KLASIFIKASI NAMA BENDA DALAM AKSARA JAWA BERBASIS ANDROID MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK

SKRIPSI

Oleh :
FIO AZIZ NUGROHO
NIM. 15650124

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2021

KLASIFIKASI NAMA BENDA DALAM AKSARA JAWA BERBASIS ANDROID MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK

SKRIPSI

Diajukan kepada:
Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
FIO AZIZ NUGROHO
NIM. 15650124

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2021

LEMBAR PERSETUJUAN

KLASIFIKASI NAMA BENDA DALAM AKSARA JAWA BERBASIS ANDROID MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK

SKRIPSI

Oleh :
FIO AZIZ NUGROHO
NIM. 15650124

Telah Diperiksa dan Disetujui untuk Diuji
Tanggal : 22 Februari 2021

Dosen Pembimbing I	Dosen Pembimbing II
A'la Syauqi, M. Kom	Hani Nurhayati, M. T
NIP. 19771201 200801 1 007	NIP. 19780625 200801 2006

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
9NIP. 19740424 200901 1 008

LEMBAR PENGESAHAN

KLASIFIKASI NAMA BENDA DALAM AKSARA JAWA BERBASIS ANDROID MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK

SKRIPSI

Oleh :
FIO AZIZ NUGROHO
NIM. 15650124

Telah Dipertahankan Di depan Dewan Penguji
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Pada Tanggal 22 Februari 2021

Susunan Penguji			**Tanda tangan**
1. Penguji Utama	:	**Dr. Cahyo Crysdian** NIP. 19740424 200901 1 008	( )
2. Ketua Penguji	:	**Ainatul Mardhiyah, M.CS** NIP. 1986033020 160801 2 075	( )
3. Sekretaris Penguji	:	**A'la Syauqi, M. Kom** NIP. 19771201 200801 1 007	( )
4. Anggota Penguji	:	**Hani Nurhayati, M.T** NIP. 19780625 200801 2 006	( )

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan dibawah ini:

Nama	: Fio Aziz Nugroho
NIM	: 15650124
Fakultas/Jurusan	: Sains dan Teknologi/Teknik Informatika
Judul Skripsi	: Klasifikasi Nama Benda Dalam Aksara Jawa Berbasis Android Menggunakan Metode *Convolutional Neural Network*.

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya sendiri, buka merupakan pengambilalihan data, tulisan atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan Skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 22 Februari 2021

Yang membuat pernyataan,

Fio Aziz Nugroho
NIM.15650124

MOTTO

“Berusaha, Berdoa, Bersyukur, Bahagia”

HALAMAN PERSEMBAHAN

اَلْحَمْدُ لِلّٰهِ رَبِّ الْعَالَمِينَ

Puji syukur kehadirat Allah SWT, shalawat dan salam bagi Rasul-Nya

Dengan kerendahan hati, saya persembahkan karya yang sederhana ini kepada:

Kedua orang tua saya yang saya cintai Bapak Trijanto Ristiyono dan Ibu Trivina Ningsih, terima kasih banyak atas kesabarannya tidak pernah lelah dalam mendoakan dan memberikan dukungan juga semangat kepada saya.

Dosen pembimbing Bapak A'la Syauqi, M. Kom dan Ibu Hani Nurhayati, M. T yang dengan sabar membimbing jalanya penelitian skripsi ini serta seluruh dosen Teknik Informatika dan ustadz-ustadzah UIN Maulana Malik Ibrahim Malang yang telah memberikan ilmu kepada saya.

Untuk semua sahabatku dan teman-temanku seperjuangan Teknik Informatika angkatan 2015 dan seluruh keluarga besar Teknik Informatika UIN Maulana Malik Ibrahim Malang serta orang yang telah membantu dan namanya tidak bisa disebut satu persatu.

KATA PENGANTAR

بِسْمِ اللهِ الرَّحْمنِ الرَّحِيمِ

Assalamu'alaikum Warahmatullahi Wabarakatuh

Segala puji bagi Allah SWT, karena atas rahmad, hidayah serta karunia-Nya, penulis dapat menyelesaikan skripsi yang berjudul "Klasifikasi Nama Benda dalam Aksara Jawa Berbasis Android Menggunakan Metode *Convolutional Neural Network*" sebagai salah satu syarat untuk memperoleh gelar sarjana pada Program Studi Teknik Informatika jenjang Strata-1 Universitas Islam Negeri Maulana Malik Ibrahim Malang. Shalawat serta salam senantiasa terlimpahkan kepada Nabi Muhammad SAW, keluarga dan para sahabat yang telah membiimbing umat dari gelapnya alam jahiliyah menuju cahaya Islam yang diridoi Allah SWT.

Penulis menyadari adanya banyak keterbatasan yang penulis miliki, sehingga ada banyak pihak yang telah memberikan bantuan baik moril maupun materil dalam menyelesaikan penelitian ini. Maka dari itu dengan segenap kerendahan hati penulis mengucapkan terima kasih kepada:

1. Prof. Dr. Abdul Haris, M.Ag selaku Rektor Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang.
2. Dr. Sri Harini,M.Si, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang.
3. Dr. Cahyo Crysdian, Selaku Ketua Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang.
4. A'la Syauqi, M. Kom, Selaku Dosen Pembimbing I yang telah senantiasa meluangkan waktu untuk membimbing, mengarahkan penulis, dan memberikan masukan dalam penyusunan skripsi ini hingga selesai.
5. Hani Nurhayati, M.T**,** Selaku Dosen Pembimbing II yang telah senantiasa meluangkan waktu untuk membimbing, mengarahkan penulis, dan memberikan masukan dalam penyusunan skripsi ini hingga selesai.
6. Seluruh Dosen Jurusan Teknik Informatika Fakultas Sains dan Teknologi UIN Maulana Malik Ibrahim Malang yang memberikan lmu dan pengetahuan serta pengalaman
7. Segenap civitas akademik Jurusan Teknik Informatika Fakultas Sains dan Teknologi UIN Maulana Malik Ibrahim Malang.
8. Kedua orang tua serta seluruh keluarga besar penulis yang senantiasa mendukung dan memberikan doa hingga skripsi ini dapat terselesaikan.

9. Teman-teman Teknik Informatika angkatan 2015 yang senantiasa memberi motivasi dan berjuang bersama selama menjadi mahasiswa.
10. Semua pihak yang telah banyak membantu dalam penyusunan skripsi ini yang tidak bisa penulis sebutkan semuanya.

Penulis menyadari dalam karya ini masih banyak kekurangan. Oleh karena itu penulis selalu menerima segala kritik dan saran dari pembaca. Semoga karya ini bermanfaat bagi seluruh pihak.

Malang, 22 Februari 2021

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Nugroho, Fio Aziz. 2020. ***Klasifikasi Nama Benda Dalam Aksara Jawa Berbasis Android Menggunakan Metode Convolutional Neural Network***. Skripsi. Jurusan Teknik Informatika Fakultas Sains Dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
Pembimbing: (I) A'la Syauqi, M. Kom. (II) Hani Nurhayati, M.T.

Kata Kunci : Citra, *Deep Learning*, *Convolutional Neural Network (CNN)*, *Keras*, *obilenet.*

Indonesia terdapat berbagai macam suku dan budaya, setiap suku menggunakan bahasa yang berbeda setiap daerahnya, misalnya bahasa jawa yang sering digunakan dalam masyarakat jawa. Saat ini aksara jawa yang salah satu warisan budaya yang patut dilestarikan karena akibat modernisasi bahasa lain sehingga aksara jawa semakin jarang digunakan. Memanfaatkan teknologi diperlukan sebuah inovasi untuk mengenalkan kata benda dalam aksara jawa, salah satunya adalah dengan mengembangkan aplikasi yang interaktif. Dengan memanfaatkan pembelajaran *Deep Learning* terdapat salah satu metode *Convolutional Neural Network (CNN)* merupakan kelas dari jaringan saraf yang sering digunakan untuk menganalisis citra visual. Dalam penerapanya dapat digunakan untuk mengklasifikasi pada objek yang diamati berdasarkan karakteristik yang telah ditentukan. Data citra visual yang digunakan adalah nama benda disekitar lingkungan rumah yang dijadikan sebagai sumber data untuk dilatih menggunakan metode CNN. Pada penelitian ini memanfaatkan library keras, terdapat 20 kelas benda yang digunakan, dengan pengujian data testing sebanyak 2400 citra benda menunjukan tingkat akurasi sebesar 95% yang dinilai telah mampu melakukan identifikasi nama kata benda.

ABSTRACT

Nugroho, Fio Aziz. 2020. ***Classification of object names in Javanese script based on Android using the Convolutional Neural Network Method***. Thesis. Department of Informatics, Faculty of Science and Technology, Maulana Malik Ibrahim State Islamic University of Malang.
Pembimbing: (I) A'la Syauqi, M. Kom. (II) Hani Nurhayati, M.T.

Keywords: Image, *Deep Learning, Convolutional Neural Network (CNN), Hard, obilenet.*

In Indonesia, there are various kinds of tribes and cultures, each tribe using a different language for each region, for example the Javanese language which is often used in Javanese society. Currently Javanese script is one of the cultural heritages that should be preserved due to the modernization of other languages so that Javanese script is rarely used. Utilizing technology requires an innovation to introduce nouns in Javanese script, one of which is by developing interactive applications. By utilizing *deep learning*, there is a *Convolutional Neural Network (CNN)* method which is a class of neural networks that is often used to analyze visual images. In its application, it can be used to classify the observed object based on predetermined characteristics. Visual image data used are the names of objects around the home environment which are used as data sources to be trained using the CNN method. In this study, using a hard library, there are 20 classes of objects used, with testing data testing as many as 2400 images of objects showing an accuracy rate of 95% which is considered capable of identifying noun names.

نبذة مختصرة

نوجروهو ، فيو عزيز .2020 **.تصنيف أسماء الكائنات في النص الجاوي بناءً على نظام ذكري المظهر ، باستخدام طريقة الشبكة العصبية التلافيفية** .مقال .قسم المعلوماتية ، كلية العلوم والتكنولوجيا .جامعة الدولة الإسلامية مولانا مالك إبراهيم مالانج .

المستشار : آلاء سياوقي ، كمبيوتر رئيسي .و هاني نورحياتي ، تقنيات إتقان

الكلمات المفتاحية :صورة ، تعلم عميق ، شبكة عصبية تلافيفية ، صلبة ، موبيلنت

يوجد في إندونيسيا أنواع مختلفة من القبائل والثقافات ، كل قبيلة تستخدم لغة مختلفة لكل منطقة ، على سبيل المثال اللغة الجاوية التي غالبًا ما تستخدم في المجتمع الجاوي .يعد الخط الجاوي حاليًا أحد التراث الثقافي الذي يجب الحفاظ عليه بسبب تحديث اللغات الأخرى بحيث نادرًا ما يتم استخدام النص الجاوي .يتطلب استخدام التكنولوجيا ابتكارًا لإدخال الأسماء في النص الجاوي ، أحدها عن طريق تطوير تطبيقات تفاعلية .من خلال استخدام التعلم العميق ، توجد طريقة شبكة عصبية تلافيفية وهي فئة من الشبكات العصبية التي تستخدم غالبًا .لتحليل الصور المرئية .في تطبيقه ، يمكن استخدامه لتصنيف الكائن المرصود بناءً على خصائص محددة مسبقًا بيانات الصورة المرئية المستخدمة هي أسماء الكائنات الموجودة حول البيئة المنزلية والتي يتم استخدامها كمصادر للبيانات ليتم تدريبها باستخدام طريقة الشبكة العصبية التلافيفية .في هذه الدراسة ، باستخدام مكتبة صلبة هناك 20 فئة من الكائنات المستخدمة ، مع اختبار البيانات لما يصل إلى 2400 صورة لكائنات تظهر معدل ، دقة 95٪ والتي تعتبر قادرة على تحديد أسماء الأسماء

BAB I

PENDAHULUAN

1.1. Latar Belakang

Indonesia mempunyai banyak beragam suku dan budaya, sehingga dengan adanya keanekaragaman suku dan budaya yang ada, berdampak pada bahasa daerah di Indonesia banyak yang beragam. Beberapa bahasa daerah di Indonesia adalah bahasa daerahnya seperti bahasa Jawa, bahasa Sunda, Bahasa Madura, bahasa Batak dan sebagainya. Di Indoensia sendiri yang biasa menggunakan bahasa daerah terbanyak adalah bahasa Jawa dengan penutur sebanyak 75,5 juta jiwa (Utari, 2013).

Saat ini bahasa aksara jawa semakin jarang digunakan karena akibat dari modernisasi, karena menggunakan bahasa lain yang dianggap maju dan modern (Priyanto, 2019). Aksara jawa merupakan wsarisan budaya yang harus dilestarikan, jika tidak maka aksara jawa akan makin tidak dikenali oleh masyarakatnya sendiri termasuk generasi muda karena secara umum masyarakat sudah menggunakan bahasa resmi yaitu bahasa Indonesia. Menurut Nisa *et al.*, (2017) dalam penelitiannya menjelaskan bahwa aksara Jawa masih menjadi mata pelajaran muatan lokal di SDN Belimbing 4 Malang. Tetapi banyak siswa yang belum paham mengenai penulisan Aksara Jawa. Sehingga untuk pengenalan aksara jawa di perlukan sebuah inovatif untuk mempelajari Aksara Jawa, salah satunya adalah memanfatkan perkembangan teknologi saat ini. Teknologi diperlukan untuk mengembangkan aplikasi dalam mempermudah pengenalan aksara jawa, dengan pemanfaatan mulimedia yang lebih interaktif dan menarik, sehingga bisa memberikan rasa menyenangkan saat menggunakan aplikasi tersebut sehingga tidak membosankan (Supriyono *et al*, 2016).

Untuk mengatasi kesulitan dalam pengenalan aksara jawa dapat menggunakan cara bermain dan belajar, karena saat bermain anak cenderung dapat meningkatkan minat serta kemampuanya dalam memahami saat belajar (Fajarizka dan Rizkiantono, 2016). Saat menggunakan aplikasi tersebut diharapkan tidak membosankan. Sehingga dampak yang bisa ditimbulkan dalah dengan adanya aplikasi ini diharapkan dapat meningkatkan motivasi dalam memahami aksara jawa.

Seiring dengan perkembangan tekonologi saat ini terutama di dalam bidang *Artificial Intelligence (AI)* yang memiliki manfaat yang besar di segala sektor (Putra *et al*, 2016). Salah satu cabang dari AI adalah *Machine Learning (ML)* yaitu mesin yang dapat mengakses data yang ada dari perintah mesin itu sendiri, mempelajari data yang ada dan melakukan tugas tertentu, dengan mempelajari metode pada algoritma dan model statistik yang diterapkan. *Deep learning* termasuk dalam salah satu bidang dari *Machine Learning (ML)* memanfaatkan jaringan syaraf tiruan untuk implementasi permasalahan pada dataset yang mampu melakukan pembelajaran secara mendalam.

Convolutional Neural Network (CNN) termasuk dalam *Deep Neural Network* karena mempunyai kedalaman jaringan yang tinggi dan digunakan dalam data pada citra digital (Putra *et al*., 2016). Dengan menggunakan metode *Convolutional Neural Network (CNN)* bisa digunakan pada data gambar, sehingga bisa digunakan untuk mendeteksi atau mengenali object pada image pada citra digital. Diharapkan dengan menggunakan *Convolutional Neural Network (CNN)* bisa mendeteksi objek benda yang di gunakan untuk mengenali benda tersebut. Penerapan aplikasi edukatif ini diharapkan dapat mengenali nama benda dalam bahasa aksara jawa, didalamnya terdapat tantangan untuk mencari benda-benda yang ada disekitarnya, dengan

mengarahkan kamera hp ke benda yang di sebut dalam aplikasi menggunakan transliterasi aksara jawa.

1.2. Pernyataan Masalah

1. Bagaimana tingkat *accuracy, precision, recall, f-measure dan prediction* pada model yang dibangun menggunakan *Deptwise Separable Convolutional Neural Network* dalam mengklasifikasikan nama benda ?
2. Bagaimana hasil pengklasifikasian nama benda dengan menggunakan *Depthwise Separable Convolutional Neural Network* ?

1.3. Tujuan Penelitian

Tujuan dari penelitian ini adalah

1. Untuk mengukur *accuracy, precision, recall, f-measure* dan *prediction* pada model yang dibuat dengan menggunakan *Depthwise Separable Convolutional Neural Network,* untuk evaluasi model yang dihasilkan menggunakan *confusion matrix*.
2. Mengetahui hasil pengklasfikasian nama benda dengan menggunakan metode *Depthwise Separable Convolutional Neural Network.*

1.4. Manfaat Penelitan

1. Bagi peneliti, Dapat mengetahui tingkat akurasi yang dihasilkan model yang menggunakan metode *Depthwise Separable Convolutional Neural Network*.
2. Bagi developer, model dapat dikembangkan menjadi game edukatif sebagai media untuk melestarikan aksawa jawa.

3. Bagi pengguna, aplikasi dapat digunakan untuk mengetahui informasi nama kata benda menggunakan aksara jawa.

1.5. Batasan Masalah

Agar penelitian ini tidak keluar dari pokok permasalahan yang dirumuskan, maka penelitian ini dibatasi sebagai berikut :

1. Data citra gambar diambil dari internet.
2. Benda yang digunakan hanya sekitar lingkungan rumah yang biasa ditemui.
3. Kalimat masukan dari label data menggunakan menggunakan huruf latin.
4. Transliterasi aksara jawa diperoleh dari penelitian sebelumnya (Ayumitha, 2016)
5. Arsitektur yang digunakan menggunakan mobilenet.

1.6. Sistematika Penulisan

Penulisan laporan skripsi ini disusun sebagai berikut ini :

BAB I : PENDAHULUAN

Dalam bab ini berisi tentang latar belakang, pernyataan masalah, batasan masalah penelitian, tujuan dari penelitian, manfaat dilakukan penelitian, dan sistematika penelitian yang digunakan.

BAB II : STUDI PUSTAKA

Dalam bab ini membahas mengenai beberapa dasar teori yang menjadi acuan dalam proses pembuatan aplikasi yang akan dibuat,

BAB III : DESAIN DAN IMPLEMENTASI

Metodologi penelitian berisikan tentang percangan aplikasi untuk bias mendeteksi objek menggunakan *convolutional neural network*.

BAB IV : UJI COBA DAN PEMBAHASAN

Perancangan dan Analisa yang berisikan tentang analisa sistem pada aplikasi.

BAB V : KESIMPULAN DAN SARAN

Dalam bab terakir ini berisikan kesimpulan dan sara yang diambil dari hasil yang telah didapatkan dari pembahasan.

BAB II

STUDI PUSTAKA

Penggunaan metode *Convolutional Neural Network* telah banyak digunakan dalam penelitian tentang citra digital. Dalam penelitian ini, peneliti menggunakan metode *Convolutional Neural Network* untuk melakukan klasifikasi pada gambar benda dan menampilkan dalam aksara jawa yang berbasis mobile. Ada beberapa penelitian terkait yang menggunakan metode *Convolutional Neural Network,* diantaranya adalah:

Penelitian yang dilakukan oleh Albawi *et al.*, (2017) membahas tentang bagaimana dengan metode *Convolutional Neural Network* dapat memliki kinerja yang baik dalam permasalahan pada machine learning, khusunya pada aplikasi yang berhubungan dengan data gambar, seperti *imagenet*, *natural language processing* (nlp). Selain itu juga berkaitan dengan parameter yang mempengaruhi efisiensi *Convolutional Neural Network.*

Pada penelitian yang dilakukan oleh Nurfita (2018) mengimplementasikan pengenalan sidik jari menggunakan sistem pengenalan sidik jari yang menggunakan metode *Convolutional Neural Network* untuk menghasilkan tingakt akurasi semakin tinggi.

Melatih data dalam mengklasfikasikan 1,2 juta gambar dalam resolusi tinggi dalam konteks *ImageNet* LSVRC-2010 kedalam 1000 kelas yang berbeda, pada pengujian ini menghasilkan tingkat kesalahan top-1 dan top-5 sebesar 37,5 % dan 17,0 % yang jauh lebih baik dibandingkan sebelumnya. Pada varian model ini dimasukan

dalam kompetisi di ILSVRC-2012 dan meraih tingkat kesalahan pengujian top-5 terbaim sebesar 15,3 % dibandingkan dengan 26,2 % terbaik didunia. (Krizhevsky *et al*., 2017)

Menurut Howard *et al.* (2017) dalam penelitianya membahas tentang arsitektur mobilenet pada neural network. Mobilenet didasarkan pada arsitektur yang menggunakan D*epthwise Separable Convolution Neural Network* yang dapat diterapkan pada platfrom mobile. Mobilenet menghasilkan model ukuran yang tepat untuk aplikasi yang ingin digunakan. Model ini dapat menunjukan akurasi dan kinerja yang lebih baik dari model popular yang lain.

Selain itu penelitian yang dilakukan oleh Chen & Su (2018) membahas tentang bagaimana deep neural network dapat di implementasikan pada perangkat mobile, karena deep neural netwok dapat mencapai akurasi tinggi untuk pengenalan gambar namun memerlukan komputasi yang tinggi. Oleh karena itu mobilenet digunakan untuk mengurangi jumlah parameter dan biaya komputasi.

Menurut Gavai *et al.* (2017) pentingnya Machine Learning dapat melakukan klasifikasi objek dalam kelas spesifiknya. Menerapkan pada studi tentang bunga merupakan subjek penting dalam bidang botani yaitu kesamaan beragam antara spesies bunga, tekstur, warna bunga dan perbedaan antara spesies bunga yang sama. Untuk mendapatkan akurasi yang tinggi menggunakan *Mobilenet* pada tensorflow untuk membuat model yang bisa meminimalkan waktu dengan akurasi tinggi.

Pada penelitan yang dilakukan oleh Ayumitha (2016) membahas tentang aksara jawa yang bersifat silabik atau bersifat kesuku-kataan sehingga menjadi masalah pada aplikasi transliterasi. Dengan menggunakan *Decision Tree* dipilih

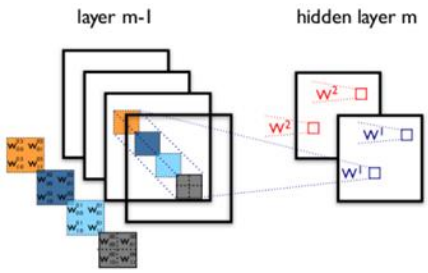
karena memudahkan dalam interprestasi yang diharapkan, mengalihkan alih aksara jawa dapat melakukan pengalihan 10 kata dengan tingkat akurasi diatas 90% dengan menggunakan metode D*ecision Tree.*

Bahasa Jawa merupakan bahasa lokal yang digunakan oleh Sebagian besar orang yang tinggal di pulau Jawa khususnya daerah Jawa Timur, Jawa Tengah dan Yogyakarta. Saat ini bahasa Jawa dan Aksara Jawa diajarkan di sekolah dasar dan sekolah menengah pertama. Tujuan dari penelitian ini adalah untuk menyediakan media pembelajaran interaktif dan permainan edukatif Aksara Jawa sebagai alat pembelajaran tambahan untuk siswa SD. Dengan hasil menunjukan 67 % responden puas bahwa aplikasi dapat meningkatkan antusiasme siswa untuk belajar aksara jawa. (Supriyono *et al*., 2016).

Citra atau gambar dalam Bahasa latin iamgo adalah adalah suatu representasi, kemiripan atau imitasi dari suatu objek atau benda. Citra dapat di bedakan menjadi citra tampak dan tidak tampak contohnya pada citra tampak seperti foto, gambar, lukisan, beda halnya dengan citra tak tampak data gambar dalam file (citra digital) yang dapat di representasikan pada komputer. Dalam bidang komputer terdapat 3 bidang studi yang berkaitan dengan data citra yaitu Grafika komputer, pengolahan citra, dan pengenalan pola. Dalam citra digital dapat didefinisikan sebagai fungsi 2 variale yaitu f(x,y) dimana x dan y adalah koordinat spasial dan nilai f(x,y) intensitas pada koordinat tersebut, pengolahan citra merupakan pemrosesan sebuah citra dengan proses numerik matrik yang di proses pada masing-masing pixel (Gazali *et al*., 2012).

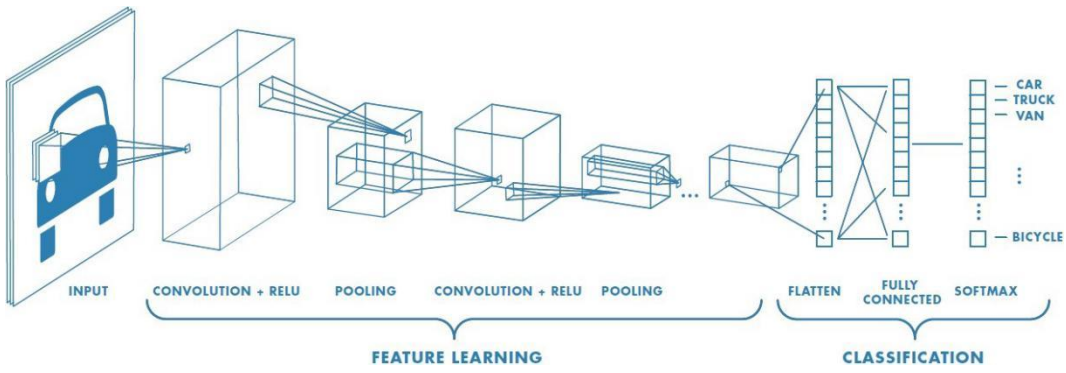
Convolutional Neural Network(CNN) merupakan salah satu pengembangan dari *Multi Layer Perceptron* (MLP) yang didesain untuk melakukan pengolahan data

dua dimensi. CNN termasuk dalam jenis Deep Neural Network karena kedalaman jaringan yang tinggi dan banyak di aplikasikan pada data citra. Cara kerja CNN memiliki kesamaan dengan MLP namun didalam CNN di setiap neuronnya dipresentasikan dalam bentuk dua dimensi, tidak seperti MLP yang setiap neuronya hanya terdapat satu dimensi. Karena di CNN mempunyai sifat konvolusi maka hanya dapat digunakan pada data yang memiliki nilai struktur data dua dimensi. Contoh konvolusi citra pada CNN (Putra *et al*., 2016) daapt dilihat pada Gambar 2.1.



Gambar 2.1. Proses Konvolusi Pada CNN

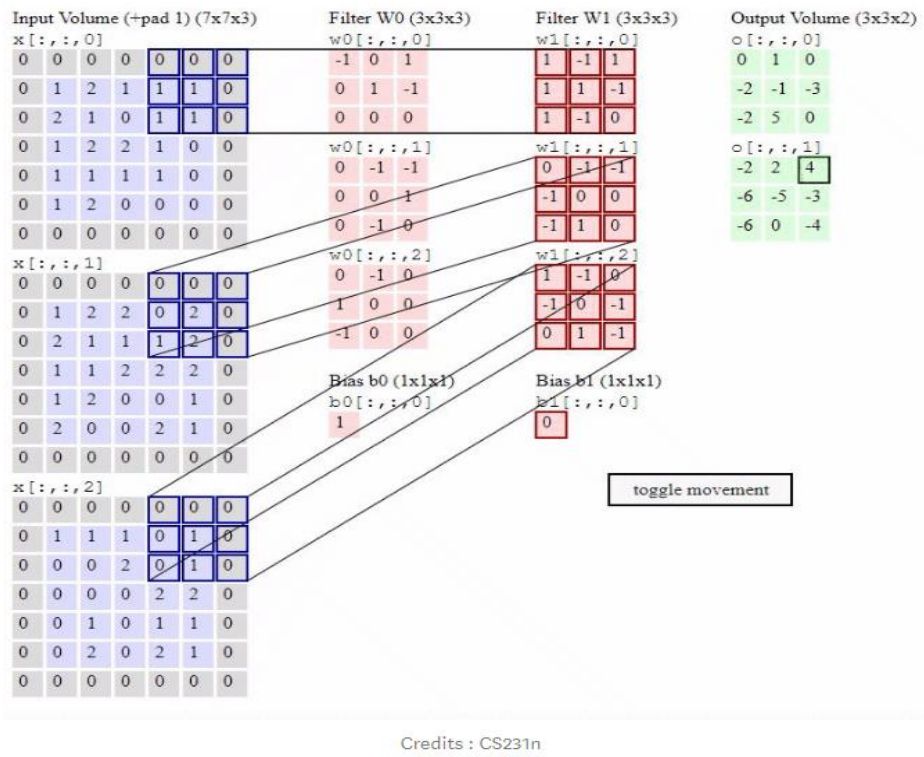
Arsitektur CNN terdiri dari beberapa layer yaitu *convolutional layer*, *pooling layer* dan *fully-connected layer*. Contoh arsitektur CNN dapat dilihat pada Gambar 2.2. berikut ini:



Gambar 2.2. Arsitektur Pada CNN

Convolutional Layer adalah tahap pada arsitektur CNN. Tahap ini melakukan operasi konvolusi pada *output* dari layer sebelumnya terdiri dari neuron yang ada lalu

membentuk sebuah filter dengan panjang dan tinggi (pixel). Layer pertama pada feature extraction layer adalah conv dengan ukuran 128 x 128 x 3 atau Panjang 128 pixel, tinggi 128 pixel, dan tebal/jumlah 3 buah sesuai dengan channel dari image tersebut. Filter akan bergeser keseluruh bagian dari gambar dengan begitu akan menghasilkan output yaitu *activation map* atau *feature map*. Didalam *convolutional Layer* terdapat stride yang merupakan parameter yang menentukan berapa jumlah pergeseran filter, jika nilainya 4 maka conv filter akan bergeser sebnyak 4 pixel secara horizontal maupun vertical. *Convolutional Layer* (Putra *et al*., 2016). Dapat dilihat pada Gambar 2.3. berikut ini:



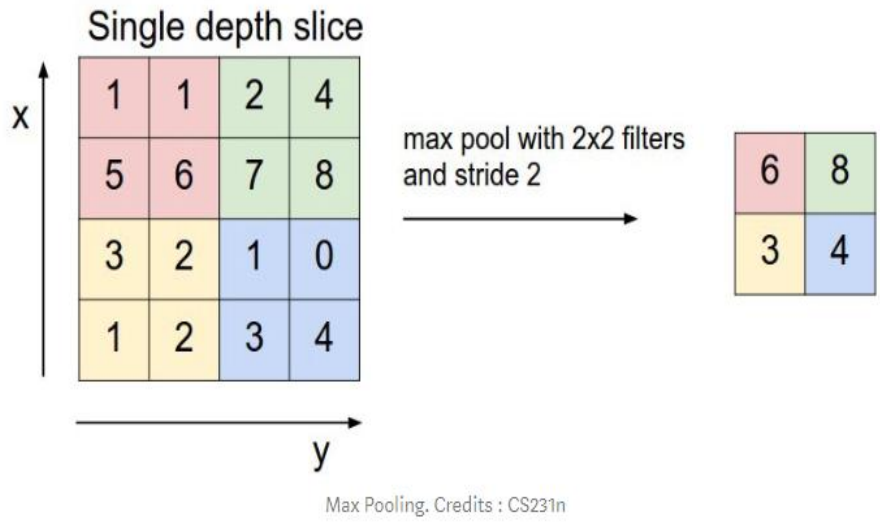
Gambar 2.3. Convolutional Layer

Untuk menghitung dimensi dari feature map bias menggunakan rumus berikut ini :

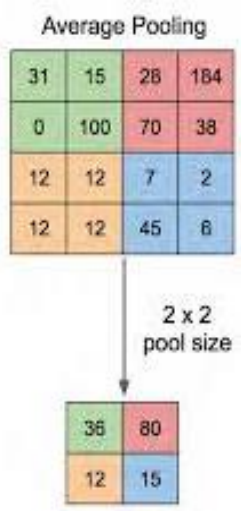
$$output = \frac{W - N + 2P}{S} + 1$$

- W = Panjang/Tinggi Input
- N = Panjang/Tinggi Filter
- P = Zero Padding
- S = Stride

Pooling layer berada setelah konvolusi layer, pooling terdiri dari sebuah filter dengan ukuran dan stride tertentu yang akan bergeser pada seluruh area *feature map*, pooling yang digunakan menggunakan *Average Pooling* yang akan memilih nilai rata-ratanya jika *Max Pooling* 4x4 dengan stride 4 maka setiap pergeseran filter nilai max nya adalah 4x4 (Putra *et al*., 2016). Proses pooling dapat dilihat pada Gambar 2.4. berikut ini:

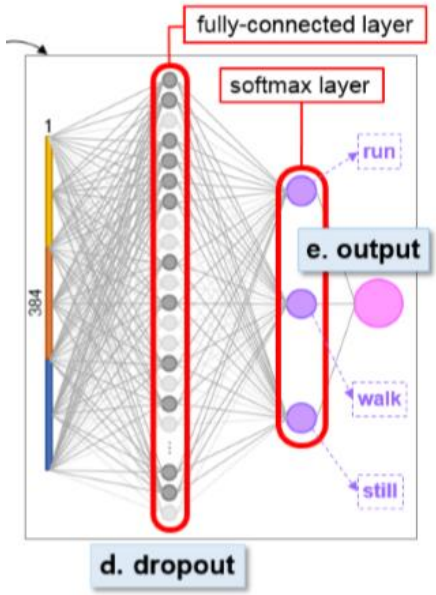


Gambar 2.4. Pooling Layer



Gambar 2.5. Average Pooling

Feature map yang telah dihitung akan menghasilkan multidimension array maka harus dilakukan flatten tau reshape feature map menjadi sebuah vector agar bias digunakan sebagai input dari *fully-connected layer* (Putra *et al*., 2016).



Gambar 2.6. Fully-Connected Layer

Tensorflow merupakan framework machine learning yang dapat digunakan dalam skala besar yang di kembangkan oleh google brain mendukung beberapa bahasa pemograman dan mendukung semua sistem operasi, tensorflow dikembangkan bertujuan untuk melakukan penelitian pada deep neural network yang dapat melatih model dataset yang berukuran besar, tensorflow juga bias menggunakan GPU untuk training secara efisien. Selain itu dalam penggunaanya dapat melakukan perhitungan pada matriks multidimensi. Sehingga pada penelitian ini akan menggunakan tensorflow untuk membuat data model.

Aksara Carakan dan Aksara Pasangannya, Aksara carakan yang digunakan dalam ejaan Bahasa jawa terdiri dari 20 aksara pokok yang bersifat kesukuan mempunyai aksara pasangan digunakan untuk menghubungkan suku kata tertutup konsonan dengan suk kata berikutnya.

Aksara Murda dan Aksara Pasangannya, Aksara murda biasa dipakai untuk menuliskan nama orang, nama lembaga, geografi dan nama gelar selain itu digunakan untuk menulis kata yang menggunakan huruf kapital di awal penulisanya dan awal paragraph atau kata.

Aksara suara biasa digunkakan untuk menuliskan aksara vokal yang menjadi saku kata dan tidak bias dijadikan aksara pasangan, selain itu bias juga digunakan untuk meulis suatu kata dari Bahasa asing untuk memepertegas objek pelafalan.

Aksara Rekanan dan Aksara Pasangannya, Aksara rekaan dapat dipakai dalam menuliskan aksara konsonan. Aksara rekan sendiri merupakan aksara yang digunakan untuk menulis huruf serapan dari bahasa asing.

BAB III

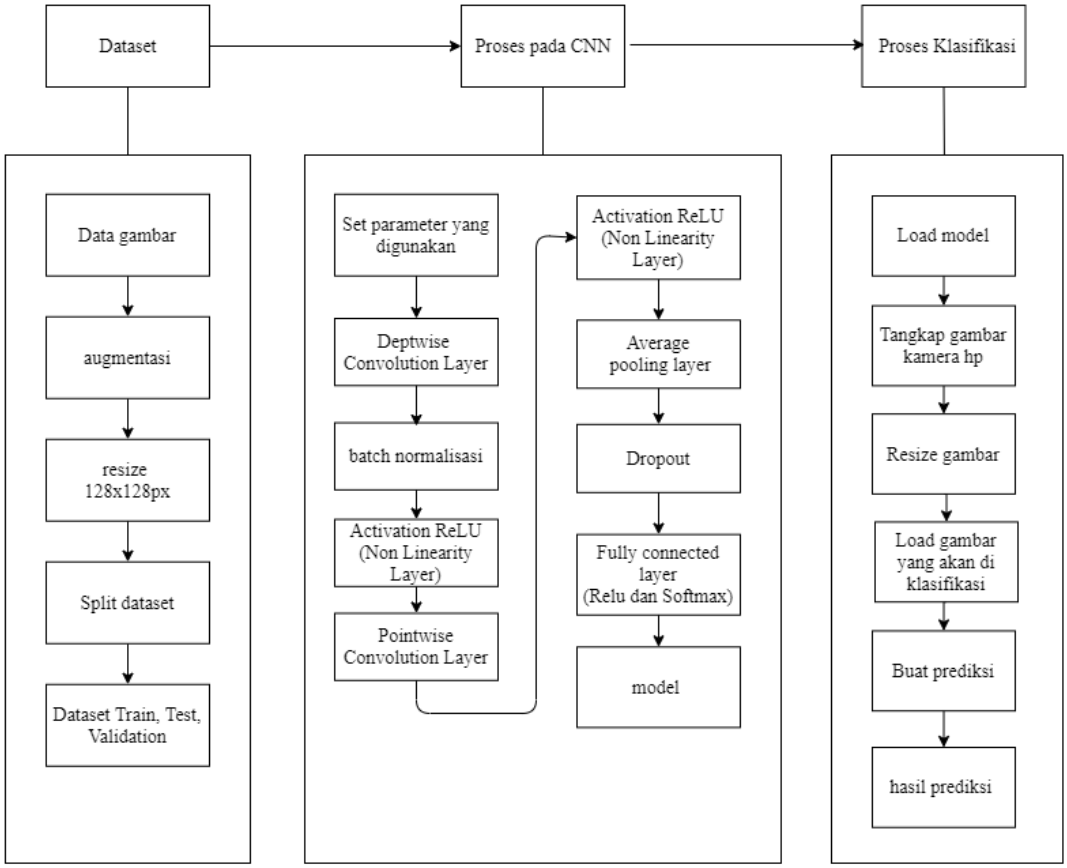
DESAIN DAN IMPLEMENTASI

3.1. Objek Penelitian

Pada penelitian ini akan dikembangkan sebuah aplikasi berbasis mobile android untuk melakukan klasifikasi pada benda dan menampilkan dalam aksara jawa menggunakan metode *neural network.* Dalam aplikasi ini pengguna akan mencari benda-benda yang ada disekitarnya. Aplikasi akan menampilkan pertanyaan nama benda yang akan dicari menggunakan aksara jawa.

3.2. Desain Sistem

Dalam penelitian ini terdiri dari beberapa langkah, dimana dalam prosesnya terdapat dua bagian yaitu proses pelatihan data dan proses pengenalan gambar. Pada proses pelatihan data menggunakan metode *Convolutional Neural Network* dengan mengimplementasikan arsitektur *Mobilenet*. Sedangkan dalam proses pengenalan gambar akan dilakukan resize dan thresholding pada inputan citra gambar, dan output yang akan dihasilkan adalah berupa hasil kata benda yang sudah di latih sebelumnya. Desain aplikasi dapat dilihat pada Gambar 3.1.



Gambar 3.1. Desain Sistem

3.3. **Pengumpulan Data**

Pada proses pelatihan data yang akan digunakan pada setiap kelas data gambar yang akan digunakan terdapat 20 jenis kelas benda. Bentuk dan jenis yang digunakan adalah gambar benda umum yang ada disekitar rumah. Gambar tersebut mempunyai format JPG, data yang digunakan untuk klasifikasi diperoleh dari *scraping* gambar dari internet. Selanjutnya setiap jenis benda dimasukan kedalam setiap kelas folder yang berisi nama-nama benda tersebut. Jumlah total data yang akan digunakan adalah sebanyak 12.000 gambar, Gambar benda yang diambil terdiri dari 20 kelas jenis benda dimana setiap jenis diambil 600 foto pada setiap kelas sehingga total gambar 20 x 600 terdapat 12.000 total gambar benda. Jumlah data yang digunakan sebagai pelatihan

yaitu sebanyak 7.200 gambar, untuk validasi 2400, dan testing 2400 . Dalam penelitian ini data gambar yang diambil diseleksi dengan beberapa kriteria sebagai berikut :

a. Data diseleksi agar menampilkan gambar yang dipilih tidak bercampur dengan benda lain.
b. Untuk meminimalisir waktu dan beban gpu maka gambar di kompres resolusinya menjadi ukuran 128x128 pixel.
c. Setiap kelas benda diisi dengan 600 buah gambar.
d. Dilakukan augmentasi citra pada semua kelas benda.

Berikut ini adalah dataset yang akan digunakan dalam pelatihan, nama benda menggunakan Bahasa jawa, dapat dilihat pada Tabel 3.1.

Tabel 3.1. Dataset

No	Nama benda (Bahasa Indonesia)	Nama Benda (Bahasa Jawa)	Jumlah gambar	Ukuran gambar (semua di kompress ukuran sama)
1	Botol	Botol	600	128 x 128 px
2	Garpu	Garpu	600	128 x 128 px
3	Gayung	Cidhuk	600	128 x 128 px
4	Cangkir	Cangkir	600	128 x 128 px
5	Jam tangan	Jam tangan	600	128 x 128 px
6	Kaos kaki	Kaos sikil	600	128 x 128 px
7	Kursi	Kursi	600	128 x 128 px
8	Mangkuk	Mangkuk	600	128 x 128 px

9	Meja	Méja	600	128 x 128 px
10	Piring	Piring	600	128 x 128 px
11	Bolpoin	Pulpén	600	128 x 128 px
12	Sandal	Sendal	600	128 x 128 px
13	Sendok	Séndhok	600	128 x 128 px
14	Sepatu	Sepatu	600	128 x 128 px
15	Tas	Tas	600	128 x 128 px
16	Tv	Tipi	600	128 x 128 px
17	Ember	émbér	600	128 x 128 px
18	Gitar	Gitar	600	128 x 128 px
19	Dompet	Dompét	600	128 x 128 px
20	Lemari	Lemari	600	128 x 128 px

3.4. **Preprosesing dan Augmentation Gambar**

Pada tahap ini gambar akan dilakukan penyesuaian yang dibutuhkan sebelum melanjukan pada tahap pelatihan data untuk membuat model. Seperti melakukan penyesuaian resolusi, pencahayaan, rotasi pada data gambar.

3.4.1. Resize Gambar

Pada tahap ini gambar akan di kompres menjadi ukuran 128 x 128 px, ukuran tersebut dipilih untuk menysesuaikan dengan spesifikasi komputer.

Tabel 3.2. Resize Gambar.

Sebelum (ukuran random)	Sesudah (ukuran 128x128)
	
Random px	128 x 128 px

3.4.2. Pengaturan pencahayaan

Pencahayaan pada gambar akan disesuaikan agar cahaya pada benda dan latar belakang tidak menyatu.

Tabel 3.3 Pengaturan Brightnes.

Sebelum	Sesudah
	
128 x 128 px	128 x 128 px

3.4.3. Augmentation

Flip dan rotasi pada gambar bertujuan untuk memvariasikan dataset

Tabel 3.4 Augmentasi.

Sebelum	Sesudah
	 
128 x 128 px	128128 px

3.5. Pelatihan Data Model

Proses pelatihan data adalah proses untuk melatih dataset yang telah disiapkan. Menurut Howard *et al*. (2017:1), Mobilenet merupakan pengembangan dari *deep neural network* yang di gunakan untuk membuat model data yang ringan, efisien dan akurasi yang tinggi sehingga cocok diimplementasikan ke aplikasi mobile. Dalam jurnal oleh Gavai *et al*. (2017:1), Pelatihan data model bertujuan untuk mengklasifikasi objek menggunakan *convolutional neural network* yang menggunakan multilayer konvolusi untuk menghasilkan fitur dan mengkombinasi layer secara otomatis sehingga untuk membuat data model menggunakan bantuan dari library keras tensorflow karena bersifat open source dan perhitungan aritmatika sehingga cocok digunakan dalam machine learning. Konfigurasi pelatihan data adalah sebagai berikut :

3.5.1. Parameter pelatihan data

Parameter diperlukan untuk menentukan pengaturan yang digunakan untuk mendapatkan hasil model yang baik, berikut ini adalah beberapa parameter yang akan digunakan untuk proses pelatihan data.

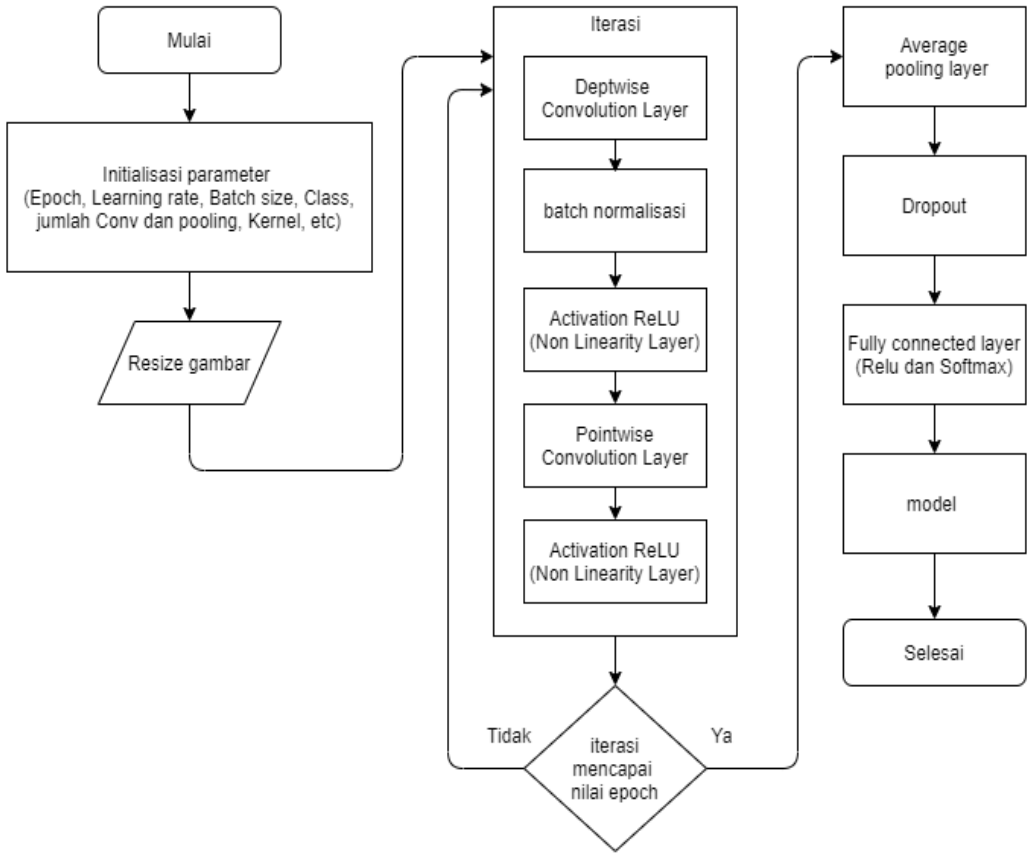
a. Epoch yang digunakan adalah Total sampel / batch.
b. Input shape = 128 x 128 px
c. Batch size = 64
d. Pooling yang digunakan = average
e. Kelas yang digunakan = 20
f. Learning rate = 0.001
g. Class mode = categorical crossentropy
h. Weight = imagenet
i. Simpan model graph = data.h5
j. Tensorboard log = epoch accuracy dan epoch loss

3.5.2. Depthwise Separable Convolution

Dalam mobilenet menggunakan konvolusi *Deptwise Separable Convolution* merupakan pengembangan dari *Convolutional Neural Network* (CNN). Terdapat beberapa tahapan dalam membangun model, dapat dilihat sebagai berikut :

a. Convolution Deptwise Layer
b. Convolution Pointwise Layer
c. Average Pooling
d. Fully Connected layer

Dalam membangun model, pelatihan data menggunakan metode C*onvolutional Neural Network* (CNN) dengan arsitektur mobilenet dapat dapat dilihat pada Gambar 3.2.

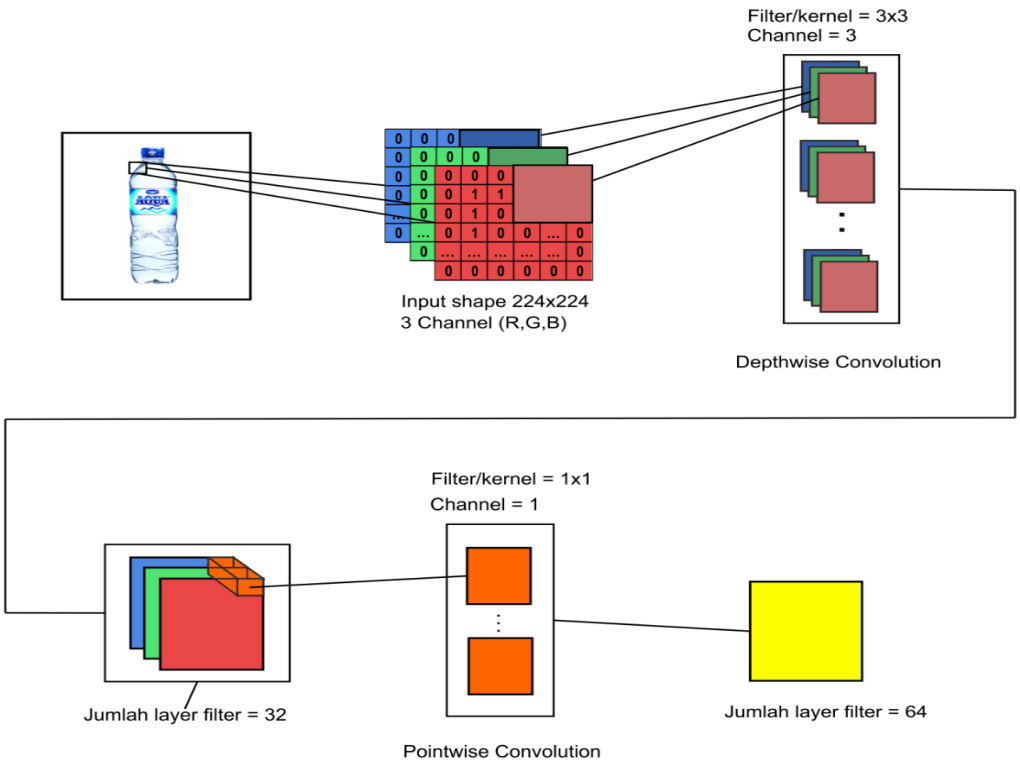


Gambar 3.2. Desain CNN

3.5.3. Desain CNN pada mobilenetv1

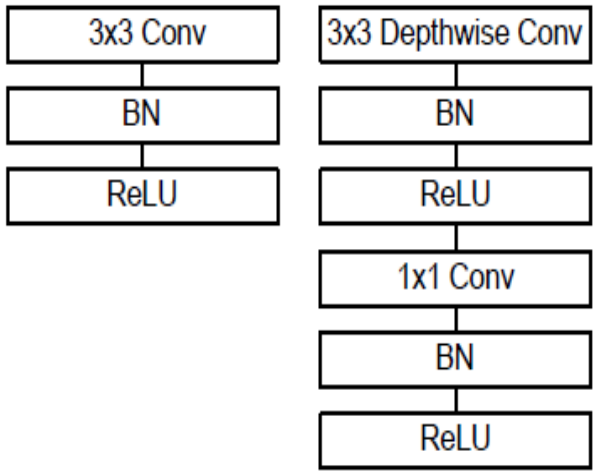
Mobilenet menggunakan *Depthwise separable convolution* digunakan untuk mengolah citra data image output dari CNN dapat digunakan untuk mendeteksi dan mengenali object pada sebuah gambar. Pada objek yang dimana didalamnya terdapat dua layer proses yaitu konvolusi *depthwise* dan konvolusi *pointwise*. Pada konvolusi *depthwise* menggunakan sebuah filter pada setiap chanel (input depth), Sedangkan

menurut .Howard *et al*. (2017:2-3). pada konvolusi *pointwise* menggunakan konvolusi sederhana 1x1 filter, lalu digunakan untuk membuat kombinasi linear pada depthwise layer. Didalam mobilenet menggunakan kombinasi layer dari batch normalisasi dan ReLU nonlinearBerikut ini adalah desain sistem pada mobilenet pada Gambar 3.3.



Gambar 3.3. Proses Konvolusi Mobilenet

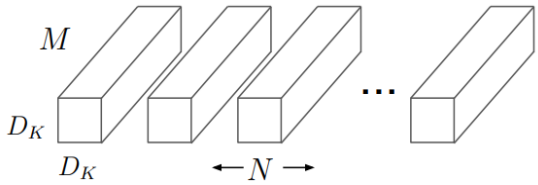
Perbedaan antara konvolusi standar dan konvolusi *Depthwise separable convolution*, dapat dilihat pada gambar alur Gambar 3.4.



Gambar 3.4. Konvolusi Standar dan Deptwise Separable

a. Konvolusi standar

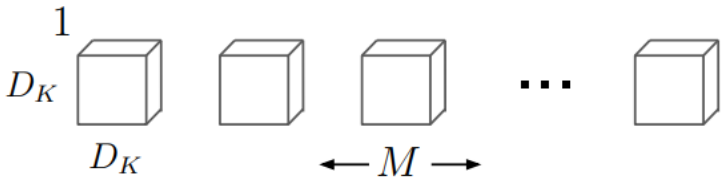
Pada proses konvolusi standar menggunakan filter yang sama pada semua chanel, dapat dilihat pada Gambar 3.5.



Gambar 3.5. Filter Konvolusi Standar

b. Konvolusi Depthwise

Pada proses konvolusi depthwise menggunakan sebuah filter tersendiri pada setiap chanel (input depth), dapat dilihat pada Gambar 3.6.



Gambar 3.6. Filter Konvolusi Depthwise

Penjelasan konvolusi depthwise dapat dijelaskan sebagai berikut :

$$\hat{\mathbf{G}}_{k,l,m} = \sum_{i,j} \hat{\mathbf{K}}_{i,j,m} \cdot \mathbf{F}_{k+i-1,l+j-1,m}$$

Keterangan :

$\hat{\mathbf{K}}$ = Kernel / filter ($D_K \times D_K \times M$)

D_K = Input width x Height

M = Input Chanel (Input depth)

F = Filter map

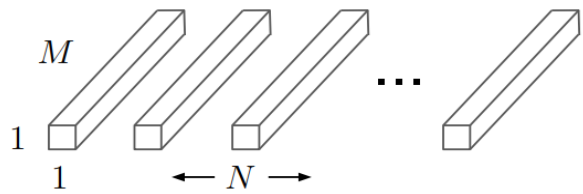
$\hat{\mathbf{G}}$ = Feature map

Sehingga dalam penulisan dalam perhitungan komputer adalah sebagai berikut :

$D_K \times D_K \times M \times D_F \times D_F$

c. Konvolusi Pointwise

Pada proses konvolusi pointwise menggunakan filter yang sama 1x1 pada semua chanel, dapat dilihat pada Gambar 3.7.



Gambar 3.7. Filter Konvolusi Pointwise

Kombinasi dari konvolusi deptwise dan pointwise menjadi depthwise separable convolution dapat dihitung seperti dibawah ini :

$D_K \times D_K \times M \times D_F \times D_F + M \times N \times D_F \times D_F$

Keterangan

D_K = Input width x Height

M = Input Chanel (Input depth)

F = Output Chanel (Output depth)

N = Output Chanel (Output depth)

Arsitektur mobilenet dapat dilihat pada Tabel 3.2.

Tabel 3.5 Arsitektur Mobilenet

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

3.5.4. Model Arsitektur CNN Depthwise Separable

Menentukan model CNN Depthwise Sparable pada Gambar 3.8 menunjukan arsitektur menggunakan 13 proses konvolusi layer, setiap layer dilakukan proses batch normalisasi dan aktivasi RELU. Batch normalisasi bertujuan untuk mempercepat proses pada saat pelatihan data karena pada neural network akan bekerja lebih cepat jika distribusi data masukan tetap sama, pada saat proses pelatihan batchnormalisasi akan menormalkan nilai input dan skala pada awal fitur layer. Sedangkan untuk proses aktivasi RELU bertujuan untuk mengubah nilai minus pada sebuah matrik dari hasil konvolusi. Lalu untuk pooling menggunakan global pooling pada semua layer lalu di

drop layer yang digunakan sebesar 0,5, untuk mengklasifikasikan menggunakan aktivasi softmax dengan total 20 kelas dapat dilihat pada Gambar 3.8.

```
model = Sequential()

model.add(InputLayer(input_shape=(128,128,3), name='InputLayer'))

model.add(ZeroPadding2D(((0,1),(0,1)), name='Conv2D_1_Pad'))

model.add(Conv2D(32, kernel_size=(3,3),
strides=(2,2),name='Conv2D_1'))

model.add(BatchNormalization(axis=1, name='Conv2D_1_Bn'))

model.add(Activation(('relu'), name='Conv2D_1_Relu'))



model.add(DepthwiseConv2D(kernel_size=(3,3),strides=(1,1),
padding='same', depth_multiplier=1, use_bias=False,
name='ConvDw2D_1'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_1_Bn'))

model.add(Activation(('relu'), name='ConvDw2D_1_Relu'))

model.add(Conv2D(64, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_1' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_1_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_1_Relu'))



model.add(ZeroPadding2D(((0,1),(0,1)), name='Conv2D_2_Pad'))

model.add(DepthwiseConv2D(kernel_size=(3,3),
strides=(2,2),depth_multiplier=1, use_bias=False,
name='ConvDw2D_2'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_2_Bn'))

model.add(Activation(('relu'), name='ConvDw2D_2_Relu'))

model.add(Conv2D(128, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_2' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_2_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_2_Relu'))



model.add(DepthwiseConv2D(kernel_size=(3,3), padding='same',
depth_multiplier=1, use_bias=False, strides=(1,1),
name='ConvDw2D_3'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_3_Bn'))
```

```
model.add(BatchNormalization(axis=1, name='ConvPw2D_3_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_3_Relu'))



model.add(ZeroPadding2D(((0,1),(0,1)), name='Conv2D_4_Pad'))

model.add(DepthwiseConv2D(kernel_size=(3,3), strides=(2,2),
depth_multiplier=1, use_bias=False, name='ConvDw2D_4'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_4_Bn'))

model.add(Activation(('relu'), name='ConvDw2D_4_Relu'))

model.add(Conv2D(256, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_4' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_4_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_4_Relu'))

model.add(DepthwiseConv2D(kernel_size=(3,3), padding='same',
depth_multiplier=1, use_bias=False, strides=(1,1),
name='ConvDw2D_5'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_5_Bn'))

model.add(Activation(('relu'), name='ConvDw2D_5_Relu'))

model.add(Conv2D(256, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_5' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_5_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_5_Relu'))



model.add(ZeroPadding2D(((0,1),(0,1)), name='Conv2D_6_Pad'))

model.add(DepthwiseConv2D(kernel_size=(3,3), strides=(2,2),
depth_multiplier=1, use_bias=False, name='ConvDw2D_6'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_6_Bn'))

model.add(Activation(('relu'), name='ConvDw2D_6_Relu'))

model.add(Conv2D(512, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_6' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_6_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_6_Relu'))

model.add(DepthwiseConv2D(kernel_size=(3,3), padding='same',
depth_multiplier=1, use_bias=False, strides=(1,1),
name='ConvDw2D_7'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_7_Bn'))
```

```
model.add(Activation(('relu'), name='ConvDw2D_7_Relu'))

model.add(Conv2D(512, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_7' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_7_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_7_Relu'))


model.add(DepthwiseConv2D(kernel_size=(3,3), padding='same',
depth_multiplier=1, use_bias=False, strides=(1,1),
name='ConvDw2D_8'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_8_Bn'))

model.add(Activation(('relu'), name='ConvDw2D_8_Relu'))

model.add(Conv2D(512, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_8' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_8_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_8_Relu'))


model.add(DepthwiseConv2D(kernel_size=(3,3), padding='same' ,
depth_multiplier=1, use_bias=False, strides=(1,1),
name='ConvDw2D_9'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_9_Bn'))

model.add(Activation(('relu'), name='ConvDw2D_9_Relu'))

model.add(Conv2D(512, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_9' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_9_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_9_Relu'))


model.add(DepthwiseConv2D(kernel_size=(3,3), padding='same',
depth_multiplier=1, use_bias=False, strides=(1,1),
name='ConvDw2D_10'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_10_Bn'))

model.add(Activation(('relu'), name='ConvDw2D_10_Relu'))

model.add(Conv2D(512, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_10' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_10_Bn'))
```

```
model.add(BatchNormalization(axis=1, name='ConvDw2D_10_Bn'))

model.add(Activation(('relu'), name='ConvDw2D_10_Relu'))

model.add(Conv2D(512, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_10' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_10_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_10_Relu'))



model.add(DepthwiseConv2D(kernel_size=(3,3), padding='same',
depth_multiplier=1, use_bias=False, strides=(1,1),
name='ConvDw2D_11'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_11_Bn'))

model.add(Activation(('relu'), name='ConvDw2D_11_Relu'))

model.add(Conv2D(512, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_11' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_11_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_11_Relu'))



model.add(ZeroPadding2D(((0,1),(0,1)), name='Conv2D_12_Pad'))

model.add(DepthwiseConv2D(kernel_size=(3,3), strides=(2,2),
depth_multiplier=1, use_bias=False, name='ConvDw2D_12'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_12_Bn'))

model.add(Activation(('relu'), name='ConvDw2D_12_Relu'))

model.add(Conv2D(1024, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_12' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_12_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_12_Relu'))

model.add(DepthwiseConv2D(kernel_size=(3,3), padding='same',
depth_multiplier=1, use_bias=False, strides=(1,1),
name='ConvDw2D_13'))

model.add(BatchNormalization(axis=1, name='ConvDw2D_13_Bn'))

model.add(Activation(('relu'), name='ConvDw2D_13_Relu'))

model.add(Conv2D(1024, kernel_size=(1,1), strides=(1,1),
name='ConvPw2D_13' ))

model.add(BatchNormalization(axis=1, name='ConvPw2D_13_Bn'))

model.add(Activation(('relu'), name='ConvPw2D_13_Relu'))
```

```
model.add(GlobalAveragePooling2D())
#fllten
model.add(Dropout(0.5))
model.add(Dense(20, activation = 'softmax'))
```

Gambar 3.8. Tampilan Arsitektur CNN Depthwise Separable

3.5.5. Blok Summary CNN Depthwise Separable

Pada Gambar 3.8 merupakan blok konvolusi yang dihasilkan menggunakan python pada Gambar 3.9 sehingga menghasilkan beberapa blok konvolusi yang akan dilakukan pelatihan data.

Layer (type)	Output Shape	Param #
input_3 (InputLayer)	(None, 128, 128, 3)	0
zero_padding2d_11 (ZeroPaddi	(None, 129, 129, 3)	0
conv2d_29 (Conv2D)	(None, 64, 64, 32)	864
batch_normalization_55 (Batc	(None, 64, 64, 32)	128
activation_55 (Activation)	(None, 64, 64, 32)	0
depthwise_conv2d_27 (Depthwi	(None, 64, 64, 32)	288
batch_normalization_56 (Batc	(None, 64, 64, 32)	128
activation_56 (Activation)	(None, 64, 64, 32)	0
conv2d_30 (Conv2D)	(None, 64, 64, 64)	2048
batch_normalization_57 (Batc	(None, 64, 64, 64)	256
activation_57 (Activation)	(None, 64, 64, 64)	0
zero_padding2d_12 (ZeroPaddi	(None, 65, 65, 64)	0
depthwise_conv2d_28 (Depthwi	(None, 32, 32, 64)	576
batch_normalization_58 (Batc	(None, 32, 32, 64)	256
activation_58 (Activation)	(None, 32, 32, 64)	0
conv2d_31 (Conv2D)	(None, 32, 32, 128)	8192
batch_normalization_59 (Batc	(None, 32, 32, 128)	512
activation_59 (Activation)	(None, 32, 32, 128)	0
depthwise_conv2d_29 (Depthwi	(None, 32, 32, 128)	1152
batch_normalization_60 (Batc	(None, 32, 32, 128)	512
activation_60 (Activation)	(None, 32, 32, 128)	0
conv2d_32 (Conv2D)	(None, 32, 32, 128)	16384
batch_normalization_61 (Batc	(None, 32, 32, 128)	512
activation_61 (Activation)	(None, 32, 32, 128)	0
zero_padding2d_13 (ZeroPaddi	(None, 33, 33, 128)	0
depthwise_conv2d_30 (Depthwi	(None, 16, 16, 128)	1152
batch_normalization_62 (Batc	(None, 16, 16, 128)	512
activation_62 (Activation)	(None, 16, 16, 128)	0
conv2d_33 (Conv2D)	(None, 16, 16, 256)	32768
batch_normalization_63 (Batc	(None, 16, 16, 256)	1024
activation_63 (Activation)	(None, 16, 16, 256)	0

depthwise_conv2d_31 (Depthwi	(None, 16, 16, 256)	2304
batch_normalization_64 (Batc	(None, 16, 16, 256)	1024
activation_64 (Activation)	(None, 16, 16, 256)	0
conv2d_34 (Conv2D)	(None, 16, 16, 256)	65536
batch_normalization_65 (Batc	(None, 16, 16, 256)	1024
activation_65 (Activation)	(None, 16, 16, 256)	0
zero_padding2d_14 (ZeroPaddi	(None, 17, 17, 256)	0
depthwise_conv2d_32 (Depthwi	(None, 8, 8, 256)	2304
batch_normalization_66 (Batc	(None, 8, 8, 256)	1024
activation_66 (Activation)	(None, 8, 8, 256)	0
conv2d_35 (Conv2D)	(None, 8, 8, 512)	131072
batch_normalization_67 (Batc	(None, 8, 8, 512)	2048
activation_67 (Activation)	(None, 8, 8, 512)	0
depthwise_conv2d_33 (Depthwi	(None, 8, 8, 512)	4608
batch_normalization_68 (Batc	(None, 8, 8, 512)	2048
activation_68 (Activation)	(None, 8, 8, 512)	0
conv2d_36 (Conv2D)	(None, 8, 8, 512)	262144
batch_normalization_69 (Batc	(None, 8, 8, 512)	2048
activation_69 (Activation)	(None, 8, 8, 512)	0
depthwise_conv2d_34 (Depthwi	(None, 8, 8, 512)	4608
batch_normalization_70 (Batc	(None, 8, 8, 512)	2048
activation_70 (Activation)	(None, 8, 8, 512)	0
conv2d_37 (Conv2D)	(None, 8, 8, 512)	262144
batch_normalization_71 (Batc	(None, 8, 8, 512)	2048
activation_71 (Activation)	(None, 8, 8, 512)	0
depthwise_conv2d_35 (Depthwi	(None, 8, 8, 512)	4608
batch_normalization_72 (Batc	(None, 8, 8, 512)	2048
activation_72 (Activation)	(None, 8, 8, 512)	0
conv2d_38 (Conv2D)	(None, 8, 8, 512)	262144
batch_normalization_73 (Batc	(None, 8, 8, 512)	2048
activation_73 (Activation)	(None, 8, 8, 512)	0
depthwise_conv2d_36 (Depthwi	(None, 8, 8, 512)	4608

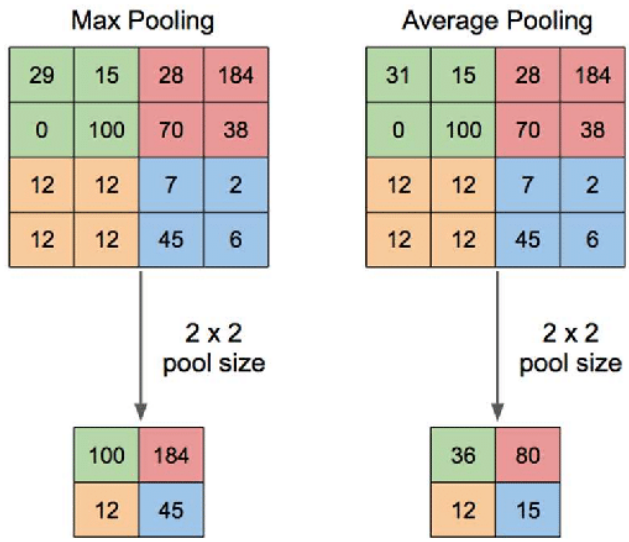
batch_normalization_74 (Batc	(None, 8, 8, 512)	2048
activation_74 (Activation)	(None, 8, 8, 512)	0
conv2d_39 (Conv2D)	(None, 8, 8, 512)	262144
batch_normalization_75 (Batc	(None, 8, 8, 512)	2048
activation_75 (Activation)	(None, 8, 8, 512)	0
depthwise_conv2d_37 (Depthwi	(None, 8, 8, 512)	4608
batch_normalization_76 (Batc	(None, 8, 8, 512)	2048
activation_76 (Activation)	(None, 8, 8, 512)	0
conv2d_40 (Conv2D)	(None, 8, 8, 512)	262144
batch_normalization_77 (Batc	(None, 8, 8, 512)	2048
activation_77 (Activation)	(None, 8, 8, 512)	0
zero_padding2d_15 (ZeroPaddi	(None, 9, 9, 512)	0
depthwise_conv2d_38 (Depthwi	(None, 4, 4, 512)	4608
batch_normalization_78 (Batc	(None, 4, 4, 512)	2048
activation_78 (Activation)	(None, 4, 4, 512)	0
conv2d_41 (Conv2D)	(None, 4, 4, 1024)	524288
batch_normalization_79 (Batc	(None, 4, 4, 1024)	4096
activation_79 (Activation)	(None, 4, 4, 1024)	0
depthwise_conv2d_39 (Depthwi	(None, 4, 4, 1024)	9216
batch_normalization_80 (Batc	(None, 4, 4, 1024)	4096
activation_80 (Activation)	(None, 4, 4, 1024)	0
conv2d_42 (Conv2D)	(None, 4, 4, 1024)	1048576
batch_normalization_81 (Batc	(None, 4, 4, 1024)	4096
activation_81 (Activation)	(None, 4, 4, 1024)	0
global_average_pooling2d_3 (	(None, 1024)	0
dropout_3 (Dropout)	(None, 1024)	0
dense_3 (Dense)	(None, 20)	20500

Gambar 3.9. Block Convolution

3.5.6. Pooling Layer (Downsampling)

Proses pooling layer berada setelah konvolusi layer, pada tahap ini bertujuan untuk mengurangi dimensi dari feature map. Sehingga akan mempercepat komputasi

karena jumlah parameter yang digunakan semakin berkurang untuk mengurangi overfitting. Pooling layer terdiri dari filter dengan ukuran dan stride tertentu yang akan bergeser pada area feature map. Proses pooling pada mobilenet menggunakan Average pooling yaitu memilih nilai rata-ratanya. Pada tabel 3.2 input yang dihasilkan adalah (7x7x1024) dengan menggunakan filter (7x7). Setelah dilakukan pooling layer menjadi (1x1x1024). Berikut ini adalah perbedaan antara max pooling dan average pooling pada Gambar 3.10.

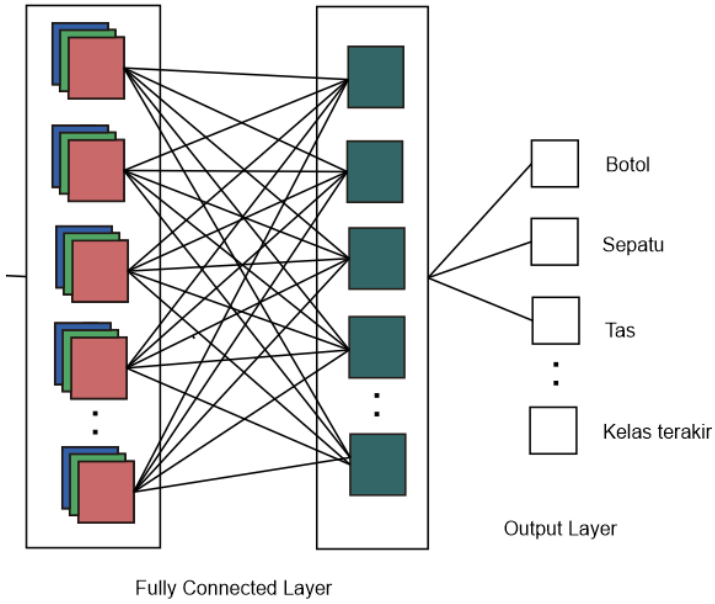


Gambar 3.10. Perbedaan Max Pooling dan Average Pooling

3.5.7. Fully Connected Layer (Flatten)

Pada feature map output yang dihasilkan masih berbentuk multidimensional array sehingga perlu dirubah menjadi vector agar bias digunakan dalam input fully connected layer. Setelah dilakukan pooling layer maka tahap selanjutnya fully connected layer. Inputan layer dapat dilihat di tabel 3.2 dimana input hasil pooling layer adalah (1x1x1024) dengan mengunakan filter 1024x16, angka 16 disini adalah

jumlah kelas yang di latih. Sehingga menghasilkan output layer (1x1x16). Berikut ini adalah proses dari fully connected layer dapat dilihat pada Gambar 3.11.



Gambar 3.11. Fully Connected Layer

3.5.8. Proses pelatihan data

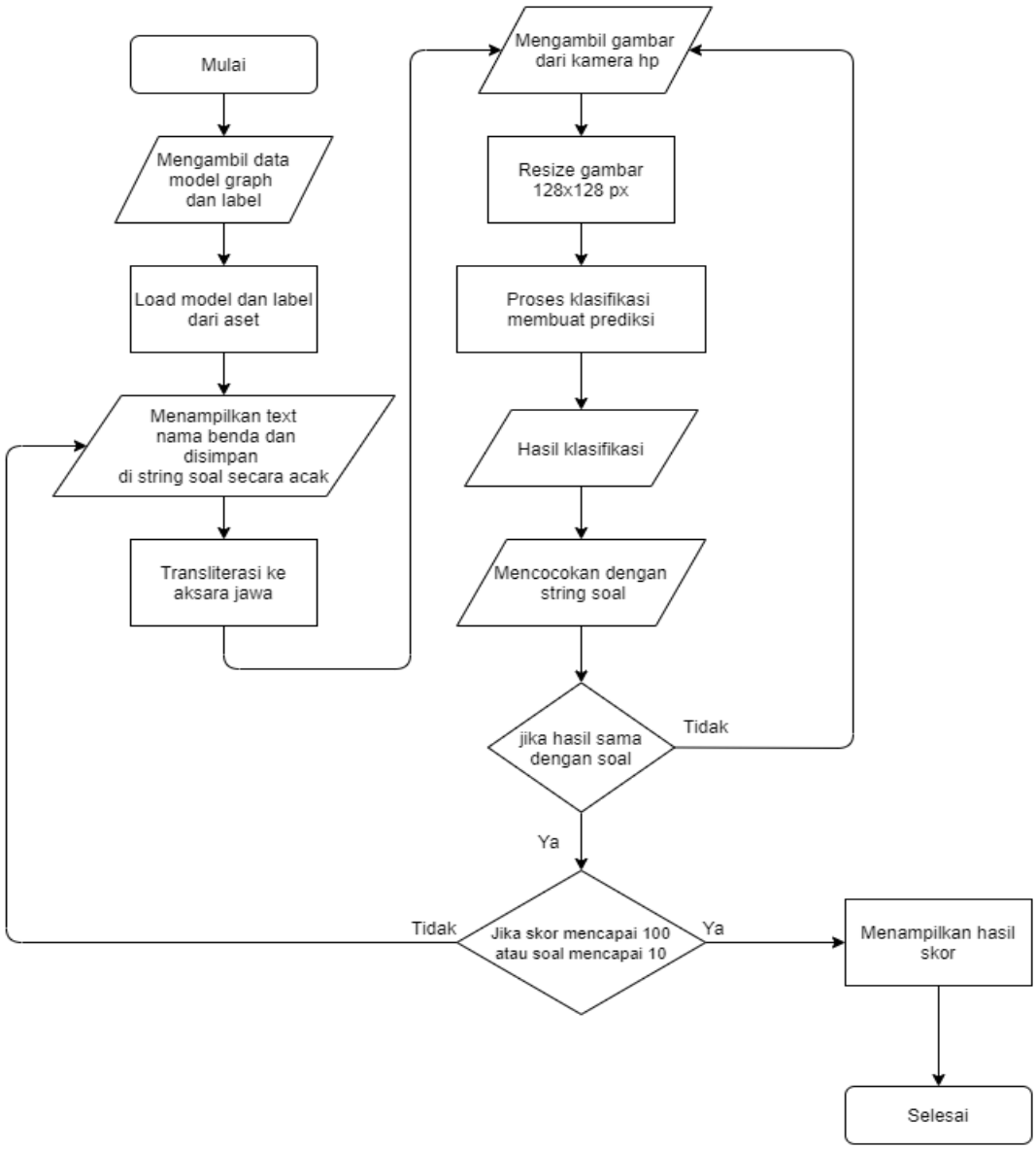
Pada langkah ini dapat diketahui bahwa semakin mendekati epoch yang dituju, akurasi semakin baik dan loss pada pelatihan semakin berkurang walaupun terkadang tidak stabil daapt dilihat pada Gambar 3.12.

```
50/50 [==============================] - 27s 535ms/step - loss: 1.1567 - accuracy: 0.6812
Epoch 2/20
50/50 [==============================] - 7s 145ms/step - loss: 0.1655 - accuracy: 0.9837
Epoch 3/20
50/50 [==============================] - 7s 146ms/step - loss: 0.0807 - accuracy: 0.9931
Epoch 4/20
50/50 [==============================] - 7s 147ms/step - loss: 0.0498 - accuracy: 0.9956
Epoch 5/20
50/50 [==============================] - 7s 150ms/step - loss: 0.0349 - accuracy: 1.0000
Epoch 6/20
50/50 [==============================] - 8s 164ms/step - loss: 0.0278 - accuracy: 1.0000
Epoch 7/20
50/50 [==============================] - 8s 151ms/step - loss: 0.0213 - accuracy: 0.9994
Epoch 8/20
50/50 [==============================] - 7s 147ms/step - loss: 0.0168 - accuracy: 1.0000
Epoch 9/20
50/50 [==============================] - 7s 145ms/step - loss: 0.0140 - accuracy: 1.0000
Epoch 10/20
50/50 [==============================] - 7s 144ms/step - loss: 0.0141 - accuracy: 1.0000
Epoch 11/20
50/50 [==============================] - 7s 144ms/step - loss: 0.0118 - accuracy: 1.0000
Epoch 12/20
```

Gambar 3.12. Proses Pelatihan Data

3.5.9. Proses prediksi citra pada aplikasi

Model yang didapatkan dari proses pelatihan selanjutnya dimasukan kedalam aplikasi untuk dilakukan pengetesan, sehingga model yang dihasilkan dapat digunakan untuk memprediksi nama benda. Untuk alur proses memuat model samapai proses bisa memprediksi nama benda pada aplikasi dapat dilihat pada Gambar 3.13.



Gambar 3.13. Proses Klasifiaksi Pada Aplikasi

BAB IV
UJI COBA DAN PEMBAHASAN

Pada bab ini akan membahas hasil dari implementasi yang telah dibahas pada Bab 3, tujuannya adalah untuk mengetahui gambar sesuai dengan perancangan sistem.

4.1. Sekenario Pengujian

Dalam penelitian ini untuk melakukan pengujian menggunakan data gambar dari benda yang ada disekitar rumah. Pada penelitian ini menggunakan dataset sebanyak 12000 gambar benda yang telah melewati proses seleksi agar dataset yang digunakan menghasilkan model yang baik lalu diklasifikasi menggunakan algoritma CNN. Sekenario pengujian menggunakan 7200 data latih, 2400 data validasi dan 2400 data tes, proses komputasi menggunakan single GPU yang memanfaatkan CUDA core. Pada dataset latih dan dataset validasi digunakan untuk proses *Training* sedangkan dataset tes digunakan untuk mengevaluasi model menggunakan *Confusion Matrix Multiclass*. Evaluasi model yang dihasilkan menggunakan *Confusion Matrix Multiclass* yang berisi informasi tentang nilai *Accuracy, Precision, Recall, F-Measure*, setelah mendapatkan model yang baik lalu dimasukan kedalam aplikasi untuk memprediksi masing-masing benda yang ada. Untuk pengujian spesifikasi Hardware yang digunakan menggunakan CPU Core i7-7700HQ 2.8Ghz, 12GB RAM, VGA Nvidia GTX 1050 2GB, NVMe 256GB, CPU Snapdragon 665, Sedangkan untuk software yang digunakan Sistem Operasi Windows 10 64-bit, Python 3.7, Atom IDE, Keras, Android 10 SDK 29.

4.2. Pengolahan Data

Groundtruth diperlukan diperlukan untuk memvalidasi akurasi dan metrik lainya sehingga dapat terlihat seberapa baik gambar tersebut tersegmentasi. Proses segmentasi digunakan untuk melakukan proses ektrasi pada ciri citra gambar, tahapan ini betujuan unutk mengektrak ciri dari suatu objek benda yang digunakan untuk membedakan antara objek yang akan digunakan dengan yang lain. Berikut ini adalah Langkah-langkah dalam segementasi pada citra gambar yang dilatih

a. Croping Gambar

Pada proses ini citra gambar akan dipilih dahulu mana objek yang akan dipilih sebagai data. Dapat dilihat pada Gambar 4.1.

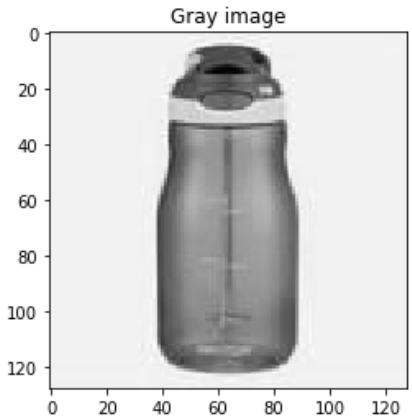
Gambar 4.1 Citra Gambar Asli.

b. Grayscale Citra Gambar

Merubah citra gambar dari citra RGB ke citra Grayscale.

```
img = cv2.imread("test_set/botol.jpg")
grey_image = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2.imwrite("test_set/botol_gray.png", grey_image)
plt.imshow(grey_image, cmap='gray')
plt.axis('on')
```

Gambar 4.2 Python Konversi RGB Menjadi Grayscale



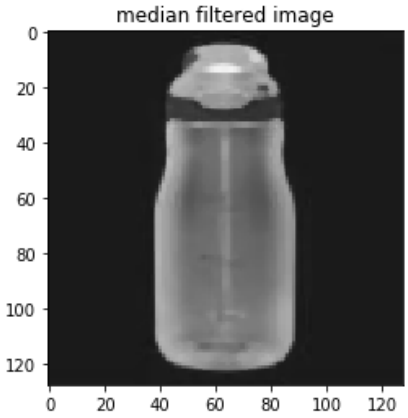
Gambar 4.3 Citra Gambar Grayscale

c. Median Filter

Median filter digunakan untuk mereduksi *noise* yang terdapat pada citra gambar

```
median_filtered = scipy.ndimage.median_filter(grayscale, size=3)
plt.imshow(median_filtered, cmap='gray')
plt.axis('on')
plt.title('median filter image')
```

Gambar 4.4 Python Median Filter



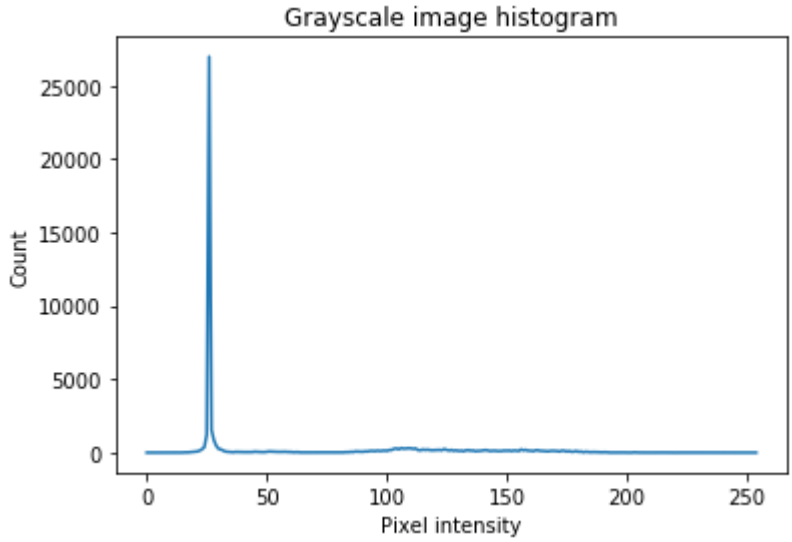
Gambar 4.5 Citra Gambar Dengan Median Filter

d. Histogram Citra Gambar

Menampilkan nilai histogram bertujuan untuk melihat nilai representasi citra dari terang atau gelap pada sebuah citra gambar dapat dilihat pada Gambar 4.7.

```
counts, vals = np.histogram(grayscale, bins=range(2 ** 8))
plt.plot(range(0, (2 ** 8) - 1), counts)
plt.title('Grayscale image histogram')
plt.xlabel('Pixel intensity')
plt.ylabel('Count')
```

Gambar 4.6 Python Histogram Citra Gambar



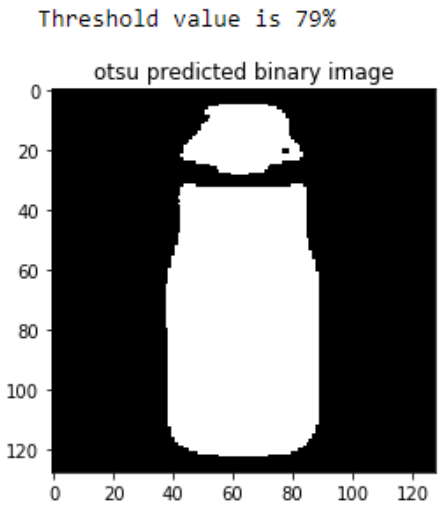
Gambar 4.7 Histogram Citra Gambar

e. Thresholding

Pada proses ini bertujuan untuk mengetahui perbedaan derajat keabuan pada citra. Metode thresholding yang diambil menggunakan *otsu* karena nilai threshold lebih tinggi yaitu mencapai 79%.

```
threshold = skimage.filters.threshold_otsu(median_filtered)
print('Threshold value is {}%'.format(threshold))
predicted = np.uint8(median_filtered > threshold) * 255
plt.imshow(predicted, cmap='gray')
plt.axis('on')
plt.title('otsu predicted image')
```

Gambar 4.8 Python *Otsu* Thresholding



Gambar 4.9 Citra *Otsu* Thresholding

Berikut ini adalah tabel perbandingan dari beberapa metode thresholding yang digunakan :

Tabel 4.1 Perbandingan Threshold

Thresholding	**Nilai**
Otsu	79
li	57
mean	58
yen	30
minium	72

4.3. Model CNN Depthwise Separable

Model *Convolutional Neural Network* yang akan digunakan untuk melatih dataset sehingga mendapatkan model yang akan dipergunakan untuk mengklasifikasikan citra gambar.

4.3.1. Prepocessing Citra

Pada citra gambar yang telah didapatkan, selanjutnya dilakukan tahap *preprocessing* citra beupa memotong citra (*cropping*) dan perubahan ukuran (*resize*). Pada proses *cropping* dilakukan unutk mendapatkan hasil object yang bagus dengan menghilangkan background pada gambar yang tidak dibutuhkan jadi hanya focus pada gambar yang dibutuhkan. Setelah proses *cropping* selesai selanjutnya adalah melakukan *resize* pada data gambar yang digunakan.

4.3.2. Proses Augmentation

Pada proses *augmentasi* data gambar akan diacak, dirotasi, dibalik sesuai dengan yang dibutuhkan tujuannya adalah agar dataset memiliki berbagai bentuk karakteristik yang berbeda sehingga menambah varian bentuk citra gambar agar mendapatkan dataset yang baik, proses Augmentation dapat dilihat pada Gambar 4.10 dibawah ini.

```
import Augmentor

p = Augmentor.Pipeline("augmenter_path/")

p.rotate(probability=0.8, max_left_rotation=8,
max_right_rotation=8)

p.rotate90(probability=0.5)

p.rotate270(probability=0.5)

p.flip_left_right(probability=0.8)

p.flip_top_bottom(probability=0.8)

p.process()
```

Gambar 4.10 Augmentasi Citra

4.3.3. Penentuan Parameter Pelatihan Data

Pada tahap ini dataset dibagi menjadi 3 bagian data train, data validation, data test. Setiap kelas dimasukan kedalam direktori setiap dataset dan ditentukan parameter yang digunakan akan dilatih dengan menggunakan python keras deep learning.

```
DATASET_DIR_TRAIN = 'zhs/train_set'

DATASET_DIR_VAL = 'zhs/val_set'

DATASET_DIR_TEST = 'zhs/test_set'

img_size = 128

num_train = 7200

num_test = 2400

num_val = 2400
```

Gambar 4.11 Parameter

Pada Gambar 4.11 terdapat parameter yang digunakan. Input gambar yang digunakan adalah 128x128px. Batch size adalah jumlah per sampel yang diambil dari dataset yang disebarkan ke Neural Network pada saat proses pelatihan, jika

num_train=jumlah sampel train / batch jika diambil dari dataset maka 7200 dan batch_size=64 maka didapatkan 112 sampel per Batch. Epoch digunakan untuk melakukan proses training pada seluruh dataset berulang kali, epoch yang digunakan adalah berjumlah epoch=20, untuk menyelesaikan setiap epoch yang dijalan kan dibagi menjadi beberapa batch pada parameter step_per_epoch(iterasi) berjumlah 64. Sedangkan jumlah untuk proses validasi adalah num_val = jumlah sampel validasi / batch jika di ambil dari dataset yang digunakan makan 2400 / 64 = 37 step.

4.3.4. Image Data

Proses preprosesing data gambar yang digunakan mengaugmentasi data gambar menjadi data preprosesing input yang dibutuhkan sebelum dilakukan pelatihan data. Seperti me*rescale* gambar rmenjadi 1/255 maksudnya adalah untuk menskala ulang nilai yang akan digunakan pemrosesan data gambar. Pada gambar asli terdiri dari RGB dalam rentang nilai 0-255 tetapi nilai tersebut terlalu tinggi untuk digunakan dalam memproses model yang akan dibuat, maka dengan menargetkan nilai pixel 0 dan 1 sehingga nilai diubah menjadi skala 1/255. Pada fungsi target size digunakan unutk menyamakan ukuran gambar menjadi 128 x 128 px selain itu batch size digunakan untuk membagi beberapa batch gambar dapat dilihat pada Gambar 4.12.

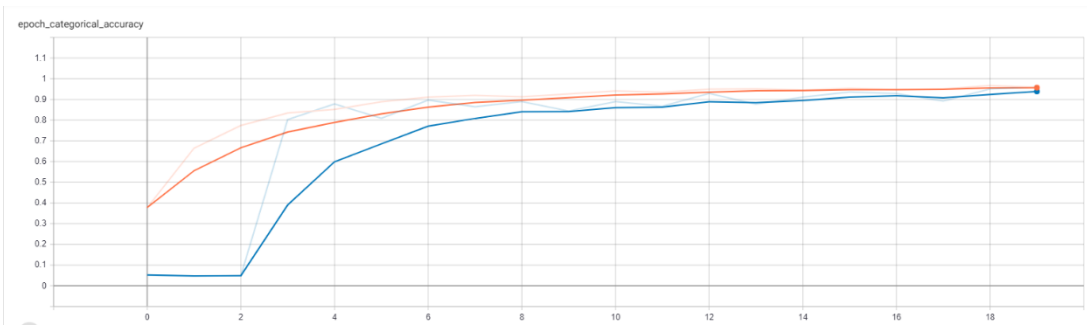
```
train_datagen=ImageDataGenerator(rescale = 1./255,
                                 shear_range=0.2,
                                 zoom_range=0.2,
                                 horizontal_flip=True)
validation_datagen=ImageDataGenerator(rescale = 1./255)
test_datagen=ImageDataGenerator(rescale = 1./255)
train_generator=train_datagen.flow_from_directory(DATASET_DIR_TRA
IN,
```

```
target_size=(img_size,img_size),

classes=label_class,

batch_size=batch_size,

class_mode='categorical')
test_generator=test_datagen.flow_from_directory(DATASET_DIR_TEST,

target_size=(img_size,img_size),

classes=label_class,

batch_size=batch_size,

class_mode='categorical')
```

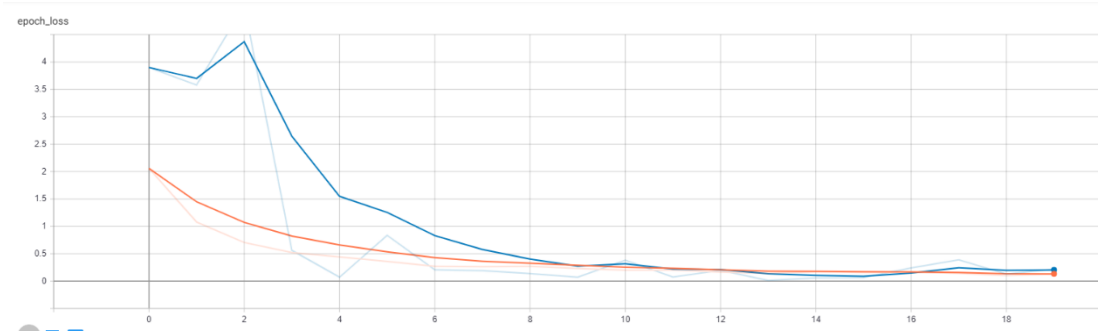
Gambar 4.12 Preprosesing Data Gambar

4.3.5. Hasil Pelatihan Data

Setelah proses pelatihan selesai maka didapatkan hasil training dan validation, Berikut ini grafik yang menampilkan jumlah epoch 20 dan nilai learning rate 0,001menggunakan log pada tensorboard pada Gambar 4.13 dan Gambar 4.14.



Gambar 4.13 Grafik akurasi training dan validation



Gambar 4.14 Grafik loss training dan validation

Berdasarkan Gambar 4.13 hasil akurasi yang diperoleh dari proses training mencapai 0,9256 dengan nilai los pada gambar 4.14 sebesar 0,2588, untuk validasi data mencapai 0,9239 dengan nilai los pada Gambar 4.13 sebesar 0.24789. Pada saat pelatihan waktu yang di butuhkan untuk menyelesaikan 20 epoch dengan 112 iterasi yaitu 30 menit 43 detik. Semakin banyak epoch dan iterasi yang digunakan maka waktu yang dibutuhkan, dengan catatan pada penelitian ini menggunakan vga dengan *Cuda Core* dengan kapasitas VRAM 2GB. Setelah pelatihan data selesai maka akan menghasilkan model.h5 untuk dimasukan kedalam android maka dikonversi ke format model.tflite dapat dilihat pada Gambar 4.15.

```
import tensorflow as tf new_model=
tf.keras.models.load_model(filepath="final_mbnet1.0.h5")
tflite_converter =
tf.lite.TFLiteConverter.from_keras_model(new_model) tflite_model
= tflite_converter.convert() open("tf_lite_model.tflite",
"wb").write(tflite_model)
```

Gambar 4.15 Klasifikasi pada tflite

4.3.6. Hasil evaluasi Model CNN Depthwise Separable

Setelah melalui proses pelatihan data maka didapatkan hasil pengelompokan citra sesuai dekan kelompok kelasnya. Cara pengujian pada model adalah menggunakan *Confusion Matrix Multiclass,* data pengujian diambil dari *Dataset Test* yang berisi 120 citra gambar pada setiap kelas, hasil dari proses training data menggunakan python dapat dilihat pada Gambar 4.16.

```
#Confusion Matrix DAN Klasifikasi Report

test_steps_per_epoch = np.math.ceil(test_generator.samples /
test_generator.batch_size)

predictions = model.predict_generator(test_generator,
steps=test_steps_per_epoch)

predicted_classes = np.argmax(predictions, axis=1)

true_classes = test_generator.classes

class_labels = list(test_generator.class_indices.keys())

report = metrics.classification_report(true_classes,
predicted_classes, target_names=class_labels)

print('Confusion Matrix')

print(confusion_matrix(test_generator.classes,
predicted_classes))

print('Classification Report')

print(report)
```

Gambar 4.16 Python Confusion Matrix

Selanjutnya untuk hasil data yang diperoleh akan menunjukan matrix yang berisi nilai jumlah data citra yang berhasil diprediksi dengan benar, untuk lebih jelasnya dapat dilihat pada Gambar 4.16 menunjukan *Confusion Matrix Multiclass* dari 20 kelas.

```
Accuracy =  0.9529687762260437
Confusion Matrix
[[120   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0 120   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0 120   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0 120   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0 120   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0 120   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  1   0   1   1   1   0 115   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   1   1   0   0 118   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0 120   0   0   0   0   0   0   0   0   0   0]
 [  1   1   5  14   2   0   0   0   1  79   2   0   8   0   0   1   0   0   4]
 [  0   0   0   1   0   0   0   0   0   0 119   0   0   0   0   0   0   0   0]
 [  0   0   0   0   6   0   0   0   0   0   0 114   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0 120   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0 120   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0 120   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 120   0   0   0]
 [  1   0   1   9   3   0   0   0  20   0   1   0   0   0   0   3   0  79   3]
 [  0   1   0  11   1   0   0   0   1   0   1   0   0   0   0   0   0   0 105]
 [  0   0   0  16   0   0   0   0   0   0   0   0   0   0   0   0   0   3 101]]
```

```
Classification Report
              precision    recall  f1-score   support

       botol       0.98      1.00      0.99       120
     cangkir       0.98      1.00      0.99       120
      cidhuk       0.94      1.00      0.97       120
      dompét       0.69      1.00      0.82       120
      émbér        0.90      1.00      0.94       120
      garpu        1.00      1.00      1.00       120
      gitar        1.00      0.97      0.98       119
  jam tangan       1.00      0.98      0.99       120
  kaos sikil       0.85      1.00      0.92       120
      kursi        1.00      0.67      0.80       118
     lemari        0.97      0.99      0.98       120
    mangkuk        1.00      0.95      0.97       120
       méja        0.94      1.00      0.97       120
     piring        1.00      1.00      1.00       120
     pulpén        1.00      1.00      1.00       120
     sendal        0.97      1.00      0.98       120
    séndhok        1.00      1.00      1.00       120
     sepatu        1.00      0.66      0.79       120
        tas        0.91      0.88      0.89       120
       tipi        1.00      0.84      0.91       120

   accuracy                             0.95      2397
  macro avg        0.96      0.95      0.95      2397
weighted avg       0.96      0.95      0.95      2397
```

Gambar 4.17 Hasil Ujicoba Confusion Matrix

Berdasarkan Gambar 4.17 dapat diambil 3 contoh kelas yang digunakan untuk cara menghitung nilai TP, TN, FP dan FN. Untuk selengkapnya dapat dilihat pada Tabel 4.10. Berikut ini adalah cara menghitung nilai TP dapat dilihat pada Tabel 4.2.

Tabel 4.2 Menghitung Nilai TP

Metrics		Predict Class																			
		botol	cangkir	cidhuk	dompét	émbér	garpu	gitar	jam tangan	Kaos Sikil	kursi	lemari	mangkok	méja	pireng	pulpén	sendal	séndhok	sepatu	tas	tipi
Actual Class	botol	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	cangkir	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	cidhuk	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	dompét	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	émbér	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	garpu	0	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	gitar	1	0	1	1	1	0	115	0	0	0	0	0	0	0	0	0	0	0	0	0
	am tanga	0	0	0	1	1	0	0	118	0	0	0	0	0	0	0	0	0	0	0	0
	kaos siki	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0
	kursi	1	1	5	14	2	0	0	0	1	79	2	0	8	0	0	1	0	0	4	0
	lemari	0	0	0	1	0	0	0	0	0	0	119	0	0	0	0	0	0	0	0	0
	mangkok	0	0	0	0	6	0	0	0	0	0	0	114	0	0	0	0	0	0	0	0
	méja	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	0
	pireng	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0
	pulpén	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0
	sendal	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0
	séndhok	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0
	sepatu	1	0	1	9	3	0	0	0	20	0	1	0	0	0	0	3	0	79	3	0
	tas	0	1	0	11	1	0	0	0	1	0	1	0	0	0	0	0	0	0	105	0
	tipi	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	101

Nilai TP (True Positive) merupakan kondisi aktual yang mampu memprediksi dengan benar atau tepat. Pada Tabel 4.2 dapat dilihat yang berwarna hijau adalah letak nilai TP pada nilai setiap kelas berisi nilai yang berbeda, untuk nilai TP selengkapnya dapat dilihat pada Tabel 4.3.

Tabel 4.3 Total Nilai TP

Kelas	botol	cangkir	cidhuk	dompét	émbér	garpu	gitar	jam tangan	kaos sikil	kursi	lemari	mangkok	méja	pireng	pulpén	sendal	séndhok	sepatu	tas	tipi
TP	120	120	120	120	120	120	115	118	120	79	119	114	120	120	120	120	120	79	105	101

Selanjutnya adalah menghitung nilai TN yang yang dapat dilihat pada Tabel 4.4 sebagai berikut.

Tabel 4.4 Menghitung Nilai TN

Metrics		Predict Class																				
		botol	cangkir	cidhuk	dompét	émbér	garpu	gitar	jam tangan	Kaos Sikil	kursi	lemari	mangkok	méja	pireng	pulpén	sendal	séndhok	sepatu	tas	tipi	TN
Actual Class	botol	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2274
	cangkir	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	cidhuk	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	dompét	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	émbér	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	garpu	0	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	gitar	1	0	1	1	1	0	115	0	0	0	0	0	0	0	0	0	0	0	0	0	
	am tanga	0	0	0	1	1	0	0	118	0	0	0	0	0	0	0	0	0	0	0	0	
	kaos sikil	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	
	kursi	1	1	5	14	2	0	0	0	1	79	2	0	8	0	0	1	0	0	4	0	
	lemari	0	0	0	1	0	0	0	0	0	0	119	0	0	0	0	0	0	0	0	0	
	mangkok	0	0	0	0	6	0	0	0	0	0	0	114	0	0	0	0	0	0	0	0	
	méja	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	
	pireng	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	
	pulpén	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	
	sendal	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	
	séndhok	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	
	sepatu	1	0	1	9	3	0	0	0	20	0	1	0	0	0	0	3	0	79	3	0	
	tas	0	1	0	11	1	0	0	0	1	0	1	0	0	0	0	0	0	0	105	0	
	tipi	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	101	

Metrics		Predict Class																				
		botol	cangkir	cidhuk	dompét	émbér	garpu	gitar	jam tangan	Kaos Sikil	kursi	lemari	mangkok	méja	pireng	pulpén	sendal	séndhok	sepatu	tas	tipi	TN
Actual Class	botol	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	cangkir	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	cidhuk	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	dompét	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	émbér	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	garpu	0	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	gitar	1	0	1	1	1	0	115	0	0	0	0	0	0	0	0	0	0	0	0	0	
	am tanga	0	0	0	1	1	0	0	118	0	0	0	0	0	0	0	0	0	0	0	0	
	kaos sikil	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	
	kursi	1	1	5	14	2	0	0	0	1	79	2	0	8	0	0	1	0	0	4	0	
	lemari	0	0	0	1	0	0	0	0	0	0	119	0	0	0	0	0	0	0	0	0	
	mangkok	0	0	0	0	6	0	0	0	0	0	0	114	0	0	0	0	0	0	0	0	
	méja	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	
	pireng	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	
	pulpén	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	
	sendal	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	
	séndhok	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	
	sepatu	1	0	1	9	3	0	0	0	20	0	1	0	0	0	0	3	0	79	3	0	
	tas	0	1	0	11	1	0	0	0	1	0	1	0	0	0	0	0	0	0	105	0	
	tipi	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	101	2277

Nilai TN (True Negative) merupakan jumlah data negative yang terprediksi benar. Pada Tabel 4.4 dapat dilihat kolom berwarna biru untuk botol dan orange untuk tipi menunjukan tiap nilai TN pada setiap kelas, untuk contoh perhitungan nilai TN diambil 2 dari 20 kelas, untuk nilai TN selengkapnya dapat dilihat pada Tabel 4.5.

Tabel 4.5 Total Nilai TN

Kelas	botol	cangkir	cidhuk	dompét	émbér	garpu	gitar	jam tangan	kaos sikil	kursi	lemari	mangkok	méja	pireng	pulpén	sendal	séndhok	sepatu	tas	tipi
TN	2274	2275	2270	2224	2263	2277	2278	2277	2255	2279	2273	2277	2269	2277	2277	2273	2277	2277	2267	2277

Setelah mendapatkan nilai TN selanjutnya adalah menghitung nilai FP pada setiap kelas dapat dilihat pada Tabel 4.6.

Tabel 4.6 Menghitung Nilai FP

Metrics		Predict Class																				
		botol	cangkir	cidhuk	dompét	émbér	garpu	gitar	jam tangan	Kaos Sikil	kursi	lemari	mangkok	méja	pireng	pulpén	sendal	séndhok	sepatu	tas	tipi	FP
Actual Class	botol	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
	cangkir	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	cidhuk	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	dompét	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	émbér	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	garpu	0	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	gitar	1	0	1	1	1	0	115	0	0	0	0	0	0	0	0	0	0	0	0	0	
	am tanga	0	0	0	1	1	0	0	118	0	0	0	0	0	0	0	0	0	0	0	0	
	kaos siki	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	
	kursi	1	1	5	14	2	0	0	0	1	79	2	0	8	0	0	1	0	0	4	0	
	lemari	0	0	0	1	0	0	0	0	0	0	119	0	0	0	0	0	0	0	0	0	
	mangkok	0	0	0	0	6	0	0	0	0	0	0	114	0	0	0	0	0	0	0	0	
	méja	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	
	pireng	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	
	pulpén	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	
	sendal	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	
	séndhok	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	
	sepatu	1	0	1	9	3	0	0	0	20	0	1	0	0	0	0	3	0	79	3	0	
	tas	0	1	0	11	1	0	0	0	1	0	1	0	0	0	0	0	0	0	105	0	
	tipi	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	101	0

Nilai FP (False Positive) merupakan data negative tetapi terdeteksi sebagai data positif. Pada Tabel 4.6 menampilkan kolom yang berisi nilai FP pada setiap kelas yang ditandai dengan warna yang berbeda pada setiap kelasnya, data kelas diambil 3 dari 20 kelas untuk selengkapnya dapat dilihat pada Tabel 4.7.

Tabel 4.7 Total Nilai FP

Kelas	botol	cangkir	cidhuk	dompét	émbér	garpu	gitar	jam tangan	kaos sikil	kursi	lemari	mangkok	méja	pireng	pulpén	sendal	séndhok	sepatu	tas	tipi
FP	3	2	7	53	14	0	0	0	22	0	4	0	8	0	0	4	0	0	10	0

Selanjutnya adalah menghitung nilai FN dapat dilihat pada Tabel 4.8.

Tabel 4.8 Menghitung Nilai FN

Metrics		Predict Class																				FN
		botol	cangkir	cidhuk	dompét	émbér	garpu	gitar	jam tangan	Kaos Sikil	kursi	lemari	mangkok	méja	pireng	pulpén	sendal	séndhok	sepatu	tas	tipi	
Actual Class	botol	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	cangkir	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	cidhuk	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	dompét	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	émbér	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	garpu	0	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	gitar	1	0	1	1	1	0	115	0	0	0	0	0	0	0	0	0	0	0	0	0	
	am tanga	0	0	0	1	1	0	0	118	0	0	0	0	0	0	0	0	0	0	0	0	
	kaos sikil	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	
	kursi	1	1	5	14	2	0	0	0	1	79	2	0	8	0	0	1	0	0	4	0	
	lemari	0	0	0	1	0	0	0	0	0	0	119	0	0	0	0	0	0	0	0	0	
	mangkok	0	0	0	0	6	0	0	0	0	0	0	114	0	0	0	0	0	0	0	0	
	méja	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	
	pireng	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	
	pulpén	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	
	sendal	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	
	séndhok	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	
	sepatu	1	0	1	9	3	0	0	0	20	0	1	0	0	0	0	0	0	79	3	0	
	tas	0	1	0	11	1	0	0	0	1	0	1	0	0	0	0	0	0	0	105	0	
	tipi	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	101	19

Nilai FN (False Negative) kebalikan dari TP merupakan data positif yang terdeteksi sebagai data negative. Pada Tabel 4.8 menampilkan data dari FN yang ditandai dengan warna yang berbeda pada setiap kelas, data kelas diambil 3 dari 20 kelas untuk selengkapnya dapat dilihat pada Tabel 4.9.

Tabel 4.9 Total Nilai FN

Kelas	botol	cangkir	cidhuk	dompét	émbér	garpu	gitar	jam tangan	kaos sikil	kursi	lemari	mangkok	méja	pireng	pulpén	sendal	séndhok	sepatu	tas	tipi
FN	0	0	0	0	0	0	4	2	0	39	0	6	0	0	0	0	0	41	15	19

Berdasarkan dari cara perhitungan tabel 4.2 sampai Tabel 4.9 untuk selengkapnya nilai TP, TN, FP dan FN hasil dari pengujian *Confusion Matrix Multiclass* dapat dilihat pada Tabel 4.10 dibawah ini.

Tabel 4.10 Hasil Perhitungan Confusion Matrix

Metrics		Predict Class																			TP	TN	FP	FN
		botol	cangkir	cidhuk	dompét	émbér	garpu	gitar	jam tangan	Kaos Sikil	kursi	lemari	mangkok	méja	pireng	pulpén	sendal	séndhok	sepatu	tas	tipi				
Actual Class	botol	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	2274	3	0
	cangkir	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	2275	2	0
	cidhuk	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	2270	7	0
	dompét	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	2224	53	0
	émbér	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	2263	14	0
	garpu	0	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	2277	0	0
	gitar	1	0	1	1	1	0	115	0	0	0	0	0	0	0	0	0	0	0	0	0	115	2278	0	4
	jam tangan	0	0	0	1	1	0	0	118	0	0	0	0	0	0	0	0	0	0	0	0	118	2277	0	2
	kaos sikil	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	120	2255	22	0
	kursi	1	1	5	14	2	0	0	0	1	79	2	0	8	0	0	1	0	0	4	0	79	2279	0	39
	lemari	0	0	0	1	0	0	0	0	0	0	119	0	0	0	0	0	0	0	0	0	119	2273	4	0
	mangkok	0	0	0	0	6	0	0	0	0	0	0	114	0	0	0	0	0	0	0	0	114	2277	0	6
	méja	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	120	2269	8	0
	pireng	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	0	120	2277	0	0
	pulpén	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	0	120	2277	0	0
	sendal	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	0	120	2273	4	0
	séndhok	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	0	0	0	120	2277	0	0
	sepatu	1	0	1	9	3	0	0	0	20	0	1	0	0	0	0	3	0	79	3	0	79	2277	0	41
	tas	0	1	0	11	1	0	0	0	1	0	1	0	0	0	0	0	0	0	105	0	105	2267	10	15
	tipi	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	101	101	2277	0	19

4.3.7. Evaluasi Model Accuracy

Akurasi adalah presentase dari total data yang diidentifikasi dan nilai benar, Berdasarkan Tabel 4.10 maka nilai akurasi dapat dihitung sebagai berikut.

Formula akurasi :

Accuracy = (TP+TN) / (TP+TN+FP+FN)

Accuracy = 47686/47939

Accuracy = 0,98 * 100%

Accuracy = 98%

Berikut ini adalah nilai total akurasi pada setiap kelas dan total keseluruhan akurasi pada semua kelas daapt dilihat pada Tabel 4.11

Tabel 4.11 Hasil Accuracy

Kelas (1-20)	TP	TN	FP	FN	Akurasi *(TP+TN) / (TP+TN+FP+FN)*	Total Akurasi
botol	120	2274	3	0	1,00	98%
cangkir	120	2275	2	0	1,00	
cidhuk	120	2270	7	0	1,00	
dompét	120	2224	53	0	0,98	
émbér	120	2263	14	0	0,99	
garpu	120	2277	0	0	1,00	
gitar	115	2278	0	4	1,00	
jam tangan	118	2277	0	2	1,00	
kaos sikil	120	2255	22	0	0,99	
kursi	79	2279	0	39	0,98	
lemari	119	2273	4	0	1,00	
mangkok	114	2277	0	6	1,00	
méja	120	2269	8	0	1,00	
pireng	120	2277	0	0	1,00	
pulpén	120	2277	0	0	1,00	
sendal	120	2273	4	0	1,00	
séndhok	120	2277	0	0	1,00	
sepatu	79	2277	0	41	0,98	
tas	105	2267	10	15	0,99	
tipi	101	2277	0	19	0,99	

4.3.8. Evaluasi Model Precision

Presisi adalah data yang diambil berdasarkan informasi yang kurang atau salah atau tidak tepat. Berdasarkan Tabel 4.12 maka nilai precision dapat dihitung sebagai berikut.

Tabel 4.12 Matrix Precision

Kelas	botol	cangkir	cidhuk	dompét	émbér	garpu	gitar	jam tangan	kaos sikil	kursi	lemari	mangkok	méja	pireng	pulpén	sendal	séndhok	sepatu	tas	tipi
TP	120	120	120	120	120	120	115	118	120	79	119	114	120	120	120	120	120	79	105	101
FP	3	2	7	53	14	0	0	0	22	0	4	0	8	0	0	4	0	0	10	0

Formulasi Precision :

Tabel 4.13 Formula Precision

FP setiap kelas	FP(kelas 1)… FP(kelas 20)
Precision tiap kelas	TP/(TP+FP)
Precision semua kelas	(Total precision setiap kelas)/20

Untuk cara menghitunya diambil contoh dari kelas pertama bernama botol :

a. Mencari FP pada setiap kelas contoh diambil kelas pertama yaitu botol :

   FP = 3

b. Mencari Precision setiap kelas contoh yang diambil adalah kelas bernama botol

   FP = TP/(TP+FP)

   FP = 120/(120+3)

   FP = 0,975609756

c. Mencari total Precision pada semua kelas adalah sebagai berikut :

   Precision = (Total precision setiap kelas) / 20

   Precision = 96%

Untuk lebih lengkapnya dari pehitungan formula Precision didapatkan hasil pada Tabel 4.14.

Tabel 4.14 Hasil Precision.

Kelas (1-20)	TP	FP	Rumus	Precision per kelas	Total Precision
			TP/(TP+FP)		
botol	120	3	120/(120+3)	0,98	96%
cangkir	120	2	120/(120+2)	0,98	
cidhuk	120	7	120/(120+7)	0,94	
dompét	120	53	120/(120+53)	0,69	
émbér	120	14	120/(120+14)	0,90	
garpu	120	0	120/(120+0)	1,00	
gitar	115	0	115/(115+0)	1,00	
jam tangan	118	0	118/(118+0)	1,00	
kaos sikil	120	22	120/(120+22)	0,85	
kursi	79	0	79/(79+0)	1,00	
lemari	119	4	119/(119+4)	0,97	
mangkok	114	0	114/(114+0)	1,00	
méja	120	8	120/(120+8)	0,94	
pireng	120	0	120/(120+0)	1,00	
pulpén	120	0	120(120+0)	1,00	
sendal	120	4	120/(120+4)	0,97	
séndhok	120	0	120/(120+0)	1,00	
sepatu	79	0	79/(79+0)	1,00	
tas	105	10	105/(105+10)	0,91	
tipi	101	0	101/(101+0)	1,00	

4.3.9. Evaluasi Model Recall

Recall adalah data yang tidak mampu di prediksi dengan benar. Berdasarkan Tabel 4.15 maka nilai akurasi dapat dihitung sebagai berikut.

Tabel 4.15 Matrix Recall

Kelas	botol	cangkir	cidhuk	dompét	émbér	garpu	gitar	jam tangan	kaos sikil	kursi	lemari	mangkok	méja	pireng	pulpén	sendal	séndhok	sepatu	tas	tipi
TP	120	120	120	120	120	120	115	118	120	79	119	114	120	120	120	120	120	79	105	101
FN	0	0	0	0	0	0	4	2	0	39	0	6	0	0	0	0	0	41	15	19

Formula recall :

Tabel 4.16 Formula Recall

FN setiap kelas	FN(kelas 1)… FN(kelas 20)
Recall tiap kelas	TP/(TP+FN)
Recall semua kelas	(Total recall setiap kelas)/20

Untuk cara menghitunya diambil contoh dari kelas pertama bernama botol :

a. Mencari FN pada setiap kelas contoh diambil kelas pertama yaitu botol :

   FN = 0

b. Mencari Precision setiap kelas contoh yang diambil adalah kelas bernama botol

   FN = TP/(TP+FN)

   FN = 120/(120+0)

   FN = 1

c. Mencari total Recall pada semua kelas adalah sebagai berikut :

   Recall = (Total Recall setiap kelas) / 20

   Recall = 95%

Sehingga dari perhitungan dari formula Recall didapatkan hasil pada Tabel 4.17 :

Tabel 4.17 Hasil Recall

Kelas (1-20)	TP	FN	Rumus *TP/(TP+FN)*	Recall per kelas	Total Recall
botol	120	0	120/(120+0)	1,00	94,72%
cangkir	120	0	120/(120+0)	1,00	
cidhuk	120	0	120/(120+0)	1,00	
dompét	120	0	120/(120+0)	1,00	
émbér	120	0	120/(120+0)	1,00	
garpu	120	0	120/(120+0)	1,00	
gitar	115	4	115/(115+4)	0,97	
jam tangan	118	2	118/(118+2)	0,98	
kaos sikil	120	0	120/(120+0)	1,00	
kursi	79	39	79/(79+39)	0,67	
lemari	119	0	119/(119+0)	1,00	
mangkok	114	6	114/(114+6)	0,95	
méja	120	0	120/(120+0)	1,00	
pireng	120	0	120/(120+0)	1,00	
pulpén	120	0	120/(120+0)	1,00	
sendal	120	0	120/(120+0)	1,00	
séndhok	120	0	120/(120+0)	1,00	
sepatu	79	41	(79/(79+41)	0,66	
tas	105	15	(105/(105+15)	0,88	
tipi	101	19	101/(101+19)	0,84	

4.3.10. Evaluasi Model F-measure

Fmeasure adalah perhitungan evaluasi dalam mengkombinasikan recall dan precision. Untuk perhitungan Fmeasure dapat dilihat pada Tabel 4.18.

Formula F-measure :

Tabel 4.18 Formula F-Measure

F-Measure setiap kelas	(2*precision*recall)/(precision+recall)
Total F-Measure	(Total Fmeasure setiap kelas)/20

Dari formula pada Tabel 4.14 dapat dihitung sebagai berikut :

Untuk cara menghitunya diambil contoh dari kelas pertama bernama botol :

a. Mencari F-Measure pada setiap kelas contoh diambil kelas pertama yaitu botol

F-Measure = (2*Precision*Recall) / (Precision+Recall)

F-Measure = (2*0,98*1) / (0, 98+1)

F-Measure = 0,99

b. Mencari total F-Measure pada semua kelas adalah sebagai berikut :

Total F-Measure = (Total F-Measure per kelas) / 20

Total F-Measure = 95%

Sehingga untuk selengkapnya perhitungan dari formula F-Measure pada Tabel 4.19 adalah sebagai berikut :

Tabel 4.19 Hasil F-Measure

Kelas (1-20)	Precision per kelas	Recall per kelas	Fmeasure per kelas	Total F-Measure
	TP/(TP+FP)	***TP/(TP+FN)***	***(2*Precision*Recall) / (Precision+Recall)***	
botol	0,98	1,00	0,99	95%
cangkir	0,98	1,00	0,99	
cidhuk	0,94	1,00	0,97	
dompét	0,69	1,00	0,82	
émbér	0,90	1,00	0,94	
garpu	1,00	1,00	1,00	
gitar	1,00	0,97	0,98	
jam tangan	1,00	0,98	0,99	
kaos sikil	0,85	1,00	0,92	
kursi	1,00	0,67	0,80	
lemari	0,97	1,00	0,98	
mangkok	1,00	0,95	0,97	
méja	0,94	1,00	0,97	
pireng	1,00	1,00	1,00	
pulpén	1,00	1,00	1,00	
sendal	0,97	1,00	0,98	
séndhok	1,00	1,00	1,00	
sepatu	1,00	0,66	0,79	
tas	0,91	0,88	0,89	
tipi	1,00	0,84	0,91	

4.3.11. Evaluasi Model Prediction

Prediksi digunakan untuk mengetahui seberapa akurat model yang dihasilkan dalam memprediksi gambar pada data test dengan mengambil data dari Tabel 4.10. Dapat diketahui bahwa total gambar yang berhasil diprediksi adalah 95% sedangkan untuk gambar yang gagal diprediksi adalah 5%. Berikut ini adalah total presentase berhasil dan gagal pada data testing yang dapat dilihat pada Tabel 4.20.

Tabel 4.20 Hasil Prediksi Pada Data Test.

Nama Kelas	Total Test dataset	Berhasil diprediksi	Gagal diprediksi	Berhasil %	Gagal %
botol	120	120	0	100%	0%
cangkir	120	120	0	100%	0%
cidhuk	120	120	0	100%	0%
dompét	120	120	0	100%	0%
émbér	120	120	0	100%	0%
garpu	120	120	0	100%	0%
gitar	120	115	5	96%	4%
jam tangan	120	118	2	98%	2%
kaos sikil	120	120	0	100%	0%
kursi	120	79	41	66%	34%
lemari	120	119	1	99%	1%
mangkok	120	114	6	95%	5%
méja	120	120	0	100%	0%
pireng	120	120	0	100%	0%
pulpén	120	120	0	100%	0%
sendal	120	120	0	100%	0%
séndhok	120	120	0	100%	0%
sepatu	120	79	41	66%	34%
tas	120	105	15	88%	13%
tipi	120	101	19	84%	16%
Total	2400	2270	130	95%	5%

Cara menghitung nilai keberhasilan prediksi perkelas diambil salah satu contoh nama kelas botol, dengan menggunakan perhitungan sebagai berikut.

Prediksi perkelas = (berhasil diprediksi / total data test) * 100%

Prediksi perkelas = (120 / 120) * 100%

Prediksi perkelas = 100%

Untuk menghitung nilai gagal dalam prediksi perkelas diambil salah satu contoh nama kelas botol, dengan menggunakan perhitungan sebagai berikut.

Prediksi perkelas = (gagal diprediksi / total data test) * 100%

Prediksi perkelas = (0 / 120) * 100%

Prediksi perkelas = 0%

Cara menghitung nilai total yang berhasil prediksi adalah menggunakan perhitungan sebagai berikut.

Total prediksi = (Total berhasil prediksi per kelas) / 20 * 100%

Total prediksi = 95 %

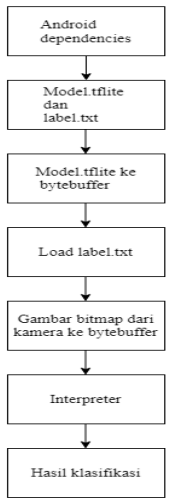
Untuk menghitung nilai total yang gagal prediksi adalah menggunakan perhitungan sebagai berikut.

Total prediksi = (Total gagal prediksi per kelas) / 20 * 100%

Total prediksi = 5 %

4.3.12. Hasil Pengujian Model

Pada pengujian pada aplikasi model akan dimasukan sebagai file assets pada android studio, file model ini yang akan digunakan dalam memprediksi gambar. Proses testing ini bertujuan untuk menguji sebuah model yang berhasil dibuat menggunakan CNN. Untuk lebih lengkapnya dapat dilihat pada Gambar 4.18 berikut ini.



Gambar 4.18 Klasifikasi Pada Tflite

a. Android dependencies

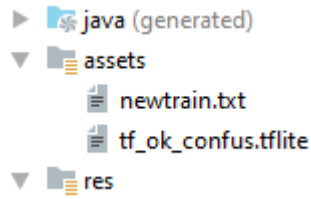
Android dependencies merupakan library yang dibutuhkan untuk ditambahkan pada build.gradle android studio, dapat dilihat pada Gambar 4.19

```
implementation 'org.tensorflow:tensorflow-lite:+'
aaptOptions {
        noCompress "tflite"
        noCompress "lite"
    }
```

Gambar 4.19 Android Dependencies

b. Load model.tflite dan label.txt

Langkah selanjutnya adalah memasukan file model.tflite dan label.txt ke file assets android studio, berikut ini adalah file assets yang telah di masukan ke android studio pada Gambar 4.20.



Gambar 4.20 Model.tflite dan Label.txt

c. Generate model.tflite ke bytebuffer

Langkah selanjutnya adalah mengenerate model.tflite menjadi bytebuffer menggunakan Mappedbytebuffer, fungsi ini digunakan untuk membuat map

memori yang diambil dari model.tflite pada folder assets untuk mendapatkan bytebuffer, untuk prosesnya dapat dilihat pada Gambar 4.21.

```
private MappedByteBuffer loadModelFile(Activity activity) throws
IOException {

    AssetFileDescriptor AFD =
activity.getAssets().openFd(lok_model);

    FileInputStream FIS = new
FileInputStream(AFD.getFileDescriptor());

    FileChannel fileChannel = FIS.getChannel();

    long startOffset = AFD.getStartOffset();

    long declaredLength = AFD.getDeclaredLength();

    return fileChannel.map(FileChannel.MapMode.READ_ONLY,
startOffset, declaredLength);

  }
```

Gambar 4.21 Generate Bytebuffer Pada Model

d. Load label.txt

Langkah selanjutnya adalah membaca label.txt pada file assets untuk dijadikan array, untuk lebih jelasnya dapat dilihat pada Gambar 4.22.

```
private List<String> loadLabelList(Activity activity) throws
IOException {

    List<String> labelList = new ArrayList<String>();

    BufferedReader BR = new BufferedReader(new
InputStreamReader(activity.getAssets().open(lok_label)));

    String line;

    while ((line = BR.readLine()) != null) {

      labelList.add(line);

    }

    BR.close();

    return labelList;
```

Gambar 4.22 Load label Dari Assets

e. Generate bitmap dari kamera ke bytebuffer

Selanjutnya adalah mengubah ukuran bitmap yang telah ditangkap oleh kamera agar sesuai dengan input model, kemudian akan dikonversi menjadi bytebuffer untuk mengeksekusi model, bitmap akan direscale menjadi 128x128px sesuai dengan inputan pada model.tflite, Untuk itu diperlukan mengubah bitmap menjadi array pixel sebelum dimasukan ke dalam bytebuffer. Selengkapnya dapat dilihat pada Gambar 4.23.

```
private void convertBitmapToByteBuffer(Bitmap bitmap) {
    if (imgData == null) {
      return;
    }
    imgData.rewind();
    bitmap.getPixels(imgValue, 0, bitmap.getWidth(), 0, 0,
bitmap.getWidth(), bitmap.getHeight());
    // Konversi data gamabr ke float
    int pixel = 0;
    long startTime = SystemClock.uptimeMillis();
    for (int i = 0; i < img_width; ++i) {
      for (int j = 0; j < img_height; ++j) {
        final int val = imgValue[pixel++];
        //maengambil nilai rgb menjadi float
        imgData.putFloat((((val >> 16) & 0xFF)-
img_mean)/img_std);
        imgData.putFloat((((val >> 8) & 0xFF)-img_mean)/img_std);
        imgData.putFloat((((val) & 0xFF)-img_mean)/img_std);
      }
    }
```

Gambar 4.23 Generate Bitmap ke Bytebuffer

f. Memulai interpreter

Setelah mendapatkan nilai bytebuffer dari bitmap, maka bisa tahap selajutnya membuat interpreter yang akan meneruskan bytebuffer bitmap ke bytebufer pada model, fungsinya adalah untuk dilakukan prediksi gambar yang telah di ambil dari kamera smartphone. Dapat dilihat pada Gambar 4.24.

```
public proses_Klasifikasi(Activity activity) throws IOException {

    interpreter = new Interpreter(loadModelFile(activity));

    //ambil label dari model

    labelList = loadLabelList(activity);

    //input image bitmap buffer

    imgData = ByteBuffer.allocateDirect(4 * batch_size *
img_width * img_height * pixel_size);

    imgData.order(ByteOrder.nativeOrder());

    labelProbArray = new float[1][labelList.size()];

    filterLabelProbArray = new
float[filter_stage][labelList.size()];

    Log.d(TAG, "Initialisasi klasifikasi pada tensorflow lite.");

  }
```

Gambar 4.24 Interpreter

g. Hasil Klasifikasi

Langkah terakir adalah menghubungkan probabilitas dengan kelas yang sesuai dengan prediksi pada Gambar 4.25.

```
public String Klasifikasi(Bitmap bitmap) {
    if (interpreter == null) {
      Log.e(TAG, "gagal initialisasi klasifikasi gambar");
      return "Uninitialized";
    }
    convertBitmapToByteBuffer(bitmap);
    interpreter.run(imgData, labelProbArray);
    // smooth the results
    applyFilter();
    // print the results
    String keStringPrediksi = printTopKLabels();
```

Gambar 4.25 Parse Klasifikasi

h. Hasil klasifikasi pada aplikasi

Berikut ini adalah hasil jika aplikasi dijalankan menggunakan model yang telah dibuat dapat dilihat pada Gambar 4.26.

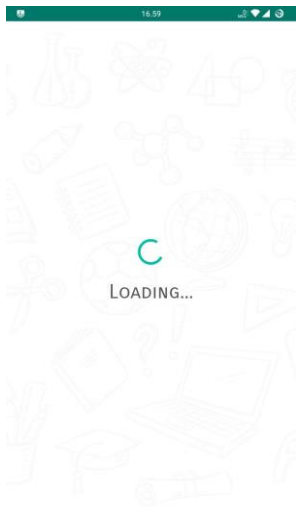
Gambar 4.26 Hasil Klasifikasi Pada Aplikasi

Dapat dilihat pada Gambar 4.23 bahwa hasil menunjukan 2 kata benda tipi menunjukan hasil 0,74 dan botol 0,17 ini menunjukan bahwa model bisa digunakan karena sesuai memprediksi kata benda yang ditangkap oleh kamera smartphone, untuk latency yang dibutuhkan pada smartphone android dengan CPU Snapdragon 665 dan ukuran kamera 64 Megapixel adalah sebesar 119 ms.

4.4. Hasil Pengujian Pada Aplikasi

Implementasi antarmuka terdiri dari tampilan beranda, tampilan kamera dan prediksi, dan tampilan hasil benda yang berhasil di ketahui.

a. Tampilan splash screen



Gambar 4.27 Tampilan Splash Screen

b. Tampilan beranda

Pada tampilan beranda merupakan tampilan yang pertama kali ditampilkan saat aplikasi dijalankan didalamnya terdapat beberapa *button* kamera, petunjuk, dan informasi dari aplikasi. Tampilan awal aplikasi dapat dilihat pada Gambar 4.28.

Gambar 4.28 Tampilan Beranda

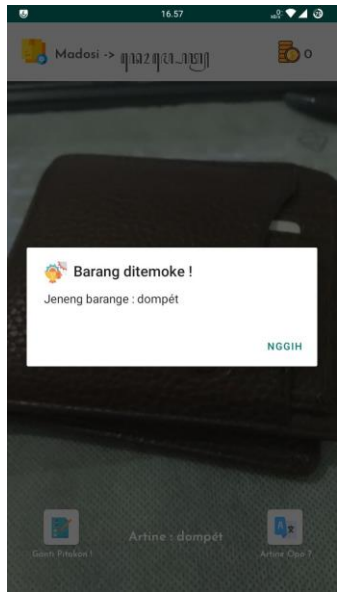
c. Tampilan petunjuk

Pada menu ini berfungsi untuk menampilkan petunjuk Langkah-langkah untuk menjalankan aplikasi ini.

Gambar 4.29 Tampilan Petunjuk Penggunaan

d. Tampilan kamera berburu benda

Tampilan kamera ini berisi fungsi untuk memanggil kamera pada *smartphone* untuk mengolah gambar yang diambil dari kamera lalu di proses untuk memprediksi hasil benda yang dituju. Selain itu terdapat indicator nama benda yang sedak di tangkap, terdapat button untuk mengganti nama benda dan pertolongan arti nama benda jka tidak paham dengan aksara jawa.



Gambar 4.30 Tampilan Kamera Prediksi Soal Benda

e. Tampilan kamera prediksi nama benda langsung

Pada menu ini akan menampilkanhasil prediksi nama benda secara langsung jika kamera diarhakan ke benda yang inginkan maka akan keluar nama benda dalam bahasa jawa dan aksara jawa, selain itu terdapat waktu yang dibutuhkan untuk melakukan prediksi pada benda yang diarahkan ke kamera.

Gambar 4.31 Tampilan Kamera Prediksi Nama Benda

f. Tampilan tentang aplikasi

Berisi tentang sumber icon yang diambil untuk digunakan dalam aplikasi.

Gambar 4.32 Tampilan Tentang Aplikasi

g. Tampilan hasil score benda yang telah diketahui

Tampilan hasil berisi tentang score yang diperoleh dari berapa banyak benda yang berhasil di cari.

Gambar 4.33 Tampilan Hasil

Dari pengujian aplikasi yang telah dilakukan dapat dilihat bawa aplikasi bisa dijalankan sesuai dengan masalah yang telah disebutkan pada penelitian ini, dengan menyebarkan ilmu ke masyarakat umum dalam rangka untuk melestarikan warisan budaya agar tidak dilupakan karena kurangnya pembahasan mengenai aksara jawa, Dalam sejarahnya aksara jawa yang sejak dahulu sudah menjadi ikonik dalam sebagian besar masyarakat Jawa yang ada di Indonesia. Kesimpulan yang dapat diambil dari pengujian menunjukan hasil yang baik, dapat dilihat pada Tabel 4.11 menunjukan tingkat akurasi mencapai 98% dari total data test yang diuji. Pada penerapan didalam aplikasi menunjukan bahwa model yang dibuat dapat memprediksi objek benda yang diamati, sehingga aplikasi yang telah dibuat dapat digunakan oleh pengguna untuk menyebarkan ilmu aksara jawa dan ikut dalam mempelestarikan budaya kemasyarakat

umum. Pada penyebaran ilmu sendiri terdapat penjelasan pada Al-Quran surah Al-Kahf ayat 60:

قَالَ لَهٗ مُوسٰى هَلْ اَتَّبِعُكَ عَلٰٓى اَنْ تُعَلِّمَنِ مِمَّا عُلِّمْتَ رُشْدًا

Musa berkata kepada Khidhr: "Bolehkah aku mengikutimu supaya kamu mengajarkan kepadaku ilmu yang benar di antara ilmu-ilmu yang telah diajarkan kepadamu?"

Terdapat penjelasan pada Tafsir Jalalyn sebagaimana:

(Musa berkata kepada Khidhir, "Bolehkah aku mengikutimu supaya kamu mengajarkan kepadaku ilmu yang benar di antara ilmu-ilmu yang telah diajarkan kepadamu?)" yakni ilmu yang dapat membimbingku. Menurut suatu qiraat dibaca Rasyadan. Nabi Musa meminta hal tersebut kepada Khidhir. karena menambah ilmu adalah suatu hal yang dianjurkan.

Selain menurut tafsir oleh Ibnu Katsir pada ayat Al-Quran surah Al Kahfi ayat 66 terdapat penjelasan sebagaimana:

Allah SWT telah menjelaskan bagaimana perkataan Musa a.s. terhadap lelaki yang bertaqwa (yakni Khidir) yang sudah diberikan keutamaan oleh Allah Swt yang berupa ilmu yang tidak diketahui oleh Musa.

قَالَ لَهٗ مُوسٰى هَلْ اَتَّبِعُكَ

Musa berkata kepadanya, "Bolehkah aku mengikutimu?"

Pertanyaan Musa mengandung nada meminta dengan cara halus, bukan membebani atau memaksa. Memang harus demikianlah etika seorang murid kepada gurunya dalam berbicara.

اَتَّبِعُكَ

Bolehkah aku mengikutimu?

Maksudnya, bolehkah aku menemanimu dan mendampingimu.

عَلٰٓى اَنْ تُعَلِّمَنِ مِمَّا عُلِّمْتَ رُشْدًا

supaya kamu mengajarkan kepadaku ilmu yang benar di antara ilmu-ilmu yang telah diajarkan kepadamu.

BAB V
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil dari pengujian Depthwise Separable Convolution Neural Network maka dapat ditarik kesimpulan sebagai berikut :

Dari hasil pengujian yang telah dilakukan, dapat disimpulkan bahwa dengan menerapkan model yang menggunakan metode Depthwise Separable Convolution Neural Network tingkat keberhasilan yang didapatkan untuk memprediksi benda yang telah ditangkap oleh kamera smartphone mencapai tingkat akurasi sebesar 94,58 %. Total precission yang didapat mencapai tingkat 95 %, Total recall yang didapatkan mencapai 94,72 % dan Total F-Measure yang didapatkan mencapai 95 %. Dari hasil pengujian diperoleh hasil klasifikasi yang baik dengan tingkat akurasi mencapai 95%, sehingga model yang dihasilkan dapat memprediksi nama benda dengan hasil baik, Terdapat beberapa faktor yang dapat mempengaruhi tingkat akurasi model dalam memprediksi gambar benda yang digunakan. Seperti dalam cara pengambilan gambar, jika dalam gambar bercampur dengan benda lain maka akan mempengaruhi tingkat akurasi model, Selain itu ukuran pixel gambar yang digunakan pada dataset juga berpengaruh jika ukuran pixel gambar semakin kecil maka akan mempengaruhi hasil yang didapatkan, Jika dataset yang digunakan banyak maka akan mempengaruhi akurasi yang akan didapatkan lebih baik, jika dataset sedikit maka akan mendapatkan tingkat akurasi yang kurang baik saat digunakan pelatihan data. Untuk hardware yang digunakan juga mempengaruhi hasil prediksi seperti besar mega pixel kamera pada

smartphone yang digunakan, semakin tinggi mega pixel kamera maka hasil prediksi juga akan semakin meningkat.

5.2. Saran

Saran untuk pengembangan dalam penelitian selanjutnya adalah sebagai berikut :

1. Dataset yang digunakan masih kurang, sehingga dibutuhkan lebih banyak lagi dataset yang dibutuhkan untuk mencapai tingkat akurasi yang lebih baik lagi.
2. Proses pelatihan data masih membutuhkan waktu yang lama, karena terbatas spesifikasi hardware yang digunakan, untuk melakukan pelatihan data yang lebih cepat dapat menggunakan sewa cloud untuk deep learning atau dengan menggunakan komputer dengan spesifiaksi VGA yang lebih tinggi.
3. Untuk kedepanya dapat dikembangkan menjadi berbagai macam aplikasi, salah satunya adalah pada bidang edukasi yang memanfaatkan deep learning.

DAFTAR PUSTAKA

Gazali, W., Soeparno, H., & Ohliati, J. 2012. *Penerapan Metode Konvolusi Dalam Pengolahan Citra Digital*. Jakarta Barat: Universitas Binus.

Ayumitha, F. H. 2014. *Transliterasi Huruf Latin ke Dalam Aksara Jawa Dengan Menggunakan Decision Tree.* Malang: Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Putra, W. S. E. 2016. *Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) pada Caltech 101*. *JTITS*, vol. 5, no. 1, Mar 2016, doi: 10.12962/j23373539.v5i1.15696.

Supriyono, H., Rahmadzani, R. F., Adhantoro M. S., & Susilo, A. K. 2016. *Rancang Bangun Pembelajaran dan Game Edukatif Pengenalan Aksara Jawa PANDAWA*. Surakarta: Universitas Muhammadiyah Surakarta.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. 2017. *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.* arXiv:1704.04861 [cs].

Albawi, S., Mohammed, T. A., & al-zawi, S. 2017. *Understanding of a convolutional neural network. International Conference on Engineering and Technology (ICET)*, Antalya, 2017, hlm. 1–6. doi: 10.1109/ICEngTechnol.2017.8308186.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. 2017. *ImageNet classification with deep convolutional neural networks. Commun. ACM*, vol. 60, no. 6, hlm. 84–90, Mei 2017, doi: 10.1145/3065386.

Gavai, N. R., Jakhade, Y. A., Tribhuvan, S. A., & Bhattad, R. 2017. *MobileNets for flower classification using TensorFlow*. *International Conference on Big Data, IoT and Data Science (BID)*, Pune, India, 2017, hlm. 154–158, doi: 10.1109/BID.2017.8336590.

Nisa, P, L, K., Maknunah, J., & Syaifulloh A. 2017. *Game Aplikasi Pengenalan Aksara Jawa HANACARAKA Berbasis Android*. Malang: Universitas Merdeka Malang.

Chen, H. Y., & Su, C. Y. 2018. An Enhanced Hybrid Mobilenet. IEEE.

Nurfita , R. D. 2018. *Implementasi Deep Learning Berbasis Tensorflow Untuk Pengenalan Sidik jari*. Surakarta: Universitas Muhammadiyah Surakarta.

Nurhikmat, T. 2018. *Implementasi Deep Learning Untuk Image Classification Menggunakan Algoritma Convolutional Neural Network (CNN) Pada Citra Wayang Golek.* Yogyakarta: Universitas Islam Indonesia.

Aribowo, E. K. 2018. *Digitialisasi Aksara Jawa dan Pemanfaatannya Sebagai Media Pemebelajaran bagi Musyawarah Guru Mata Pelajaran Bahasa Jawa SMP Kabupaten Klaten*. Klaten: Universitas Widya Dharma.

Melania, N. 2018. *Analisis Kemampuan Membaca Bahasa Jawa Pada Siswa Kelas II.* Yogyakarta: Universitas Negeri Yogyakarta.

Utari, N. R. D. *Kemampuan Berbahasa Jawa Pada Siswa Sekolah Dasar di SDN Tandes Kidul I/110 Surabaya*. vol. 1, no. 3, hlm. 10.

Priyanto, A. S. 2019. *Aksara Jawa Ca, Ra, Ka Sebagai Sumber Ide Penciptaan Katya Keramik*. Yogyakarta: Institut Seni Indonesia Yogyakarta.