | 11 | 6 |
| **Topi Orisinal** | 0 | 0 | 1 | 0 | 10 | 9 |

Tabel 4. Hasil *Confusion Matriks* MobileNetV2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **True Positive** | **False Positive** | **True Negative** | **False Negative** |
| **Kaos Tiruan** | 17 | 16 | 3 | 41 |
| **Kaos Orisinal** | 5 | 3 | 17 | 39 |
| **Sepatu Tiruan** | 16 | 22 | 1 | 34 |
| **Sepatu Orisinal** | 19 | 19 | 1 | 38 |
| **Topi**  **Tiruan** | 11 | 10 | 9 | 25 |
| **Topi Orisinal** | 9 | 11 | 6 | 25 |
| **Total** | **77** | **81** | **37** | **202** |

Tabel 4. Hasil *Accuracy* MobileNetV2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F1-Score** | **Accuracy** |
| **Kaos Tiruan** | 0.5152 | 0.5500 | 0.5116 |  |
| **Kaos Orisinal** | 0.6250 | 0.8500 | 0.6415 |  |
| **Sepatu Tiruan** | 0.4211 | 0.2500 | 0.5517 |  |
| **Sepatu Orisinal** | 0.2000 | 0.8000 | 0.0800 | **0.5333** |
| **Topi Tiruan** | 0.5238 | 0.0500 | 0.5366 |  |
| **Topi Orisinal** | 0.6000 | 0.4500 | 0.5143 |  |
| **Average** | 0.4808 | 0.4917 | 0.4469 |  |

### 4.2.3 Arsitektur ResNet50V2

Hasil pengujian menggunakan arsitektur ResNet50V2 didapatkan True Positif atau berhasil mendeteksi 55 citra *brand* orisinal atau tiruan. Diperoleh nilai *accuracy* sebesar 0,49%.

Tabel 4. *Confusion Matrix* ResNet50V2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Kaos Tiruan** | **Kaos Orisinal** | **Sepatu Tiruan** | **Sepatu Orisinal** | **Topi Tiruan** | **Topi Orisinal** |
| Kaos Tiruan | 10 | 8 | 0 | 0 | 0 | 2 |
| Kaos Orisinal | 10 | 9 | 0 | 0 | 1 | 0 |
| Sepatu Tiruan | 0 | 0 | 15 | 3 | 0 | 2 |
| Sepatu Orisinal | 0 | 0 | 15 | 2 | 0 | 3 |
| Topi Tiruan | 1 | 0 | 0 | 0 | 11 | 9 |
| Topi Orisinal | 0 | 0 | 0 | 0 | 12 | 8 |

Tabel 4. Hasil *Confusion Matriks* ResNet50V2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **True Positive** | **False Positive** | **True Negative** | **False Negative** |
| **Kaos Tiruan** | 10 | 11 | 10 | 45 |
| **Kaos Orisinal** | 9 | 11 | 10 | 39 |
| **Sepatu Tiruan** | 15 | 21 | 20 | 54 |
| **Sepatu Orisinal** | 2 | 3 | 18 | 36 |
| **Topi**  **Tiruan** | 11 | 13 | 10 | 43 |
| **Topi Orisinal** | 8 | 16 | 12 | 40 |
| **Total** | **55** | **75** | **80** | **257** |

Tabel 4. Hasil *Accuracy* ResNet50V2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F1-Score** | **Accuracy** |
| **Kaos Tiruan** | 0.5000 | 0.5000 | 0.5000 |  |
| **Kaos Orisinal** | 0.5294 | 0.4500 | 0.4865 |  |
| **Sepatu Tiruan** | 0.5000 | 0.7500 | 0.6000 |  |
| **Sepatu Orisinal** | 0.4000 | 0.1000 | 0.1600 | **0.4000** |
| **Topi Tiruan** | 0.4583 | 0.5500 | 0.5000 |  |
| **Topi Orisinal** | 0.3333 | 0.4000 | 0.3636 |  |
| **Average** | 0.4535 | 0.4583 | 0.4350 |  |

### 4.2.4 Arsitektur DenseNet121

Hasil pengujian menggunakan arsitektur MobileNetV2 didapatkan True Positif atau berhasil mendeteksi 70 citra *brand* orisinal atau tiruan. Diperoleh nilai *accuracy* sebesar 0,52%.

Tabel 4. *Confusion Matrix* DenseNet121

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Kaos Tiruan** | **Kaos Orisinal** | **Sepatu Tiruan** | **Sepatu Orisinal** | **Topi Tiruan** | **Topi Orisinal** |
| **Kaos Tiruan** | 17 | 3 | 0 | 0 | 0 | 0 |
| **Kaos Orisinal** | 14 | 5 | 0 | 0 | 1 | 0 |
| **Sepatu Tiruan** | 0 | 0 | 13 | 7 | 0 | 0 |
| **Sepatu Orisinal** | 0 | 0 | 18 | 2 | 0 | 0 |
| **Topi Tiruan** | 1 | 0 | 0 | 0 | 6 | 14 |
| **Topi Orisinal** | 0 | 0 | 0 | 0 | 2 | 18 |

Tabel 4. Hasil *Confusion Matriks* DenseNet121

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **True Positive** | **False Positive** | **True Negative** | **False Negative** |
| **Kaos Tiruan** | 17 | 3 | 0 | 0 |
| **Kaos Orisinal** | 14 | 5 | 0 | 1 |
| **Sepatu Tiruan** | 13 | 7 | 0 | 0 |
| **Sepatu Orisinal** | 2 | 18 | 0 | 0 |
| **Topi**  **Tiruan** | 6 | 14 | 1 | 0 |
| **Topi Orisinal** | 18 | 2 | 0 | 0 |
| **Total** | **70** | **49** | **1** | **1** |

Tabel 4. Hasil *Accuracy* DenseNet121

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F1-Score** | **Accuracy** |
| **Kaos Tiruan** | 0.5458 | 0.8500 | 0.6667 |  |
| **Kaos Orisinal** | 0.6250 | 0.2500 | 0.3571 |  |
| **Sepatu Tiruan** | 0.4828 | 0.7000 | 0.5714 |  |
| **Sepatu Orisinal** | 0.4545 | 0.2500 | 0.3226 | **0.5253** |
| **Topi Tiruan** | 0.5278 | 0.9500 | 0.6786 |  |
| **Topi Orisinal** | 0.6000 | 0.1500 | 0.2400 |  |
| **Average** | 0.5397 | 0.5250 | 0.4747 |  |

### 4.2.5 Arsitektur Xception

Hasil pengujian menggunakan arsitektur MobileNetV2 didapatkan True Positif atau berhasil mendeteksi 62 citra *brand* orisinal atau tiruan. Diperoleh nilai *accuracy* sebesar 0,49%.

Tabel 4. *Confusion Matrix* Xception

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Kaos Tiruan** | **Kaos Orisinal** | **Sepatu Tiruan** | **Sepatu Orisinal** | **Topi Tiruan** | **Topi Orisinal** |
| **Kaos Tiruan** | 11 | 9 | 0 | 0 | 0 | 0 |
| **Kaos Orisinal** | 10 | 10 | 0 | 0 | 1 | 0 |
| **Sepatu Tiruan** | 0 | 0 | 15 | 5 | 0 | 0 |
| **Sepatu Orisinal** | 0 | 0 | 16 | 4 | 0 | 0 |
| **Topi Tiruan** | 0 | 0 | 0 | 1 | 13 | 6 |
| **Topi Orisinal** | 0 | 0 | 0 | 0 | 11 | 9 |

Tabel 4. Hasil *Confusion Matriks* Xception

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **True Positive** | **False Positive** | **True Negative** | **False Negative** |
| **Kaos Tiruan** | 11 | 10 | 9 | 39 |
| **Kaos Orisinal** | 10 | 10 | 1 | 32 |
| **Sepatu Tiruan** | 15 | 16 | 4 | 26 |
| **Sepatu Orisinal** | 4 | 6 | 16 | 27 |
| **Topi**  **Tiruan** | 13 | 7 | 12 | 47 |
| **Topi Orisinal** | 9 | 6 | 11 | 42 |
| **Total** | **62** | **55** | **53** | **213** |

Tabel 4. Hasil *Accuracy* Xception

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Precision** | **Recall** | **F1-Score** | **Accuracy** |
| **Kaos Tiruan** | 0.5238 | 0.5500 | 0.5366 |  |
| **Kaos Orisinal** | 0.5263 | 0.5000 | 0.5128 |  |
| **Sepatu Tiruan** | 0.4893 | 0.7500 | 0.5882 | **0.5023** |
| **Sepatu Orisinal** | 0.4000 | 0.2000 | 0.2667 |  |
| **Topi Tiruan** | 0.4517 | 0.6500 | 0.5909 |  |
| **Topi Orisinal** | 0.6000 | 0.4000 | 0.5143 |  |
| **Average** | **0.5126** | **0.5167** | **0.5016** |  |

### 4.2.6 Perbandingan Hasil Arsitektur CNN

Setelah dilakukan proses pengujian dengan lima arsitektur diperoleh akurasi dari setiap arsitektur, yaitu:

Tabel 4. Perbandingan Hasil Arsitektur

|  |  |
| --- | --- |
| Arsitektur | Akurasi |
| VGG16 | 49% |
| MobileNetV2 | 53% |
| ResNet50V2 | 49% |
| DenseNet121 | 52% |
| Xception | 50% |

Hasil akurasi dari lima arsitektur diatas menunjukkan akurasi yang tertinggi terdapat arsitektur MobileNetV2 dengan 53%.

Berdasarkan hasil training model dari lima arsitektur diatas, didapatkan kesimpulan bahwa *accuracy* tertinggi terdapat pada arsitektur MobileNetV2 dengan nilai *accuracy* sebesar 0,53% dan accuracy terendah terdapat pada arsitektur VGG16 dengan nilai *accuracy* sebesar 0.49%.

Gambar 4. Hasil Akurasi Pelatihan Model

## 4.3 Hasil Implementasi Aplikasi

Setelah didapatkan hasil *accuracy* dari pengujian model pada bab 4.2 selanjutnya proses pembuatan aplikasi android dilakukan dengan menggunakan model Arsitektur MobileNetV2 dari *Google Colab*, di *export* dalam format tflite dan dilakukan proses *load* model dalam Android Studio.Model yang telah di load dalam Android Studio diproses dalam pengkodean aplikasi mobile agar mendapatkan hasil klasifikasi dari input yang diberikan user. Berikut hasil dari klasifikasi *Brand Fashion* yang telah diproses dalam aplikasi mobile:

A picture containing diagram

Description automatically generated

Gambar 4. Tampilan Awal Splash Screen

Pada Gambar 4.2 merupakan tampilan awal saat menjalankan aplikasi mobile. Setelah membuka aplikasi akan ada *splash screen* yang berjalan selama 3 detik.

A picture containing icon

Description automatically generated

Gambar 4. Tampilan Pilihan Media

Setelah tampilan splash screen akan ada tampilan untuk pilihan jenis media yang akan digunakan untuk mengklasifikasi seperti pada Gambar 4.3. Terdapat perintah “Deteksi dengan kamera” dan “Deteksi dari Galeri”, sistem memberi perintah kepada user untuk memilih media yang digunakan dalam melakukan klasifikasi. Terdapat button kamera dan gallery yang akan tersambung dengan local akses smartphone user.

A pair of red shoes

Description automatically generated with low confidence

Gambar 4. Tampilan Klasifikasi



Gambar 4. Tampilan Deteksi dengan Kamera

Pada Gambar 4.4 merupakan hasil dari inputan user menggunakan deteksi dari *galery* Hasil menampilkan nama serta informasi mengenai objek yang di klasifikasi.

Jika user menekan tombol deteksi dengan kamera, maka akan masuk ke kamera *smartphone* seperti pada gambar 4.5dan terdapat beberapa menu dari hasil prediksi.

# BAB V PENUTUP

## 5.1 Kesimpulan

Berdasarkan hasil penelitian dan uraian pembahasan serta kajian temuan setelah dilakukan implementasi metode *deep learning* yaitu CNN atau *Convolution Neural Network* pada *brand fashion* didapatkan suatu kesimpulan bahwa metode CNN:

1. Dari pengujian lima arsitektur model CNN pada klasifikasi *brand fashion* berdasarkan kondisi barang orisinal atau tiruan yang mendapatkan hasil *accuracy* tinggi yaitu MobileNetV2 dengan *accuracy* 53%, *Precision 48%, Recall 49%, dan F1-Score 44%.*
2. Pengujian menggunakan arsitektur VGG16 dan ResNet50V2 mendapatkan hasil *accuracy* 49%.
3. Pengujian menggunakan arsitektur DenseNet121 dengan *accuracy* 52% dan untuk arsitektur Xception mendapatkan hasil *accuracy* 50%.

## 5.2 Saran

Adapun saran dari penulis pada penelitian yang telah dilakukan, yakni :

1. Memperbanyak lagi jumlah dataset atau *merk brand* yang akan digunakan dan memperbaiki tingkat akurasi.
2. Memperbaiki tampilan aplikasi serta lebih mempermudah user menggunakan aplikasi.

# DAFTAR PUSTAKA

Alwanda, M. R., Ramadhan, R. P. K., & Alamsyah, D. (2020). Implementasi Metode Convolutional Neural Network Menggunakan Arsitektur LeNet-5 untuk Pengenalan Doodle. *Jurnal Algoritme*, *1*(1), 45–56. https://doi.org/10.35957/algoritme.v1i1.434

Arther Sandag, G., Waworundeng Universitas Klabat, J., Arnold Mononutu, J., & -Minahasa Utara, A. (2021). Identifikasi Foto Fashion Dengan Menggunakan Convolutional Neural Network (CNN). *202.62.11.57*, *7*(2), 2021. <http://202.62.11.57/index.php/cogito/article/view/340>

Eka Putra, W. S. (2016). Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) pada Caltech 101. *Jurnal Teknik ITS*, *5*(1). https://doi.org/10.12962/j23373539.v5i1.15696

Rangkuti, A. H. (2014). Klasifikasi Motif Batik Berbasis Kemiripan Ciri dengan Wavelet Transform dan Fuzzy Neural Network. *ComTech: Computer, Mathematics and Engineering Applications*, *5*(1), 361. https://doi.org/10.21512/comtech.v5i1.2630

Wujaya, M. C., & Santoso, L. W. (2021). Klasifikasi Pakaian Berdasarkan Gambar Menggunakan Metode YOLOv3 dan CNN. *Jurnal INFA*, *9*(1), 2–7.

# LAMPIRAN

Link *Dataset Training, Validation, dan Test* pada Google Drive

https://drive.google.com/drive/folders/13PZX1MMNLSgi-LBdWbcjJ6JD\_ziHQCb8?usp=sharing

**Google Colaboratory**

Import Dataset

#Memuat semua gambar ke memori untuk pertama kali

##gambar = batch size x step(1200= 10x120)

#Memuat dataset pelatihan

IMAGE\_SIZE = 300

BATCH\_SIZE = 10

train\_dir = os.path.join('/content/drive/MyDrive/skripsi/bismillahbagus/train')

datagen = tf.keras.preprocessing.image.ImageDataGenerator(

horizontal\_flip=True,

brightness\_range=[0.2,0.8],

vertical\_flip=True,

rescale = 1./255,

)

train\_generator = datagen.flow\_from\_directory(

train\_dir,

target\_size=(IMAGE\_SIZE, IMAGE\_SIZE),

batch\_size=BATCH\_SIZE,

class\_mode='categorical',

subset='training')

val\_dir = os.path.join('/content/drive/MyDrive/skripsi/bismillahbagus/validation')

val\_generator = datagen.flow\_from\_directory(

val\_dir,

target\_size=(IMAGE\_SIZE, IMAGE\_SIZE),

batch\_size=BATCH\_SIZE,

class\_mode='categorical',

)

test\_dir =os.path.join('/content/drive/MyDrive/skripsi/bismillahbagus/test')

test\_generator = datagen.flow\_from\_directory(

test\_dir,

target\_size=(IMAGE\_SIZE, IMAGE\_SIZE),

batch\_size=BATCH\_SIZE,

class\_mode='categorical',

shuffle=False)

Menyimpan Label

IMG\_SHAPE = (300, 300, 3)

base\_model = tf.keras.applications.DenseNet121(input\_shape=IMG\_SHAPE,

include\_top=False,

weights='imagenet')

Ekstraks Fitur

base\_model.trainable = False

base\_model.summary()

Pembuatan Model

IMG\_SHAPE = (300, 300, 3)

from tensorflow.keras.models import Sequential

from tensorflow.python.keras.layers import Dense, Flatten, Dropout

model = Sequential()

# Membuat model dasar (base model) dari pre-trained model VGG16

base\_model = tf.keras.applications.Xception(input\_shape=IMG\_SHAPE,

include\_top=False,

weights='imagenet')

#Agar pembobotan pada layer tertentu tidak mengalami update pada saat training

for layer in base\_model.layers:

layer.trainable=False

model.add(base\_model)

model.add(Flatten())

model.add(Dense(1024, activation='relu'))

model.add(Dense(512, activation='relu'))

model.add(Dropout(0.2))

model.add(Dense(256, activation='relu'))

model.add(Dropout(0.2))

model.add(Dense(128, activation='relu'))

model.add(Dense(64, activation='relu'))

model.add(Dropout(0.2))

model.add(Dense(32, activation='relu'))

model.add(Dense(16, activation='relu'))

model.add(Dropout(0.2))

model.add(Dense(8, activation='relu'))

model.add(Dense(6, activation='softmax'))

model.compile("adam",

loss="categorical\_crossentropy",

metrics=["acc"])

model.summary()

Callback untuk menghindari Overfitting

import tensorflow.keras as kr

early\_stop = kr.callbacks.EarlyStopping(monitor='val\_loss', min\_delta=1e-3, patience=30,

verbose=1, mode='min', restore\_best\_weights=True)

Model fit dengan epochs 100

history = model.fit(train\_generator,

epochs=100,

validation\_data=val\_generator,

callbacks=[early\_stop])

Menggambarkan Hasil Grafik

plt.plot(history.history["acc"],label="Akurasi Pelatihan")

plt.plot(history.history["val\_acc"],label="Validasi Akurasi")

plt.legend()

plt.show()

plt.plot(history.history["loss"],label="Kesalahan Pelatihan")

plt.plot(history.history["val\_loss"],label="Validasi Kesalahan")

plt.legend()

plt.show()

Menggunakan Model

from tensorflow.keras import metrics

CATEGORIES = ["kaosfake", "kaosoriginal", "sepatufake", "sepatuoriginal", "topifake", "topioriginal"]

true\_classes = test\_generator.classes

class\_labels = CATEGORIES

import numpy

predictions = model.predict(test\_generator, steps=12)

import numpy as np

from sklearn.metrics import multilabel\_confusion\_matrix

y\_true = np.array(true\_classes)

y\_pred = np.argmax(predictions, axis=-1)

Confusion Matriks

from sklearn.metrics import classification\_report, confusion\_matrix

print(confusion\_matrix(y\_true, y\_pred))

Accuracy, Precesion, Recall, f1-Score

report = classification\_report(y\_true, y\_pred, target\_names=CATEGORIES, digits=4)

print(report)

Menyimpan Model dan Konversi ke tensorflowlite

saved\_model\_dir = 'save/model'

tf.saved\_model.save(model, saved\_model\_dir)

converter = tf.lite.TFLiteConverter.from\_saved\_model(saved\_model\_dir)

tflite\_model = converter.convert()

with open('nike.tflite', 'wb') as f:

f.write(tflite\_model)

# BIODATA PENULIS

|  |  |  |
| --- | --- | --- |
| I. Data Diri |  |  |
| * Foto | : |
| * Nama Lengkap | : | Mohamad Fajar Zulkarnain |
| * Tempat, Tanggal Lahir | : | Bojonegoro, 31 Juli 2000 |
| * Agama | : | Islam |
| * Alamat | : | Jl. Brawijaya, Ds. Widang, Kec. Widang, Kab. Tuban |
| * Jenis Kelamin | : | Laki – Laki |
| * Status | : | Belum Menikah |
| * Email | : | [mohamad.zulkarnain18@student.uisi.ac.id](mailto:mohamad.zulkarnain18@student.uisi.ac.id) |
| * No. Telp | : | 082131212741 |
| II. Jenjang Pendidikan |  |  |
| 2006 – 2012 | : | SDN Babat VIII |
| 2012 – 2015 | : | MTSN Model Babat |
| 2015 – 2018 | : | SMKN 2 Bojonegoro |