KLASIFIKASI NOMINAL UANG KERTAS RUPIAH UNTUK TUNANETRA MENGGUNAKAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK*

SKRIPSI

Oleh

Adwitiya Sadhu Prayastita
NIM 192410101054

PROGRAM STUDI SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS JEMBER
2023

KLASIFIKASI NOMINAL UANG KERTAS RUPIAH UNTUK TUNANETRA MENGGUNAKAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK*

SKRIPSI

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat untuk
menyelesaikan Program Studi Sistem Informasi (S1)
dan mencapai gelar Sarjana Komputer

Oleh

Adwitiya Sadhu Prayastita
NIM 192410101054

PROGRAM STUDI SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS JEMBER
2023

PERSEMBAHAN

Skripsi ini saya persembahkan untuk :

1. Allah S.W.T yang senantiasa telah memberikan Rahmat dan hidayah-Nya sehingga diberi kelancaran dan kemudahan dalam jenjang studi perkuliahan hingga penulisan Skripsi sebagai tugas akhir;
2. Ibunda Tunis Dinafsia Ratri dan Ayah Agus Riyono serta keluarga;
3. Seluruh dosen serta mentor yang telah memberikan ilmu kepada penulis;
4. Sahabat serta teman yang memberikan doa serta dukungannya;
5. Almamater Program Studi Sistem Informasi, Fakultas Ilmu Komputer, Universitas Jember;

MOTTO

"*You Don't Have to be Great to Start, But You Have to Start to be Great.*"

- Zig Ziglar -

"You Can Do It, Enjoy World, Enjoy Life."

- Sakura Miko -

"The Most Important Things Seem to be Invisible To Eye."

- Hoshimachi Suisei -

PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Adwitya Sadhu Prayastita

NIM : 192410101054

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul "Klasifikasi Nominal Uang Kertas Rupiah Untuk Tunanetra Menggunakan Algoritma *Convolutional Neural Network*" adalah benar-benar hasil karya saya sendiri, kecuali kutipan yang telah disebutkan sumbernya, belum pernah diajukan pada institusi mana pun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa ada paksaan dan tekanan dari pihak mana pun serta bersedia mendapat sanksi akademik jika di kemudian hari pernyataan ini tidak benar

Jember,

Yang Menyatakan

Adwitya Sadhu P.

NIM 192410101054

SKRIPSI

KLASIFIKASI NOMINAL UANG KERTAS RUPIAH UNTUK TUNANETRA MENGGUNAKAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK*

Oleh

Adwitiya Sadhu Prayastita
NIM 192410101054

Pembimbing:

Dosen Pembmbing Utama : Nelly Oktavia Adiwijaya S.Si.,MT.

Dosen Pembimbing Pendamping : Januar Adi Putra S.Kom., M.Kom

HALAMAN PERSETUJUAN

Skripsi berjudul *Klasifikasi Nominal Uang Kertas Rupiah Untuk Tunanetra Menggunakan Algoritma Convolutional Neural Network* telah diuji dan disahkan oleh Fakultas Ilmu Komputer Universitas Jember pada:

Hari : Kamis

Tanggal : 15 Juni 2023

Tempat : Fakultas Ilmu Komputer Universitas Jember

Pembimbing — Tanda Tangan

1. Pembimbing Utama
   Nama : Nelly Oktavia Adiwijaya S.Si.,MT. (.....)
   NIP : 198410242009122008
2. Pembimbing Anggota
   Nama : Januar Adi Putra S.Kom., M.Kom (.....)
   NIP : 760017015

Penguji

1. Penguji Utama
   Nama : Muhamad Arief Hidayat S.Kom,.M.Kom (.....)
   NIP : 198101232010121003

   Penguji Anggota 1
   Nama : Gayatri Dwi Santika S.SI., M.Kom (.....)

ABSTRAK

Penggunaan *blind code* pada uang kertas Rupiah Indonesia dinilai tidak dapat membantu penyandang tunanetra. Maka dari itu diperlukan sebuah terobosan lain, salah satunya dengan mengajarkan mesin agar dapat melihat layaknya makhluk biologis atau biasa disebut *Computer Vision*. *Convolutional Neural Network* (CNN) dapat menjalankan tugas tersebut yang mana CNN dapat memproses kedalaman atau *depth* citra. Selain itu diperkirakan latarbelakang akan memengaruhi peforma model sehingga diperlukan sebuah metode untuk mengisolasikan objek dari citra atau memisahkan objek dengan latar belakang, dalam kasus ini digunakan Segmentasi dengan menggunakan arsitektur U-Net. Selain metode yang digunakan, diperlukan pula perangkat yang *mobile* sehingga penyandang dapat menggunakannya sewaktu waktu, maka dari itu digunakan pula arsitektur klasifikasi MobileNetV2 yang diklaim memiliki arsitektur ringan sehingga cocok digunakan pada perangkat *mobile* seperti *smartphone*. Maka dari itu penelitian ini akan membandingkan 2 buah skenario utama yaitu, Skenario klasifikasi MobileNetV2 tanpa segmentasi U-Net dengan skenario klasifikasi MobileNetV2 dengan data tersegmentasi U-Net. Data yang digunakan akan dibagi menjadi 3 jenis data, dimana pada data jenis 1 didapatkan dengan menggunakan webcam msi gf63thin dengan ukuran 224x224 pixel. Data jenis 2 didapatkan dengan menggunakan kamera xiaomi redmi 9 dengan ukuran 2304x4096 pixel. Dan data jenis 3 yang didapatkan melalui kaggle secara open source dengan ukuran 1600x900 pixel. Data jenis 3 digunakan untuk memperbanyak citra atau memperkaya tiap kelas citra. Setelah dilakukan percobaan dan beberapa *tuning* hasil terbaik didapatkan pada skenario utama klasifikasi MobileNetV2 tanpa data tersegmentasi U-Net dengan pembagian data 75% data latih, 25% data validasi. Model tersebut berhasil mendapatkan metrik 99% *accuracy*, 99% *precision*, 99% *recall*, 99% *f1-score*.

Kata kunci : Convolutional Neural Network, Currency Classification, MobileNetV2, U-Net Segmentation, Visually Impaired

RINGKASAN

Klasifikasi Nominal Uang Kertas Rupiah Untuk Tunanetra Menggunakan Algoritma Convolutional Neural Network; Adwitya Sadhu Prayastita; 62 Halaman; Program Studi Sistem Informasi Fakultas Ilmu Komputer.

Sebanyak 1% dari total penduduk Indonesia mengalami kebutaan. *Blind code* yang dikeluarkan oleh Bank Indonesia (BI) dinilai kurang efektif dengan masih banyaknya penyandang tunanetra yang belum mengenali *blind code* itu sendiri dan kondisi uang seperti lusuh, kusut, dan sebagainya. Tidak setiap saat penyandang tunanetra menemui transaksi yang benar benar jujur dan tidak setiap saat juga *guardian* penyandang tunanetra tersedia, maka dari itu penyandang tunanetra perlu untuk dapat mengenali nominal itu sendiri. Melalui machine learning, dimana data *input* yang digunakan dipelajari pola sehingga dapat menghasilkan model yang dapat melakukan tugasnya. Deep learning merupakan sub-kriteria dari machine learning, dimana proses analisis pola dilakukan oleh mesin sepenuhnya. Salah satu metode yang dapat digunakan adalah *Convolutional Neural Network* (CNN). Dalam penelitian ini dirancang dua skenario utama yaitu klasifikasi MobileNetV2 disertai menggunakan segmentasi U-Net dan tanpa segmentasi U-net. Segmentasi diharapkan dapat membantu tugas klasifikasi agar dapat lebih tepat dengan cara mengekstrak target objek dari citra. Hasil skenario terbaik yang didapatkan adalah klasifikasi MobileNetV2 tanpa menggunakan segmentasi U-Net. Dengan skenario pembagian data 75% data latih, 25% data validasi. Model tersebut berhasil mendapatkan metrik 99% *accuracy*, 99% *precision*, 99% *recall*, 99% *f1-score*.

PRAKATA

Puji dan syukur kehadirat Allah Subhanallahu wa Ta'ala atas segala rahmat serta karunia-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul "Klasifikasi Nominal Uang Kertas Rupiah Untuk Tunanetra Menggunakan Algoritma *Convolutional Neural Network*" dengan baik dan selesai pada waktu yang diharapkan. Perancangan skripsi ini dilakukan untuk memenuhi salah satu syarat untuk menyelesaikan jenjang studi Pendidikan Strata 1 (S1) program studi Sistem Informasi fakultas Ilmu Komputer Universitas Jember.

Perancangan skripsi ini tentu tidak lepas dari bantuan beberapa pihak, maka dari itu penulis menyampaikan ucapan terimakasih kepada :

1. Ibunda Dra. Tunis Dinafsia Ratri dan Ayahanda Drs. Agus Riyono serta keluarga yang selalu memberikan *support* dan mendoakan saya untuk kelancaran saya dalam menempuh pendidikan,
2. Prof. Drs. Antonius Cahya Prihandoko, M.App.Sc, Ph.D, selaku Dekan Fakultas Ilmu Komputer,
3. Ibu Nelly Oktavia Adiwijaya S.Si.,MT selaku dosen pembimbing utama dan Bapak Januar Adi Putra S.Kom., M.Kom selaku dosen pembimbing pendamping yang selalu meluangkan waktu dan pikiran, serta dengan sabar telah membimbing dan memberikan pengetahuan sehingga perancangan skripsi dapat dilaksanakan hingga selesai,
4. Bapak Muhamad Arief Hidayat S.Kom,.M.Kom dan Gayatri Dwi Santika S.SI., M.Kom selaku Dosen Pembahas,
5. Sakura Miko dan Hoshimachi Suisei yang telah menghibur saya selama perancangan skripsi,
6. Sahabat serta teman teman yang memberikan dukungan serta doa,
7. Seluruh pihak dan orang-orang baik yang telah membantu dan tidak dapat disebutkan satu-persatu.

DAFTAR ISI

Halaman

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1. PENDAHULUAN

1.1. Latar Belakang

Manusia menggunakan lebih dari 80% informasi yang didapatkan dari indera visual (INFODATIN, 2018). Maka dari itu penyandang tunanetra tidak dapat melaksanakan kegiatan yang berkaitan dengan transaksi menggunakan uang. Hal ini dibuktikan dengan banyaknya penyandang tunanetra yang masih kesulitan dalam mengenali uang dalam bertransaksi (Pamungkas, et al., 2021).

Secara global terdapat setidaknya 2.2 miliar orang dengan gangguan pengelihatan, sekitar 50% diantaranya mengalami kebutaan (WHO, 2022). Penyandang tunanetra di Indoensia telah mencapai 1% dari total penduduk. Bank Indonesia (BI) telah menerbitkan *blind code* sebagai upaya membantu penyandang tunanetra, namun *blind code* belum dikenali secara meneyeluruh (Pamungkas et al., 2022). *Blind code* juga dinilai susah dikenali apabila uang kertas dalam keadaan basah, lusuh, dan terlipat (Pertuni, 2016). Maka dari itu dilakukan penelitian terkait identifikasi nominal uang dengan cara melatih mesin agar dapat mengidentifikasi nominal uang ketas (Pamungkas, et al., 2021). Identifikasi nominal uang kertas untuk tunanetra dapat dilakukan dengan metode klasifikasi citra menggunakan algoritma *Convolutional Neural Network*. Metode ini telah digunakan di berbagai domain yang memiliki hubungan erat dengan citra, seperti *face recognition, object detection, gesture recognition* (Dhillon & Verma, 2020) dan terbukti unggul dalam pemrosesan citra (Sharma, Jain, & Mishr, 2018). Hal ini dikarenakan CNN mampu memproses *depth* pada citra (Ajit et al., 2020).

Klasifikasi citra menggunakan CNN sejatinya telah memberikan performa yang bagus, hal ini dibuktikan dengan banyaknya penelitian terkait pengklasifikasian citra khususnya citra yang terkait dengan nominal uang. Penelitian terhadap deteksi nominal uang kertas Rupee Pakistan dengan menggunakan arsitektur AlexNet mampu mendapatkan akurasi sebesar 82% dalam mengidentifikasikan nominal uang Rupee Pakistan (Imad et al., 2020).

Implementasi CNN terkait identifikasi nominal uang juga dilakukan pada perangkat *smartphone* dengan mencipatakan sebuah jaringan ringan yang didasarkan dari *dense-connection*, *multi dilatation*, dan *depth-wise separable*. Arsitektur buatan tersebut mampu mengidentifikasikan nominal uang kertas India dengan hasil akurasi 96.75% (Singh et al., 2022). Keberagaman latar belakang dapat memengaruhi peforma model yang dirancang, maka dari itu diperlukan sebuah teknik untuk menyisihkan objek target dari citra. Penelitian terkait identifikasi nominal uang dengan menggunakan gabungan teknik antara deteksi objek yang dilakukan dengan Faster R-CNN guna mengisolasi objek dan ResNet, sebuah arsitektur CNN sebagai *feature extractor* sekaligus klasifikasi. Kombinasi tersebut membuahkan hasil nilai *f1-score* sebesar 96% pada uang kertas dan koin Korea dan *f1-score* sebesar 94.78% pada uang kertas dan koin Jordania (Park et al., 2020).

Penelitian ini menggunakan segmentasi karena segmentasi sejatinya dapat mengisolasi objek yang kita inginkan dari citra (Tamilselvan & Murugesan, 2018) yang secara langsung memiliki kesamaan tujuan dengan teknik deteksi objek. Berdasarkan informasi sebelumnya diharapkan akurasi pada klasifikasi citra dapat meningkat dengan adanya segmentasi. Segmentasi juga dapat dilakukan dengan menggunakan arsitektur CNN, yaitu U-Net (Vaze et al., 2020). U-Net telah memberikan banyak kontribusi terutama pada dunia medis dan biologis dengan kemampuan segmentasinya yang baik (Quoc et al., 2020). Karena segmentasi merupakan sebuah teknik yang berbeda dari klasifikasi maka memerlukan metric pengukuran selain *confusion matrix*. Salah satunya ialah *Intersection over Union* (IoU) yang merupakan metric yang biasa digunakan untuk tugas *semantic segmentation* seperti U-Net (Shukla et al., 2023).

Berdasarkan informasi yang telah dijabarkan diatas, pokok bahasan penelitian ini akan menggunakan arsitektur CNN, MobileNetV2 pada klasifikasi citra. MobileNetV2 memiliki performa yang baik pada perangkat *mobile* (Sandler et al., 2018). Segmentasi dilakukan dengan menggunakan U-Net, dengan pertimbangan bahwa U-Net juga menerapkan *convolutional network* serta U-Net yang dapat berjalan secara efisien meskipun data relatif kecil (Quoc et al., 2020).

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah dijabarkan sebelumnya, berikut rumusan masalah yang dirancang dalam penelitian ini :

1. Bagaimana hasil peforma klasifikasi nominal uang dengan arsitektur *convolutional neural network* U-Net dan arsitektur *convolutional neural network* MobileNetV2?
2. Bagaimana perbandingan dari performa klasifikasi menggunakan arsitektur MobileNetV2 dengan segmentasi arsitektur U-Net dan tanpa menggunakan segmentasi?

1.3. Tujuan Penelitian

Tujuan umum dirancangnya penelitian ini adalah untuk menambah wawasan dan pengetahuan dalam penggunaan pengidentifikasian citra menggunakan metode *Convolutional Neural Network*. Sedangkan tujuan khusus penelitian ini adalah sebagai berikut :

1. Mengetahui hasil peforma identifikasi nominal uang dengan arsitektur *convolutional neural network* U-Net dan arsitektur *convolutional neural network* MobileNetV2
2. Mengetahui perbandingan dari performa identifikasi menggunakan arsitektur MobileNetV2 dengan segmentasi arsitektur U-Net dan tanpa menggunakan segmentasi

1.4. Manfaat Penelitian

Manfaat yang diharapkan dari dirancangnya penelitian ini adalah :

a. **Bagi Fakultas Ilmu Komputer**, hasil penelitian ini diharapkan dapat berguna untuk dijadikan bahan diskusi mata kuliah bidang ilmu terkait
b. **Bagi penyandang Tunanetra**, hasil penelitian ini diharapkan dapat membantu dalam pengidentifikasian nominal uang rupiah
c. **Bagi mahasiswa**, hasil penelitian ini diharapkan dapat membantu dan memenuhi penulisan tugas akhir Skripsi

d. **Bagi peneliti selanjutnya**, hasil penelitian ini diharapkan dapat dijadikan bahan informasi, acuan untuk dilakukannya penelitian lebih lanjut dikemudian hari

1.5. Batasan Masalah

Batasan masalah dirancang agar penelitian tidak meluas dan pembahasan fokus pada linkup yang telah ditetapkan.

1. Jenis uang yang digunakan adalah uang kertas dengan mata uang Rupiah Indonesia.
2. Klasifikasi nominal uang terbatas pada uang kertas tahun emisi 2016 tidak termasuk UPK (pecahan Rp.75.000).
3. Citra uang yang digunakan didapatkan secara mandiri dengan menggunakan kamera handphone Xiaomi Redmi 9 dan webcam MSI GF63 thin yang diberi jarak $\pm$20 cm terhadap objek serta citra yang didapat dari kaggle secara open source, dengan total citra sebanyak 1378 citra.
4. Penelitian berfokus pada penemuan model terbaik uang dengan perpaduan MobileNetV2 dan U-Net disertai implementasinya pada perangkat *smartphone*.
5. Penelitian berfokus untuk identifikasi besaran nominal, tidak termasuk deteksi keaslian atau pembeda terhadap uang palsu atau mainan.

BAB 2. TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

Penelitian terdahulu terkait identifikasi nominal uang kertas untuk membantu penyandang tunanetra menggunakan *convolutional neural network* (CNN) yang digunakan sebagai referensi adalah penelitian yang dilakukan oleh Imad Muhammad et al (2020). Penelitian ini menunjukkan bagaimana CNN bagaimana CNN dilakukan dalam mengidentifikasikan besaran nominal uang kertas dengan menggunakan arsitektur Alex-Net dan Support Vector Machine (SVM) dengan uang kertas dan koin Rupee Pakistan dengan pecahan nominal Rs 10, Rs 20, Rs 50, Rs 100, Rs 500, Rs 1000, dan Rs 5000 dengan total data sebanyak 154483 citra. Penelitian ini juga menunjukkan augmentasi atau rekayasa data dilakukan untuk memperkaya data latih, dengan teknik *addition, scaling, blur,* dan *rotation*. Secara garis besar penelitian ini dibagi menjadi 2 tahap, yaitu proses training dengan pre-trained Alex-Net dan penggunaan kamera yang dapat menangkap video agar dapat diidentifikasikan dengan model yang telah dirancang sebelumnya. *Training* model membuahkan 99.96% sedang pada saat ekperimen menggunakan kamera mendapatkan *overall accuracy* 96.8%.

Penelitian lainnyta yang dilakukan Singh Mandhatya et al (2022) menunjukkan MobileNet, sebuah arsitektur CNN yang ringan sehingga dapat dijalankan pada perangkat *mobile*. Penelitian ini merancang sebuah *framework* IPCRNet yang menggabungkan arsitektur MobileNet sebagai *feature extractor* dan *dense-connection* serta *multi-dilatation* sebagai *front-end* dan Contextual block sebagai *back-end*. Menggunakan *Indian Paper Currency Dataset* (IPCD) sebagai dataset yang diambil dengan bermacam variasi *smartphone* dengan pendekatan uang diletakkan pada luar ruangan, dalam ruangan, keadaan uang terlipat, terpotong dan ukuran penuh dengan jumlah akhir 13.399 citra. Selain pengambilan mandiri dataset juga dikumpulkan melalui *crowd-sourcing* dengan jumlah akhir 109,550 citra. Seleksi citra dilakukan dengan jumlah akhir 50,263 gambar. Penelitian ini

juga menerapkan rekayasa data dengan teknik *shift* secara horizontal dan vertikal, *zoom-in* dan *zoom-out* secara random (80%-120%) dan *rotation* sebesar 20 derajat searah jarum jam serta diperhalus. Model dilatih dengan menggunakan 40 epoch, 8, 16 dan 32 batch size dan learning rate sebesar 10^{-3} serta hyperparameter *depth multiplier, alpha* dan droput yang di atur masing masing 1,1, dan 10^{-3}. Uji coba dilakukan pada IPCD dataset dengan menggunakan average accuracy sebagai evaluasi. Rancangan penelitian ini membuahkan hasil *average accuracy* sebesar 96.75%, penelitian ini juga diimplementasikan ke aplikasi "Roshni-Currency Recognizer"

Penelitian selanjutnya dilakukan Park Chanhum et al (2020) dengan judul memberikan gambaran serta membuktikan bagaimana pengaruh latar belakang objek yang bervariasi dapat memengaruhi peforma model. Penelitian ini merancang dengan tiga tahapan utama dalam deteksi nominal uang, yaitu deteksi objek dengan Faster R-CNN yang dibekali VGG-16 sebagai feature extraktor. Pada tahap kedua yaitu *geometric constrain,* hal ini dilakukan dengan cara menghapus hasil deteksi pada tahap pertama yang terdeteksi sebagai *false positive*, cara penghapusannya sendiri dengan menggunakan rasio dari tinggi dan lebar dari *detection box* atau dengan membandingkan *ground truth* dari target uang koin dan uang kertas. Cara kedua dengan berdasarkan besar *detection box*. Pada tahap ketiga kembali menggunakan Faster R-CNN namun dengan ResNet-18 sebagai fitur ekstraktornya. Eksperimen dilakukan dengan menggunakan dataset Jordanian Dinar (JOD) dan 6.400 citra Won Korea yang diambil dengan *smartphone* Galaxy Note 5 dengan hasil nilai *f1-score* hasil masing masing pada dataset sebesar 96% dan 94.78%.

Penelitian lainnya terkait metode alternatif dari deteksi objek untuk menghilangkan latar belakang yang dilakukan Vaze Sagar et al (2022) menunjukkan bagaimana segmentasi diterapkan pada perangkat *mobile* dengan mengenalkan '*thin*' U-Net, U-net yang telah dimodifikasi pada blok konvolusinya dengan cara mereduksi jumlah *filters* pada arsitektur U-Net yang asli. Selain itu, penelitian ini tidak hanya merancang *thin* U-Net saja, akan tetapi juga merancang *thin* U-Net yang menggunakan *depth-wise separable convolution*, yang dapat

mereduksi ukuran model secara signifikan serta dirancangnya *thin* U-Net yang menggunakan *knowledge distillation*. Objek pada penelitian ini adalah citra *Ultrasound* yang didapatkan dari kaggle, dengan membandingkan ke empat arsitektur penelitian membuahkan hasil 98.89% akurasi pada original U-Net, 98.89% akurasi pada *thin* U-Net, 98.72% akurasi pada *thin separable* U-Net, 98.83% akurasi pada *distilled thin* U-Net.

Berdasarkan penjabaran penelitian terdahulu sebelumnya *Tabel 2.1* menunjukkan perbandingan dengan penelitian yang dilakukan

Tabel 2.1 Tabel Perbandingan Penelitian Terdahulu

No	Peneliti (Tahun)	Judul Penelitian	Parameter Penelitian	Hasil Penelitian	Gap Penelitian
1.	Imad, Ullah, Hassan dan Naimullah (2020)	*Pakistani Currency Recognition to Assist Blind Person Based on Convolutional Neural Network*	Uang pecahan Rupee Rs 10, Rs 20, Rs 50, Rs 100, Rs 500, Rs 1.000, dan Rs 5.000. Teknik augmentasi *addition, scaling, blur,* dan *rotation*	Menunjukkan penggunaan CNN terhadap nominal uang Rupee dan penerapannya. Serta berhasil mendapatkan akurasi klasifikasi sebesar 96.85% menggunakan Alex-Net dan *support vector machine* sebagai pembeda uang koin dan kertas dengan akurasi sebesar 76% untuk koin dan 82% untuk uang kertas	Penggunaan arsitektur cnn yang berbeda, dimana pada penelitian ini tidak menggunakan arsitektur yang mobile friendly
2.	Singh, Joohi, Kanroo, Verma dan Goyal (2022)	*IPCRF : An End-To-End Indian Paper Currency Recognition Framework For Blind And Visually Impaired People*	IPCD (Indian Paper Currency Dataset) yang terdiri dari pecahan keluaran lama dan baru 10 Rupee,20 Rupee, 50 Rupee, 100 Rupee dan	Menunjukkan CNN juga bisa diterapkan pada perangkat *mobile* seperti *smartphone* dengan menggunakan arsitektur MobileNetV2. Penelitian ini menghasilkan	Menunjukkan adanya arsitektur yang mobile-friendly dan dipergunakan beberapa hyperparameter untuk perancangan model

No	Peneliti (Tahun)	Judul Penelitian	Parameter Penelitian	Hasil Penelitian	Gap Penelitian
			pecahan baru 200 rupee, 500 rupee dan 1000 rupee	akurasi sebesar 96.75%	
3.	Park, Cho, Baek, Choi, dan Park (2020)	*Deep Feature-Based Three-Stage Detection of Banknotes and Coins for Assisting Visually Impaired People*	Jordanian Dinar (JOD) dan 6.400 citra Won Korea dan latar belakang objek	Latar belakang terbukti memengaruhi peforma dari model yang ada. Pada penelitian ini model yang dirancang menghasilkan nilai *f1-score* sebesar 94.78% pada uang koin dan kertas yang menunjukkan peningkatan pada model yang tidak menyisihkan latar belakang	Menggunakan segmentasi sebagai alternatif deteksi objek. Karena deteksi objek telah dibuktikan memberikan peforma yang baik
4.	Vaze, Xie dan Namburete (2020)	*Low-Memory CNNs Enabling Real-Time Ultrasound Segmentation Towards Mobile Deployment*	*Ultrasound Images* dari Kaggle Competition dan variasi arsitektur U-Net	Menunjukkan bagaimana segmentasi dapat dilakukan pada perangkat *mobile*. Penelitian ini merancang empat model dan masing masing memberikan hasil akurasi sebesar 98.89% akurasi pada original U-Net, 98.89% akurasi pada *thin* U-Net, 98.72% akurasi pada *thin separable* U-Net, 98.83% akurasi pada *distilled thin* U-Net.	Rancangan arsitektur serta hyperparameter digunakan dalam perancangan model segmentasi. Pada penelitian ini tidak dilakukan klasifikasi, hanya segmentasi U-Net

2.2. Mata Uang Rupiah Indonesia

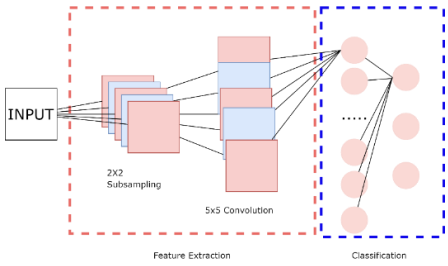
Uang Rupiah Indonesia dibedakan menjadi dua jenis, logam dan kertas. Tentunya uang yang telah menjadi salah satu media transaksi yang sah menjadi bagian dari kehidupan sehari hari, Bank Indonesia menerbitkan beberapa keluaran atau yang disebut Tahun Emisi (TE) seperti TE 2006, TE 2016, dan TE 2020. Bank Indonesia juga menerbitkan *blind code* pada setiap tahun emisi agar penyandang tunanetra dapat menggunakannya.

2.3. Deep Learning

Deep learning (DL) atau kerap disebut *deep neural network* sejatinya merupakan bagian dari *machine learning*, perbedaan antar dua pendekatan tersebut terletak pada teknik yang berbeda dimana DL menggunakan banyak lapisan *artificial neural network* (ANN) sehingga terciptanya '*deep*' lapisan. (Li et al., 2023). Selain memiliki lapisan yang dapat memproses data secara non-linear, DL memiliki aspek lain dimana DL dapat mempelajari fitur atau merepresentasikan ciri dari input secara otomatis, dengan kata lain *deep learning* dapat mengekstrak fitur yang diperlukan secara mandiri (Lauzon, 2012).

2.4. Convolutional Neural Network

Penelitian dari Hubel dan Weisel (1995) adalah penelitian yang mendasari CNN, mereka mempelajari bahwa adanya sel yang dapat mendeteksi cahaya pada hewan.(Pamungkas et al., 2022)



Gambar 2.1 Struktur *Convolutional Neural Network* (Sharma et al., 2018)

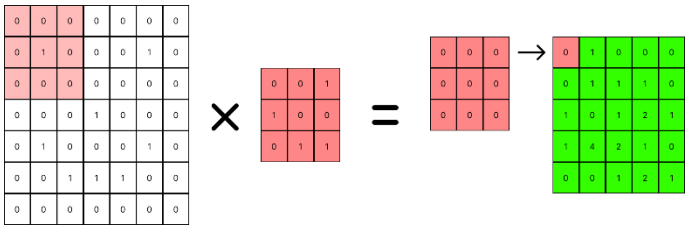
Arsitektur CNN memiliki beberapa komponen lapisan dasar seperti pada Gambar 2.1 yang terdiri dari *input layer*, *convolution layer* dan *pooling layer* sebagai *feature extractor* dan *fully connected layer* sebagai klasifikasi.

2.4.1. Input Layer

Lapisan pertama pada CNN adalah Input layer, seperti pada gambar 2. Perlu dilakukan resize pada gambar terlebih dahulu sebelum lanjut ke *feature extraction* layer

2.4.2. Convolution layer

Merupakan lapisan dasar dari CNN yang juga menjadi identitas dari CNN itu sendiri.



Gambar 2.2 Proses Konvolusi (Ajit et al., 2020)

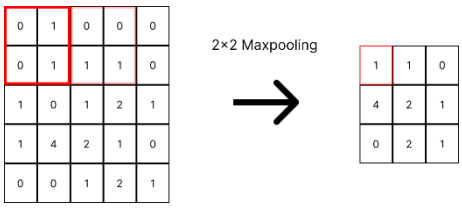
Pada *Gambar 2.2* memberikan ilustrasi bagaimana proses konvolusi dioperasikan, dimana dua matrix (matrix input dan matrix kernel dikalikan) dan ditambahkan setiap elemennya untuk setiap besar kernel. Konvolusi sendiri dapat dinotasikan kedalam persamaan matematika seperti pada *Persamaan 2.1*

$$y_n = (X \times W)_n = \sum_{K=-\infty}^{n} X_K \times W_{n-K} \qquad \text{Persamaan 2.1}$$

Dengan X sebagai matriks input dan W sebagai kernel serta y sebagai hasil konvolusi dari kedua matriks tersebut, dengan n merupakan index output dan K merupakan index input.

2.4.3. Pooling Layer

Lapisan selanjutnya adalah *pooling layer*, tujuan utama dari lapisan ini adalah untuk mengurangi dimensi spasial dari *feature* map yang didapat setelah konvolusi tanpa menghilangkan informasi penting (Ajit et al., 2020).



Gambar 2.3 Proses pooling dengan *Max pooling* (Ajit et al., 2020)

Ilustrasi *Gambar 2.3* menunjukkan bagaimana *pooling layer* bekerja, dengan mengambil nilai yang terbesar dalam matrix dengan besar *n*. Dengan begitu untuk setiap *n*×*n* kernel pooling akan menghasilkan 1 nilai, jika dilihat pada Gambar 2.3 maka untuk satu kotak merah (2x2 kernel Maxpooling) akan menghasilkan nilai ‘1’ pada matriks kanan pada kotak merah.

2.4.4. Fully Connected Layer

Lapisan terakhir adalah *Fully Connected Layer* (FC) dimana lapisan ini merupakan tugas klasifikasi dilakukan pada CNN, dengan kata lain lapisan ini sama dengan *neural network*. Sebelum masuk ke FC *output* dari CNN terlebih dahulu di *flatten* agar menjadi array 1-d atau vector, dikarenakan output dari CNN adalah tensor (array n-dim).

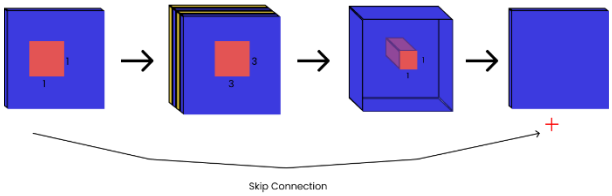
2.4.5. MobileNetV2 Architecture

MobileNetV2 diciptakan oleh Google sebagai pengembangan dari MobileNetV1. Pada versi MobileNetV2 digunakan *depthwise separable convolution* yang mana merupakan peningkatan dari konvolusi tradisional.

$$Conv = {D_r}^2 \times M \times {D_v}^2 \times N$$ Persamaan 2.2

$$D - wise\ Conv = {D_r}^2 \times {D_v}^2 \times M + M \times {D_v}^2 \times N \qquad \text{Persamaan 2.3}$$

Pada kedua perhitungan biaya komputasi diatas dapat dihitung dengan *input* $(D_r \times D_r \times M)$ yang sama dan filter D_v yang sama, *Persamaan 2.3* cenderung memiliki biaya yang lebih rendah. Dengan *M* adalah *depth* dari input citra dan *N* adalah banyak filter.

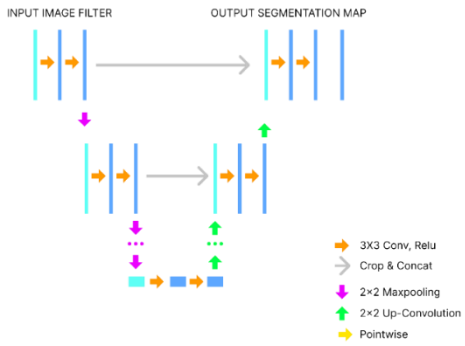


Gambar 2.4 Ilustrasi Blok Inverted Residual

Selain *D-Wise*, MobileNetV2 juga menggunakan *Inverted Residual* dan *Linear Bottlenecks*. Seperti pada *Gambar 2.4* input akan diproses dengan *pointwise* untuk meningkatkan jumlah channel kemudian melewati *depthwise-separable* yang akan bertugas sebagai feature extraktor, setelah itu akan diproses dengan *pointwise* kembali untuk mengembalikan jumlah channel ke semula. Output dari proses tersebut kemudian ditambahkan (*element-wise*) terhadap tensor input sebelum diproses untuk menciptakan output baru. Pada *pointwise* terakhir digunakan *linear activation* guna mengurangi biaya komputasi (Sandler et al., 2018).

2.4.6. U-Net

U-Net adalah arsitektur CNN yang berjalan pada tugas segmentasi, U-Net juga merupakan arsitektur yang dapat berjalan secara efisien pada data yang relative kecil (Quoc et al., 2020)



Gambar 2.5 Struktur U-Net (Siddique et al., 2021)

Cara kerja U-Net sendiri sama dengan cara kerja *conv network* pada umumnya dimana input dilakukan operasi konvolusi sebesar 3x3 dan dilakukan *maxpooling*, proses tersebut diulangi sebanyak empat kali pada desain aslinya (Siddique et al., 2021). Namun tidak menutup kemungkinan untuk dapat melebihi atau kurang bergantung pada kebutuhan, proses ini dinamakan *contracting path* yang digunakan untuk extrak fitur dan mengurangi resolusi citra. Pada U-Net juga terdapat *expansive path*, pada proses ini citra dilakukan up-convolutional layer untuk menaikkan resolusi citra.

2.5. Confusion Matrix

Dalam pengembangan model *machine learning* (ML) diperlukan evaluasi sebagai tolak ukur suatu model yang telah melewati proses training, salah satunya adalah *confusion matrix* (CM) (Pamungkas et al., 2022).

Tabel 2.2 Tabel *confusion matrix* biner (Pamungkas et al., 2022)

Class	Actual (Positive)	Actual (Negative)
Predicted (True)	True Positive	True Negative
Predicted (False)	False Negative	False Positive

Confusion matrix pada *Tabel 2.2* memiliki beberapa kelas, yaitu True Positive (TP), proporsi dimana keadaan prediksi dan aktual diklasifikasikan masuk kedalam kelas yang sama. True Negative (TN), proporsi dimana keadaan prediksi dan aktual diklasifikasikan masuk kedalam kelas B untuk target kelas A. False Positive (FP), proporsi untuk target kelas A dimana keadaan aktual diklasifikasikan kedalam kelas B, namun prediksi diklasifikasikan kedalam kelas A. False Negative

(FN), proporsi untuk target kelas A dimana keadaan aktual diklasifikasikan kedalam kelas B, namun prediksi diklasifikasikan kedalam kelas B. (T.K. et al., 2021). Dari beberapa kelas tersebut, kita dapat menentukan akurasi, recall, presisi dan f1-score untuk mengevaluasi bagaimana peforma model yang dikembangkan.

a. Akurasi

$$Akurasi = \frac{(TP + TN)}{(TP + TN + FN + FP)} \times 100\% \qquad \text{Persamaan 2.4}$$

Merujuk pada *persamaan 2.7*, akurasi didapatkan dengan mencari persentase dari kejadian yang diprediksi benar secara aktual (*True Positive*) dan kejadian yang diprediksi benar namun secara aktual salah (*True Negative*) terhadap seluruh kejadian.

b. Presisi

Metric yang digunakan untuk mengukur proporsi dari prediksi positif yang diidentifikasikan secara tepat. Presisi dapat dihitung pada *Persamaan 2.5*.

$$Presisi = \frac{(TP)}{(TP + FP)} \qquad \text{Persamaan 2.5}$$

Metric presisi mengukur kemampuan model untuk dapat menghasilkan prediksi yang presisi

c. Recall

Metric yang digunakan untuk mengukur proporsi dari kejadian aktual yang diprediksi benar oleh model. Recall dapat dihitung dengan *Persamaan 2.6*.

$$Recall = \frac{(TP)}{(TP + FN)} \qquad \text{Persamaan 2.6}$$

Metric recall mengukur kemampuan model untuk mengidentifikasi kelas yang relevan secara tepat.

d. F1-Score

Metric F1-score menghitung weight average dari presisi dan recall, dan merupakan ukuran akurasi secara keseluruhan model. F-1 Score dapat dihitung pada *Persamaan 2.7*.

$$F1 - score = 2 * \frac{(Presisi * Recall)}{(Presisi + Recall)} \quad \text{Persamaan 2.7}$$

F-1 score lebih dipercaya ketika terdapat ketidakseimbangan jumlah antara kejadian positif dan kejadian negatif. Jika jumlah kejadian positif lebih sedikit dari pada jumlah kejadian negatif, maka model cenderung memprediksi kejadian menjadi kejadian negatif sehingga mengakibatkan memiliki akurasi tinggi, namun recall rendah.

2.5.2. Multi-class Confusion Matrix

Pada pengembangan model machine learning dengan kelas lebih dari dua, dibutuhkan perhitungan khusus agar dapat mengevaluasi setiap kelas secara akurat. *Multi-class Confusion Matrix* ditandai dengan jumlah kelas lebih dari dua dan subkelas berjumlah satu. *Multi-class Confusion Matrix* (MCCM) juga memiliki tabelnya sendiri seperti pada *Tabel 2.3*.

Tabel 2.3 Contoh tabel *Multi-class Confusion Matrix* (Grandini et al., 2020)

		PREDIKSI			
	Kelas	**A**	**B**	**C**	**Total**
Aktual	**A**	1	2	4	7
	B	5	5	0	10
	C	2	1	9	12
	Total	8	8	13	58

$$MacroAveragePrecision = \frac{\sum_{m=1}^{K} Precision_m}{K} \quad \text{Persamaan 2.8}$$

$$MacroAverageRecall = \frac{\sum_{m=1}^{K} Recall_m}{K} \quad \text{Persamaan 2.9}$$

$$MacroF-1Score = 2 \times \frac{(MAP \times MAR)}{(MAP + MAR)}$$ Persamaan 2.10

Dengan K merupakan jumlah kelas, maka average, precision dan akurasi merupakan perhitungan rerata untuk setiap kelas dan disebut Macro Average, sedangkan untuk Macro F-1 Score merupakan perhitungan antara Macro Average Recall dan Macro average Precision, dengan rumus yang sama seperti pada *Persamaan 2.10*.

2.6. Intersection over Union

Intersection over Union atau IoU merupakan perhitungan performa standar yang biasa digunakan pada pemrosesan citra yang berkaitan dengan menggolongkan area dari sebuah citra seperti segmentasi dan deteksi objek. Konsep dibalik IoU adalah mencari keterkaitan atau persamaan antara area yang terprediksi dengan area yang benar atau area yang dijadikan target yang tersedia pada citra (Rahman & Wang, 2016). Mean IoU dapat dihitung dari rerata IoU untuk setiap kelasnya, IoU dapat dihitung dengan *Persamaan 2.11*.

$$IoU = \frac{area\ irisan}{area\ gabungan}$$ Persamaan 2.11

BAB 3. METODE PENELITIAN

3.1. Jenis Penelitian

Penelitian ini dirancang dengan jenis penelitian eksperimen yang bertujuan untuk menemukan skenario dan model yang paling baik dan efektif dalam identifikasi nominal uang untuk membantu tunanetra menggunakan CNN. Agar dapat mencapai tujuan ini, diperlukan beberapa kali skenario dan percobaan. Selain eksperimen, penelitian ini juga termasuk penelitian terapan. Menurut Sukardi (2010) penelitian terapan dilakukan berkenaan dengan penerapan yang berfungsi untuk mencari solusi tentang masalah yang spesifik.

3.2. Jenis dan Sumber Data

Data yang digunakan dalam penelitian ini adalah uang kertas rupiah yang diterbitkan oleh Bank Indonesia dengan tahun emisi (TE) 2016. Pecahan nominal yang digunakan berupa Rp1.000, Rp10.000, Rp100.000, Rp2.000, Rp20.000, Rp5.000, Rp50.000.

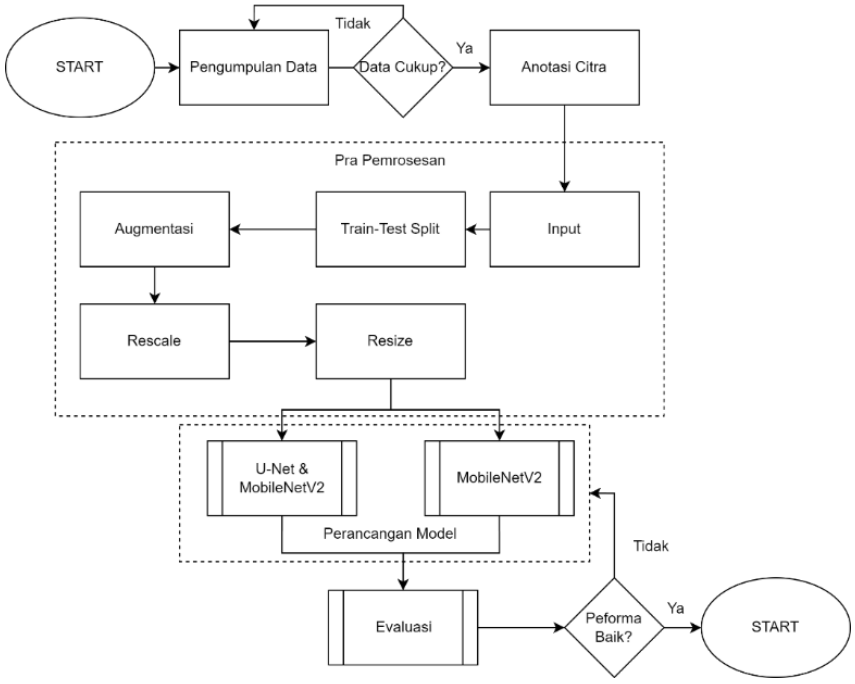
Tabel 3.1 Tabel Deskripsi Data

No	Nominal	Terbilang	Jumlah			Total
			Jenis I	Jenis II	Jenis III	
1.	Rp 1.000	Seribu Rupiah	115 Citra	67 Citra	20 citra	202 Citra
2.	Rp 2.000	Dua ribu Rupiah	131 Citra	34 Citra	20 citra	185 Citra
3.	Rp 5.000	Lima ribu Rupiah	98 Citra	29 citra	20 citra	147 Citra
4.	Rp 10.000	Sepuluh ribu Rupiah	137 Citra	44 citra	20 citra	201 Citra
5.	Rp 20.000	Duapuluh ribu Rupiah	140 Citra	60 citra	20 citra	220 Citra
6.	Rp 50.000	Limapuluh ribu Rupiah	161 Citra	16 citra	20 citra	197 Citra
7.	Rp 100.000	Seratus ribu Rupiah	161 Citra	45 citra	20 citra	226 Citra
TOTAL			943	295	140	1378 Citra

Gambar dengan ‘Jenis 1’ memiliki ukuran pixel sebesar 224x224 yang didapatkan dengan menggunakan webcam dengan bantuan tools kamera dari *teachablemachine* agar gambar dapat diambil secara kolektif. Gambar dengan ‘Jenis 2’ memiliki ukuran 2304x4096 pixel yang diambil menggunakan kamera Xiaomi Redmi 9. Dan gambar ‘Jenis 3’ memiliki ukuran 1600x900 pixel yang didapatkan melalui Kaggle secara *open source*.

3.3. Tahapan Penelitian

Dalam penelitian diperlukan kerangka agar penelitian dilaksanakan secara sistematis, dan diharapkan menjawab keseluruhan permasalahan yang diberikan serta mencapai tujuan dari penelitian yang dilakukan. Tahapan pada penelitian ini dapat dilihat pada *Gambar 3.1*



Gambar 3.1 Rancangan Tahapan penelitian

3.3.1. Pengumpulan Citra

Pengumpulan gambar dilakukan dengan menggunakan kamera *smartphone* Xiaomi Redmi 9 dan webcam MSI GF63 Thin dengan jarak ±20 centimeter.

Gambar diambil dengan posisi depan, belakang, terlipat, terpotong, dan kombinasi diantara posisi-posisi tersebut. Pengambilan citra dilakukan dengan beberapa skenario, yaitu uang dalam keadaan terlipat tampak depan, terlipat tampak belakang, terlipat dipegang oleh tangan, terpotong kamera.

3.3.2. Anotasi Citra

Langkah selanjutnya adalah pemberian label pada masing masing citra yang telah dikumpulkan. Memberikan *masking* atau menandai daerah yang spesifik pada citra, dengan begitu *ground truth* dapat diperoleh.

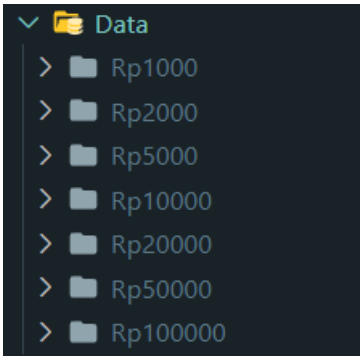
Gambar 3.2 Citra Awal (kiri), masking label(tengah), Citra berlabel(kanan)

3.3.3. Prapemrosesan Citra

Pra-pemrosesan (*preprocessing)* merupakan tahapan yang penting di berbagai bidang yang menerapkan ilmu *machine learning*, dalam penelitian ini diberlakukan teknik prapemrosesan citra, dikarenakan objek penelitian merupakan citra/gmbar. Tahap pra-pemrosesan dilakukan dengan penyiapan dokumen mentah menjadi dokumen atau representatif dokumen yang siap diproses untuk langkah selanjutnya. Dalam penelitian ini tahap preprocessing dilakukan dengan cara sebagai berikut

a. Input

Pada tahapan ini digunakan direktori induk yang berisikan folder tiap kelas.



Gambar 3.3 Struktur folder data

Gambar 3.3 merupakan bagaimana struktur data disimpan. Maka dari itu dengan menggunakan direktori induk yaitu folder 'Data', direktori serta target kelas bisa didapatkan serta dapat dijadikan kolom untuk dataframe yang akan digunakan.

b. Splitting

Setelah direktori induk didapatkan selanjutnya data dibagi menjadi data training dan validasi. Splitting data dibagi dengan skema rasio 70:30, 75:25, dan 80:30. Proses *splitting* data ini dapat dilakukan dengan menggunakan library pada python yaitu *split-folders.*

c. Augmentasi

Augmentasi merupakan sebuah teknik manipulasi data tanpa menghilangkan atau menghapus inti dari data itu sendiri. Misalnya jika data berupa sebuah gambar, maka gambar tersebut dapat diputar, diperbesar, diperkecil dan sebagainya. Pada penelitian ini teknik yang digunakan adalah rotasi, digeser, dan diperbesar dan diperkecil.

d. Rescale

Dalam *computer vision, rescaling* adalah salah satu langkah yang dapat dibilang krusial. Karena koefisien warna pada citra mulanya berisi nilai dengan range 0-255, dengan nilai range sebesar itu model akan bekerja lebih keras dan banyak risiko *loss*. Salah satu caranya adalah dengan menggunakan perhitungan

seperti *Persamaan 3.1*, dengan begitu koefisien warna berada pada *range* 0-1. Hal ini akan mempermudah model untuk proses trainingnya

$$x_{citra} = x_{citra} \times \frac{1}{255} \qquad \text{Persamaan 3.1}$$

e. Resize

Selain *rescaling*, mengubah size juga diperlukan dalam penelitian ini, karena semakin kecil dimensi input, maka semakin kecil pula komputasi yang dibutuhkan. Selain itu *resizing*, dilakukan dengan pertimbangan input size dari pre-trained MobileNetV2.

3.3.4. Pengembangan Model Segmentasi dan Klasifikasi

Setelah data berhasil dikumpulkan dan dilakukan pra-pemrosesan, tahapan selanjutnya berupa eksperimen pengembangan model identifikasi nominal uang kertas. Adapun skenario eksperimen yang diterapkan berupa perbandingan antara model yang diproses dengan U-Net dan MobileNetV2 dengan model yang diproses dengan MobileNetV2 tanpa U-Net.

a. MobileNetV2

Untuk dapat menemukan model klasifikasi nominal uang terbaik, diperlukan beberapa skema pada *epoch* dan *batch size*. Model akan dilakukan *training* pada 100 epoch dengan *batch size* 8,16, dan 32. Selain *epoch* dan learning rate serta dropout juga diperhatikan dengan kondisi learning rate sebesar 10^{-3} dan nilai dropout sebesar 10^{-3}.

b. U-Net dan MobileNetV2

Untuk dapat menggunakan U-Net kita memberikan label pada dataset atau anotasi yang sebelumnya sudah dilakukan. Kemudian dibuatlah dataset yang berisikan data yang telah tersegmentasi yang didapatkan dari prediksi U-Net setelah training model. Skenario training dilakukan pada 100 epoch dengan 8,16, dan 32 batch.

3.3.5. Evaluasi Model

Pada tahapan ini evaluasi dilakukan dengan dua cara, karena evaluasi untuk segmentasi dan klasifikasi membutuhkan teknik khusus untuk dapat mengevaluasi model yang dirancang.

a. Evaluasi Segmentasi U-Net

Pada tugas pengembangan model segmentasi U-Net evaluasi dilakukan dengan menggunakan metric Mean IoU, dengan membandingkan *ground truth* dan area prediksi peforma model dapat diobservasi sehingga dapat dilakukan penyesuaian baik dari segi hyperparameter seperti epoch, batch size dan terhadap jumlah network pada rancangan arsitektur. Mean IoU didapatkan dari rata rata IoU untuk setiap citra. IoU dapat dihitung dengan *Persamaan 2.11*

b. Evaluasi Klasifikasi MobileNetV2

Sedangkan pada pengembangan model klasifikasi MobileNetV2, evaluasi dilakukan dengan menggunakan multi-class confusion matrix. Pada multi-class confusion matrix terdapat macro average akurasi, presisi, recal dan f1-score. Untuk mendapatkan ke empat metric tersebut dapat dihitung dengan mencari rata rata akurasi, presisi, recal dan f1-score. Persamaan Macro Average F1-score dapat dilihat pada *Persamaan 2.10*.

3.3.6. Deployment

Deployment merupakan tahapan dimana model yang telah dirancang dipublikasiakan agar dapat digunakan oleh publik secara leluasa. Tahapan deployment dilakukan dengan membuat aplikasi bernama “Lihat.in” dan melakukan integrasi antara server side dan *mobile apps*. Pembuatan aplikasi dilakukan dengan menggunakan *flutter*.

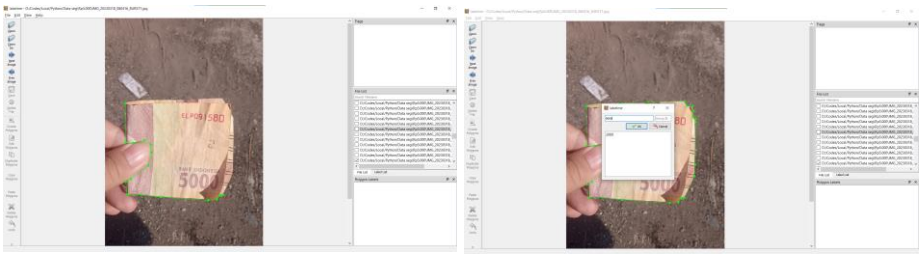
BAB 4. HASIL DAN PEMBAHASAN

4.1. Pengumpulan Citra

Pengambilan gambar dilakukan dengan menggunakan kamera dari perangkat smartphone Xiaomi Redmi 9 dan webcam MSI GF63 Thin serta beberapa gambar yang didapatkan dari kaggle secara open source. Pengambilan dengan webcam dibantu dengan tools *teachablemachine* agar citra dapat diambil secara kolektif. Citra diambil kedua sisi dengan cara objek diluar ruangan, objek didalam ruangan, terlipat, dipegang, terpotong, terlentang.

4.2. Anotasi Citra

Anotasi citra dilakukan dengan menggunakan package *labelme* dari python. Anotasi dilakukan untuk mendapatkan *ground truth* pada citra agar pada saat segmentasi dapat dilakukan evaluasi. Menggunakan *labelme* anotasi citra dapat dilakukan cukup mudah, dengan memberikan batasan pada objek.



Gambar 4.1 pemberian bentuk masking (kiri) dan pelabelan masking (kanan)

Pemberian label pada citra dilakukan dengan memberikan garis batas pada daerah objek yang diinginkan atau biasa disebut *masking*. Dengan *labelme* pemberian garis batas ini dapat dilakukan cukup mudah, mulanya direktori induk data perlu dimuat kemudian untuk tiap citra dapat diberi garis batas dengan menggunakan ‘create polygon’ kemudian diberikan label seperti pada Gambar 4.1 (kanan).

4.3. Prapemrosesan Citra

4.3.1. Input dan train-test split

Dalam penelitian ini dibuat dataframe yang berisikan direktori dari citra serta masing masing kelasnya sebagai input. Train test split dilakukan bersamaan pada input guna mendapatkan dataframe yang telah dibagi sesuai skenario splitting

```
#Split Scenario

scene = [
    [0.7,0.3],
    [0.75,0.25],
    [0.8,0.2]
]

#For god sake, doit once
for i,scenerios in enumerate(scene):
    splitfolders.ratio(base_dir, output=f"splitted_data/skenario-{i+1}", ratio=(scenerios[0],scenerios[1]), group_prefix=None,seed=42)

#Skenario 1
train_dir1 = "splitted_data/skenario-1/train"
val_dir1 = "splitted_data/skenario-1/val"

#Skenario 2
train_dir2 = "splitted_data/skenario-2/train"
val_dir2 = "splitted_data/skenario-2/val"

#Skenario 3
train_dir3 = "splitted_data/skenario-3/train"
val_dir3 = "splitted_data/skenario-3/val"
```

Gambar 4.2 potongan source code input dan train test-split

Mulanya dengan memanggil ‘base_dir’ sebagai direktori induk yang berisikan folder-folder kelas seperti pada Gambar 3.3. Kemudian mempersiapkan skenario train-test split agar dapat dilakukan looping untuk mendapatkan pembagian data dan direktori setelah dilakukan splitting. Setelah dapat direktori dari masing masing skenario, dipersiapkan variabel untuk menampung direktori tersebut, sebagai contoh ‘train_dir1’ untuk direktori data training dengan skenario pertama (0.7;0.2) dan ‘val_dir1’ untuk direktori data validasi. Setelah mendapatkan tiap pembagian skenario, dibuatlah dataframe tiap skenario tersebut.

```
def built_dataframe(dir = base_dir,):

    '''
        A function to create a Dataframe containing Paths and Classes
    '''


    df = pd.DataFrame({
        'Path':[],
        'Class':[]
    })

    for label in os.listdir(dir):
        for data in os.listdir(dir+f"/{label}"):
            df = df.append({
                'Path':f"{dir}/{label}/{data}",
                'Class':label
            },ignore_index = True)

    return df
```

Gambar 4.3 potongan source code untuk mendapatkan dataframe perskenario

Fungsi ‘built_dataframe’ digunakan untuk membuat sebuah dataframe yang memuat direktori tiap citra dan kelasnya masing masing. Pada penelitian ini fungsi tersebut digunakan pada tiap skenario. Meski dengan ‘base_dir’ sebagai default parameter, pada penerapannya menggunakan data yang telah dibagi per-skenario seperti pada Gambar 4.2 untuk tiap skenario split data.

4.3.2. Augmentasi & Rescaling

Augmentasi dilakukan dengan menggunakan *Imagedatagenerator* dari library Keras dari python. Selain augmentasi *imagedatagenerator* juga dapat melakukan rescaling, maka dari itu rescaling dan augmentasi dilakukan bersamaan pada penelitian ini.

```
datagen = ImageDataGenerator(
    rescale=1./255,
    fill_mode='nearest',
    rotation_range= 15,
    width_shift_range= 10.0,
    height_shift_range= 10.0,
    )

val_gen = ImageDataGenerator(
    rescale=1./255,
    )
```

Gambar 4.4 Potongan source code Imagedatagenerator untuk augmentasi dan rescaling

Pada penelitian ini dipersiapkan dua imagedatagenerator, variabel ‘datagen’ digunakan untuk data training, dimana parameter augmentasi seperti ‘rotation_range’, ‘width_shift_range’ dan ‘height_shift_range’ didapatkan dari penelitian sebelumnya. Sedangkan variabel ‘val_gen’ digunakan untuk memvalidasi data training saat training dilakukan. Sedangkan parameter rescale digunakan untuk men-*scaling* ulang sesuai dengan nilai dari parameter tersebut, dalam penelitian ini menggunakan nilai ‘1./255’.

4.3.3. Resizing

Resizing diterapkan saat pengaplikasian augmentasi dan rescaling, dengan menggunakan *imagedatagenerator.flow* dari package keras, resizing dapat dilakukan. Dengan parameter 'target_size' sebagai parameter resizing pada *Gambar 4.5.*

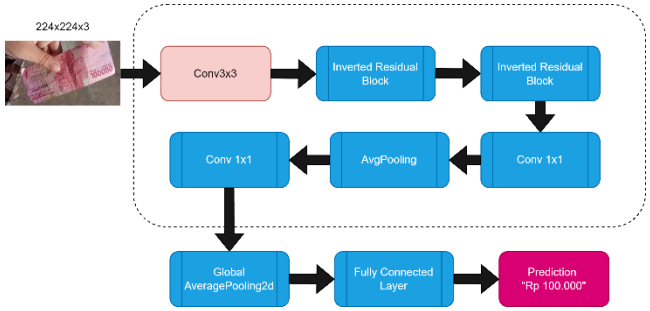
```python
def generators(data_train,data_val,train_datagen,val_datagen,bs,resize_shape=(224,224)):

    '''
        A function to crete the generator
    '''

    train_generator = train_datagen.flow_from_dataframe(
                dataframe=data_train,
                x_col='Path',
                y_col='Class',
                target_size=resize_shape,
                batch_size=bs,
                class_mode='categorical'
            )


    val_generator = val_datagen.flow_from_dataframe(
                dataframe=data_val,
                x_col='Path',
                y_col='Class',
                target_size=resize_shape,
                batch_size=bs,
                shuffle=False,
                class_mode='categorical'
            )

    return train_generator, val_generator
```

Gambar 4.5 Source code resizing

Dengan menggunakan *ImageDataGenerator* kita dapat mengubah citra yang masih berbentuk jpg/png ke bentuk tensor, dilakukan dengan 'datagen' sebagaimana data tersebut diterapkan *scaling*, augmentasi. Kemudian dengan menggunakan method 'flow_from_dataframe' sebagai method untuk menghasilkan *batches* dan kemana 'datagen' diterapkan, dalam penelitian ini menggunakan 'data_train' sebagai data latih dan 'data_val' sebagai data validasi yang didapat dari *splitting* per-skenario.

4.4. Implementasi MobileNetV2

MobileNetV2 digunakan sebagai arsitektur CNN untuk tugas klasifikasi pada penelitian ini. MobileNetV2 sendiri menggunakan *Inverted Residual Block* yang merupakan salah satu tipe *Residual Block*. Flowchart MobilNetV2 yang digunakan pada penelitian ini dapat dilihat pada *Gambar 4.6.*



Gambar 4.6 Flowchart MobileNetV2

Layaknya arsitektur lainya, arsitektur MobileNetV2 menerima input terlebih dahulu, dan jika kita lihat pada flowchart *Gambar 4.8*. Dengan ciri khas arsitektur CNN, MobileNetV2 memproses input dengan *convolution layer*.

```
IMG_SHAPE = (224, 224, 3)

mobileNetV2_model = tf.keras.applications.MobileNetV2(
    input_shape=IMG_SHAPE, include_top=False,
    alpha=1.0, weights='imagenet',)

mobileNetV2_model.trainable = False
```

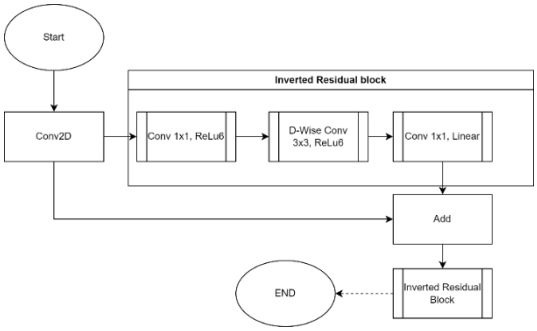
Gambar 4.7 Source code MobileNet

MobileNetV2 dapat dipanggil dengan menggunakan tensorflow, dengan menggunakan pretrained mobilenetV2 yang menggunakan *weights* 'imagenet' seperti pada gambar *Gambar 4.7*.

4.4.1. Flowchart Arsitektur

a. Flowchart Inverted Residual Block

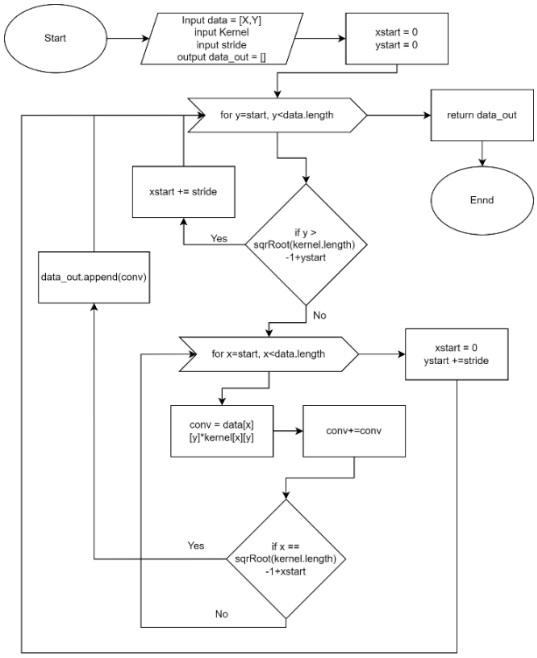
Inverted Residual Block with Linear Bottleneck merupakan ciri khusus arsitektur MobileNetV2. Blok ini merupakan kebalikan dari residual blok pada umumnya. Pada blok ini konvolusi dimulai dengan menaikkan dimensi atau *expand* tidak seperti pada residual blok yang menurunkan dimensinya setelah konvolusi pertama. Flowchart *Inverted residual block* dapat dilihat pada Gambar 4.8



Gambar 4.8 Flow chart Inverted Residual Block

b. Flowchart Convolution

Konvolusi adalah ciri khusus dari CNN itu sendiri, dimana secara garis besar konvolusi adalah operasi seperti dot matrix antara input *M* dan kernel *K* sehingga menciptakan matrix baru atau fitur. Apablika konvolusi dilakukan tanpa padding, maka matrix baru cenderung memiliki dimensi yang lebih kecil. Pada Gambar 4.9 merupakan flowchart konvolusi.



Gambar 4.9 Flowchart Konvolusi

4.4.2. Peforma model MobileNetV2

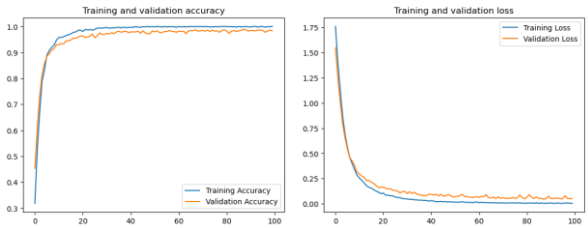
Model diuji dengan beberapa skema yang melibatkan rasio data training, nilai learning rate dan batch size. Training model dilakukan pada spesifikasi perangkat sebagai berikut :

Processor : Intel i3-12100F
GPU : NVIDIA GeForce RTX 3050
Memory : 16 GB RAM

```python
def fitting_model(EPOCHS,train,val,base_model = mobileNetV2_model,num_classes =7):
    model = tf.keras.models.Sequential()

    model.add(base_model)

    model.add(tf.keras.layers.GlobalAveragePooling2D()),
    model.add(tf.keras.layers.Dense(128, activation='relu')),
    model.add(tf.keras.layers.Dropout(0.001)),
    model.add(tf.keras.layers.Dense(32, activation='relu')),
    model.add(tf.keras.layers.Dropout(0.001)),
    model.add(tf.keras.layers.Dense(num_classes, activation='softmax'))

    model.compile(loss='categorical_crossentropy',
                  optimizer=Adam(learning_rate=0.0001),
                  metrics=['categorical_accuracy'])

    history = model.fit(
      train,
      epochs=EPOCHS,
      validation_data=val,
      verbose=0)


    plot_accLost(history)


    return model

```

Gambar 4.10 Fitting Model

Perancangan model dilakukan dengan membuat method ‘fitting_model’ yang berisikan arsitektur model secara keseluruhan dengan variabel ‘base_model’ adalah arsitektur mobileNetV2 seperti pada *Gambar 4.7*. Training model dilakukan dengan nilai dropout sebesar 10^{-3} dan learning rate sebesar 10^{-4}, nilai tersebut didapatkan dari penelitian sebelumnya dengan pencocokan nilai *learning rate* yang semula bernilai 10^{-3}.



Gambar 4.11 Grafik Training & Validation skenario 70:30 batch size 16 learning rate 10^{-4}

Skenario 70:30 dengan nilai dropout sebesar 10^{-3} dan learning rate sebesar 10^{-4} didapatkan hasil yang terbaik pada *batch size* 16 dan 32. Meski memiliki hasil peforma yang tidak jauh beda, namun grafik dengan indikasi *overfit* dan *underfit* paling minimum didapatkan pada *batch size* 16.

Tabel 4.1 Tabel Peforma Macro Average untuk skenario 70:30 dengan learning rate 10^{-4}

Batch Size	Macro Precision	Accuracy	Macro Recall	Macro F-1 Score
8	0.97	0.97	0.97	0.97
16	0.98	0.98	0.98	0.98
32	0.98	0.98	0.98	0.98

Tabel 4.1 menunjukkan peforma dengan skenario 70:30, hasil peforma yang didapatkan merupakan *Weighted Macro Average* yang dapat dihitung melalui *Persamaan 2.8, Persamaan 2.9,* dan *Persamaan 2.10*.

Training dilanjutkan dengan nilai dropout sebesar 10^{-3} dan learning rate sebesar 10^{-4} untuk untuk skenario *train-test split data* 75:25 dan skenario 80:20 dan didapatkan peforma hasil training pada Tabel 4.2.

Tabel 4.2 Peforma klasifikasi MobileNetV2

Skenario Data	Batch Size	Accuracy	Macro Precision	Macro Recall	Macro F-1 Score
70:30	8	0.97	0.97	0.97	0.97
70:30	16	0.98	0.98	0.98	0.98
70:30	32	0.98	0.98	0.98	0.98
75:25	8	0.98	0.98	0.98	0.98
75:25	16	0.99	0.99	0.99	0.99
75:25	32	0.98	0.98	0.98	0.98
80:20	8	0.99	0.99	0.99	0.99
80:20	16	0.98	0.98	0.98	0.98
80:20	32	0.98	0.98	0.98	0.98

Pada Tabel 4.2 skenario 75:25 dan 80:20 dengan nilai dropout sebesar 10^{-3} dan learning rate sebesar 10^{-4} didapatkan hasil yang terbaik pada *batch size* 16 untuk skenario 75:25 dan *batch size* 8 untuk skenario 80:20. Meskipun pada *batch size* 8 untuk kedua skenario tersebut memiliki peforma yang sangat tinggi, namun grafik

training untuk kedua skenario tersebut memiliki grafik yang terbaik pada *batch size* 32 untuk *epoch* 100.

4.5. Implementasi U-Net dan MobileNetV2

Pada dasarnya U-Net merupakan arsitektur CNN yang dirancang khusus untuk tugas semantic segmentasi. Tidak seperti MobileNetV2 yang memiliki blok residual, U-Net hanya berbekal lapisan konvolusi, lapisan pooling, skip connection dan konvolusi transpose. Implementasi U-Net dilakukan dengan pembuatan dataset yang telah disegmentasi terlebih dahulu, kemudian diterapkan tahapan sebagaimana tahapan klasifikasi menggunakan mobileNetV2.

4.5.1. Tahapan Segmentasi U-Net

a. Persiapan data

Persiapan data pada tahapan ini meliputi *input* data dan pra pemrosesan yang meliputi *resizing*, *rescaling*, dan rekayasa data

```
def image_generator(files, batch_size = 32, sz = (256,256)):
  while True:
    batch = np.random.choice(files, size = batch_size) #Pilih batch random untuk training

    #Pengumpulan batch input dan output
    batch_x = []
    batch_y = []

    for f in batch:
      mask = Image.open(f'dataset/SegmentationClassPNG/{f[:-4]}.png')
      mask = np.array(mask.resize(sz))
      mask[mask != 0] = 1
      batch_y.append(mask)

      #Preprocess gambar mentah
      raw = Image.open(f'dataset/JPEGImages/{f}')
      raw = raw.resize(sz)
      raw = np.array(raw)

      #Cek chanel gambar RGBA atau GRAY
      if len(raw.shape) == 2:
        raw = np.stack((raw,)*3, axis=-1)
      else:
        raw = raw[:,:,0:3]

      batch_x.append(raw)

    batch_x = np.array(batch_x)/255.
    batch_y = np.array(batch_y)
    batch_y = np.expand_dims(batch_y,3)

    yield (batch_x,batch_y)
```

Gambar 4.12 Potongan source code data generator

Data generator digunakan untuk mempersiapkan atau mengubah data input yang mulanya gambar ke tensor, hal ini dilakukan agar dapat dilakukan baik pemrosesan citra maupun pra-pemrosesan. Dengan ‘batch_y’ sebagai penampung tensor untuk data label atau *masking* dan ‘batch_x’ sebagai penampung dari data yang mentah atau tidak dilakukan *masking*.

b. Perancangan model Segmentasi

Perancangan model U-net dilakukan dengan 100 epoch dan dirancang pada 8, 16, dan 32 *batch size* dan nilai dropout sebesar 0.2. Training dilakukan dengan spesifikasi perangkat yang sama pada proses training MobileNetV2.

```
def unet(sz = (256, 256, 3)):
  x = Input(sz)

  inputs = x

  #Downsampling
  f = 8
  layers = []

  for i in range(0,6):
    x = Conv2D(f, 3, activation='relu', padding='same') (x)
    x = Conv2D(f, 3, activation='relu', padding='same') (x)

    layers.append(x)
    x=MaxPooling2D() (x)
    f = f*2
  ff2 = 256

  #bottleneck
  j = len(layers) - 1
  x = Conv2D(f, 3, activation='relu', padding='same') (x)
  x = Conv2D(f, 3, activation='relu', padding='same') (x)
  x = Conv2DTranspose(ff2, 2, strides=(2, 2), padding='same') (x)
  x = Concatenate(axis=3)([x, layers[j]])
  j = j -1

  #Upsampling
  for i in range(0,5):
    ff2 = ff2//2
    f = f // 2
    x = Conv2D(f, 3, activation='relu', padding='same') (x)
    x = Conv2D(f, 3, activation='relu', padding='same') (x)
    x = Conv2DTranspose(ff2, 2, strides=(2, 2), padding='same') (x)
    x = Concatenate(axis=3)([x, layers[j]])
    x = Dropout(0.2)(x)
    j = j -1

  #Classification
  x = Conv2D(f, 3, activation='relu', padding='same') (x)
  x = Conv2D(f, 3, activation='relu', padding='same') (x)
  outputs = Conv2D(1, 1, activation='sigmoid') (x)

  #model creation
  model = Model(inputs=[inputs], outputs=[outputs])
  model.compile(optimizer = 'rmsprop', loss = 'binary_crossentropy', metrics = [me
an_iou])

  return model
```

Gambar 4.13. Source code arsitektur U-Net

Potongan kode diatas merupakan arsitektur U-Net, dengan kedalaman 4 lapisan untuk *contracting path* dan *expanding path*. Karena pada dasarsnya U-Net hanya menggunakan *conv layer* maka dari itu banyak baris kode yang menggunakan *Conv2D*. Pada *downsampling* atau *contracting* U-net hanya perlu menggunakan *Conv2D* dan *maxpooling* dengan masing masing besar kernel *Conv2D* sebesar 3x3 dan filter yang digunakan pada awalnya sebesar 8. Nilai ini didapatkan mengingat U-net yang dirancang akan digunakan pada perangkat mobile, maka dari itu nilai yang digunakan relatif kecil. Pada *Upsampling* atau *expanding path Conv2DTranspose* digunakan, hal ini dikarenakan U-net perlu meningkatkan dimensi dari citra setelah dilakukan *downsampling*.

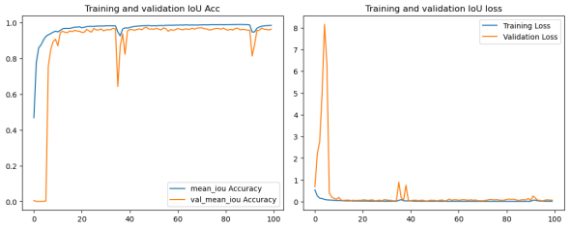
Untuk tugas segmentasi, evaluasi dilakukan dengan *mean intersection over union* (Mean IoU) maka dari itu diperlukan function yang dapat melakukan perhitungan metric tersebut.

```python
def mean_iou(y_true, y_pred):
    yt0 = y_true[:,:,:,0]
    yp0 = K.cast(y_pred[:,:,:,0] > 0.5, 'float32')
    inter = tf.math.count_nonzero(tf.logical_and(tf.equal(yt0, 1), tf.equal(yp0, 1)))
    union = tf.math.count_nonzero(tf.add(yt0, yp0))
    iou = tf.where(tf.equal(union, 0), 1., tf.cast(inter/union, 'float32'))
    return iou
```

Gambar 4.14 Potongan source code mean iou

Pada baris ke-2 slicing untuk mendapatkan *ground truth*, sedang pada baris 3 slicing serta casting untuk hasil prediksi ke tipe float 32 bit, karena format untuk *ground truth* juga menggunakan float 32 bit. Pada baris ke 4 digunakan untuk mencari bagian *intersection* sedang pada baris ke 5 digunakan utnuk mencari bagian *union*. Setelah mendapatkan keduanya, maka IoU dapat dicari mengikuti Persamaan 2.11.

Dengan membandingkan hasil prediksi ‘y_pred’ dan *ground truth* ‘y_true’ maka IoU didapatkan. Kemudian model dilatih dengan split data 80%.



Gambar 4.15 Grafik Mean IoU akurasi dan loss

Pada *Gambar 4.15* merupakan hasil training model segmentasi yang dilakukan, dengan sumbu axis sebagai epoch ke-n dan sumbu ordinat sebagai nilai akurasi/loss.

Tabel 4.3 Sampel grafik training dan validasi mean IoU epoch 96-100

EPOCH	Mean IoU Acc	Mean IoU Loss	Mean IoU Val Acc	Mean IoU Val Loss
96	0.980767	0.019831	0.969734	0.034374
97	0.982629	0.017397	0.964077	0.065820
98	0.983995	0.016602	0.962824	0.084135
99	0.984838	0.014852	0.958917	0.069824
100	0.985419	0.014613	0.964708	0.067273

c. Pembuatan dataset segmentasi untuk training MobileNetV2

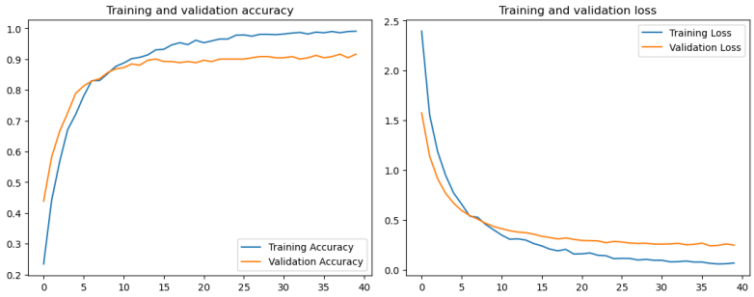
```
1  def predict(path,name,loaded_model = loaded_model,output_dir="/Codes/Local/Python/Lihat.in/Segmented Data/"):
2      raw = Image.open(path)
3      raw = np.array(raw.resize((256,256)))/255.
4      raw = raw[:,:,0:3]
5
6
7      #Predict Mask
8      pred = loaded_model.predict(np.expand_dims(raw, 0))
9
10     msk = pred.squeeze()
11     msk = np.stack((msk,)*3, axis=-1)
12     msk[msk >= 0.5] = 1
13     msk[msk < 0.5] = 0
14
15     # combined = np.concatenate([raw, msk, raw*msk], axis = 1)
16     result = (raw*msk)
17     image = Image.fromarray(((result)*255).astype(np.uint8))
18     image.save(output_dir+name)
```

Gambar 4.16 potongan code membuat dataset citra tersegmentasi

Dengan potongan kode diatas, dataset yang sebelumnya tidak tersegmentasi dapat diubah atau membuat sebuah dataset baru yang telah tersegmentasi. Dengan menggunakan model segmentasi yang telah dirancang sebelumnya. Pada baris 2 hingga 4 merupakan baris kode untuk pra pemrosesan citra sebelum di lakukan segmentasi, sedang pada baris ke-8 merupakan prediksi *mask* citra dan pada baris ke-10 hingga 13 merupakan pengubahan hasil prediksi yang sebelumnya tidak dapat divisualisasikan menjadi dapat divisualisasikan dengan baris 16 dan baris 17 merupakan hasil citra segmentasi serta normalisasi dan pada baris 18 merupakan langkah untuk menyimpan citra hasil segmentasi.

4.5.2. Peforma model U-Net dan MobileNetV2

Training MobileNetV2 dengan dataset yang telah dilakukan segmentasi serta menggunakan arsitektur sama seperti pada sebelumnya termasuk skenario, namun terdapat perbedaan pada learning rate dan drop out pada saat dilakukan training MobileNetV2 dengan dataset yang telah disegmentasi. Hal ini dikarenakan dengan menggunakan learning rate sebelumnya pada grafik terindikasi underfitting sehingga learning rate yang digunakan sebesar 10^{-5} dan dropout sebesar 0.2.



Gambar 4.17 Grafik Training & Validation skenario 80:20 batch size 16 learning rate 10^{-5} dengan U-Net

Hasil peforma terbaik dengan MobileNetV2 dengan melakukan segmentasi U-Net didapatkan pada skenario 80:20 dengan *batch size* 16 dropout sebesar 0.2 dan learning rate 10^{-5}. Hal ini baik dari segi peforma seperti yang ditunjukkan pada Tabel 4.4 maupun hasil training pada *epoch* 100. Dilihat pada grafik *accuracy* dan *loss* pada Gambar 4.17 menunjukkan peforma yang baik pada saat epoch diatas 20, hal ini ditunjukkan dengan tidak adanya grafik *validation* naik atau turun secara signifikan, melainkan mendekati datar dan cenderung mendekat terhadap *training*.

Tabel 4.4 Peforma Klasifikasi Segmentasi U-Net dan MobileNetV2

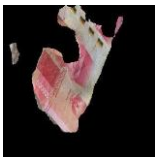
Skenario Data	Batch Size	Accuracy	Macro Precision	Macro Recall	Macro F-1 Score
70:30	8	0.89	0.90	0.89	0.89
70:30	16	0.91	0.91	0.90	0.91
70:30	32	0.88	0.89	0.88	0.88
75:25	8	0.90	0.91	0.90	0.90
75:25	16	0.89	0.89	0.88	0.89
75:25	32	0.88	0.88	0.88	0.88
80:20	8	0.91	0.92	0.91	0.91
80:20	16	0.92	0.92	0.91	0.92
80:20	32	0.89	0.89	0.89	0.89

Mulanya model dilakukan training dengan menggunakan skenario yang digunakan pada training MobileNetV2 tanpa segmentasi, dengan beberapa pencocokan untuk *epoch* 100 didapatkan peforma terbaik pada skenario data 80:20 dengan *batch size* 16 dropout sebesar 0.2 dan learning rate 10^{-5}.

Dilihat pada *Gambar 4.15* grafik mean IoU memiliki akurasi serta loss yang tampak kacau, dimana nilai akurasi optimal atau nilai terbaik yang bisa didapatkan

dari mean IoU adalah 1 berdasarkan Persamaan 2.11. Setelah dilakukan observasi didapatkan model gagal segmentasi secara baik pada data jenis dua dan jenis tiga berdasarkan Tabel 3.1 degan error segmentasi terbanyak pada jenis dua pada nominal ‘Rp 100.000’ dan jenis tiga secara merata pada tiap nominal. Hal ini terjadi akibat beberapa faktor yang didapatkan setelah observasi secara manual :

1. Tedapat bagian uang yang memiliki pencahayaan rendah atau tinggi sehingga memiliki kemiripan pada latar belakang dan jari sebagai objek yang tidak diinginkan, maka bagian uang tersebut terdeteksi bukan sebagai target objek sehingga disisihkan oleh model seperti yang ditunjukkan pada Gambar 4.18 a dan Gambar 4.18 b.
2. Terdapat bagian latar belakang atau objek jari yang memiliki kemiripan dengan objek target atau uang maka bagian tersebut terhitung sebagai uang sehingga tidak disisihkan oleh model seperti yang ditunjukkan pada Gambar 4.18 c dan Gambar 4.18 d.
3. Kondisi citra yang memiliki kecacatan seperti blur dan resolusi yang berbeda sehingga menyebabkan informasi yang bisa diambil berkurang (Thambawita et al., 2021).



(a)

(b)

(c)

(d)

Gambar 4.18 Perbandingan citra usai segmentasi dan sebelum segmentasi

Gambar 4.18 a dan Gambar 4.18 menunjukkan hasil segmentasi citra yang kurang tepat yang mana seharusnya citra yang dihasilkan hanya menunjukkan uang sebagai objek, namun hasil segmentasi masih menunjukkan latar belakang atau bahkan justru menghilangkan bagian dari objek itu sendiri.

Disisi lain peforma klasifikasi MobileNetV2 tanpa segmentasi U-Net memberikan peforma yang cukup tinggi yaitu dengan nilai 99% *accuracy*, 99% *precision*, 99% *recall*, 99% *f1-score*. Hasil tersebut menunjukkan peningkatan dari penelitian sebelumnya(Singh et al., 2022) dengan hasil *average accuracy* sebesar 96.75%. Peningkatan peforma tersebut didapatkan dari arsitektur berikut:

1. Menggunakan *GlobalAveragePooling2D* daripada *flatten*. Hal ini dikarenakan *GlobalAveragePooling2D* lebih dapat menurunkan resiko overfitting (Kumar et al., 2021).
2. *Learning rate* yang digunakan lebih kecil yaitu 10^{-4} daripada 10^{-3}
3. Adanya penambahan dense layer dengan neuron unit 128 dan 32.

Berdasarkan hasil observasi yang telah dijabarkan diatas, model yang digunakan adalah klasifikasi MobileNetV2 tanpa segmentasi U-Net.

4.6. Deployment

Sebelum dilakukan deployment, model diubah format menjadi tflite. Hal ini juga dapat mempercepat proses untuk mendapatkan hasil prediksi dibanding menggunakan *flask*, meskipun cara proses input berbeda yang dapat mengakibatkan berbedanya hasil yang didapatkan.

```
converter = tf.lite.TFLiteConverter.from_keras_model(model_for_export)
pruned_tflite_model = converter.convert()

_, pruned_tflite_file = tempfile.mkstemp('.tflite')

with open(pruned_tflite_file, 'wb') as f:
  f.write(pruned_tflite_model)

print('Saved pruned TFLite model to:', pruned_tflite_file)
```

Gambar.19 Mengubah model tflite

Pada baris 1 dan 2 model yang sebelumnya dirancang untuk dikonversi ke tflite. Sedangkan pada baris ke 4 merupakan pembuatan tempfile. Pada baris ke-6 dan baris 7, model tempfile sebelumnya diletakkan pada direktori yang ditetapkan pada saat pembuatan tempfile. Dengan menggunakan potongan kode diatas model yang mulanya berformat ‘.h5’ menjadi ‘.tflite’ sehingga dapat digunakan pada aplikasi yang dirancang menggunakan *flutter* tanpa menggunakan koneksi internet.

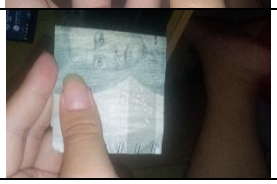
Deployment dilakukan dengan pembuatan aplikasi berbasis *mobile* (utamanya android) yang dirancang dengan menggunakan flutter. Versi flutter yang digunakan adalah Flutter 3.3.5 Stable dan versi android studio Bumblebee 2021. Android studio di-*install* utamanya agar mendapatkan AndroidSDK (Software Development Kit).

4.7. Uji Coba Aplikasi

Setelah dilakukan deployment, maka aplikasi dapat dilakukan uji coba untuk melihat sejauh mana model yang dirancang dapat diaplikasikan pada perangkat *mobile*.

Tabel 4.5 Tabel Uji coba aplikasi

No	Gambar	Aktual	Prediksi	Keterangan
1.		Rp 100.000	• Rp 100.000 (99,31%)	Tepat
2.		Rp 100.000	• Rp 100.000 (99,75%)	Tepat
3.		Rp 100.000	• Rp 5.000 (52,62%)	Tidak Tepat
4.		Rp 10.000	• Rp 10.000 (80,84%)	Tepat
5.		Rp 10.000	• Rp 10.000 (96.27%)	Tepat
6.		Rp 10.000	• Rp 10.000 (77.61%)	Tepat

7.		Rp 5.000	• Rp 5.000 (88,52%)	Tepat
8.		Rp 5.000	• Rp 5.000 (99,4%)	Tepat
9.		Rp 2.000	• Rp 2.000 (84,4%)	Tepat
10.		Rp 2.000	• Rp 2.000 (94%)	Tepat

Uji coba tersebut dilakukan dengan perangkat Xiaomi Redmi 9 dilakukan dengan menggunakan *blitz* atau *flash* kamera. Hasil uji coba tersebut menggunakan kemungkinan yang didapatkan disertai prediksi terdekat kedua dengan menggunakan tingkat keyakinan yang didapat dalam bentuk prosentase. Dari tabel uji coba dapat ditarik kesimpulan bahwa aplikasi dapat mendeteksi nominal uang, meskipun tidak dapat 100% mendeteksi secara tepat. Sebagai contoh pada nomor 3 pada tabel menunjukkan *tone* warna yang sedikit mendekati uang dengan pecahan 5000 seperti pada nomor 7 dan 8 pada tabel sehingga tidak tepat terprediksi.

BAB 5. Kesimpulan dan Saran

5.1. Kesimpulan

Identifikasi nominal uang rupiah kertas dapat dilakukan dengan MobileNetV2 secara baik dengan skenario data 75:25 dengan *batch size* 16, nilai dropout $2x10^{-1}$ dan learning rate 10^{-4}. Adapun kesimpulan yang dapat ditarik pada penelitian ini adalah :

1. Klasifikasi nominal uang kertas rupiah yang dilakukan dengan arsitektur MobileNetV2 mendapatkan hasil peforma diatas 95% secara keseluruhan dengan hasil paling tinggi didapatkan pada angka akurasi 99% dan f1-score 99%. Pada arsitektur yang melibatkan U-Net mendapatkan hasil peforma keseluruhan dibawah 95% dengan hasil terbaik yang dihasilkan berada pada akurasi 92% dan f1-score 92%.
2. Klasifikasi nominal uang kertas rupiah menggunakan arsitektur MobileNetV2 tanpa segmentasi U-Net menunjukkan hasil peforma yang lebih baik dengan perbandingan akurasi 99%:92%.

5.2. Saran

Berdasarkan hasil penelitian, pembahasan dan kesimpulan, penulis mengajukan saran :

1. Disarankan pada pengambilan citra menggunakan perlakuan atau teknik pengambilan yang sama untuk setiap kelas serta menggunakan jumlah citra yang tidak berbeda jauh.
2. Disarankan untuk menggunakan arsitektur lain seperti Mask R-CNN, DeepLab sebagai pembanding terhadap arsitektur U-Net yang utamanya dirancang untuk tujuan biomedis.

DAFTAR PUSTAKA

Ajit, A., Acharya, K., & Samanta, A. (2020). A Review of Convolutional Neural Networks. *2020 International Conference on Emerging Trends in Information Technology and Engineering (Ic-ETITE)*, 1–5. https://doi.org/10.1109/ic-ETITE47903.2020.049

Dhillon, A., & Verma, G. K. (2020). Convolutional neural network: a review of models, methodologies and applications to object detection. *Progress in Artificial Intelligence*, *9*(2), 85–112. https://doi.org/10.1007/s13748-019-00203-0

Grandini, M., Bagli, E., & Visani, G. (2020). *Metrics for Multi-Class Classification: an Overview*. https://arxiv.org/abs/2008.05756

Imad, M., Ullah, F., Hassan, M. A., & Naimullah. (2020). Pakistani Currency Recognition to Assist Blind Person Based on Convolutional Neural Network. *Journal of Computer Science and Technology Studies*, *2*(2), 12–19.

INFODATIN. (2018). *Situasi Gangguan Pengelihatan*. Kementerian Kesehatan Republik Indonesia.

Kumar, R. L., Kakarla, J., Isunuri, B. V., & Singh, M. (2021). Multi-class brain tumor classification using residual network and global average pooling. *Multimedia Tools and Applications*, *80*(9), 13429–13438. https://doi.org/10.1007/s11042-020-10335-4

Lauzon, F. Q. (2012). An introduction to deep learning. *2012 11th International Conference on Information Science, Signal Processing and Their Applications (ISSPA)*, 1438–1439. https://doi.org/10.1109/ISSPA.2012.6310529

Li, Z., Gao, E., Zhou, J., Han, W., Xu, X., & Gao, X. (2023). Applications of deep learning in understanding gene regulation. *Cell Reports Methods*, *3*(1), 100384. https://doi.org/10.1016/j.crmeth.2022.100384

Pamungkas, O. E., Rahmawati, P., Supriadi, D. M., Khalika, N. N., Maliyano, T., Pangestu, D. R., Nugraha, E. S., Afandi, M. A., Wulandari, N., Goran, P. K., & Wicaksono, A. (2022). Classification of Rupiah to Help Blind with The Convolutional Neural Network Method. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *6*(2), 259–268. https://doi.org/10.29207/resti.v6i2.3852

Park, C., Cho, S. W., Baek, N. R., Choi, J., & Park, K. R. (2020). Deep Feature-Based Three-Stage Detection of Banknotes and Coins for Assisting Visually Impaired People. *IEEE Access*, *8*, 184598–184613. https://doi.org/10.1109/ACCESS.2020.3029526

Pertuni. (2016, December 25). *Aksesibilitas untuk Tunanetra pada Uang Baru NKRI*. Https://Pertuni.or.Id/Aksesibilitas-Untuk-Tunanetra-Pada-Uang-Baru-Nkri/.

Quoc, T. T. P., Linh, T. T., & Minh, T. N. T. (2020). Comparing U-Net Convolutional Network with Mask R-CNN in Agricultural Area Segmentation on Satellite Images. *2020 7th NAFOSTED Conference on Information and Computer Science (NICS)*, 124–129. https://doi.org/10.1109/NICS51282.2020.9335856

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). *MobileNetV2: Inverted Residuals and Linear Bottlenecks*. http://arxiv.org/abs/1801.04381

Sharma, N., Jain, V., & Mishra, A. (2018). An Analysis Of Convolutional Neural Networks For Image Classification. *Procedia Computer Science*, *132*, 377–384. https://doi.org/10.1016/j.procs.2018.05.198

Shukla, A., Bhardwaj, S., & Singh, M. (2023). Segmentation for Lumbar Spinal Stenosis Using Convolutional Neural Networks. *Procedia Computer Science*, *218*, 2210–2223. https://doi.org/10.1016/j.procs.2023.01.197

Siddique, N., Paheding, S., Elkin, C. P., & Devabhaktuni, V. (2021). U-Net and Its Variants for Medical Image Segmentation: A Review of Theory and

Applications. *IEEE Access*, *9*, 82031–82057. https://doi.org/10.1109/ACCESS.2021.3086020

Singh, M., Chauhan, J., Kanroo, M. S., Verma, S., & Goyal, P. (2022). IPCRF: An End-to-End Indian Paper Currency Recognition Framework for Blind and Visually Impaired People. *IEEE Access*, *10*, 90726–90744. https://doi.org/10.1109/ACCESS.2022.3202007

Tamilselvan, K. S., & Murugesan, G. (2018). Image Segmentation. In *Medical and Biological Image Analysis*. InTech. https://doi.org/10.5772/intechopen.76428

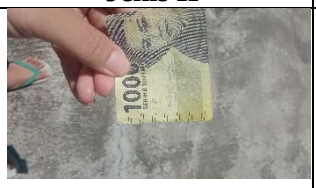
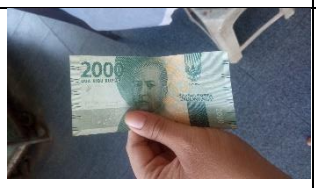
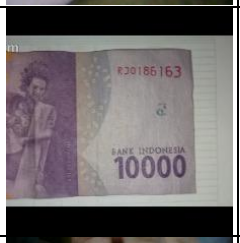
Thambawita, V., Strümke, I., Hicks, S. A., Halvorsen, P., Parasa, S., & Riegler, M. A. (2021). Impact of Image Resolution on Deep Learning Performance in Endoscopy Image Classification: An Experimental Study Using a Large Dataset of Endoscopic Images. *Diagnostics*, *11*(12), 2183. https://doi.org/10.3390/diagnostics11122183

T.K., B., Annavarapu, C. S. R., & Bablani, A. (2021). Machine learning algorithms for social media analysis: A survey. *Computer Science Review*, *40*, 100395. https://doi.org/10.1016/j.cosrev.2021.100395

Vaze, S., Xie, W., & Namburete, A. I. L. (2020). Low-Memory CNNs Enabling Real-Time Ultrasound Segmentation Towards Mobile Deployment. *IEEE Journal of Biomedical and Health Informatics*, *24*(4), 1059–1069. https://doi.org/10.1109/JBHI.2019.2961264

WHO. (2022, October 13). *Blindness and vision impairment*. Https://Www.Who.Int/News-Room/Fact-Sheets/Detail/Blindness-and-Visual-Impairment.

LAMPIRAN

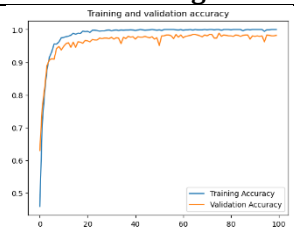
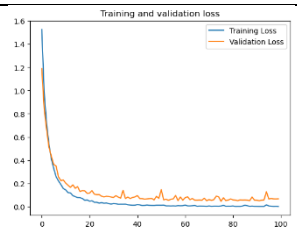
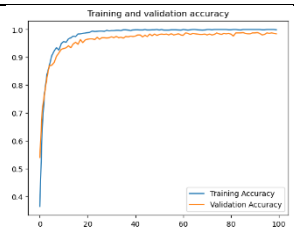
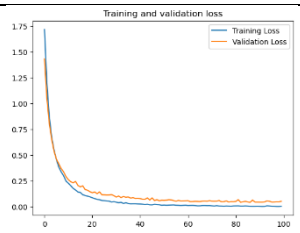
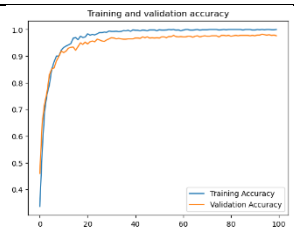
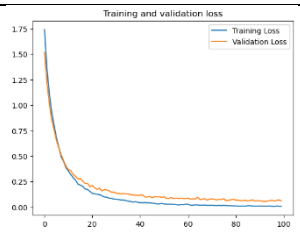
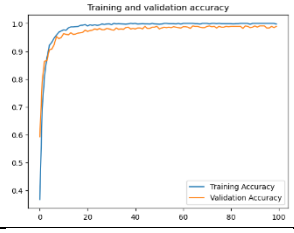
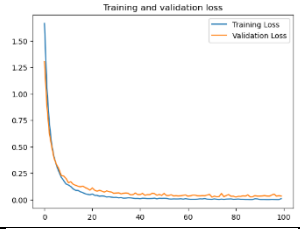
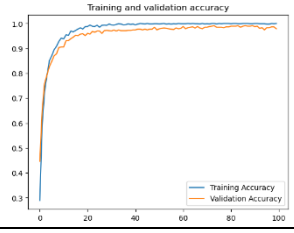
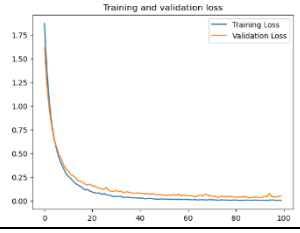
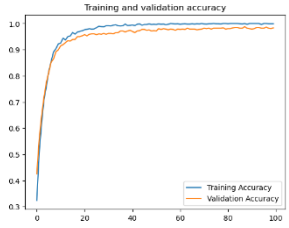
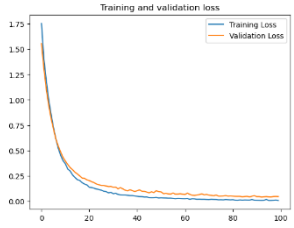
Nominal	Jenis I	Jenis II	Jenis II
Rp 1.000			
Rp 2.000			
Rp 5.000			
Rp 10.000			
Rp 20.000			
Rp 50.000			

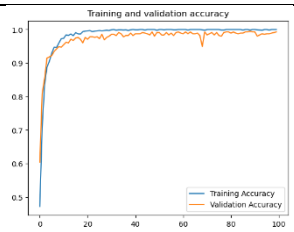
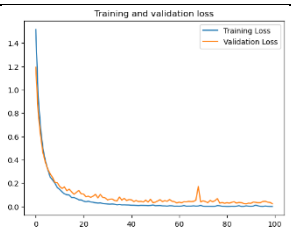
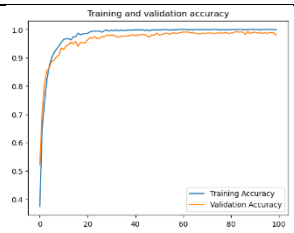
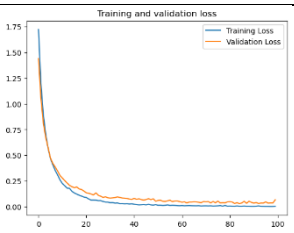
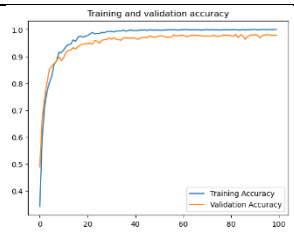
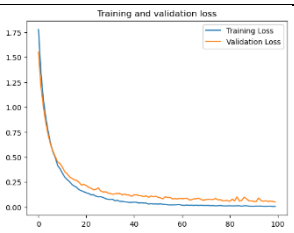
Rp 100.000			

Lampiran 1 Sampel Data tiap nominal

Lampiran 2 Tautan Dataset (QR)

Lampiran 3 Tautan Source Code

Skenario Data	Batch Size	Grafik	
		Training	Validation
70 : 30	8		
	16		
	32		
75 : 25	8		
	16		
	32		

80 : 20	8		
	16		
	32		

Lampiran 4 Grafik Traning dan Validasi MobileNetV2