PENDETEKSIAN PENYAKIT *DIABETIC RETINOPHATY* MELALUI CITRA FUNDUS MENGGUNAKAN MODEL *CNN INCEPTIONRESNET-V2*

(Skripsi)

Oleh

RENATA ADISTI PRATIWI
2015061025

FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

PENDETEKSIAN PENYAKIT *DIABETIC RETINOPHATY* MELALUI CITRA FUNDUS MENGGUNAKAN MODEL *CNN INCEPTIONRESNET-V2*

Oleh

Renata Adisti Pratiwi

Skripsi

Sebagai Salah Satu Syarat untuk Mencapai Gelar
SARJANA TEKNIK

Pada

Jurusan Teknik Elektro
Fakultas Teknik Universitas Lampung

FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

ABSTRAK

PENDETEKSIAN PENYAKIT *DIABETIC RETINOPHATY* MELALUI CITRA FUNDUS MENGGUNAKAN MODEL *CNN INCEPTIONRESNET-V2*

Oleh

Renata Adisti Pratiwi

Diabetic Retinopathy (DR) adalah komplikasi serius diabetes melitus yang dapat menyebabkan kebutaan. Deteksi dini melalui analisis citra fundus sangat penting, tetapi keterbatasan jumlah *ophthalmologist* mendorong pengembangan metode otomatis berbasis kecerdasan buatan. Penelitian ini mengembangkan model deteksi DR berbasis *CNN InceptionResNet-V2* untuk mengklasifikasikan lima tingkat keparahan DR. Model dianalisis menggunakan *input* 3 kanal RGB, kanal hijau (murni dan RGB tiruan), serta dampak prapemrosesan citra seperti CLAHE*, Sharpening,* dan *Super Resolution.*Evaluasi kinerja dilakukan dengan akurasi*, sensitivitas, spesifisitas, AUC, F1-score, Confusion Matrix,* dan waktu komputasi. Percobaan dengan berbagai *batch size* (8, 16, 32, 64) menguji stabilitas dan efisiensi model. Model tanpa prapemrosesan (3 kanal RGB) mencapai akurasi 86%, tetapi kesalahan signifikan terjadi pada kelas mayoritas dan minoritas dibanding percobaanlainnya. Sebaliknya, model dengan kanal hijau RGB tiruan mencapai akurasi 96% dan *F1-score* 95,91%, dengan stabilitas terbaik pada *batch size* 16 sebagai konfigurasi optimal. Kombinasi kanal hijau RGB tiruan dengan CLAHE dan *Sharpening* memberikan kinerja terbaik. Sementara itu, kanal hijau murni menawarkan waktu komputasi lebih cepat tetapi kinerja kurang konsisten. Penelitian ini menegaskan pentingnya pemilihan teknik prapemrosesan dan konfigurasi *batch size* dalam meningkatkan akurasi deteksi otomatis DR. Hasil ini mendukung pengembangan sistem diagnosis berbasis CNN yang efisien untuk membantu deteksi dini DR dalam pengambilan keputusan medis.

Kata kunci : *Diabetic Retinopathy (DR),InceptionResNet-V2*, Citra fundus, Kanal Hijau

ABSTRACT

DETECTION OF DIABETIC RETINOPATHY THROUGH FUNDUS IMAGES USING THE CNN INCEPTIONRESNET-V2 MODEL

By

Renata Adisti Pratiwi

Diabetic Retinopathy (DR) is a serious complication of diabetes mellitus that can lead to blindness. Early detection through fundus image analysis is crucial, but the limited number of ophthalmologists drives the development of automated methods based on artificial intelligence. This study developed a DR detection model using CNN InceptionResNet-V2 to classify five levels of DR severity. The model was analyzed using 3-channel RGB inputs, green channel (pure and synthetic RGB), as well as the impact of image preprocessing techniques such as CLAHE, Sharpening, and Super Resolution.Performance evaluation was conducted using accuracy, sensitivity, specificity, AUC, F1-score, Confusion Matrix, and computation time. Experiments with various batch sizes (8, 16, 32, 64) tested the model's stability and efficiency. The model without preprocessing (3-channel RGB) achieved an accuracy of 86%, but significant errors were observed in majority and minority classes compared to other experiments. In contrast, the model with synthetic RGB green channel achieved an accuracy of 96% and an F1-score of 95.91%, demonstrating the best stability with batch size 16 as the optimal configuration. The combination of synthetic RGB green channel with CLAHE and Sharpening produced the best performance. Meanwhile, the pure green channel offered faster computation time but less consistent performance. This study highlights the importance of selecting appropriate preprocessing techniques and batch size configurations to enhance the accuracy of automated DR detection. The findings support the development of efficient CNN-based diagnostic systems to aid early DR detection in medical decision-making.

Keywords: Diabetic Retinopathy (DR), InceptionResNet-V2, Fundus Image, Green Channel

Judul Skripsi	:	**PENDETEKSIAN PENYAKIT DIABETIC RETINOPHATY MELALUI CITRA FUNDUS MENGGUNAKAN MODEL CNN INCEPTIONRESNET-V2**
Nama mahasiswa	:	Renata Adisti Pratiwi
Nomor Pokok Mahasiswa	:	2015061025
Program Studi	:	Teknik Informatika
Jurusan	:	Teknik Elektro
Fakultas	:	Teknik

MENYETUJUI

1. Komisi Pembimbing

Pembimbing Utama	Pembimbing Pendamping
Ir. Ing. Hery Dian Septama, S.T.,IPM.	**Ir.Titin Yulianti,S.T.,M.Eng.IPM.**
NIP. 19830915200812100	NIP. 198807092019032015

2. Mengetahui

Ketua Jurusan Teknik Elektro	Ketua Program Studi Teknik Informatika
Herlinawati, S.T.,M.T.	**Yessi Mulyani, S.T.,M.T.**
NIP. 197103141999032001	NIP. 197312262000122001

MENGESAHKAN

1. Tim Penguji

   Ketua : **Ir. Ing. Hery Dian Septama, S.T.,IPM.**

   Sekertaris : **Ir.Titin Yulianti,S.T.,M.Eng.IPM.**

   Penguji : **Yessi Mulyani, S.T.,M.T.**

2. Dekan Fakultas Teknik

Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc.
Nip. 197509282001121002

Tanggal Lulus Ujian Skripsi : **03 Januari 2025**

SURAT PERNYATAAN

Yang bertanda tangan dibawah ini:

Nama : Renata Adisti Pratiwi

NPM : 2015061025

Dengan ini menyatakan bahwa skripsi saya dengan judul "Pendeteksian Penyakit *Diabetic Retinophaty* Melalui Citra Fundus Menggunakan Model *CNN InceptionResNet-V2*" dibuat oleh saya sendiri. Semua Hasil yang termuat dalam skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti bahwa skripsi ini merupakan salinan atau dibuat oleh orang lain, maka saya bersedia menerima sanksi sesuai dengan ketentuan hukum atau akademik yang berlaku.

Bandar Lampung, 23 Januari 2025

Pembuat pernyataan

Renata Adisti Pratiwi

NPM. 2015061025

RIWAYAT HIDUP

Penulis lahir di Kota Pariaman pada tanggal 17 Januari 2002, sebagai anak kedua dari dua bersaudara dari pasangan Bapak Albudri dan Ibu Endang Kusmiati. Penulis menyelesaikan sekolah dasar di SDN 1 Sukadana Baru pada tahun 2014, dilanjutkan di SMPN 2 Sekampung pada tahun 2017, dan menyelesaikan pendidikan menengah di SMAN 1 Sekampung pada tahun 2020.

Pada tahun yang sama, penulis berhasil masuk sebagai mahasiswa program studi Teknik Informatika melalui jalur undangan atau SNMPTN. Selama perjalanan kuliah, penulis aktif terlibat dalam berbagai kegiatan. Menjadi anggota Himpunan Mahasiswa Teknik Elektro Universitas Lampung, terlibat dalam departemen Pengembangan Keteknikan periode 2020/2022. Pada Forum Silaturahmi dan Study Islam Fakultas Teknik, terlibat dalam departemen Akademik dan Riset (AKRIS) sebagai anggota di periode 2021/2022, kemudian menjadi sekertaris departemen AKRIS di periode 2022/2023. Selain itu, penulis menjadi penerima beasiswa Kartu Indonesia Pintar (KIP-K) selama masa perkuliahan yaitu 2020 sampai 2024.

Dalam konteks akademis, penulis juga terlibat sebagai asisten laboratorium Teknik Komputer pada tahun 2022 sampai 2023. Pengembangan diri penulis tidak hanya terbatas pada lingkup kampus, namun juga mencakup program Studi Independen di mitra Orbit Future Academy dengan fokus pada domain AI Mastery. Pada tahun 2023, penulis menjalani program magang bersertifikat Kampus Merdeka Batch 5 di Diterktorat Jendral Pendididkan Tinggi, Riset dan Teknologi Kementrian Pendidikan, Kebudayaan, Riset dan Teknologi sebagai Blockchain / Web3 Developer selama periode 14 Agustus hingga 31 Desember 2023. Selain itu, pengalaman praktis penulis juga terlihat dalam pelaksanaan Kuliah Kerja Nyata di Desa Ulok Mukti, Kecamatan Ngambur, Pesisir Barat, Lampung.

MOTTO

مَا وَدَّعَكَ رَبُّكَ وَمَا قَلَى

“Tuhanmu tiada meninggalkan kamu dan tiada (pula) benci kepadamu.”
(QS. Ad-Dhuha : 3)

فَإِنَّ مَعَ الْعُسْرِ يُسْرًا , إِنَّ مَعَ الْعُسْرِ يُسْرًا

“Maka sesungguhnya bersama kesulitan ada kemudahan. Sesungguhnya bersama kesulitan ada kemudahan”
(QS. Al-Insyirah: 5-6).

“Pengalamanmu tidak mendefinisikan siapa dirimu. Apa yang penting adalah bagaimana kamu belajar dan tumbuh dari pengalaman itu. Proses penyembuhan mungkin membutuhkan waktu, tetapi kamu berhak untuk merasa baik tentang diri sendiri dan bergerak maju dengan kehidupanmu.”.
(*Anonym*)

“Belajar meyakini bahwa tidak ada takdir yang buruk. Semua takdir adalah baik. Tidak ada yang Allah rencanakan bagi hidup kita kecuali kebaikan. Tidak ada yang Allah kehendaki bagi kehidupan kita kecuali kebaikan, karena Allah Maha Baik. Mustahil Allah yang Maha Baik menyakiti hambaNya. (*back to first motto*)
(Ust. Hanan Attaki)

" Anything is possible. Your happiness is a choice you can make. Don't hesitate, and please don't regrets, no matter what happens."
(Renata Adisti Pratiwi)

PERSEMBAHAN

Dengan tulus, karya ini kupersembahkan kepada.Kedua Orang Tuaku tercinta. Mamaku tercinta **Endang Kusmiati** sang lentera yang doanya tak pernah padam mengiringi setiap langkahku, sandaranku yang tanpa lelah memberikan semangat dan motivasi untuk terus maju. Kuucapkan terima kasih sebesar-besarnya karena telah memberikan banyak kebaikkan yang tidak akan pernah bisa terbalaskan. Kupersembahkan karya ini dengan tulus untuk almarhum papa tercinta **Albudri** yang selalu menjadi alasanku untuk terus bertahan dan maju demi membuktikan kata yang selalu beliau ucapkan "Adis anak kebanggaan Papa". Skripsi ini merupakan bukti kecil dari baktiku kepada mereka.

Tidak lupa juga kepada abangku **Reynaldi Indra Pratama** yang selalu menemani memberikan semangat, do'a ,saran dan motivasi untukku.

Juga kepada seluruh dosen, rekan-rekan seperjuangan, dan civitas Jurusan Teknik Elekro Universitas Lampung. Untuk mereka yang bertanya, *"Kapan lulus?"* dengan bangga kupersembahkan karya ini sebagai jawabannya..

Akhir kata, terima kasih untuk diriku sebagai nakhoda yang tak menyerah dan tenggelam meski badai menghadang.

SANWACANA

Alhamdulillahirabbil'alamin, puji syukur selalu terpanjatkan kepada Allah Subhanahu Wa Ta'ala atas segala rahmat, karunia, dan hidayah-Nya sehingga dapat menyelesaikan skripsi dengan judul "**Pendeteksian Penyakit Diabetic Retinophaty Melalui Citra Fundus Menggunakan Model CNN InceptionResNet-V2**". Shalawat serta salam senantiasa tercurah kepada kekasih Allah SWT panutan seluruh umat yakni Baginda Rasulullah SAW yang telah memperbaiki ahlak budi pekerti manusia dari zaman kebodohan sampai saat ini. Dalam proses pelaksanaan pembuatan skripsi ini tidak lepas dari bantuan, bimbingan serta dukungan baik secara moril maupun materil dari berbagai pihak. Oleh karena itu, diucapkan terima kasih kepada semua pihak yang telah berkonstribusi baik secara langsung maupun tidak langsung pada pembuatan skripsi ini, khususnya kepada :

1. Dekan Fakultas Teknik Univeritas Lampung, Bapak Dr. Eng. Helmy Fitriawan, S.T., M.Sc.,.
2. Ibu Herlinawati, S.T., M.T. selaku ketua Jurusan Teknik Elektro Universitas Lampung.
3. Ibu Yessi Mulyani, S.T., M.T selaku Ketua Program Studi Teknik Informatika Universitas Lampung sekaligus dosen pembimbing akademik,serta dosen penguji skripsi yang selalu memberikan ilmu, bimbingan dan masukkan selama menjalani masa perkuliahan.
4. Bapak Ir. Ing. Hery Dian Septama, S.T.,IPM selaku pembimbing utama yang telah banyak membantu dalam proses pengerjaan skripsi ini dengan memberikan bimbingan, semangat, energi positif dan idenya dalam pengerjaan skripsi ini.
5. Ibu Ir. Titin Yulianti, S.T., M.Eng selaku dosen pembimbing pendampung yang telah meluangkan waktu untuk membimbing ditengah kesibukannya, memberikan ide baru, serta dukungan yang sangat baik dalam pengerjaan skripsi ini.

6. Terutama dan paling utama kepada mama dan abang yang tiada hentinya memberikan semangat, dukungan moril dan materil, doa dan hingga skripsi ini dapat terselesaikan dengan baik.
7. Untuk diri sendiri, karena telah menyelesaikan skripsi ini dengan semua rintangan yang dilalui.
8. Dwindy Monica, Afifah Lutfi Anisa, Putri Pratiwi, Zeri Nurtilia, Dwi Putri Oktaviani, Annisa Fitriani, Elda Aqil Usrotin, Adinda Tasya Amelia, dan Maylan yang senantiasa menemani, berbagi keluh-kesah, memberikan dukungan, meluangkan waktu , memberikan masukan dan bertukar pandangan selama masa perkuliahan serta pengerjaan skripsi sehingga dapat terselesaikan dengan baik.
9. 20*****003 yang pernah menemani, membersamai dan memberikan pengalaman serta pelajaran berharga dalam hidup saat masa perkuliahan dan pengerjaan skripsi.
10. Serta pihak-pihak yang tidak bisa saya sebutkan satu persatu

Penelitian ini masih jauh dari kata sempurna, dikarenakan terbatasnya pengalaman dan pengetahuan yang dimiliki. Oleh karena itu, diharapkan segala bentuk saran serta masukan bahkan kritik yang membangun dari berbagai pihak. Semoga skripsi ini dapat bermanfaat bagi pada pembaca dan semua pihak khususnya dalam bidang pengembangan *machine learning* dan *deep learning*.

Bandar Lampung, 23 Januari 2025

Penulis

Renata Adisti Pratiwi

DAFTAR ISI

Halaman

DAFTAR GAMBAR

DAFTAR TABEL

I. PENDAHULUAN

1.1 Latar Belakang

Diabetes Melitus (DM) merupakan salah satu penyakit yang menghantui kesehatan global, menurut data dari *International Diabetes Federation* (IDF) tahun 2021, DM merupakan penyebab dari 6,7 juta kematian di dunia. *Diabetes Melitus* ditandai dengan tingginya tingkat gula darah penderita yang disebabkan kurangnya kadar insulin di tubuh, sehingga mengakibatkan kerusakan yang serius terutama pada pembuluh darah [1]. Dikategorikan sebagai gangguan metabolik kronis diabetes menyebabkan berbagai komplikasi serius yang menjalar ke berbagai organ tubuh, salah satunya organ mata, diabetes jenis ini disebut dengan *Diabetic retinophaty* (DR). Komplikasi *mikrovaskular* merupakan akumulasi kerusakan pembuluh darah di retina dalam jangka panjang yang menjadi penyebab dari DR, dampak terburuk yang mungkin dialami penderita DR adalah kehilangan pengelihatan [1].

Jumlah penderita *Diabetic retinophaty* pada tahun 2020 mencapai 103,12 juta jiwa dan terus meningkat dan diproyeksikan pada tahun 2045 peningkatan mencapai 55,9% dibandingkan dengan tahun 2020. Mengingat tingginya jumlah penderita penyakit ini, teknologi untuk pendeteksian penyakit *Diabetic retinophaty* telah banyak dikembangkan demi mencegah dampak terburuk bagi penderita, pendeteksian ini dilakukan melalui pengamatan dan analisis citra biomedis yaitu melalui citra fundus yang dilakukan oleh *ophthalmologist* [2]. Namun demikian jumlah dari *ophthalmologist* sendiri masih terbatas terutama di wilayah afrika dan asia tenggara.

Citra biomedis merupakan pemanfaatan teknologi yang digunakan untuk mendeteksi berbagai penyakit atau kondisi abnormal pada tubuh manusia berdasarkan ciri – ciri visualnya. Kemudian, dilakukan analisis berdasarkan ciri-ciri tersebut oleh ahli untuk kepentingan diagnosis atau pemantauan kondisi kesehatan.

Citra biomedis yang digunakan untuk mendeteksi berbagai penyakit seperti *Diabetic retinopathy*, *glaucoma*, *age-related macular degeneration*, *cardiovascular*, dan *hypertensive retinopathy,* memanfaatkan kamera fundus dalam pengambilannya sehingga disebut citra fundus [3] . Berdasarkan citra fundus tersebut, dilakukan diagnosis oleh *ophthalmologist* secara konvensional.

Namun, seperti yang disampaikan sebelumnya, jumlah dari *ophthalmologist* sendiri masih terbatas di beberapa wilayah, diagnosis secara konvensional dinilai kurang efisien karena memerlukan waktu yang lama dan biaya yang tidak sedikit. Untuk mengatasi hal tersebut beberapa tahun terakhir dikembangkan teknologi pendeteksian otomatis penyakit pada retina menggunakan *Artificial Intelligence* (AI). Pendeteksian otomatis penyakit pada retina dilakukan dengan menafaatkan AI pada bidang pemrosesan citra untuk memahami data visual sehingga dapat mendeteksi *microaneurysms, haemorrhages, retinal exudates, cotton wool spot*, dan *macular edema*. Secara umum langkah-langkah pendeteksiannya meliputi *image capture, image processing* yang terdiri dari *enhancement*, restorasi, segmentasi*, image registration* dan klasifikasi [6].

Dalam pemrosesan citra, algoritma yang sering digunakan untuk klasifikasi pola atau ciri dari suatu citra adalah arsitektur *Convolutional Neural Network* (CNN). Pada penelitian ini digunakan arsitektur *Convolutional Neural Network* yang telah dikembangkan yaitu model *CNN InceptionResNet-V2*, model ini menggabungkan keunggulan dari dua arsitektur yaitu *Inception* dan *restnet* yang memungkinkan ekstraksi fitur pada berbagai skala sehingga membantu model mengenali objek dengan lebih baik, terutama pada citra yang kompleks. Dengan menggabungkan *residual connections*, model ini mengatasi masalah *vanishing gradient* yang sering terjadi pada jaringan yang sangat dalam dan memungkinkan pelatihan jaringan yang lebih dalam tanpa mengorbankan kinerja. Mempertimbangkan hal tersebut pada penelitian ini dilakukan perbandingan dengan membangun sistem pengenalan tingkatan DR menggunakan model *CNN InceptionResNet-V2* untuk mengenali 5 kelas sekaligus dan disertai beberapa peningkatan pada *Input* gambar (*prapemrosessan images*) untuk menunjang kinerja model menggunakan beberapa pendekatan yang bereda.

Citra masukan dengan kualitas yang baik dapat meningkatkan efisiensi model dalam mengenali pola suatu citra, sehingga dapat dikatakan prapemrosesan citra yang baik akan menghasilkan model pengenalan (klasifikasi) yang baik pula. Berdasarkan hasil penelitian sebelumnya oleh Nugroho (2014) [7] yang memfokuskan pada ruang warana RGB, kanal hijau (Green *channel*) merupakan kanal dengan kontras terbaik bagi model untuk mengenali fitur yang ada pada citra fundus. Sehingga dalam penelitian ini kanal tersebut pula yang diekstraksi kemudian di-*enhancement* sebelum dilakukan ekstraksi fitur demi meningkatkan kualitas citra. *Enhancement* citra dilakukan dengan meregangkan *contrast* menggunakan algoritma CLAHE dan mempertimbangkan penambahan *Sharpening* dan *Super Resolution*. Namun cara pengambilan kanal hijau sendiri ada berbagai macam, yaitu dapat dilakukan dengan menginisiasikan nilai 0 pada kanal Red dan Blue dan nilai 1 pada kanal Green yang menghasilkan kanal hijau murni, ada pula cara dengan menginisiasikan kanal hijau ke dalam ketiga ruang warna yang menghasilkan citra dengan kanal hijau tiruan. Perbedaan ini tentunya membawa dampak pada performa model, sehingga pengaruh kedua cara pengambilan (pendekatan) kanal hijau ini terhadap performa model menjadi salah satu fokus pada penelitian ini.

Kemudian juga dilakukan evaluasi sebagai parameter penguji keberhasilan dari model. Evaluasi dilakukan berdasarkan hasil menghitung nilai *sensitivitas (recall), spesifisitas, akurasi, Area Under the ROC Curve (AUC), F1 score* dan waktu komputasi. Dengan melakukan evaluasi ini diharapkan dapat memberikan citraan komprehensif tentang performa relatif antara model *CNN InceptionResNet-V2* yang menggunakan teknik prapemrosesan citra input RGB, Green Chanel murni dan Green Chanel RGB tiruan.

1.2 Rumusan Masalah

1. Bagaimana membangun model pendeteksian penyakit *Diabetic retinophaty* menggunakan *CNN InceptionResNet-V2* berdasarkan citra fundus?
2. Bagaimana pengaruh praproses citra terhadap performa pemodelan *CNN InceptionResNet-V2 ?*

3. Manakah pendekatan yang menghasilkan performa terbaik dari model *CNN InceptionResNet-V2* untuk pendeteksian penyakit *Diabetic retinophaty* ?

1.3 Tujuan

Adapun tujuan dari penelitian ini adalah sebagai berikut :

1 Membangun model klasifikasi untuk pendeteksian penyakit *Diabetic retinophaty* berdasarkan citra fundus dengan menggunakan model *CNN InceptionResNet-V2* untuk mengenali 5 tingkatan DR sekaligus .

2 Membandingkan dan menganalisis pengaruh pendekatan pada citra *input* , seperti penggunaan 3 kanal RGB, kanal hijau murni, dan kanal hijau RGB tiruan, serta tambahan praproses seperti CLAHE, *sharpening* dan *super resolution*. terhadap performa model *CNN InceptionResNet-V2*.

3 Mengevaluasi performa model *CNN InceptionResNet-V2* berdasarkan nilai sensitivitas (recall), spesifisitas, akurasi, AUC, F1 score, Confusion Matrix, dan waktu komputasi guna menentukan pendekatan terbaik untuk model *CNN InceptionResNet-V2* pada klasifikasi 5 tingkatan DR.

1.4 Manfaat Penelitian

Adapun manfaat yang diharapkan dari penelitian ini adalah sebagai berikut :

1. Manfaat Teoritis

Penelitian ini diharapkan dapat membantu perkembangan ilmu pengetahuan terkait pemanfataan model *CNN InceptionResNet-V2* dalam klasifikasi penyakit *Diabetic retinophaty* sehingga dapat digunakan sebagai bacaan maupun acuan dalam pengembangan maupun penelitian serupa di kemudian hari.

2. Manfaat Praktis

- Bagi penulis, penulis mendapatkan wawasan lebih mengenai model *CNN InceptionResNet-V2* serta kemampuan dalam penerapannya untuk klasifikasi penyakit *Diabetic Retinophaty*
- Bagi peneliti atau akademinsi , penelitian ini dapat menjadi refrensi

perkembangan ilmu pengetahunan terkait klasifikasi penyakit *Diabetic retinophaty* serta mengetahu perbandingan penggunaan model antara model *CNN InceptionResNet-V2* dengan tambahan prapemrosesan pada citra *Input* dan pendekatan pengambilan kanal hijau, dalam klasifikasi tingkat keparahan Diabetic Retinophaty.

- Bagi dunia kedokteran , hasil penelitian ini dapat menjadi bahan pengembangan *Machine learning* untuk diagnosis penyakit *Diabetic retinopathy,* sehingga memudahkan dokter dalam pertimbangan dan pengambilan keputusan terkait diagnosis kemungkinan awal adanya penyakit *Diabetic retinopathy* .

1.5 Batasan Masalah

Penelitian difokuskan pada pembangunan model klasifikasi dengan model *CNN InceptionResNet-V2* untuk pendeteksian *Diabetic retinopathy*. Dilakukan pula perbandingan kinerja model *CNN InceptionResNet-V2* dengan penggunaan prapemrosesan citra yang berbeda. Evaluasi dilakukan dengan menghitung akurasi, sensitivitas, spesifisitas, Area under ROC, *F1 score, confusion matrix* dan waktu komputasi. Data yang digunakan dalam penelitian ini adalah citra fundus *MESSIDOR 2* . Penelitian ini tidak mencakup pembahasan lebih lanjut terkait *design* UI dan UX dari visualisasi sistem pendeteksian *Diabetic retinopathy* dengan model *CNN InceptionResNet-V2* .

1.6 Sistematika Penulisan Laporan

Laporan ini dibagi menjadi beberapa bab untuk memudahkan dalam penguraian, antara lain :

BAB I : PENDAHULUAN

Bab ini membahas tentang latar belakang dan rumusan masalah dari pentingnya dilakukan perbandingan kinerja model antara model *CNN InceptionResNet-V2* dalam klasifikasi penyakit *Diabetic Retinophaty* dengan pendekatan

berbeda pada praproses citra *input* , menjelaskan tujuan dilakukannya perbandingan performa antara model *CNN InceptionResNet-V2* dengan pendekatan yang berbeda, batasan masalah dari penelitian ini , serta sistematika penulisan laporan.

BAB II : TINJAUAN PUSTAKA

Bab ini akan membahas mengenai dasar-dasar teori yang digunakan dalam klasifikasi penyakit *Diabetic Retinophaty,* perbandingan penggunaan model antara model *CNN InceptionResNet-V2* dengan prapemrosesan citra yang berdeda sebagai pedoman dan acuan dalam pemecahan masalah berdasarkan buku, jurnal, skripsi dan penelitian-penelitian sebelumnya yang terkait dengan topik penelitian ini,

BAB III : METODE PENELITIAN

Pada bab ini membahas tentang tahap-tahap sistematis dan logis dalam pembuatan sistem klasifikasi penyakit *Diabetic retinophaty* menggunakan model *CNN InceptionResNet-V2* framework OSEMN yang terdiri dari tahap *obtain, srubbing, exporating, modeling* dan *intepreter*. Masing-masing tahapan menjelaskan secara detail proses yang dilakukan.

BAB IV : HASIL DAN PEMBAHASAN

Pada bab ini membahas tentang hasil yang didapatkan pada pembuatan dan pengujian sistem klasifikasi penyakit *Diabetic retinophaty* menggunakan model *CNN InceptionResNet-V2* dengan citra *input*an meliputi penggunakan 3 kanal RGB, menggunakna beberapa tenik prapemrosesan citra pengambilan kanal hijau murni dan RGB tiruan, CLAHE, Superesolution, dan *Sharpening*. Hasil yang dipaparkan berupa perbandingan nilai evaluasi yang dilakukan

berdasarkan parameter berupa akurasi, sensitivitas, spesifisitas, Area under ROC, dan waktu komputasi.

BAB V : KESIMPULAN DAN SARAN

Pada bab ini membahas kesimpulan dari hasil penelitian perbandingan penggunaan antar praproses citra pada model *CNN InceptionResNet-V2* dalam klasifikasi penyakit *Diabetic retinophaty* beserta saran dari hasil penelitian yang diharapkan dapat meningkatkan wawasan serta kemajuan dalam pengembangan penelitian selanjutnya.

II. TINJAUAN PUSTAKA

2.1 Diabetes

Diabetes adalah penyakit kronis yang terjadi ketika tubuh tidak dapat menghasilkan cukup insulin atau tidak dapat menggunakan insulin secara efektif. Insulin adalah hormon yang diproduksi oleh pankreas yang memungkinkan sel-sel tubuh untuk mengambil glukosa (gula) dari darah dan menggunakannya sebagai energi. Tanpa insulin yang cukup atau jika tubuh tidak dapat menggunakan insulin dengan baik, glukosa menumpuk dalam darah, menyebabkan kadar gula darah tinggi yang dapat merusak berbagai organ dan sistem tubuh[8].

Diabetes diklasifikasikan menjadi dua tipe utama yaitu diabetes Tipe 1 dan diabetes Tipe 2. Sistem kekebalan tubuh menyerang sel-sel beta pankreas yang berfungsi memproduksi insulin dan menghancurkannya hal ini terjadi pada diabetes tipe 1 sehingga dapat dikatakan diabetes tipe 1 merupakan keadaan autoimun. Karena sel beta pangkreas yang menghasilkan insulin hancur, penderita diabetes tipe 1 perlu menggunakan insulin setiap hari agar kadar gula darah tetap seimbang. Berbeda dengan penderita Diabetes Tipe 2, kondisi ini adalah saat tubuh tidak memproduksi insulin yang cukup serta tidak dapat menggunakan insulin dengan baik. Jenis diabetes ini sering terkait dengan obesitas dan gaya hidup yang tidak sehat dan umum terjadi[8]. Tiap tiap penderita diabetes mengalami kekurangan insulin yang berfungsi untuk memecah glukosa dalam tubuh untuk diubah menjadi energi, kelebihan kadar glukosa ini dalam jangka waktu tertentu dapat mengakibatkan berbagai komplikasi salah satunya adalah *Diabetic retinopathy*[9].

2.2 *Diabetic Retinopathy*

Diabetic retinopathy adalah komplikasi mata pada penderita *diabetes mellitus* yang berefek pada gangguan penglihatan bahkan hingga kebutaan. Kerusakan dan sumbatan pada pembuluh darah halus merupakan tanda suatu *mikroangiopati progresif* yang disebut *Diabetic retinopathy* . Kelainan *patologis* paling dini dari kondisi ini adalah adanya penebalan membran *basalis endotel kapiler* serta pengurangan total banyaknya *perisit*. Bahkan terdapat 60-70% pasien penderita *diabetes mellitus* yang juga menderita *Diabetic retinopathy* setelah sepuluh tahun [10]. *Diabetic retinopathy* dapat disebabkan oleh beberapa keadaan abnormal sebagai berikut [11]:

2.2.1 Microneurysms

Merupakan ciri pertama yang muncul pada *Diabetic retinopathy* [10] yaitu adanya bintik merah kecil yang letaknya tidak jauh dari pembuluh darah terutama *polus posterior* yang diakibartkan oleh penonjolan daerah kapiler terutama daerah vena.

2.2.2 Hard Exudates

Merupakan *infiltrasi lipid* atau lemak yang masuk ke dalam retina dengan citraan ireguler atau bentuknya tidak tetap, berwarna kekuning-kuningan yang awalny *exudate* ini membesar dan bergabung. *Exudate* dapat timbul kemudian hilang dalam beberapa minggu.

2.2.3 Soft Exudates atau *Cotton Wool Spots (CWS)*

Adalah *iskemia retina* yang mana pada pemeriksaan *optalmoskopi* terlihat bercak berwarna kuning dan putih yang bersifat difus. Bagian tepi atau daerah *non-perfusi* merupakan tempat biasanya di temukan *Soft exudate*

2.2.4 *Hemorrages* (HEM)

Merupakan bercak berwarna merah dengan tepi yang tidak beraturan yang terjadi karena kebocoran kapiler.

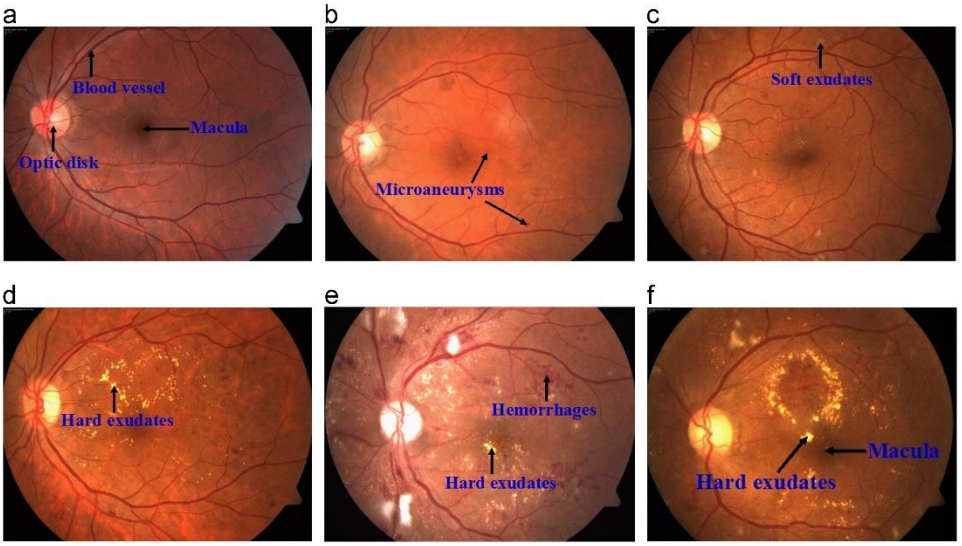
2.2.5 Neovascularization (NV)

Pada retina, pembuluh darah biasanya berada di permukaan jaringan. Pembuluh-pembuluh ini tampak berkelok-kelok, dalam, berkelompok, dan memiliki pola yang

tidak teratur. Awalnya, pembuluh darah ini berada di dalam jaringan retina, kemudian dapat tumbuh ke area preretina dan menuju ke badan kaca. Jika neovaskularisasi di area ini pecah, hal tersebut dapat menyebabkan pendarahan pada retina, preretina (subhyaloid), atau vitreus (badan kaca).NV merupakan bagian dari kasus ringan disebut juga *Intraretinal microvascular abnormalities* (IRMA). IRMA mewakili pertumbuhan pembuluh darah baru dalam retina atau, lebih mungkin, pembuluh yang sudah ada dengan proliferasi sel endotel yang berfungsi sebagai *shunts* melalui area *nonperfusi*. Pembuluh darah baru, baik di atau dekat *diskus optik* (NVD) atau di tempat lain di retina (NVE), menunjukkan adanya *retinopati diabetik proliferatif*, dengan peningkatan risiko kehilangan penglihatan akibat perkembangan perdarahan *vitreous* atau *ablasi retina traksi* [12].

2.2.6 Macular Edema (ME)

Merupakan *edema* retina yang dapat diketahui melalui hilangnya citraan retina terutama pada daerah makula yang berdampak pada gangguan ketajaman penglihatan.



Gambar 2. 1 Ciri oftalogis penyakit pada retina[13]

Keadaan abnormal seperti yang ditunjukan oleh gambar 2,1 mengidentifikasikan perkembangan tingakat keparahan dari DR. *Early Treatment Diabetic retinopathy Study* (ETDRS) *Diabetic retinopathy* secara klinis melakukan standarisasi untuk meverifikasi tingkatan DR, yaitu, *Non proliferative Diabetic retinopathy* (NPDR) dan *Proliferative Diabetic retinopathy* (PDR). Seperti ditunjukkan pada Tabel 2. **1**

Tipe *Diabetic retinopathy* dan ciri klinis Tingkatan ini dimulai dari *Non proliferative Diabetic retinopathy* (NPDR) ringan, yang ditandai dengan peningkatan permeabilitas vaskular, ke NPDR sedang dan berat, dengan penutupan *vaskular*, atau juga dikenali berdasarkan karakteristik kemunculan *microaneurysm*, *exudate*, *hemorrhanges* dan *microinfarcts.* Kemudian *Proliferative Diabetic retinopathy* (PDR), dengan kemunculan pertumbuhan pembuluh darah baru pada retina dan permukaan *posterior vitreous*. PDR merupakan stadium lanjut dari *Diabetic retinopathy* [10].

Tabel 2. 1 Tipe *Diabetic retinopathy* dan ciri klinis

Tipe DR	**Level**	**Ciri patologi**
Non-proliferative Diabetic retinopathy (NPDR)	ringan ke sedang	*Microaneurysms*, tidak ada atau sedikit ditemukan intra-retinal *hemorrhages*, *hard exudate*
	Sedang	*Microaneurysms*, intra-retinal *hemorrhages*, *hard exudate* dan *macular edema*
	sedang ke berat	*extensive intra-retinal hemorrhages* dan/atau *microaneurysms* dan/atau *cotton wool spots, venous beading* (rangkaian vena yang seperti manik-manik) atau *intra-retinal microvascular abnormalities* (IRMA)
	berat ke sangat berat	plus *cotton wool spots*, *venous beading*, dan IRMA, semuanya minimal berada pada dua kuadran retina. *Intra-retinal hemorrhages* dalam empat kuadran, *venous beading* dalam dua kuadran dan IRMA yang parah dalam satu

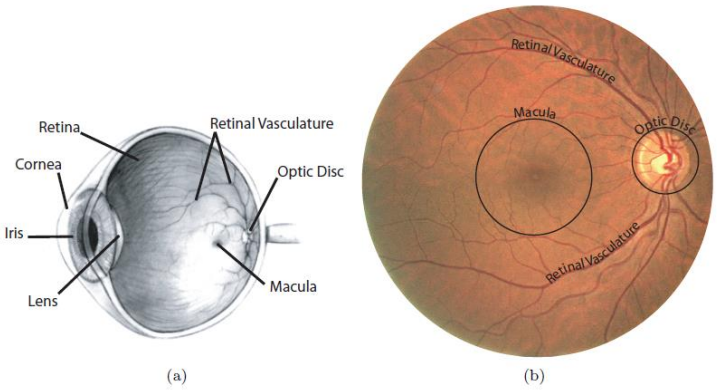
		kuadran.
Proliferative retinopathy (PDR)	Resiko tinggi	Ditandai dengan adanya *vitreous hemorrhage*, *blood vessels* baru pada *disc* > 1/3 DD (faktor penting penyebab kehilangan penglihatan pada penderita DR), munculnya pembuluh darah baru di area lain > ½ DD *tractional retinal detachment, neovascularization* pada iris

Tingkat *retinopati* tidak selalu berkorelasi dengan fungsi visual dan DR dengan tingkat berat awalnya dapat terjadi tanpa kehilangan penglihatan yang signifikan. Mengidentifikasi tingkat keparahan *retinopati diabetik* penting untuk menentukan risiko progresi dan perawatan yang tepat untuk menjaga penglihatan. Setiap tingkat NPDR dikaitkan dengan risiko yang sesuai untuk progresi ke PDR dan risiko selanjutnya yaitu kehilangan penglihatan yang berat [12].

2.3 Retina dan Citra Fundus

Retina merupakan lapisan dari jaringan dengan tebal hampir 0,1 - 0,23 mm yang melapisi bagian terdalam bola mata dengan lapisan yang paling tipis pada *fovea* sentral yaitu bagian tengah *macula*. Jaringan berkembang dari sebuah kantong embrio otak depan, oleh karena itu retina disebut sebagai bagian dari otak. Retina mencakup *neuron sensorik* yang merespon cahaya dan jaringan saraf yang berfungsi menerima bayangan visual, sebagian dianalisisnya, dan informasi yang telah dimodifikasi ini diteruskan ke saraf optik menuju otak [8].

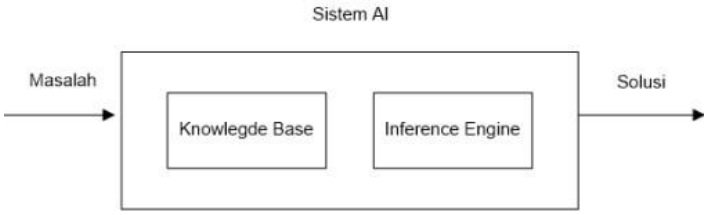
Untuk keperluan pendeteksian dan diagnosis penyakit pada retina diperlukan citra fundus yang diambil dengan menggunakan kamera fundus. Citra fundus yang diambil dengan kamera fundus merupakan citra 2-D yang intensitasnya merepresentasikan jumlah cahaya yang dipantulkan [3]. Struktur citra fundus secara umum terdiri dari *blood vessels*, *macula* dan *optic disc/optic nerve*, seperti terlihat pada gambar 2.2



Gambar 2. 2. Struktur citra fundus[14]

2.4 Artificial intelegent

Artificial Intelegent (AI) merupakan ilmu pengetahuan dibidang komputer yang mengkaji cara bagaimana komputer dapat membantu manusia dalam mengerjakan berbagai tugas layaknya manusia itu sendiri. AI pada sistem diharapkan menghasilkan keluaran (*output*) berupa solusi dari permasalahan dengan bekal sekumpulan pengetahuan (*knowlagde*) yang sebelumnya telah diperlajari oleh sistem tersebut.



Gambar 2. 3 Sistem menggunakan *Artificial Intelligence*

Pada Gambar 2.3, Sistem berbasis AI menerima masukan berupa masalah. Di

dalamnya, sistem AI harus memiliki basis pengetahuan yang berisi kumpulan informasi atau pengetahuan. Mesin inferensi kemudian digunakan untuk menarik kesimpulan berdasarkan fakta atau informasi yang ada dalam basis pengetahuan tersebut. Hasil akhir yang dihasilkan oleh sistem AI berupa solusi untuk permasalahan yang dianalisis melalui proses inferensi.

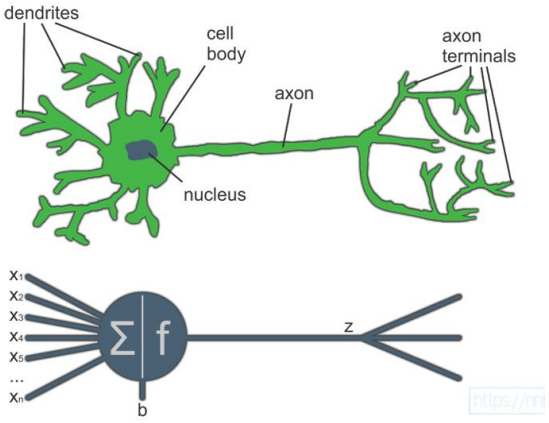
2.5 *Machine learning*

Machine learning (ML) merupakan salah satu teknik populer dalam *Artificial intelligence* (AI). AI itu sendiri merupakan salah satu bagian dari Revolusi Industri 4.0. Machine Learning (ML) adalah sebuah metode yang memungkinkan program untuk belajar dari sejumlah besar data. Pendekatan ini sangat berbeda dari komputer tradisional yang menggunakan metode statis dan tidak memiliki kemampuan belajar mandiri. Proses pembelajaran dalam ML meniru cara manusia belajar, yaitu dengan mempelajari banyak contoh yang telah dialami sebelumnya. Contoh-contoh tersebut kemudian dianalisis untuk membantu menjawab berbagai pertanyaan di masa mendatang[15]. Beberapa tahun terakhir *Machine learning* tengah banyak digunakan dalam berbagai bidang salah satunya yaitu biomedik. Hal ini juga digunakan untuk pendeteksian *Diabetic retinopathy*. Pada Gambar 2.4 ditunjukkan bagaimana *Machine learning style* dalam penelitian terkait klasifikasi *Diabetic retinopathy*.

Gambar 2.4. *Machine learning* style untuk klasifikasi *Diabetic retinopathy*[16]

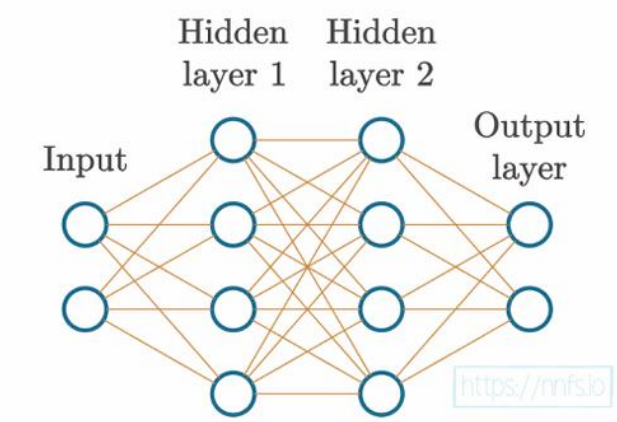
2.6 Jaringan Syaraf Tiruan (*artificial neural networks*)

Jaringan Syaraf Tiruan (JST) atau Neural Networks adalah model pembelajaran mesin yang dirancang untuk meniru cara kerja otak manusia. Model ini terdiri dari neuron-neuron yang terhubung dalam beberapa lapisan seperti lapisan masukan (*input layer*), lapisan tersembunyi (*hidden layers*), dan lapisan keluaran (*output layer*). Setiap neuron dalam jaringan menerima masukan, menerapkan bobot (*weights*) dan bias (*biases*), lalu menghasilkan keluaran melalui fungsi aktivasi. Proses ini memungkinkan JST mempelajari pola kompleks dari data dan diaplikasikan dalam berbagai bidang seperti klasifikasi gambar, pengenalan suara, dan prediksi data.



Gambar 2. 5 Perbandingan jaringan saraf asli dengan jaringan saraf tiruan[17]

Pada gambar 2.5, *impulse* atau masukan bagi jaringan syaraf direpresentasikan sebagai x yang dalam JST disebut *input*, kemudian *input* ini akan dikalikan dengan nilai bobot (*weight*) masing- masing kemudian di tambahkan dengan nilai bias (b). Selanjutnya nilai bobot akan dikenakan fungsi aktivasi (f) yang kemudian *output* nya akan berupa nilai bobot baru (z).



Gambar 2. 6 Jaringan syaraf tiruan sederhana[17]

Pada gambar 2.6 menunjukan jaringan saraf tiruan sederhana yang terdisi dari *input* layer, 2 hidden layer dan output layer. Setiap garis membawa nilai bobot yang berubah seiring penerapan perhitungan matematis demi menghitung kemungkinan berupa prediksi sebagai tujuan akhir dari pembuatan JST tersebut.

Dalam JST terdapat beberapa variabel yang terlibat, variabel ini adalah komponen yang diolah selama pelatihan dan inferensi. Variable yang dimaksdu mencakup Masukan (*Inputs*) yang merupakan data numerik berupa fitur-fitur yang akan diproses jaringan, contohnya adalah nilai pixel dalam klasifikasi gambar atau nilai

numerik dari sensor. Keluaran (*Outputs*) atau hasil akhir dari jaringan setelah masukan diproses, seperti probabilitas untuk klasifikasi atau nilai kontinu untuk regresi. Kemudian juga terdapat *Gradien* adalah nilai turunan dari fungsi *loss* terhadap parameter. Gradien digunakan untuk memperbarui bobot dan bias dalam *optimisasi*. Matriks Bobot (*Weights Matrix*) yang menyimpan bobot untuk semua koneksi antar neuron di dua lapisan bertetangga. Vektor Bias (*Bias Vector*), vektor yang menyimpan bias untuk setiap neuron dalam lapisan, serta *Feature Map* yang merupakan representasi data hasil lapisan tersembunyi setelah proses aktivasi

Parameter dalam JST

Parameter adalah nilai-nilai internal dalam model yang dioptimalkan selama proses pelatihan. Parameter dalam jaringan saraf tiruan, didalamnya termasuk:

1. Bobot (*Weights*):

Bobot menentukan seberapa besar pengaruh setiap masukan terhadap keluaran neuron. Bobot dikalikan dengan nilai masukan dan dijumlahkan kemudian di tambarkan nilai bias sebelum diterapkan fungsi aktivasi. Bobot dioptimalkan selama pelatihan menggunakan algoritma optimisasi seperti Gradient Descent.[17]

2. Bias (*Biases*):

Bias adalah nilai tambahan untuk menggeser hasil perhitungan, memungkinkan jaringan menangkap hubungan yang lebih fleksibel. Bias menambahkan *offset* sehingga jaringan tidak hanya bergantung pada hasil perkalian bobot dan masukan.[17] Nilai bias dalam jaringan saraf tiruan (*neural network*) juga diinisialisasi dengan nilai acak pada awalnya, mirip dengan bobot. Selama proses pelatihan, bias ini disesuaikan bersama dengan bobot menggunakan algoritma optimasi seperti *backpropagation*. Tujuan dari penyesuaian ini adalah untuk meminimalkan kesalahan antara prediksi model dan nilai sebenarnya.

Jadi, baik bobot maupun bias adalah parameter yang dapat dilatih dan disesuaikan selama proses pelatihan model untuk meningkatkan kinerja jaringan saraf dalam memprediksi data baru.

Hyperparameter dalam JST

Hyperparameter adalah nilai-nilai yang tidak dioptimalkan selama pelatihan, melainkan ditentukan sebelum proses pelatihan dimulai. Hyperparameter mencakup:

1. Learning Rate

Mengontrol seberapa besar langkah perubahan bobot dan bias selama pembaruan. Nilai learning rate yang terlalu tinggi dapat menyebabkan model gagal konvergen, sementara yang terlalu rendah memperlambat pelatihan.[17]

2. Jumlah Epoch

Epoch adalah jumlah iterasi di mana seluruh dataset dilalui selama pelatihan.

3. Batch Size

Jumlah sampel data yang diproses dalam satu iterasi pelatihan. Batch size yang lebih besar meningkatkan stabilitas pembaruan parameter, tetapi membutuhkan lebih banyak memori.

4. Arsitektur Jaringan:

Termasuk jumlah lapisan tersembunyi (*hidden layers*) dan jumlah neuron di setiap lapisan. Desain arsitektur ini memengaruhi kapasitas model untuk menangkap pola dalam data.

5. Fungsi Aktivasi

Fungsi yang menentukan keluaran dari setiap neuron, seperti ReLU atau Softmax untuk klasifikasi multi-kelas. Fungsi aktivasi diterapkan pada *output* neuron (atau lapisan neuron), yang memodifikasi output. Fungsi aktivasi yang non-linear memungkinkan jaringan saraf dengan dua atau lebih lapisan tersembunyi untuk memetakan fungsi non-linear.

Secara umum, jaringan saraf Anda akan memiliki dua jenis fungsi aktivasi. Yang pertama adalah fungsi aktivasi yang digunakan di lapisan tersembunyi, dan yang kedua digunakan di lapisan output. Biasanya, fungsi aktivasi yang digunakan untuk neuron tersembunyi akan sama untuk semua neuron, tetapi tidak harus demikian.

6. Fungsi Loss

Fungsi yang digunakan untuk mengukur kesalahan atau eror antara keluaran jaringan dan target. Penghitungannya dapat menggunkan *Mean Squared Error* (MSE) atau pula *Cross-Entropy Loss*. MSE digunakan untuk tugas regresi, mengukur rata-rata kuadrat perbedaan antara prediksi dan nilai sebenarnya. *Cross-Entropy Loss* digunakan untuk tugas klasifikasi, mengukur perbedaan antara distribusi probabilitas prediksi dan distribusi probabilitas sebenarnya.

7. Optimizer

Algoritma untuk memperbarui bobot dan bias berdasarkan gradien loss. Optimazer sendiri memiliki beberapa jenis seperti *Stochastic Gradient Descent* (SGD), *Adam*, atau *RMSprop*.

8. Regularisasi

Teknik seperti *Dropout* atau *L2 Regularization* untuk menghindari *overfitting.*

Dengan memahami peran parameter, hyperparameter, dan variabel dalam JST, kita dapat mengoptimalkan kinerja model sesuai kebutuhan. Kombinasi yang tepat dari semua komponen ini membantu jaringan menghasilkan keluaran yang akurat dan mampu menggeneralisasi data baru.

2.7 Deep learning

Deep Learning adalah salah satu cabang dari Machine Learning yang memproses pembelajaran secara hierarkis. Pembelajaran ini melibatkan berbagai lapisan (layers), dimulai dari lapisan awal yang mengekstraksi fitur-fitur sederhana hingga lapisan akhir yang menghasilkan fitur-fitur yang lebih kompleks. Deep Learning digunakan dalam berbagai aplikasi seperti klasifikasi, klasterisasi, segmentasi, dan pengenalan (recognition). Teknologi ini sangat cocok untuk menangani data yang tidak terstruktur, seperti teks, audio, dan gambar [18].

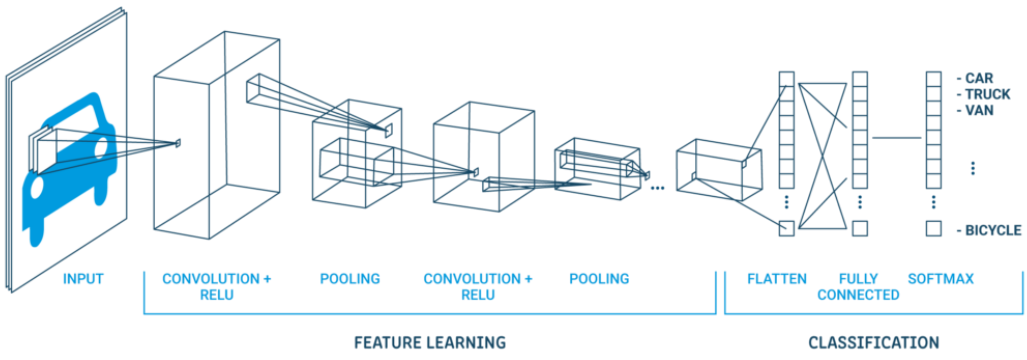
Dalam penelitian ini teknik *deep learning* dipilih karena data yang digunakan merupakan data tidak teratur yaitu berupa data citra. Teknik klasifikasi, *clustering*, *segmentasi* pada *deep learning* akan sangat berguna untuk menentukan pola penyakit diabetes retinopati pada citra. Model *Deep learning* yang paling umum untuk memproses data jenis ini adalah *Confolutinal Neural Netwok* (CNN) sehingga CNN dipilih untuk digunakan dalam penelitian ini.

2.8 *Convolutional Neural Network*

Convolutional Neural Network (CNN) adalah algoritma Deep Learning yang merupakan pengembangan dari *MultiLayer Perceptron (MLP),* dirancang khusus untuk memproses data dalam format dua dimensi seperti gambar atau audio. CNN

berfungsi untuk mengklasifikasi data berlabel dengan pendekatan *supervised learning.* Dalam *supervised learning,* model dilatih menggunakan data dengan label target tertentu, sehingga tujuannya adalah mengelompokkan data baru berdasarkan kategori yang sudah ada. Struktur CNN terdiri dari lapisan-lapisan neuron berbentuk tiga dimensi yang mencakup lebar, tinggi, dan kedalaman. Lebar dan tinggi biasanya memiliki ukuran yang sama, sedangkan kedalaman menunjukkan jumlah lapisan dalam jaringan. Sebagai contoh, dalam matriks RGB, nilai *channel* menentukan kedalaman *(depth),* yaitu tiga *channel* yang merepresentasikan warna merah, hijau, dan biru.

CNN merupakan model yang memiliki banyak lapisan bahkan hingga ratusan lapisan dengan tujuan mempelajari dan mendeteksi berbagai gambar. CNN juga menerapkan berbagai resolusi yang menerapkan pengolahan citra sendiri pada setiap citra latih, kemudian *output* dari masing-masing gambar tadi kemudian diolah untuk selanjutnya menjadi *Input* pada lapisan berikutnya. Pengolahan citra yang di terapkan pada fitur yang sangat sederhana seperti contohnya pada intensitas kecerahan dan tepi atau menaikan kompleksitas pada fitur yang secara unik menentukan objek sesuai ketebalan lapisan. Jaringan arsitektur *Convolutional Neural Network* dapat dilihat pada ilustrasi berikut [19]:



Gambar 2. 7 Arsitektur CNN [16]

Pada Gambar 2.7, ketika *Input* dimasukkan, arsitektur sistem terbagi menjadi dua bagian, yaitu *feature learning* dan *classification*. Pada *feature learning*, terdapat tiga lapisan proses untuk ekstraksi fitur, di mana lapisan-lapisan ini memproses data guna mengenali pola tertentu secara spesifik. Sementara itu, pada bagian

classification, model digunakan untuk mengklasifikasikan objek berdasarkan kategorinya..

2.7.1 Konvolusi

Konvolusi adalah istilah dalam matematika yang merujuk pada penerapan suatu fungsi terhadap *output* fungsi lain secara berulang. Dalam konteks pengolahan citra, konvolusi diterapkan dengan menggunakan kernel, yang merupakan matriks kecil berukuran lebih kecil dibandingkan matriks citra yang diproses. Kernel, yang juga disebut *Filter*, memainkan peran penting dalam mengekstraksi fitur spesifik dari gambar, seperti tepi, tekstur, atau pola tertentu[20].

Pada penerapannya di *machine learning*, *input* citra biasanya berbentuk array dua dimensi, sementara kernel adalah parameter yang juga direpresentasikan sebagai array multidimensi. Konvolusi tidak terbatas pada dua dimensi; teknik ini dapat diterapkan pada berbagai dimensi data sesuai dengan kebutuhan model. Misalnya, dalam kasus gambar dua dimensi sebagai *input*, kernel juga berbentuk dua dimensi untuk menghasilkan hasil konvolusi yang sesuai. Dengan cara ini, konvolusi memungkinkan model untuk menangkap dan mempelajari pola dalam data secara efektif. Jika diterapkan pada gambar dua dimensi, misalnya I sebagai *Input*, maka kernel K juga berbentuk dua dimensi, persamaannya dapat ditulis seperti pada persamaan 1 berikut :

$$S\,(i,j) = (I * K)(i,j) = \sum_{m}\sum_{n} I(m,n)K(i-m,j-n) \qquad (1)$$

Keterangan :

S (i,j) = Fungsi hasil konvolusi

I = *Input*

K = Kernel atau *Filter*

i,j = *Pixel Input*

m,n = *Pixel Filter*

Konvolusi juga memiliki sifat komutatif, sehinggga persamaan (1) dapat pula

ditulis sebagai berikut:

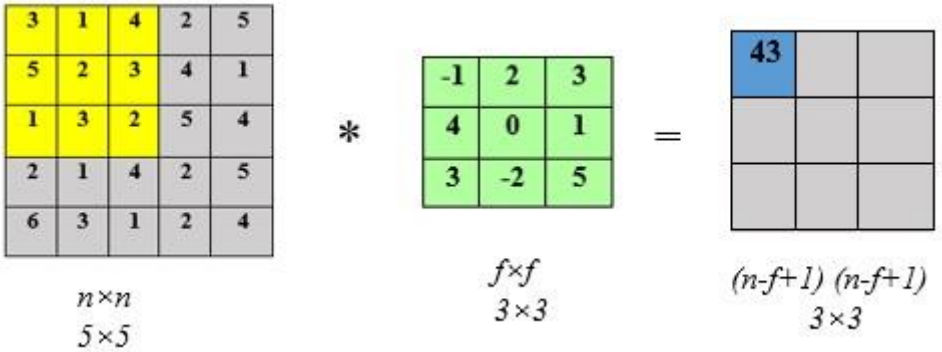
$$S\,(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K\,(m,n) \tag{2}$$

Sifat komutatif muncul setelah dibalik relatif terhadap *Input*. Sifat komutatif ini berguna untuk menulis bukti (*proof*), namun pada implementasinya dalam jaringan saraf tiruan sifat ini bukan merupakan hal yang esensial. *Cross correlation* merupakan fungsi yang mirip dengan sfiat komutatif seperti konvolusi namun tanpa membalik kernel, sehingga dapat dilakukan penerapan pada persamaan 2 menjadi :

$$S\,(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n)K\,(m,n) \tag{3}$$

Kebanyakan *library machine learning* yang menerapkan *cross correlation* namun menyebutnya konvolusi. Sehingga untuk menyamakan peresepsi, fungsi tersebut kali ini juga disebut konvolusi [21].

Konvolusi senidri pada data citra berperan dalam mengekstraksi fitur dari citra *Input*. Pada gambar 2.8 diilustrasikan sebuah matriks *Input* berukuran 5 ×5 dikonvolusikan dengan kernel/*Filter* berukuran 3 × 3, dengan *stride* (parameter yang menentukan berapa jumlah pergeseran *Filter*) sebanyak 1, dan *zero padding*. Hasil dari konvolusi tersebut ditunjukkan pada matriks *result* atau biasa disebut sebagai *Activation map* atau *Activation Map*[22].



cara hitung *result* =
(3*-1) + (1*2) + (4*3) +
(5*4) + (2*0) + (3*1) +

(1*3) + (3*-2) + (2*5) = 43

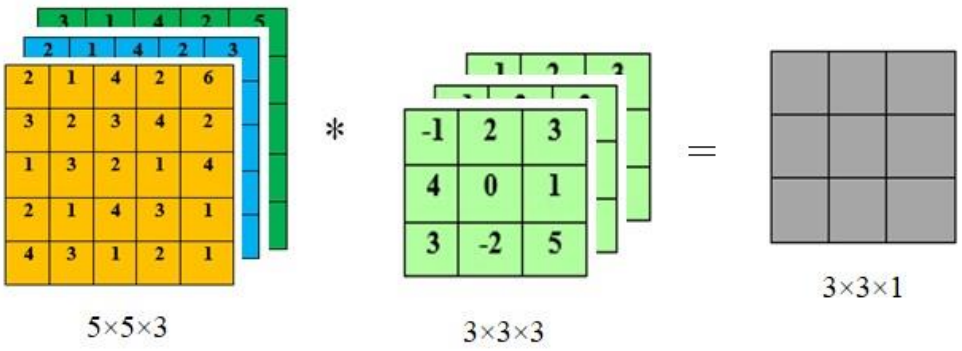
Gambar 2. 8. Proses konvolusi [22]

Gambar 2.8 menunjukkan konvolusi pada citra yang hanya memiliki satu *channel*. Namun, ketika konvolusi dilakukan pada citra RGB dengan 3 *channel*, jumlah *channel* pada *Filter* juga harus disesuaikan, yaitu sama dengan jumlah *channel* pada citra *input*. Dengan kata lain, dalam proses konvolusi, jumlah *channel* pada *input* harus identik dengan jumlah *channel* pada kernel. Jika citra *input* memiliki dimensi $n \times n \times n_c$ dengan n_c adalah jumlah *channel* atau lapisan pada citra *input*, maka persamaan (3) dapat dinyatakan dengan menyesuaikan parameter tersebut. Seperti berikut :

$$S\,(i,j) = (I * K)(i,j) = \sum_{c}^{nc} \sum_{m} \sum_{n} I(i+m, j+n, c) K\,(m,n,c) \qquad (4)$$

dengan c = 1….n_c

Pada konvolusi pada *Input* yang memiliki 3 *channel*, hasil dari konvolusi ketiga *channel* ini hanya akan menghasilkan satu *channel* matriks saja. Masing-masing *channel* matriks akan dikonvolusikan dengan *Filter*. Kemudian hasil konvolusi dari masing-masing *channel* dijumlahkan untuk mendapatkan hasil dari matriks konvolusi atau *result*. Pada gambar 2.9 diilustrasikan proses konvolusi 3 *channel.*



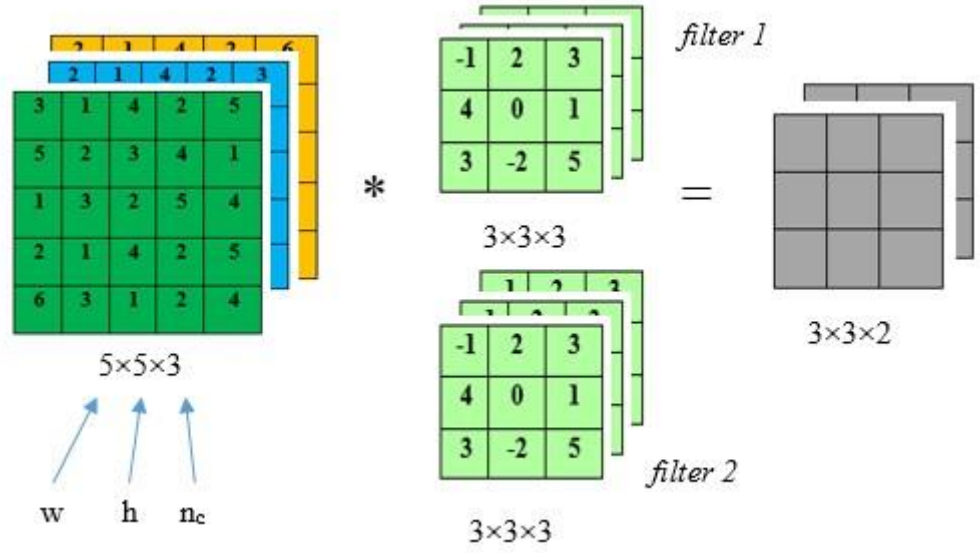
Gambar 2. 9. Contoh operasi konvolusi matriks citra yang memiliki 3 *channel*[22]

Operasi konvolusi tidak hanya menggunakan 1 *Filter* khususnya dalam CNN yang dijalankan dengan lebih dari 1 *Filter*. Operasi jenis ini dinamakan *multiple* kernel. Prinsip kerjanya masih sama, namun hasil dari konvolusi *multiple* kernel memiliki

jumlah *channel* yang sama dengan jumlah *Filter*nya. Jika nf adalah jumlah *Filter* yang digunakan, dan k = 1…nf maka persamaan (4) akan menjadi seperti berikut :

$$S\,(i,j) = (I * Kk)(i,j) = \sum_{c}^{nc} \sum_{m} \sum_{n} I(i+m, j+n, c) K\,k(m,n,c) \quad (5)$$

Jika memiliki lebih banyak matriks yang memiliki 3 *channel*, misalkan pada gambar 2.10 memiliki matriks *Input* berukuran 5 × 5 dengan *channel* sebanyak 3, dikonvolusikan dengan 2 *Filter* yang masing-masing memiliki *channel* sebanyak 3. Sehingga hasil konvolusi memiliki matriks dengan ukuran 3 × 3 dengan *channel* sebanyak 2 [21].

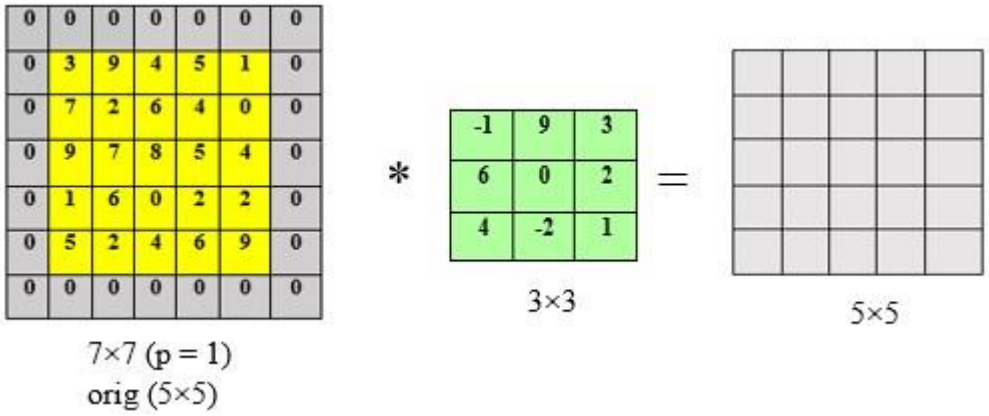


Gambar 2. 10. Contoh operasi konvolusi *multiple* kernel [21]

2.7.2 Padding

Padding atau *Zero padding* merupakan tindakan memasukan jumlah *pixel* (berisi nilai 0) yang akan ditambahkan ke setiap sisi *Input* gunakan untuk memanipulasi dimensi *output* lapisan konvolusi (*Activation map*). Lapisan konvolusi akan selalu lebih kecil dari *Input* (kecuali penggunaan *Filter* 1 × 1 *stride* 1) sehingga pada prosesnya banyak informasi terbuang yang tidak diperlukan saat proses konvolusi sedang berjalan [23]. Selain itu, pada *padding* nol dimensi lapisan keluaran diatur

agar tetap sama dengan dimensi masukan atau setidaknya tidak berkurang drastis.

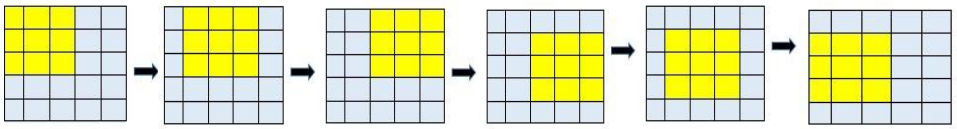


Gambar 2.11.Penambahan *padding* pada *Input* [23]

Pada gambar 2.11 diilustrasikan sebuah matriks *Input* dengan ukuran asli sebesar 5 × 5 dengan *Filter* sebesar 3 × 3. *Input* ini diberi *padding* sebanyak satu sehingga membuat ukuran matriks *Input* berubah ke 7 × 7. Matriks yang telah diberi *padding* lalu dikonvolusikan dengan *Filter* yang menghasilkan *Activation map* dengan ukuran yang sama dengan ukuran *Input* aslinya yaitu 5 × 5.

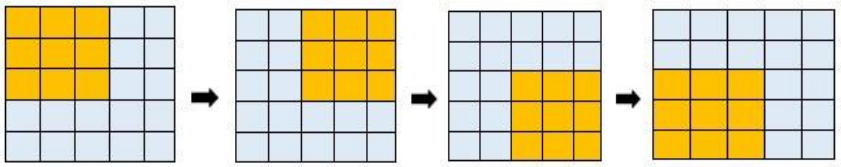
2.7.3 *Stride*

Stride merupakan parameter yang menentukan jumlah pergeseran *Filter* dalam *pixel* gambar. Jika *Stride* nilainya 1, *Filter* konvolusi akan bergeser 1 *pixel* secara *horizontal* dan *vertikal*. Lebih kecil nilai *stride*, model akan menangkap informasi yang lebih rinci dari gambar *Input*, tetapi membutuhkan lebih banyak perhitungan jika dibandingkan dengan langkah besar. Nilai *Stride* kecil tidak selalu menghasilkan detail informasi *pixel* yang lebih baik, tetapi dengan nilai langkah kecil mencegah penumpukan informasi *pixel* yang tidak digunakan. Sebaliknya, jika stride ditetapkan dengan nilai yang lebih besar, seperti 2 atau 3, kernel akan bergerak lebih jauh di setiap langkah. Hal ini dapat mengurangi jumlah komputasi yang diperlukan, tetapi model akan kehilangan beberapa detail kecil dari citra. Oleh karena itu, pemilihan nilai stride menjadi keseimbangan antara efisiensi komputasi dan tingkat detail yang ingin dipertahankan.



Gambar 2. 12. *Stride* 1 yang bergeser secara *horizontal* [7]

Pada gambar 2.12 diilustrasikan sebuah *Filter* yang diaplikasikan pada *Input*. Pergeseran *Filter* ditentukan oleh besarnya ukuran *stride*. Nilai *Stride* pada gambar 2.10 adalah 1, maka pergeseran kernel pada *Input* hanya sebanyak 1 *pixel* saja per tahapannya. Sedangkan untuk *Stride* 2 illusntarsinya dapat dilihat pada gambar 2.11 dibawh ini.



Gambar 2.13. *Stride* 2 yang bergeser secara *horizontal* [7]

Dalam gambar 2.13 pergeseran matrik *Input* pertama dan kedua adalah sesuai dengan nilai *stride* yaitu 2 atau secara umum ditentukan oleh nilai parameter *stride* dalam tiap lapisan. Proses ini diulang untuk konvolusi selanjutnya, sehingga menghasilkan matrik dengan ukuran yang lebih kecil karena pergeseran tadi.

2.7.4 *Batch Normalization*

Batch normalization adalah teknik yang digunakan untuk meningkatkan efisiensi pelatihan model dan stabilitas pembelajaran. Teknik ini bekerja dengan menormalkan nilai *input* di setiap lapisan atau dapat dibilang menyamakan/menyusun nilai , sehingga distribusinya menjadi lebih rata dan konsisten meskipun parameter pada lapisan sebelumnya terus berubah selama proses pelatihan. Dengan demikian, *batch normaliza*tion membantu model belajar lebih cepat dan efisien. [24]. Operasi *Batch Normalization* dirumuskan pada persamaan (5), (6), (7), dan (8).

- Hitung *mini bacth mean* dengan persamaan:

$$\mu_{\mathcal{B}} = \frac{1}{m}\sum_{i=1}^{m} x_i$$ ……..….(5)

- Hitung *Batch varian* dengan persamaan

$$\sigma_{\mathcal{B}}^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2$$ ……(6)

- Hitung normalisasi dengan persamaan:

$$\hat{x_i} = \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$$ ……...…….…..….(7)

- Hitung *scale and shift* dengan persamaan:

$yi \leftarrow \gamma x\hat{i} + \beta = BN\gamma,\beta(xi)$ ……(8)

Keterangan:

X_i = *Input* data berupa tensor (array multi dimensi dari porses konvolusi *Input*)

μB = merupakan nilai rata-rata dari *Batch*

σ = standar deviasi dari *Input* data yang dihitung dari *Batch Input* data yang diberikan pada setiap langkah *forward pass*

m = jumlah dataset pelatihan

ϵ = merupakan nilai konstan yang meningkatkan stabilitas numerik ketika *Batch* variance sangat kecil , Nilai ϵ\epsilonϵ yang sering digunakan adalah 10^{-5}atau 10^{-6} untuk nilai yang mendekati nol

γ dan β = parameter pada saat pelatihan ,

γ = parameter skala (1) ,dan

β = 0parameter pergeseran (0).

Selain beberapa rumus diatas terdapat teknik normalisasi lain yang digunakan dalam penelitian ini, yaitu Z-Score Normalisasi (Standarisasi). Z-score normalisasi adalah teknik statistik yang digunakan untuk mengubah data mentah menjadi skala yang sama, sehingga data memiliki rata-rata 0 dan simpangan baku 1 (nilai dapat

berubah tergantung nilai parameter yang digunakan). Proses ini sering disebut dengan standarisasi. Tujuan utama dari normalisasi ini adalah untuk membuat data lebih mudah dibandingkan dan dianalisis, terutama ketika data berasal dari distribusi yang berbeda-beda. Dengan mentransformasikan data ke dalam skala yang sama dapat meningkatkan kualitas model dan memperoleh hasil yang lebih baik. Jika suatu fitur memiliki rentang nilai yang jauh lebih besar daripada fitur lainnya, fitur tersebut dapat mendominasi proses pembelajaran. Normalisasi membantu mengurangi efek dominasi ini. Rumus Z-Score Normalisasi dapat dilihat dibawah ini [25].

$$Z = (X - \mu) / \sigma$$

Keterangan:

Z = Nilai yang telah dinormalisasi (z-score)

X = Nilai asli dari data

M = Rata-rata (mean) dari seluruh data

σ = Simpangan baku (standar deviasi) dari seluruh data

2.7.5 Fungsi Aktivasi

Fungsi aktivasi dihitung setelah operasi konvolusi (jika tidak menggunakan *Batch normalization*). Fungsi ini menambahkan elemen *non-linearitas* pada model, memungkinkan jaringan saraf mempelajari hubungan kompleks dalam data. Beberapa fungsi aktivasi yang umum digunakan meliputi *ReLU (Rectified Linear Unit), sigmoid, tanh,* dan *softmax. Softmax* merupakan fungsi aktivasi yang digunakan pada *Layer output*. Pada *Layer output* terdapat banyak kesamaan dengan *fully-connected Layer* , kedua *Layer* ini dibedakan dengan penggunaan fungsi aktivasi *softmax* pada *Layer output* dan fungsi aktivasi *ReLU* pada *fully-connected Layer* [26]

Bentuk fungsi *ReLU* :

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} \quad (9)$$

Fungsi *ReLU* bekerja dengan menghasilkan *output* sebesar nilai *input* (x) jika nilai *input* tersebut lebih besar atau sama dengan 0. Artinya, jika nilai aktivasi *input* adalah 0 atau positif, *output* neuron akan sama dengan nilai *input* tersebut. Sebaliknya, jika nilai *input* kurang dari 0 atau bernilai negatif, *output* neuron akan menjadi 0.

Sementara itu, fungsi *softmax* bertugas menghitung probabilitas untuk setiap kelas target dari semua kelas yang mungkin. Fungsi ini membantu model menentukan kelas target yang paling sesuai berdasarkan *input* yang diberikan, dengan memastikan bahwa total probabilitas untuk seluruh kelas adalah 1 [22].

Bentuk fungsi *Softmax* :

$$S\ (x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \tag{10}$$

Keterangan :

S = softmax

e = basis dari logaritma natural (sekitar 2.718)

x_i = *Input* elemen ke -i

n = jumlah kelas dalam *classifier*

i = Indeks elemen yang probabilitasnya sedang dihitung

j = Indeks elemen yang digunakan dalam penjumlah untuk normalisasi

Berdasarkan fungsi dari *softmax*, dapat diketahui bahwa setiap probabilitas dalam hasil berada dalam rentang 0...1, dan jumlah probabilitasnya adalah 1.

2.7.6 *Cross Entropy*

Cross entropy merupakan fungsi untuk kerugian dan gradien digunakan dalam *multi-classification.* Untuk mengukur *entropi* relatif antara dua distribusi probabilitas pada data yang sama. *Cross entropy* fungsi kerugian yang digunakan ketika melatih model *neural network*. Cara kerjanya mengurangi log negatif dari dataset. *Cross entropy* terdapat persamaan (11). [27]

$$L_{cross-entropy} = -\sum_{j} y_i \log(\hat{y}) \quad \text{.....(11)}$$

Keterangan:

$\hat{y}$ = hasil dari *softmax* activation

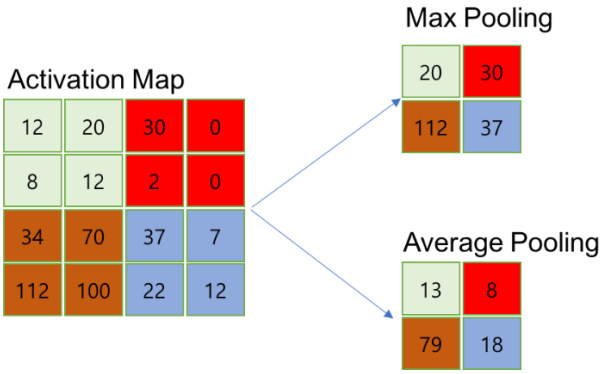
y_i = nilai dari kelas, bernilai 1 jika kelas yang benar dan bernilai 0 jika kelas yang salah

j = kelas ke-j(1,2)

2.7.7 *Pooling Layer*

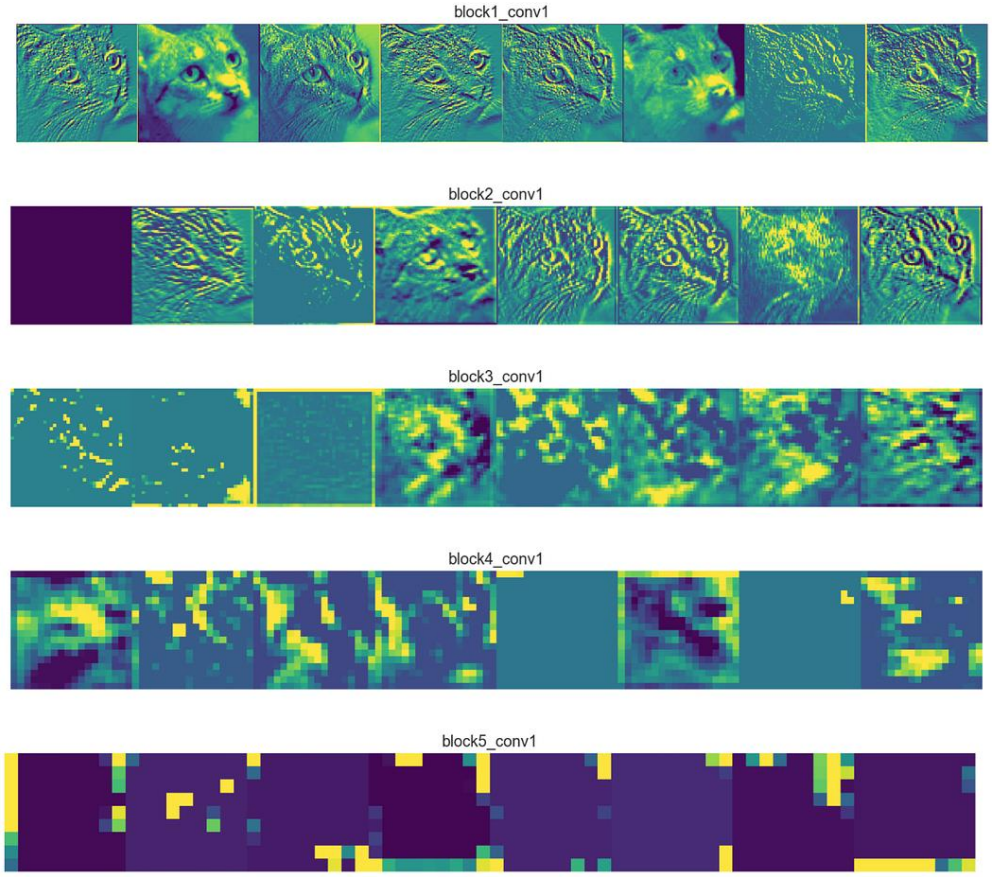
Pooling, pada dasarnya, mirip dengan proses pengambilan sampel data dalam statistik. Namun, perbedaannya terletak pada *pooling* yang tidak menyebabkan hilangnya informasi. Tujuan utama *pooling* adalah memperkecil ukuran matriks, sehingga proses komputasi menjadi lebih cepat dan efisien. *Pooling* layer berperan dalam mengurangi dimensi spasial dari fitur hasil konvolusi, atau dikenal sebagai *downsampling*. Dengan mengurangi ukuran *activation map*, *pooling* mengurangi kebutuhan sumber daya komputasi dan jumlah parameter yang perlu diperbarui selama pelatihan, sehingga mempercepat proses pelatihan model.

Selain itu, *pooling* membantu mengekstraksi fitur yang paling menonjol dari data, membuat model lebih efektif dalam memahami pola-pola penting. Ada beberapa jenis *pooling* yang digunakan, tetapi dua jenis yang paling umum adalah *Max pooling* dan average *pooling*. *Max pooling* memilih nilai maksimum dari area gambar yang diliputi oleh kernel, sementara average *pooling* menghitung rata-rata nilai dalam area tersebut. Umumnya, *pooling* layer menggunakan *Filter* berukuran 2 × 2 yang diterapkan dengan langkah (*stride*) sebesar 2, yang kemudian diaplikasikan pada setiap bagian *input*. Contoh penggunaan *Max pooling* dan average *pooling* dapat dilihat pada Gambar 2.14.



Gambar 2. 14. Contoh operasi *Max pooling* dan *average pooling*[28]

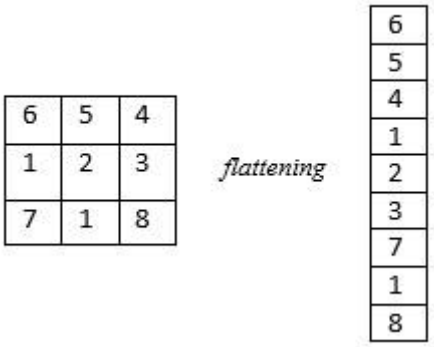
Pooling layer merupakan *layer* akhir pada *feature learning* sebelum memasuki *flatten*. Layer yang bertumpuk mengakibatkan citra *Input* mengalami Abstraksi Fitur. Semakin dalam lapisan konvolusi, fitur yang dideteksi semakin abstrak. Fitur-fitur ini mungkin tidak lagi memiliki korelasi langsung dengan piksel-piksel pada gambar *Input* asli. Selain itu, dikarenakan setiap layer konvolusi memiliki dimensi yang berbeda, tergantung pada ukuran *Filter*, stride, dan padding, membuat sulit untuk menyusun kembali semua layer konvolusi menjadi sebuah gambar dengan dimensi yang sama dengan gambar *Input*.



Gambar 2. 15. Hasil Feature Learning[29]

2.7.8 *Flatten*

Flatten atau pemampatan adalah proses pengubahan matriks menjadi vektor satu dimensi. Proses *flatten* bertujuan mengubah *Activation map* yang telah diperoleh dari *Layer* sebelumnya menjadi vektor satu dimensi agar dapat diklasifikasikan dengan *fully-connected Layer* dan *softmax* karena keduanya hanya menerima masukan berupa vektor 1 dimensi[26]. Proses *flatten* diilustrasikan pada Gambar 2.16.



Gambar 2. 16. Proses *flattening* [26]

Backpropagation Layer merupakan lapisan dimana vektor satu dimensi diubah kembali menjadi matriks dengan dimensi seperti semula untuk selanjutnya dapat dilakukan proses perubahan bobot *Filter*.

2.7.9 *Fully connected Layer*

Fully Connected Layer adalah lapisan yang sering digunakan dalam arsitektur MLP *(Multilayer Perceptron)* untuk mentransformasikan dimensi data sehingga dapat diklasifikasikan secara linear. Disebut *fully connected* karena setiap neuron pada lapisan ini terhubung dengan semua neuron di lapisan sebelumnya maupun berikutnya. Lapisan ini berfungsi sebagai komponen klasifikasi dalam jaringan, yang pada dasarnya merupakan implementasi metode *feed-forward neural network.* Sebelum data dari lapisan konvolusi dimasukkan ke *Fully Connected Layer,* data tersebut harus diubah menjadi vektor satu dimensi. Proses ini, meskipun penting, menyebabkan hilangnya informasi spasial pada data, sehingga transformasi ini tidak dapat dibalik. Oleh karena itu, *Fully Connected Layer* biasanya hanya digunakan di bagian akhir jaringan. Sebagai alternatif, lapisan konvolusi dengan kernel berukuran 1 × 1 dapat melakukan fungsi serupa dengan *Fully Connected Layer* sambil tetap mempertahankan informasi spasial, yang membuat penggunaan *Fully Connected Layer* dalam *CNN* modern menjadi kurang umum[30].

.

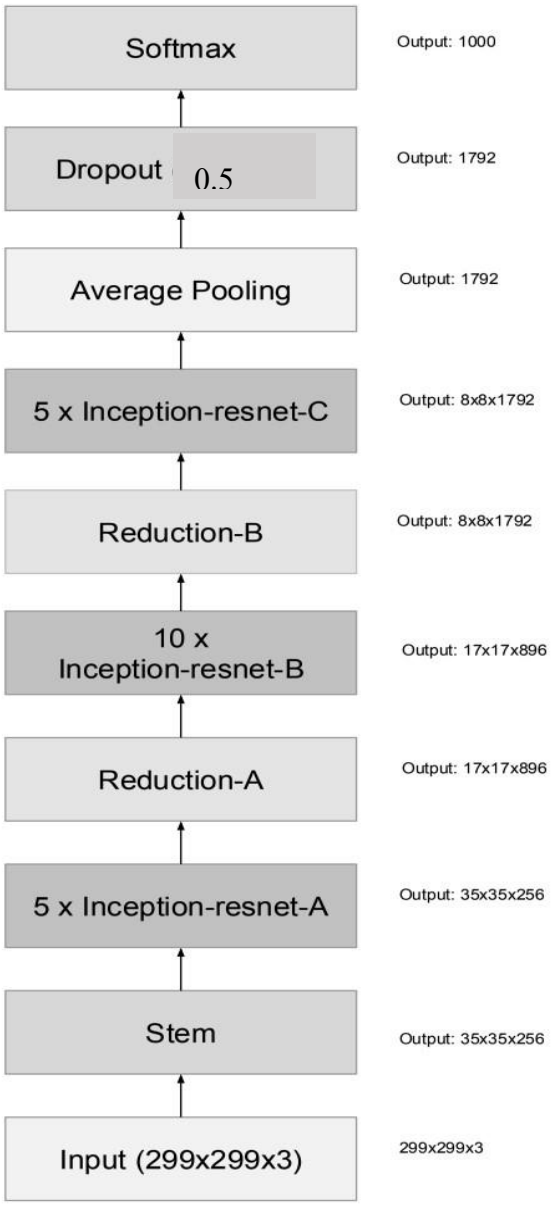
Pada penelitian ini, arsitektur CNN yang akan digunakan adalah *InceptionResNet-V2.* Beberapa penyesuaian akan dilakukan pada arsitektur ini untuk mendapatkan hasil yang optimal sesuai dengan kebutuhan penelitian.

2.9 *InceptionResnetV2*

Jaringan *AlexNet* telah diterapkan pada berbagai tugas visi komputer, seperti deteksi objek, segmentasi, pendeteksian gerakan manusia, klasifikasi video, pelacakan objek, dan super resolusi sejak memenangkan *kompetisi ImageNet 2012 oleh Krizhevsky et al*. [30]. Sebagai pengembangan lebih lanjut *dari Convolutional Neural Network* (CNN), Szegedy pada tahun 2014 memperkenalkan arsitektur *Inception* melalui makalahnya yang berjudul *"Going Deeper with Convolutions"* [31]. Dalam arsitektur ini, *convolution* dilakukan untuk mengekstraksi fitur dari gambar melalui proses *Filterisasi* menggunakan kernel matriks, yang bergeser dengan nilai *stride* tertentu pada gambar masukan. Hasil dari proses *convolution* ini kemudian diteruskan ke bagian *fully connected* untuk proses klasifikasi [30]. Tujuan utamanya adalah menciptakan jaringan yang lebih andal dan meningkatkan akurasi klasifikasi melalui penggunaan arsitektur *Inception-ResNet*.

Pada tahun 2017, Szegedy mempelajari kombinasi koneksi *residual* dengan versi terbaru arsitektur *Inception*. Dalam penelitian ini, dilakukan perbandingan berbagai varian *Inception,* seperti *Inception-V3, Inception-V4*, dan *versi hybrid Inception-ResNet*.[31] Kendala utama yang dihadapi adalah memastikan parameter dan kompleksitas komputasi model tetap serupa dengan model *non-residual*. Hasilnya menunjukkan bahwa *Inception-ResNet-V2* memiliki kinerja yang sangat baik, bahkan melampaui model *single-frame* tercanggih pada dataset validasi *ImageNet*.

Penelitian tersebut hanya membahas dua versi *residual* dari arsitektur *Inception*. Versi pertama, yaitu *Inception-ResNet-V1*, memiliki biaya komputasi yang hampir setara dengan *Inception-V3*. Sementara itu, *Inception-ResNet-V2* memiliki tingkat kompleksitas yang setara dengan *Inception*-V4. Lihat Gambar 2.17 untuk skema arsitektur *Inception-ResNet-V2*.



Gambar 2. 17. Skema untuk arsitektur *Inception ResNet-V2*[31]

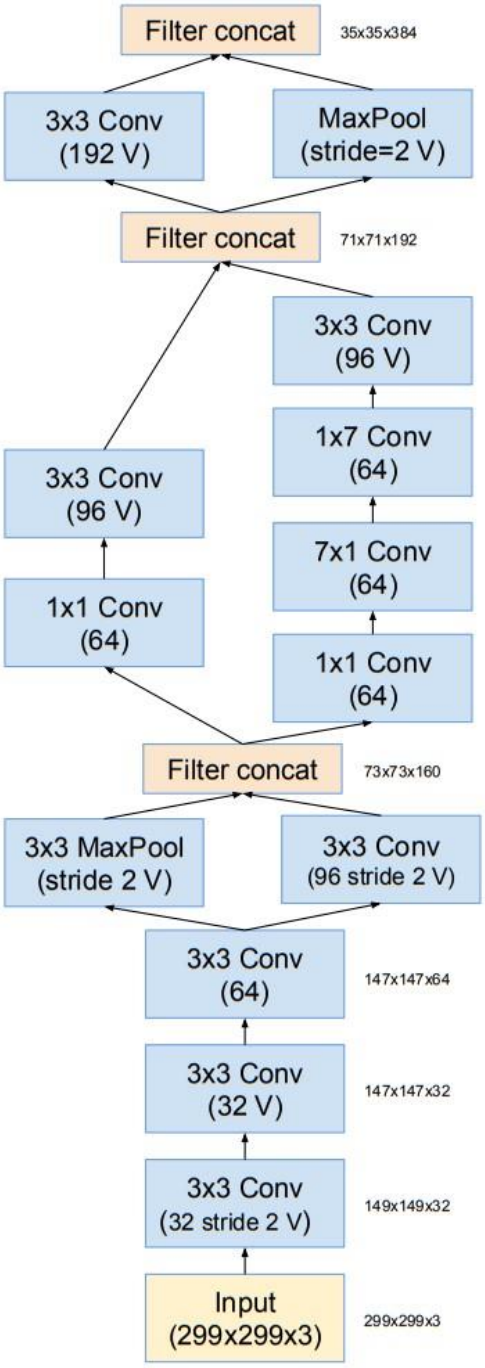
Arsitektur rinci dari *Inception ResNet-V2* yang dikelompokkan berdasarkan *layer* dapat dilihat pada tabel 2.2. Jumlah *channel Input* pada masing-masing *layer* ditentukan oleh jumlah kernel pada *layer* sebelumnya yang sudah ditentukan oleh arsitektur *Inception ResNet-V2* sendiri. Pada *Input* awal citra ukuran 299x299 *pixel* dipilih karena desain arsitektur, efisiensi komputasi, dan performa optimal pada tugas-tugas klasifikasi gambar. Ukuran ini memberikan cukup ruang bagi model untuk mengekstrak fitur yang lebih detail, terutama pada data yang lebih kompleks. Untuk total layer yang ada dalam *InceptionResnetV2* sendiri ada sangat banyak

diperkirakan ada sekitar 300-350 layer yang terlibat dalam datu kali batch pelatihan model. Lebih lengkapnya tahapan dapat dilihat pada tabel 2.2.

Tabel 2. 2 Arsitektur *Inception ResNet-V2*

Type	***Kernel size/stride Or remarks***	***Input Size***
Stem		299×299×3
5× Inception-resnet A		35×35×384
Reduction-A		35×35×384
10× Inception-resnet-B		17×17×1152
Reduction-B		17×17×1152
5× Inception-resnet-C		8×8×2144
Average Pooling	8×8	8×8×2144
Dropout (0,5)		1×1×2144
Softmax		1×1×2144

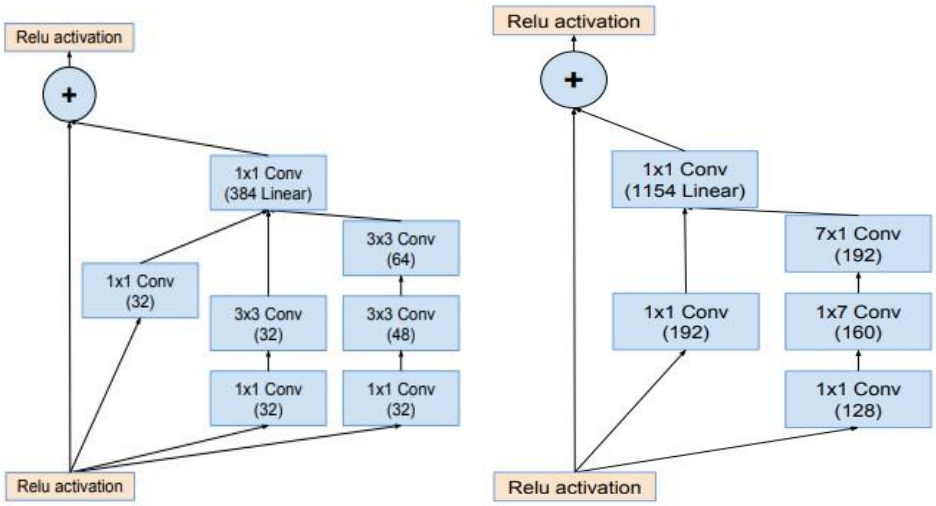
Penerapan arsitektur model pada tabel 2.2 dapat dilihat pada Gambar 2.16 yang menunjukkan arsitektur lengkap dari *Inception ResNet-V2.* Model ini dimulai dengan mamasukan *Input* ke *stem*. *Stem*, dalam hal ini mengacu pada operasi *initial set* yang dilakukan sebelum memperkenalkan blok-blok *Inception ResNet-V2*. Untuk lebih lengkapnya tahapan stem ada pada gambar 2.18.



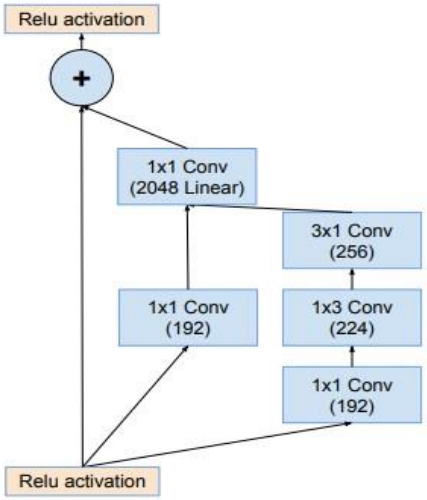
Gambar 2. 18. Proses *stem Inception ResNet-V2*[31]

Pada gambar 2.18, *convolution* yang ditandai dengan "V" menggunakan *valid padding* secara *default* karena lebih efisien tanpa adanya elemen tambahan yang harus dihitung, sehingga memungkinkan komputasi yang lebih cepat sehingga dimensi *output* akan sama dengan dimensi *Input*. Sebaliknya, *convolution* yang tidak ditandai dengan "V" menggunakan *same padding*.

Pada *Inception ResNet-V2* konvolusi dibagi menjadi 3 modul, yaitu modul A, B, dan C yang terdapat pada gambar 2.19 (modul A dan B) dan 2.20. modul-modul ini terlihat mirip dengan *Inception ResNet-V1*.

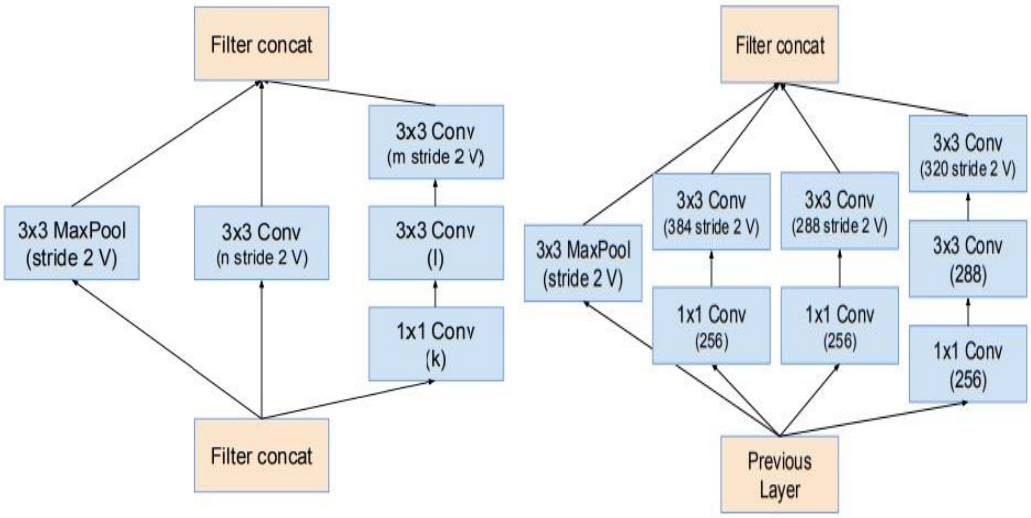


Gambar 2. 19. Modul A dan Modul B *Inception Resnet-V2*[31]



Gambar 2. 20. Modul C *Inception Resnet-V2*[31]

Pada *Inception ResNet-V2* diperkenalkan *reduction blocks* khusus yang gunanya untuk mengubah lebar dan tinggi *grid*. Versi sebelumnya (*Inception ResNet)* tidak secara eksplisit memiliki *reduction blocks*, tetapi fungsionalitas ini juga diterapkan. Sturktur *Reduction blocks* bisa dilihat pada gambar 2.21.



Gambar 2. 21. *Reduction block* A dan *Reduction block* B *Inception Resnet-V2* [31]

2.10 *Filter concatenation*

Filter concatenation adalah teknik yang digunakan dalam arsitektur jaringan konvolusional (CNN), terutama dalam model-model seperti *Inception*, untuk menggabungkan hasil dari beberapa *Filter* konvolusi yang berbeda dalam satu layer. Pada dasarnya, *Filter-Filter* ini bekerja secara paralel untuk menangkap berbagai jenis informasi spasial dalam citra, dan hasilnya digabungkan (*concatenate*) dalam dimensi *channel*. Misalnya, dalam satu blok *Inception*, satu jalur konvolusi mungkin menggunakan *Filter* 1x1, jalur lain menggunakan *Filter* 3x3, dan jalur lainnya menggunakan *Filter* 5x5. Hasil dari masing-masing jalur ini kemudian digabungkan untuk membentuk tensor *output* yang lebih kaya, menggabungkan berbagai informasi fitur dari ukuran *Filter* yang berbeda. Teknik ini memungkinkan model untuk menangkap berbagai skala spasial dalam citra secara simultan tanpa meningkatkan terlalu banyak jumlah parameter, karena penggunaan *Filter* kecil (seperti 1x1) memungkinkan komputasi yang lebih efisien.[32] *Filter concatenation* membantu meningkatkan kapasitas representasi model, memungkinkan model belajar fitur yang lebih beragam dan lebih kompleks, yang dapat meningkatkan akurasi model dalam tugas seperti klasifikasi citra, deteksi objek, dan segmentasi.

Setelah konvolusi dilakukan pada setiap jalur dan dimensi spasial (H dan W) *outputnya* disesuaikan, *output* dari setiap jalur digabungkan sepanjang dimensi *channel*. Dengan kata lain, tensor yang dihasilkan oleh masing-masing jalur dengan dimensi (H x W x C_i) dimana i adalah nomor jalur digabungkan menjadi tensor *output* akhir dengan dimensi H x W x($C_1 + C_2 + C_3$) Secara matematis, ini dapat ditulis sebagai:

$$\text{Concatenated Output} = [\text{Output}_1, \text{Output}_2, \text{Output}_3]$$

Di mana:

- *Output*$_1$ adalah hasil konvolusi dengan *Filter* 1x1 dengan dimensi H x W x C_1

- *Output*$_2$ adalah hasil konvolusi dengan *Filter* 3x3 dengan dimensi H x W x C_2

- *Output*$_3$ adalah hasil konvolusi dengan *Filter* 5x5 dengan dimensi H x W x C_3

Setelah *concatenation, output* akhir akan memiliki dimensi H x W x ($C_1 + C_2 + C_3$) Sebagai contoh, jika jalur pertama dengan *Filter* 1x1 menghasilkan H x W x 64 jalur kedua dengan *Filter* 3x3 menghasilkan H x W x 128 dan jalur ketiga dengan *Filter* 5x5 menghasilkan H x W x 32 . Maka *output* akhir setelah *concatenation* akan memiliki dimensi H x W x 224. Keuntungan utama dari *Filter* concatenation adalah kemampuan model untuk menangkap berbagai fitur spasial dengan berbagai ukuran *Filter*, yang meningkatkan kapasitas representasi dan meningkatkan akurasi model dalam tugas-tugas seperti klasifikasi citra dan deteksi objek.

2.11 *Adam Optimizer*

Adam optimization diperkenalkan oleh Diederik Kingma dari OpenAI dan Jimmy Ba dari University of Toronto pada *paper* ICLR tahun 2015 berjudul "Adam: A Method for Stochastic Optimization"[33]. Adam merupakan singkatan dari *Adaptive Moment Estimation* [33]. *Adam*, yang merupakan singkatan dari *Adaptive Moment Estimation* [31], adalah algoritma optimisasi yang dirancang untuk menggantikan metode *stochastic gradient descent* klasik. Algoritma ini

memperbarui bobot jaringan secara iteratif tanpa memerlukan perubahan pada nilai *learning rate*.

2.12 Kernel *Filter* dalam *InceptionResnetV2*

Pada arsitektur *Inception*-ResNet-v2, *Filter* yang digunakan memiliki berbagai ukuran, seperti 1x1, 3x3, dan 5x5. *Filter* ini digunakan untuk mengekstraksi fitur dari *input* citra dengan berbagai tingkat kompleksitas dan skala. *Filter* ini bisa dianggap sebagai "kernel" yang menggeser di atas gambar atau fitur untuk menghitung nilai output.

1. *Filter* 1x1 (*bottleneck*):
    - *Filter* 1x1 adalah *Filter* dengan ukuran terkecil, yaitu hanya satu piksel di setiap dimensi ruangnya (lebar dan tinggi). Fungsi utama dari *Filter* 1x1 adalah untuk mereduksi jumlah saluran (*channel*) atau fitur dalam suatu layer tanpa mengubah ukuran spasial (lebar dan tinggi) gambar atau tensor. Ini digunakan untuk mengurangi dimensi dalam komputasi yang lebih efisien dan menghindari overfitting.
    - Secara teknis, *Filter* 1x1 bekerja dengan cara mengambil linear combination dari *channel input* tanpa mengubah dimensi spasial (H x W). Misalnya, dalam *Inception*, *Filter* 1x1 digunakan untuk mengurangi jumlah *channel* sebelum melakukan operasi konvolusi yang lebih besar (seperti 3x3 atau 5x5). Ini memungkinkan model untuk mengurangi jumlah parameter dan komputasi, membuatnya lebih efisien.
2. *Filter* 3x3
    - *Filter* 3x3 adalah *Filter* yang memiliki ukuran 3x3 piksel, yang berarti bahwa setiap elemen *Filter* 3x3 akan mengoperasikan pada grid 3x3 piksel dari gambar *input*. *Filter* ini digunakan untuk menangkap informasi spasial yang lebih luas dibandingkan dengan *Filter* 1x1. *Filter* 3x3 sering digunakan dalam banyak arsitektur CNN untuk

mengekstraksi fitur tingkat menengah, seperti tekstur dan pola yang lebih besar.

- Keuntungan dari *Filter* 3x3 adalah kemampuannya untuk menangkap pola yang lebih besar dalam citra, namun tetap menjaga sejumlah besar parameter dan komputasi yang dapat dikelola. *Filter* ini memungkinkan model untuk mengenali pola yang lebih kompleks, seperti sudut atau garis yang lebih besar dalam citra.

3. *Filter* 5x5:

- *Filter* 5x5 adalah *Filter* yang memiliki ukuran 5x5 piksel, dan biasanya digunakan untuk menangkap fitur dengan skala yang lebih besar lagi, seperti pola tekstur yang lebih besar atau objek dalam citra. *Filter* ini memiliki lebih banyak parameter dibandingkan dengan *Filter* 3x3, tetapi juga dapat menangkap informasi spasial yang lebih kompleks.
- Namun, karena *Filter* ini lebih besar, ia membutuhkan lebih banyak komputasi dan lebih banyak parameter, yang bisa menjadi mahal secara komputasi. Untuk itu, biasanya *Filter* 5x5 dipasangkan dengan pengurangan dimensi (misalnya, dengan *Filter* 1x1 sebelumnya) untuk mengurangi jumlah *channel*, yang membantu efisiensi komputasi.

Pada *Inception*-ResNet-v2, *Filter-Filter* ini digunakan dalam berbagai jalur *(pathways)* yang paralel untuk menangkap berbagai informasi spasial pada skala yang berbeda. *Output* dari jalur-jalur ini kemudian digabungkan (*concatenated*) untuk membentuk *output* yang lebih kaya dari jaringan.

Setiap elemen dalam *Filter* adalah angka yang mengontrol bagaimana *Filter* beroperasi pada gambar *input* saat proses konvolusi. Nilai-nilai dalam *Filter* ini biasanya diinisialisasi secara acak pada awal pelatihan dan kemudian disesuaikan selama proses pelatihan menggunakan algoritma optimisasi seperti *gradient descen*t.

Untuk *Inception-ResNet-v2,* secara umum nilai-nilai dalam *Filter* akan bergantung pada cara *Filter* tersebut diinisialisasi dan dipelajari selama pelatihan dengan menggunakan metode tertentu, seperti *Xavier Initialization* (juga dikenal sebagai

Glorot *initialization*) atau He *initialization*. Ini memastikan bahwa nilai *Filter* di awal pelatihan tidak terlalu besar atau kecil, sehingga membantu pelatihan menjadi lebih stabil. Setelah itu, selama pelatihan, nilai-nilai dalam *Filter* diupdate melalui backpropagation berdasarkan gradien error yang dihitung.

Jenis Inisialisasi yang Sering Digunakan:

- *Xavier Initialization*: Menyesuaikan nilai *Filter* agar distribusinya memiliki *varians* yang seimbang antara layer-layer dalam jaringan. Ini penting untuk menghindari gradien yang terlalu kecil atau terlalu besar.
- *He Initialization*: Lebih fokus pada inisialisasi *Filter* untuk layer dengan fungsi aktivasi ReLU, yang memungkinkan distribusi nilai *Filter* lebih cocok untuk mencegah masalah vanishing gradient.

Bagaimana Nilai dalam *Filter* Diperoleh:

1. Proses Inisialisasi: Saat model dimulai, nilai *Filter* biasanya diinisialisasi secara acak, mengikuti distribusi tertentu (misalnya distribusi normal atau uniform).
2. Proses Pelatihan: Selama pelatihan, nilai-nilai dalam *Filter* diperbarui menggunakan gradient descent untuk meminimalkan fungsi loss. Ini berarti nilai-nilai dalam *Filter* akan berubah sesuai dengan bagaimana jaringan belajar dari data.

Namun, nilai-nilai ini akan berubah seiring dengan proses pelatihan untuk mengoptimalkan jaringan. Proses ini memungkinkan *Filter* untuk menyesuaikan diri dengan fitur-fitur yang ada dalam data, seperti garis, tekstur, atau pola yang relevan untuk tugas tertentu (misalnya klasifikasi citra).

Metode *Xavier Initialization* (atau *Glorot initialization*) dan *He Initialization* adalah teknik untuk menginisialisasi bobot (nilai dalam *Filter*) dalam jaringan saraf untuk mencegah masalah yang bisa terjadi saat pelatihan, seperti *vanishing gradients* (gradien yang menghilang) atau *exploding gradients* (gradien yang meledak). Berikut penjelasan tentang cara kerja kedua metode tersebut:

A. *Xavier Initialization* (Glorot Initialization)

Metode ini dirancang untuk layer dengan fungsi aktivasi *sigmoid* atau *tanh*, yang sensitif terhadap skala gradien. *Xavier Initialization* bertujuan untuk menjaga agar *varians* aktivasi tetap konstan pada setiap layer dalam jaringan, yang penting agar sinyal (baik aktivasi maupun gradien) tidak hilang atau berkembang terlalu besar.[34]

Tujuan metode ini menjaga agar *varians output* layer tetap konstan, mencegah hilangnya atau meluapnya gradien. Bagaimana cara kerjanya: Untuk layer dengan fungsi aktivasi *sigmoid* atau *tanh*, bobot pada layer tersebut diinisialisasi dengan distribusi acak yang memiliki varians:

$$\mathrm{Var}(w) = \frac{2}{n_{\text{input}} + n_{\text{output}}}$$

di mana:

n_{input} adalah jumlah neuron di layer sebelumnya,

n_{output} adalah jumlah neuron di layer berikutnya.

Biasanya, *Xavier Initialization* menggunakan distribusi uniform atau normal dengan mean 0 dan *varians* seperti yang dijelaskan di atas.

Contoh: Jika sebuah layer memiliki 100 neuron *input* dan 50 neuron output, maka bobot-bobot diinisialisasi dengan distribusi yang memiliki *varians* 2/(100+50) = 2/150.

B. He Initialization

Metode *He Initialization* lebih cocok untuk layer dengan fungsi aktivasi ReLU atau variannya (seperti *Leaky* ReLU), yang sering kali memiliki masalah dengan aktivasi yang tidak cukup kuat di layer awal. Tujuan dari metode ini adalah menjaga agar gradien tidak hilang atau meledak, dan mengakomodasi karakteristik fungsi aktivasi ReLU yang memiliki kemiringan (*gradient*) yang besar pada bagian positif dan 0 pada bagian negatif. Cara kerjanya untuk layer dengan fungsi aktivasi ReLU, bobot diinisialisasi dengan distribusi yang memiliki varians. [35]

$$\mathrm{Var}(w) = \frac{2}{n_{\mathrm{input}}}$$

Di mana:

n_{input} adalah jumlah neuron pada layer sebelumnya.

He Initialization mengasumsikan bahwa ReLU lebih aktif dan memungkinkan nilai gradien yang lebih besar, sehingga membutuhkan inisialisasi dengan *varians* yang lebih besar dibandingkan dengan Xavier.

Contoh: Jika sebuah layer memiliki 100 neuron *input*, maka bobot-bobot diinisialisasi dengan distribusi yang memiliki *varians* 2/100.

Xavier Initialization digunakan untuk fungsi aktivasi *sigmoid/tanh*, yang lebih sensitif terhadap masalah vanishing gradient. Ini mencoba menjaga agar *varians* tetap seimbang di seluruh layer. Gradien yang sangat kecil saat backpropagation, menyebabkan bobot sulit diperbarui, yang memperlambat pelatihan atau bahkan menghentikannya *He Initialization* digunakan untuk fungsi aktivasi ReLU yang cenderung memiliki gradient lebih besar, memungkinkan lebih banyak informasi untuk mengalir tanpa masalah vanishing gradient. Gradien yang sangat besar, menyebabkan pembaruan bobot yang sangat besar dan tidak stabil.

2.13 *Agregasi Chanel*

Agregasi channel dalam konteks pengolahan data atau sinyal adalah proses menggabungkan informasi dari beberapa *channel* (saluran) menjadi satu representasi gabungan. Konsep ini digunakan untuk mengurangi dimensi data sekaligus menangkap informasi penting dari seluruh saluran. Misalnya, dalam sistem berbasis AI untuk penginderaan dan agregasi data pada *edge computing*, *agregasi channel* membantu meningkatkan akurasi inferensi dengan menggabungkan fitur dari berbagai sensor atau saluran sinyal menjadi satu

representasi yang lebih kaya [36].

Dalam konteks visualisasi citra berbentuk tuple agregasi, seperti maximum projection atau mean projection di sepanjang dimensi *channel*, tujuannya adalah untuk mereduksi data multidimensi (misalnya, gambar 3D dengan beberapa *channel*) menjadi gambar 2D. Berikut cara membaca dan memahami proses tersebut. Dalam data gambar, *channel* biasanya merujuk ke saluran informasi yang berbeda (misalnya, RGB, saluran intensitas tertentu dalam citra medis, atau spektrum warna. *Agregasi channel* dilakukan untuk mereduksi data dari banyak *channel* menjadi satu representasi gambar 2D.

Ada dua cara pertama Maximum Projection, maksimum projection bekerja dengan memilih nilai maksimum dari semua *channel* untuk setiap piksel dengan tujuan menonjolkan fitur dengan intensitas tertinggi. Kedua, Mean Projection, bekerja dengan mengambil rata-rata nilai intensitas di semua *channel* untuk setiap piksel dengan memberikan informasi rata-rata keseluruhan. Hasil Proyeksi Maximum Projection mewakili intensitas tertinggi dari semua *channel* di setiap posisi piksel sehingga visualisasi Piksel yang cerah berarti ada *channel* yang memiliki intensitas tinggi pada posisi tersebut. Pada Mean Projection yang memberikan gambaran rata-rata intensitas dari semua *channel* sehingga piksel cenderung memiliki tingkat kecerahan lebih rata. Area nilai intensitas tinggi Mewakili piksel di mana nilai rata-rata RGB cukup tinggi. Ini bisa menunjukkan area dengan warna dominan terang atau putih. Area nilai intensitas rendah Mewakili piksel dengan rata-rata RGB rendah, biasanya pada area yang gelap atau kurang informasi warna.

Rumus Mean Projection :

$$I(i,j) = \frac{R(i,j) + G(i,j) + B(i,j)}{3}$$

i = posisi tinggi *pixel* ,

j = posisi panjang *pixel*

I (i,j) = rata – rata intensitas di setiap kanal

R(i,j) = intensitas piksel dari kanal merah

G(i,j) = intensitas piksel dari kanal hijau

B(i,j) = intensitas piksel dari kanal biru

Pada penelitian ini agregasi chanel dengan mean projection digunakan dalam visualisai tensor menjadi gambar dari hasil keluaran tiap tahapan *InceptionResnetV2*.

2.14 Evaluasi

Nilai yang digunakan untuk menentukan performa model dalam penialaian kualitas citra fundus divisualisasikan dalam bentuk *confusion matrix* yang divisualisasikan dalam bentuk tabel. *Confusion matrix* adalah tabel yang secara spesifik menunjukkan performa model atau algoritma klasifikasi. Setiap baris pada tabel ini mewakili kelas aktual dari data, sedangkan setiap kolom menunjukkan kelas prediksi dari data (atau sebaliknya). Terdapat empat istilah utama dalam *confusion matrix* yang menggambarkan hasil klasifikasi, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN).

Jika sebuah instance sebenarnya positif dan diprediksi sebagai positif, maka disebut *True Positive* (TP). Sebaliknya, jika instance yang sebenarnya positif diprediksi sebagai negatif, disebut *False Negative* (FN). Untuk instance yang sebenarnya negatif, jika diprediksi sebagai negatif disebut *True Negative* (TN), dan jika diprediksi sebagai positif disebut *False Positive* (FP).

Confusion matrix dapat digunakan untuk menghitung berbagai metrik kinerja (*performance metrics*) yang membantu mengevaluasi model, seperti *accuracy*, *precision*, *recall*, dan *F1-score*.

Accuracy mengukur seberapa tepat model dalam mengklasifikasikan data secara keseluruhan. Nilai ini dihitung sebagai rasio jumlah prediksi benar terhadap total data, menunjukkan tingkat kesesuaian prediksi dengan nilai aktual.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (12)$$

Precision mengukur tingkat akurasi model dalam memprediksi kelas positif, dihitung sebagai rasio prediksi benar positif terhadap semua hasil yang diprediksi positif.

$$precision = \frac{TP}{TP + FP} \tag{13}$$

Recall (juga dikenal sebagai *sensitivity*) menunjukkan kemampuan model dalam mendeteksi kelas positif dengan benar, yaitu rasio prediksi benar positif terhadap total data positif yang sebenarnya.

$$recall = \frac{TP}{TP + FN} \tag{14}$$

F1-score adalah kombinasi *precision* dan *recall* yang memberikan gambaran tentang keseimbangan antara keduanya dalam mengevaluasi performa model.

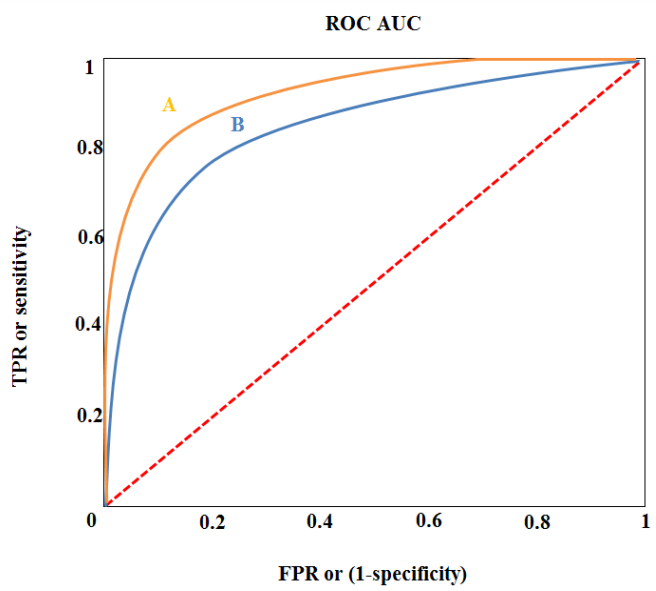
$$F1\text{-}Score = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)} \tag{15}$$

Spesifisitas adalah perhitungan pada probabilitas citra, pada penelitian ini merupakan citra yang dikatakan bukan *Diabetic retinopathy* dan pada kenyataannya citra tersebut memang bukan *Diabetic retinopathy*. Spesifisitas dihitung dengan persamaan (16).

$$Spesifisitas = \frac{TN}{FP + TN} \tag{16}$$

Untuk evaluasi lebih lanjut, digunakan Receiver Operating Characteristics (ROC), sebuah teknik visualisasi yang populer untuk menilai performa model klasifikasi diagnostik. *Kurva ROC* adalah grafik dua dimensi, dengan sumbu *X* menunjukkan *False Positive Rate (FPR)* dan sumbu *Y* menunjukkan *True Positive Rate (TPR).* Pada grafik ini terdapat garis diagonal yang membagi *area sensitivitas* dan *spesifisitas* menjadi dua bagian yang sama, dikenal sebagai *random guess*.

Sebagai tambahan, *Area Under the ROC Curve* (AUC) sering digunakan untuk menilai akurasi klasifikasi. AUC mengukur luas area di bawah kurva ROC, dengan nilai berkisar antara 0,0 hingga 1,0. Semakin dekat nilai AUC ke 1, semakin tinggi akurasi klasifikasi model..



Gambar 2.22. Kurva ROC AUC

Gambar 2.22 memperlihatkan perbandingan performa dua algoritme klasifikasi 'A' dan 'B' berdasarkan ROC AUC. Berdasarkan kurva ROC AUC maka dapat disimpulkan bahwa algoritme 'A' memiliki keakuratan klasifikasi yang lebih baik dibandingkan algoritme 'B', karena AUC dari algoritma 'A' mendekati nilai 1.

2.15 *Praprocessing Images*

Praprocessing gambar adalah langkah penting dalam *pipeline machine learning*, khususnya dalam tugas-tugas yang melibatkan pengenalan pola atau analisis gambar. Tujuannya adalah untuk meningkatkan kualitas citra atau mengekstraksi fitur yang lebih informatif sebelum digunakan dalam pelatihan model[37]. Pada penelitian ini digunakan beberapa teknik untuk *praprocessing images.*

2.12.1 Ruang warna

Sebelum membahas Teknik prosesing gambar ada baiknya untuk mengenal dasar dari pengenalan citra itu sendiri, salah satunya mengenai ruang warna. Beberapa ruang warna yang paling populer adalah RGB, YUV, HSV, Lab, dan sebagainya. Setiap ruang warna memberikan keuntungan yang berbeda. Pemilihan ruang warna yang didasarkan dari masalah yang diangkat, pemilihan ruang warna ini penting terutaman untuk penyesuaian dengan model *machine learning* yang akan digunakan. Ruang warna dalam citra memiliki fokus informasi yang berbeda,

beberapa informasi yang diberikan [38]:

- RGB**,** merupakan ruang warna yang paling populer. RGB adalah singkatan dari *Red, Green,* dan *Blue* (Merah, Hijau, dan Biru). Dalam ruang warna ini, setiap warna diwakili sebagai kombinasi tertimbang dari merah, hijau, dan biru. Jadi, setiap nilai *pixel* diwakili sebagai tupel dari tiga angka yang sesuai dengan merah, hijau, dan biru. Setiap nilai berkisar antara 0 dan 255.
- YUV, Meskipun RGB bagus untuk banyak tujuan, RGB cenderung sangat terbatas untuk banyak aplikasi kehidupan nyata. Orang-orang mulai memikirkan metode berbeda untuk memisahkan informasi intensitas dari informasi warna. Oleh karena itu, mereka mengembangkan ruang warna YUV. Y mengacu pada *luminance* atau intensitas, dan saluran U/V mewakili informasi warna. Ini bekerja dengan baik di banyak aplikasi karena sistem visual manusia merasakan informasi intensitas sangat berbeda dari informasi warna.
- HSV digunakan karena ternyata bahkan YUV masih belum cukup baik untuk beberapa aplikasi. Jadi, orang-orang mulai berpikir tentang bagaimana manusia melihat warna dan mereka mengembangkan ruang warna HSV. HSV adalah singkatan dari *Hue, Saturation*, dan *Value* (Nada, Kejenuhan, dan Nilai). Ini adalah sistem silinder yang memisahkan tiga properti warna yang paling utama dan merepresentasikannya menggunakan saluran yang berbeda. Ini sangat terkait dengan bagaimana sistem visual manusia memahami warna. Ini memberi kita banyak fleksibilitas tentang bagaimana kita dapat menangani gambar.

Berdasarkan hasil penilitian sebelumnya oleh Nugroho (2014) [7] yang memfokuskan pada ruang warana RGB, kanal hijau (Green *channel*) merupakan kanal dengan kontras terbaik bagi model untuk mengenali fitur yang ada pada citra fundus. Sehingga dalam penelitian ini kanal tersebut pula yang diekstraksi kemudian di-*enhancement* sebelum dilakukan ekstraksi fitur. *Enhancement* citra dilakukan dengan meregangkan *contrast* menggunakan algoritma CLAHE.

2.12.2 *Contrast Limited Adaptive Histogram Equalization (CLAHE)*

Contrast Limited Adaptive Histogram Equalization (CLAHE) merupakan metode pemrosesan citra yang digunakan untuk meningkatkan kontras, terutama pada area gambar dengan rentang intensitas sempit atau area yang bersifat homogen. Teknik ini merupakan pengembangan dari *Adaptive Histogram Equalization* (AHE) yang lebih umum. CLAHE dirancang untuk mengatasi masalah noise yang sering muncul pada AHE dengan membatasi peningkatan kontras di area homogen. Area homogen ini biasanya ditandai dengan puncak tinggi pada histogram lokal, karena banyak piksel memiliki tingkat keabuan yang serupa.

Dalam CLAHE, peningkatan kontras dibatasi dengan mengatur skema penugasan tingkat keabuan agar tidak terlalu curam. Hal ini dilakukan dengan membatasi jumlah maksimum piksel pada setiap bin di histogram lokal. Setelah histogram dipotong (*clipping*), piksel yang berlebih didistribusikan secara merata ke seluruh histogram untuk menjaga total jumlah piksel tetap sama [35].

CLAHE bekerja dengan tahapan yaitu pertama melakukan Identifikasi Area Homogen, CLAHE membagi gambar menjadi blok-blok kecil. Untuk setiap blok, histogram intensitas dihitung. Area dengan puncak histogram yang tinggi menunjukkan area homogen. Kemudian dilakukan pembatasan kontras untuk mencegah peningkatan kontras yang berlebihan pada area homogen, CLAHE membatasi jumlah *pixel* maksimum pada setiap *bin* histogram. *Pixel* yang melebihi batas ini akan "dipotong" dan didistribusikan kembali ke *bin-bin* lainnya. Setelah pembatasan kontras, histogram yang telah dimodifikasi digunakan untuk memetakan nilai intensitas *pixel* asli ke nilai intensitas baru, sehingga menghasilkan peningkatan kontras secara keseluruhan.

CLAHE dipilih untuk digunakan karena teknik ini sangat efektif dalam meningkatkan detail pada area dengan kontras rendah, seperti bayangan atau area yang *overexposed*. Dengan membatasi peningkatan kontras pada area homogen, CLAHE dapat mengurangi *noise* yang sering muncul pada teknik peningkatan kontras lainnya. Selain itu CLAHE juga menyesuaikan peningkatan kontras secara lokal untuk setiap blok gambar, sehingga menghasilkan hasil yang lebih natural. Namun demikian berdasarkan percobaan, hasil citra dari hanya menerapkan

CLAHE masih agak buram, karenanya pada penelitian ini dipertimbangkan untuk menambahkan proses pengolahan citra lainnya yaitu *Sharpening* dan *Super Resolution*.

2.12.3 *Sharpening*

Sharpening Filter (*Filter* penguat tepi) digunakan untuk mengekstrak dan menonjolkan detail halus dari sebuah gambar, serta meningkatkan detail yang buram. Penggunaan khas dari *Filter* semacam ini adalah untuk menghilangkan kekaburan pada gambar, sehingga menghasilkan tepi yang lebih tajam dan detail dalam gambar [39]. Dengan menerapkan *Filter* ini, gambar yang awalnya buram dapat terlihat lebih tajam dan jelas. Proses ini dilakukan dengan cara mengurangi nilai *Pixel* di sekitar tepi (tepat perubahan intensitas warna yang tiba-tiba) dalam gambar dan kemudian menekankan tepi tersebut dengan meningkatkan nilai *Pixel* di area tepi itu.

2.12.4 *Super Resolution*

Super-Resolusi adalah teknik yang digunakan untuk mendapatkan citra yang beresolusi tinggi dari kumpulan citra yang beresolusi rendah. Citra resolusi tinggi didapat dari sekumpulan resolusi rendah yang diambil dari *scene* (adegan) yang sama. Karena dari *scene* yang sama akan menyediakan informasi yang mungkin dapat digunakan untuk merekonstruksi citra resolusi tinggi [40]. Dalam konteks ini, *super-resolution* merujuk pada teknik yang digunakan untuk meningkatkan resolusi gambar, sehingga menghasilkan gambar yang lebih besar dan lebih jelas. *Super Resolution* secara umum terdiri dari tiga tahap algoritma yaitu *registrasi, interpolasi* dan *rekonstruksi* [40]. Dalam penelitian ini, metode yang digunakan adalah *interpolasi kubik*.

Interpolasi kubik adalah metode yang menggunakan polinomial derajat tiga untuk menghitung nilai *Pixel* baru berdasarkan nilai *Pixel* di sekitarnya. Dengan demikian, *Interpolasi kubik* meningkatkan ukuran gambar untuk menghasilkan gambar yang lebih besar dengan detail yang lebih baik. Metode ini memberikan hasil yang lebih halus dibandingkan *interpolasi bilinear*, karena mempertimbangkan lebih banyak titik data saat menghitung nilai *Pixel* baru. Meskipun tidak ada rumus tunggal untuk *interpolasi kubik*, prosesnya melibatkan

penggunaan polinomial untuk menghitung nilai *Pixel* baru. Dalam konteks citra, *Interpolasi kubik* dapat dinyatakan pada persamaan (17):

$$f(x) = a_00 + a_11\ x + a_22\ x^2 + a_33\ x^3 \text{.....(17)}$$

Dengan f(x) adalah nilai *Pixel* baru yang dihitung berdasarkan nilai *Pixel* di sekitarnya, dan $a_00 + a_11\ x + a_22\ x^2 + a_33\ x^3$ adalah koefisien yang ditentukan berdasarkan titik-titik data yang ada.

2.13 OSEMN *Framework*

OSEMN (*Obtain, Scrub,Explore, Model, Interpret)* adalah salah satu kerangka kerja dalam *data science* yang terdiri dari serangkaian tahapan untuk menyelesaikan masalah terkait pengolahan data *[24].* Kerangka kerja ini membantu mengarahkan proses *data science* secara sistematis. Diagram alur dari *framework OSEMN* dapat dilihat pada Gambar 2.2.

Gambar 2. 23.OSEMN *Framework*[41]

2.13.1 ***Obtain***

Tahap pertama dalam *framework* OSEMN adalah pengumpulan data dari berbagai sumber yang tersedia dalam berbagai format. Data yang dikumpulkan harus sesuai untuk dianalisis dalam proses *data science* [29]. Proses pengumpulan data dapat dilakukan dengan menggunakan *web API* atau dengan cara tradisional, seperti mengunduh file dalam format Excel (.csv) atau gambar [24]

2.13.2 ***Scrub***

Tahap berikutnya setelah data dikumpulkan adalah proses pembersihan atau *scrubbing* data. Pada tahap ini, data dibersihkan, di*Filter*, diganti, atau digabungkan [24]. Beberapa langkah yang dilakukan meliputi menghapus data yang tidak

relevan, mengisi nilai yang hilang, menghapus duplikasi, mengubah data ke dalam format tertentu, dan menggabungkan semua data ke dalam satu format seperti CSV [30]. Tujuan utama dari proses ini adalah memastikan data dapat dibaca dan dipahami oleh mesin untuk keperluan modeling.

2.13.3 ***Explore Data***

Tahap *eksplorasi data* menjadi penghubung antara pembersihan data *(scrubbing)* dan pemodelan *(modeling).* Tujuan dari *eksplorasi data* adalah mencari informasi penting dari kumpulan data yang akan digunakan dalam proses pemodelan, sekaligus membantu membangun hipotesis [39]. Dalam tahap ini, data dianalisis dengan beberapa perspektif [39] :

- Perspektif pertama yaitu memeriksa data beserta propertinya dengan cara melihat nama variabel, tipe data variabel, dan visualisasi data sebelum dan sesudah pra-pemrosesan untuk memahami distribusi data dan efek dari teknik pra-pemrosesan (misalnya *cropping*) [30].
- Perspektif kedua yaitu penggunaan algoritma ekstraksi fitur untuk mengambil informasi yang relevan dari gambar. Proses pemilihan atau ekstraksi informasi relevan dari data, seperti tekstur, bentuk, warna, atau fitur-fitur tingkat tinggi yang dapat ditangkap oleh algoritma seperti HOG, SIFT, atau deep *learning*. [30].
- Perspektif ketiga yaitu visualisasi data yang berfungsi untuk mengidentifikasi tren dan pola dari data tersebut sehingga didapatkan citraan yang lebih mudah dipahami dalam bentuk grafik sederhana [30].

2.13.4 ***Model***

Pada tahap ini, model dipilih dan dibangun untuk melakukan prediksi. Proses pemodelan mencakup pembuatan deskripsi abstrak tentang jenis model prediksi yang akan digunakan pada dataset. Selain itu, model juga dilatih agar mampu mengklasifikasikan data berdasarkan label yang diberikan [30]..

2.13.5 ***Interpret***

ahap terakhir dari framework OSEMN adalah interpretasi hasil. Pada tahap ini, data dan model yang telah dibuat dianalisis untuk menghasilkan interpretasi yang mudah

dipahami. Hasil tersebut kemudian dideskripsikan dengan jelas dan dibandingkan dengan penelitian lainnya. Interpretasi adalah fase penting yang berfokus pada penyederhanaan dan meringkas hasil dari semua model yang dibangun. Wawasan yang dihasilkan digunakan untuk menarik kesimpulan rasional dan menentukan langkah selanjutnya [29].

2.14 Perangkat Lunak (*Software*)

Perangkat lunak ataupun *software* adalah bagian dari sistem komputer sebagai sarana penghubung dalam interaksi pengguna dengan perangkat keras. Pada penelitian ini berbagai perangkat lunak digunakan sebagai penunjang dalam proses penelitian.

2.14.1 ***Microsoft Excel***

Microsoft excel adalah aplikasi yang digunakan untuk mengolah angka, menganalisis data, dan membuat visualisasi informasi. Dikembangkan oleh Microsoft, program ini sering digunakan untuk melakukan perhitungan, pengolahan data, dan membantu pengambilan keputusan.

2.14.2 ***Visual Studio Code***

Visual Studio Code merupakan editor teks yang ringan namun andal, dikembangkan oleh Microsoft untuk berbagai sistem operasi seperti Windows, Linux, dan macOS. Editor ini mendukung pengembangan perangkat lunak untuk berbagai platform, termasuk aplikasi web, desktop, dan seluler.

2.14.3 ***Python***

Python adalah bahasa pemrograman tingkat tinggi yang populer dan serbaguna. Dikembangkan oleh Guido van Rossum pada tahun 1991, *Python* dikenal karena sintaksisnya yang sederhana dan keterbacaan kode. Bahasa ini digunakan secara luas dalam berbagai bidang seperti teknologi informasi, *data science*, pengembangan web, *game development*, dan lainnya.

Python merupakan pilihan utama dalam data science, khususnya untuk deep

learning dan machine learning, karena komunitasnya yang besar dan dukungan banyak pustaka. Berikut adalah beberapa pustaka *Python* yang sering digunakan dalam pengolahan data [31] :

- Numpy

Numpy adalah pustaka yang digunakan untuk memproses array dan matriks multidimensi, dengan dukungan untuk aljabar linier, bilangan acak, dan *transformasi Fourier*.

- Pandas

Pandas adalah pustaka yang dirancang untuk manipulasi dan analisis data. Struktur data seperti *dataframe* yang disediakan sangat membantu dalam bekerja dengan data tabula.

- Matplotlib

Matplotlib igunakan untuk visualisasi data, seperti membuat berbagai jenis grafik (misalnya histogram, scatterplot, dan diagram lingkaran).

- *TensorFlow*

TensorFlow adalah pustaka yang sering digunakan untuk pemodelan *machine learning*. *Library* ini mendukung berbagai perangkat keras dan digunakan untuk membangun, melatih, serta menerapkan model *deep learning*.

- *Keras*

Keras menyediakan antarmuka tingkat tinggi untuk membangun dan melatih model jaringan saraf. *Library* ini mendukung berbagai komponen jaringan saraf seperti lapisan, fungsi aktivasi, dan pengoptimal..

- *OpenCV*

OpenCV adalah perpustakaan pemrograman yang dirancang untuk memvasilitasi pengolahan gambar dan visi komputer.. Pustaka ini menyediakan berbagai fungsi untuk manipulasi gambar, deteksi objek, analisis video, dan banyak lagi, serta mendukung berbagai bahasa pemrograman seperti *Python* , C++, dan Java.OpenCV banyak digunakan dalam aplikasi seperti pengawasan keamanan, kendaraan otonom, dan augmented reality. Dengan komunitas pengguna yang besar dan

dokumentasi yang komprehensif, OpenCV menjadi alat yang sangat berguna dalam bidang visi komputer [42].

- *Scikit-learn*

Scikit-learn merupakan perangkat lunak *open source* yang digunakan dalam mengatasi permasalahan terkait *supervice* dan *unsupervised* dalam pembelajaran mesin. Dibangun seluruhnya dalam bahasa *Python* dan memanfaatkan perpustakaan populer seperti numpy,scipy, karenannnya memiliki kemudahan penggunaan dan integrasinya dengan ekosistem *Python* lainnya seperti NumPy, SciPy, dan pandas. Alasan utama mengapa scikit-learn sangat populer berasal dari fakta bahwa sebagian besar algoritme pembelajaran mesin paling populer di dunia dapat diimplementasikan dengan cukup cepat dalam format plug and play setelah mengetahui seperti apa core pipeline-nya. Alasan lainnya adalah algoritma populer untuk klasifikasi seperti regresi logistik dan mesin vektor dukungan ditulis dalam Cython. Cython digunakan untuk memberikan kinerja seperti C pada algoritma ini dan dengan demikian membuat penggunaan scikit-learn cukup efisien dalam prosesnya [43].

2.14.4 ***PyTorch***

PyTorch adalah sebuah perpustakaan pembelajaran mesin untuk *Python* yang dibangun di atas perpustakaan Torch. PyTorch secara luas digunakan sebagai alat pembelajaran mendalam baik untuk penelitian maupun pengembangan aplikasi industri. Perpustakaan ini terutama dikembangkan oleh Meta. PyTorch menjadi pesaing bagi perpustakaan pembelajaran mendalam terkenal lainnya, TensorFlow, yang dikembangkan oleh Google. Perbedaan awal antara keduanya adalah PyTorch menggunakan *Eksekusi eager* sedangkan TensorFlow dibangun berdasarkan grafik komputasi tertunda. Meskipun demikian, TensorFlow kini juga menyediakan mode *eksekusi eager*.

Eksekusi eager pada dasarnya adalah mode pemrograman imperatif dengan operasi matematika dihitung segera. Mode eksekusi tertunda akan menyimpan semua operasi dalam sebuah grafik komputasi tanpa perhitungan langsung, kemudian seluruh grafik akan dievaluasi nanti. *Eksekusi eager* dianggap lebih menguntungkan karena alasan seperti alur yang intuitif, mudah di-debug, dan lebih

sedikit kode bantu [44].

PyTorch memiliki sintaks/antarmuka mirip NumPy, namun lebihnya PyTorch menyediakan kemampuan komputasi tensor dengan akselerasi yang kuat menggunakan GPU. Tensor sendiri merupakan unit komputasi, sangat mirip dengan *array* NumPy namun tensor diakselerasikan dengan GPU untuk mempercepat komputasi. Dengan komputasi yang dipercepat dan fasilitas untuk membuat grafik komputasi dinamis, PyTorch menyediakan kerangka kerja pembelajaran mendalam yang lengkap. Selain itu, PyTorch benar-benar bersifat *Python ic*, yang memungkinkan pengguna PyTorch untuk memanfaatkan semua fitur yang disediakan *Python* , termasuk ekosistem ilmu data *Python* yang luas. PyTorch memiliki berbagai modul yang memperluas berbagai fungsionalitas dalam membantu memuat data, membangun model, dan menentukan jadwal optimasi selama pelatihan model. Berikut adalah beberapa fitur utama PyTorch[44]:

1. *Tensor Computation:* PyTorch menyediakan kelas Tensor, yang merupakan struktur data utama untuk menyimpan dan mengoperasikan data. Tensors mirip dengan *array* NumPy tetapi juga dapat dijalankan di GPU untuk komputasi yang lebih cepat.
2. *Dynamic Computation Graphs:* Salah satu fitur yang paling menonjol dari PyTorch adalah penggunaan computational graph yang dinamis. Ini berarti bahwa grafik komputasi dibangun secara langsung saat eksekusi, memungkinkan lebih banyak fleksibilitas dan kemampuan untuk melakukan debugging yang lebih mudah.
3. *Autograd:* PyTorch memiliki fitur autograd yang memungkinkan perhitungan derivatif secara otomatis. Ini sangat penting untuk pelatihan model *Deep learning* menggunakan teknik backpropagation.
4. *Neural Network Module:* PyTorch menyediakan modul torch.nn, yang berisi berbagai lapisan neural network, fungsi aktivasi, loss functions, dan optimizers untuk membangun dan melatih model neural network.
5. *Optimizers:* PyTorch menyediakan berbagai algoritma optimisasi dalam modul torch.optim, termasuk SGD, Adam, RMSprop, dan lainnya.

6. *Data Utilities:* PyTorch menyediakan kelas Dataset dan DataLoader untuk memudahkan pemuatan dan pengelolaan data.
7. *CUDA Integration:* PyTorch memiliki dukungan yang sangat baik untuk menjalankan komputasi pada GPU melalui CUDA, yang memungkinkan pelatihan model *Deep learning* dengan kecepatan yang jauh lebih cepat dibandingkan CPU.
8. *Ecosystem:* PyTorch memiliki ekosistem yang luas dan aktif yang mencakup pustaka untuk visi komputer (torchvision), pemrosesan teks (torchtext), dan banyak lagi.

2.16 Penelitian Terkait

Beberapa penelitian berbasis *Machine learning* dan *Deep Learning* untuk pendeteksian *Diabetic retinopathy* tengah banyak dilakukan beberapa tahun terakhir. Penelitian ini didasarkan pada ilmu yang didapat dari penelitian penelitian sebelumnya.

Penelitian yang dilakukan oleh Pardede, et al., pada tahun 2020[45], memfokuskan pada penggunaan dataset melanoma yang terdiri dari 5341 citra melanoma dan 1780 citra non-melanoma. Dalam penelitian ini, berbagai model telah diterapkan, termasuk *ResNet, LeNet, Support Vector Machine* (SVM), dan *DenseNet*121, dengan tujuan untuk membandingkan performa masing-masing model dalam prediksi kanker kulit. Hasil penelitian menunjukkan bahwa *DenseNet*121 secara signifikan mengungguli model lainnya. Model ini berhasil mencapai nilai akurasi sebesar 0.94, *precision* sebesar 0.95, dan *recall* sebesar 0.92. Hasil ini menunjukkan bahwa *DenseNet*121 memiliki kemampuan yang lebih baik dalam mengidentifikasi kasus melanoma dibandingkan dengan *ResNet, LeNet*, dan SVM, yang menjadikannya pilihan yang sangat baik untuk aplikasi klinis dalam mendeteksi kanker kulit. Penelitian ini memberikan landasan kuat untuk mengeksplorasi model *deep learning* lainnya dalam mendeteksi penyakit kulit dengan akurasi yang lebih tinggi.

Penelitian yang dilakukan oleh Wardani, et al., pada tahun 2020[46], memanfaatkan

data citra fundus mata digital dari database STARE, yang terdiri dari 90 citra retina fundus dalam format PNG. Penelitian ini mengimplementasikan algoritma *hybrid* dari SVM dan *K-Nearest Neighbor (KNN*) untuk diagnosis penyakit mata. Dengan konversi skala abu-abu dan deteksi tepi *Canny* untuk mendeteksi *mikroaneurisma*, algoritma SVM mencapai akurasi sebesar 77,39%. Namun, dengan mengombinasikan algoritma SVM dan KNN, akurasi meningkat signifikan hingga 94,67%, menunjukkan efektivitas pendekatan *hybrid* ini dalam mengklasifikasikan Retinopati Diabetes dengan tingkat akurasi yang tinggi. Hasil penelitian ini relevan dalam mengembangkan metode yang lebih baik untuk diagnosis penyakit mata melalui analisis citra fundus.

Selanjutnya, penelitian oleh Sarker, et al., pada tahun 2020[47], menggunakan dataset COVIDx yang mencakup 13,800 citra radiologi dada dari 13,725 pasien. Dataset ini merupakan sumber daya terbuka yang digunakan untuk mendeteksi COVID-19. Penelitian ini menerapkan arsitektur *DenseNet*-121 dengan koneksi lapisan langsung dan efek regularisasi, serta menggunakan transfer pembelajaran dari model ChexNet. Hasilnya menunjukkan bahwa model ini mencapai akurasi 91% melalui validasi silang 10 kali lipat, memperlihatkan potensi besar dari *DenseNet*-121 dalam deteksi COVID-19 secara akurat berdasarkan data radiologi. Temuan ini menjadi dasar bagi penelitian lebih lanjut dalam menggunakan arsitektur *DenseNet* untuk deteksi penyakit lain dengan dataset citra medis yang beragam.

Odeh, et al., pada tahun 2021[48], memanfaatkan dataset MESSIDOR dan menerapkan pendekatan ensemble learning dengan kombinasi algoritma *Random Forest, Neural Network*, dan SVM. Penelitian ini bertujuan untuk meningkatkan akurasi deteksi penyakit mata dengan memanfaatkan kekuatan dari berbagai model. Menggunakan dua metode seleksi fitur, yaitu *InfoGainEval* dan *WrapperSubsetEval*, hasilnya menunjukkan bahwa akurasi mencapai 70,7% dengan *InfoGainEval* dan meningkat menjadi 75,1% dengan *WrapperSubsetEval*. Penelitian ini menyoroti pentingnya pemilihan fitur yang tepat dan penggabungan berbagai algoritma untuk meningkatkan kinerja model dalam tugas klasifikasi penyakit mata, yang menjadi referensi dalam pengembangan model yang lebih kompleks dan akurat.

Sharma, et al., pada tahun 2021[37], mengembangkan model untuk mengklasifikasikan retinopati berdasarkan *mikroaneurisma* dan *eksudat* menggunakan dataset DIARETDB0 dan DIARETDB1. Dalam penelitian ini, metode Wavelet dan SVM diterapkan. Hasilnya menunjukkan bahwa metode Weighted KNN mencapai akurasi sebesar 85,8%, sementara metode *Cubic* yang menggunakan fitur kemiringan dan wavelet level 5 mencapai akurasi tertinggi. Sementara itu, SVM multiclass One-Against-All mencapai akurasi 72% menggunakan parameter kemiringan, menunjukkan bahwa teknik pengolahan citra yang canggih dapat meningkatkan akurasi dalam diagnosis retinopati. Penelitian ini memberikan dasar kuat dalam penggunaan teknik pengolahan citra lanjutan untuk diagnosis penyakit mata, khususnya retinopati.

Pada tahun 2022, Joshi, et al.[49], melakukan penelitian menggunakan dataset Kaggle yang terdiri dari 3,662 foto fundus. Dari jumlah tersebut, 2,930 foto digunakan untuk pelatihan dan 732 foto untuk validasi. Penelitian ini menggunakan model CNN U-Net dan menerapkan berbagai teknik *augmentasi* data seperti rotasi, zoom, flip, buram, kecerahan, dan saturasi untuk meningkatkan variasi data dan kinerja model. Selain itu, pem*Filter*an Gabor digunakan untuk mendeteksi piksel frekuensi tertentu guna memperjelas garis syaraf pada citra fundus. Transformasi morfologi pembukaan digunakan untuk mengurangi noise pada citra akhir. Hasilnya, jaringan saraf ini berhasil mencapai spesifisitas sebesar 0.94 dan akurasi 0.83 dalam deteksi Retinopati Diabetes (DR), menunjukkan keefektifan pendekatan yang komprehensif ini. Penelitian ini menjadi dasar penting dalam pengembangan teknik *augmentasi* data dan pengolahan citra untuk meningkatkan akurasi model dalam diagnosis penyakit mata.

Penelitian yang dilakukan oleh Bakti, et al., pada tahun 2023[50], berfokus pada penggunaan dataset X-Ray dada untuk pelatihan dan validasi dalam mengklasifikasikan Pneumonia. Penelitian ini menguji kumpulan data yang terdiri dari 4000 citra dengan menggunakan arsitektur CNN *InceptionResNet*-V2. Hasil penelitian menunjukkan bahwa model ini berhasil mencapai akurasi sebesar 0.98, yang menandakan bahwa arsitektur *InceptionResNet*-V2 sangat efektif dalam mengklasifikasikan pneumonia berdasarkan citra X-Ray dada. Keberhasilan ini menunjukkan potensi besar dari model ini untuk digunakan dalam aplikasi klinis

dalam diagnosis pneumonia, dan memberikan dasar untuk mengeksplorasi penggunaan arsitektur CNN dalam mendeteksi penyakit lainnya.

Pada tahun 2023, Waasiu, et al.[20], melakukan penelitian yang memanfaatkan dataset dari Kaggle yang terdiri dari 204 citra tulisan tangan, spiral, dan sketsa gelombang. Penelitian ini menggunakan arsitektur *Inception-ResNet-v2* untuk mengklasifikasikan citra penyakit Parkinson. Untuk meningkatkan variasi dan kinerja model, *augmentasi* data diterapkan. Hasil penelitian menunjukkan bahwa model ini mencapai akurasi sebesar 0.9138, *precision* sebesar 0.85, *recall* sebesar 0.80, dan skor F1 sebesar 0.82. Penelitian ini menunjukkan bahwa arsitektur *Inception-ResNet-v2* yang digabungkan dengan teknik *augmentasi* data mampu memberikan hasil yang sangat baik dalam klasifikasi penyakit Parkinson. Hasil ini menjadi dasar untuk mengembangkan model serupa untuk klasifikasi penyakit lain dengan dataset yang lebih kompleks.

Penelitian oleh Nadakumar, et al., pada tahun 2023[51], menggunakan dataset APTOS 2019 yang terdiri dari 3,662 citra pelatihan, serta dataset Kaggle 2015 yang terdiri dari 35,126 citra pemindaian retina. Dalam penelitian ini, teknik fusi data diterapkan untuk prediksi Retinopati Diabetes (DR) dengan menggunakan model *DenseNet*. Arsitektur *DenseNet* dengan fusi fitur diterapkan untuk meningkatkan deteksi DR, yang didukung oleh proses pemrosesan citra klinis dan teknik deteksi yang canggih. Hasil penelitian menunjukkan bahwa model ini berhasil mencapai akurasi pelatihan sebesar 93,51%, dengan *precision* dan *recall* sebesar 0.98, menunjukkan efektivitas teknik fusi data dalam meningkatkan kinerja model dalam deteksi DR. Penelitian ini memberikan dasar yang kuat untuk pengembangan model dengan teknik fusi fitur dalam diagnosis penyakit mata.

Penelitian terbaru oleh Vardhan, et al., pada tahun 2024[52], menggunakan dataset MESSIDOR yang terdiri dari 400 data citra retina yang dikategorikan menjadi No DR dan DR. Dataset ini kemudian disesuaikan untuk klasifikasi CNN *biner* dan *multi-kelas*. Dalam penelitian ini, model yang digunakan adalah kombinasi (*hybrid*) dari *InceptionResnetV2* dan *DenseNet*121. Hasil penelitian menunjukkan bahwa model *InceptionResnetV2* berhasil mencapai akurasi pelatihan sebesar 0.9782, akurasi pengujian sebesar 0.9421, dan AUC pelatihan sebesar 0.9968. Di sisi lain,

model *DenseNet*121 mencapai akurasi pelatihan sebesar 0.7911, akurasi pengujian sebesar 0.7511, dan AUC pelatihan sebesar 0.8466. Penelitian ini menunjukkan bahwa kombinasi dari dua arsitektur yang kuat dapat menghasilkan kinerja yang sangat baik dalam klasifikasi penyakit mata berdasarkan citra retina. Penelitian ini juga menggarisbawahi pentingnya integrasi berbagai teknik dan model dalam meningkatkan efektivitas diagnosis berbasis citra dalam bidang kesehatan.

Dari berbagai penelitian tersebut, terlihat bahwa penggunaan model *deep learning* seperti *DenseNet*, *InceptionResNet*, dan kombinasi *hybrid* dengan berbagai teknik *augmentasi* dan fusi fitur dapat secara signifikan meningkatkan akurasi dan efektivitas dalam diagnosis berbagai penyakit berdasarkan citra medis. Penelitian-penelitian ini menjadi landasan penting untuk pengembangan lebih lanjut dalam menggunakan teknologi *deep learning* untuk aplikasi klinis, khususnya dalam bidang diagnosis penyakit mata dan penyakit lainnya yang dideteksi melalui citra medis. Rangkuman dari penelitian yang terkait dengan penelitian ini dapat dilihat pada tabel 2.3 berikut.

Tabel 2. 3. Penelitian Terkait

No	Peneliti dan tahun	Data Citra	Metode	Hasil
1	Pardede,et.al., 2020[45]	Dataset melanoma dan non-Melanoma.	*ResNet, LeNet, Support Vector Machine DenseNet121*	*DenseNet121* 0.94 untuk nilai akurasi , 0.95 *precision*, dan 0.92 *recall*
2	Wardani, et. al., 2020[46]	STARE Berjumlah 90 citra retina fundus.	SVM , *K-Nearest Neighbor ,*Canny	Akurasi algoritma SVM: 77,39% Akurasi algoritma SVM-KNN: 94,67%
3	Sarker , et. al., 2020[47]	Dataset COVIDx 13,800 citra	*DenseNet*-121	Akurasi 91% menggunakan

No	Peneliti dan tahun	Data Citra	Metode	Hasil
		radiologi dada atau citra CT pasien COVID_19		validasi silang 10 kali lipat
4	Odeh et. al., 2021 [48]	MESSIDOR	Ensemble *learning Random forest, Neural Network*, SVM	Akurasi 70,7% dengan seleksi fitur InfoGainEval, 75,1% dengan seleksi fitur WrapperSubset Eval
5	Sharma et. al. 2021 [37]	DIARETDB0 dan DIARETDB1	Metode Wavelet dan SVM mengklasifikasikan retinopati berdasarkan mikroaneurisma dan eksudat.	Akurasi KNN 85,8%, SVM akurasi 72%
6	Joshi et. al., 2022[49]	Kaggle fundus.	CNN U-Net, Augmentasi data dengan rotasi, zoom, flip, buram, kecerahan, saturasi Pem*Filter*an Gabor untuk deteksi *pixel* frekuensi tertentu (untuk memperjelas garis syaraf)	Spesifisitas 0.94, dan akurasi 0.83 dalam deteksi DR.
7	Bakti, et. al,.	Dataset X-Ray Dada	CNN	Akurasi 0,98.

No	Peneliti dan tahun	Data Citra	Metode	Hasil
	2023[50]	penderita Pneumonia	*Inception*ResNet-V2	
8	Waasiu,al.2023[20]	Tulisan tangan, spiral, dan sketsa gelombang dari Kaggle dengan 204 citra ciri penyakit Parkinson	*Inception*-ResNet-v2	akurasi 0.9138, presisi 0.85, *recall* 0.80, dan skor F1 0.82.
9	Nadakumar, et. al., 2023 [51]	APTOS 2019 citra retina.	*DenseNet*.	akurasi 93,51% dengan presisi dan recall 0,98.
10	Vardhan et. al., 2024[52]	MESSIDOR	Kombinasi (hybrid) *InceptionResnetV2* Dan *DenseNet121*	*InceptionResnet V2*: Train *Accuracy* 0.9782, Test *Accuracy* 0.9421, Train AUC 0.9968. *DenseNet121*: Train *Accuracy* 0.7911, Test *Accuracy* 0.7511, Train AUC 0.8466.

Pada Tabel diketahui terus dilakuakan pengembangan melalui penelitian yang berkaitan dengan DR menggunkan berbagai arsitektur maupun model. Selain itu pula model *InceptionResnetV2* digunakan pula pada berbagai masalah dan data citra yang berbeda. Berdasarkan penelitian yang telah dilakukan sebelumnya, belum terdapat pendeteksian DR dengan model *InceptionResnetV2* yang secara spesifik menunjukan pembagian menjadi multi kelas berupa 5 kelas yang berbeda serta

prepemrosesan citra tambahan yang digunakan terutama Teknik pengaambilan kanal hijau, *Sharpening*, dan Super pesolution. Perbandingan ini dilakukan dengan tujuan mengetahui mana metode yang menghasilkan performa yang paling efisien dalam pendeteksian penyakit *Diabetic retinophaty* berdasarkan citra fundus.

Pada penelitian akan dilakukan pendeteksian *Diabetic retinopathy* dengan deep *learning*. Data citra yang akan digunakan yaitu MESSIDOR 2. Praproses dilakukan dengan *cropping* citra untuk mengambil regiont of interest citra untuk diagnosis Diabetic retinopathy. Selanjutnya dilakukan enhancement citra menggunakan CLAHE pada Green *channel* serta *Sharpening* dan *Super-resolusi*. Klasifikasi akan dilakukan dengan menggunakan *InceptionResnetV2*. Evaluasi dilakukan dengan menghitung akurasi, sensitivitas, spesifisitas, Area under ROC, *F1 score* dan waktu komputasi.

III. METODOLOGI PENELITIAN

3.1 Waktu dan Tempat

Penelitian dan pembuatan skripsi ini dilakukan pada :

Waktu : July 2024 - Desember 2024

Tempat : Laboratorium Teknik Komputer Universitas Lampung

Penelitian akan dilakukan berdasarkan jadwal kegiatan pada Tabel 3.1

Tabel 3. 1 Jadwal Kegiatan Penelitian

No	Aktivitas	2024																						
		Juli				Aug				Sept				Okt				Nov				Des		
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3
1	Studi Literatur																							
2	Persiapan Alat dan Bahan																							
3	Proses pembuatan model Framework OSEMN																							
4	Pengerjaan Laporan																							

3.2 Alat dan Bahan

3.2.1 Alat

Perangkat Keras *(Hardware),* digunakan Laptop dengan spesifikasi AMD Ryzen 5 (5600H) with Radeon Graphics 12 CPUs, ~3.3GHz sebagai perangkat media untuk merancang sistem. Perangkat Lunak *(Software)* yang digunakan pada penelitian ini dapat dilihat pada table 3.2 berikut.

Tabel 3. 2 Perangkat Lunak (*Software*)

No	Nama *Software*	Versi	Keterangan
1.	*Python*	3.11.5	Bahasa pemrograman yang akan digunakan dalam pembuatan algoritma klasifikasi. Pada bahasa pemrograman ini menggunakan beberapa *library* yang digunakan dalam proses klasifikasi menggunakan *Inceptionresnet2*.
	a. NumPy	1.26.4	*Library* yang berguna dalam pemrosesan data numerik. *library* ini menyediakan *array* multidimensi yang efisien dan berbagai fungsi matematika untuk bekerja dengan data.
	b. Pandas	2.2.2	*Library Python* yang digunakan untuk manipulasi dan analisis data. *library* ini menyediakan struktur data seperti *dataframe* yang sangat berguna untuk bekerja dengan data tabular. Digunakan untuk membaca file label citra dan

			menyimpannya dalam bentuk DataFrame.
	c. Matplotlib	3.10.0	*Library* untuk membuat grafik,visualisasi data, dan menampilkan hasil pemrosesan citra dalam mode debug.
	d. *PyTorch*		Sebagai framework utama untuk membangun, melatih, dan menguji model jaringan saraf. Pada penelitian ini digunakna untuk membuat arsitektur model *InceptionResnetV2* dan *DenseNet121* serta untuk definisi dataset, data loader, dan operasi tensor lainnya
	e. Torchvision	0.15.2	*Library pytorch* yang digunakan untuk transformasi citra dan model pra-terlatih. *Torchvision* digunakan untuk transformasi citra seperti konversi citra ke tensor dan normalisasi
	f. OpenCV	4.10.0	*Library machine learning* yang popular yang menyediakan antarmuka tingkat tinggi untuk untuk

			pemrosesan citra. Pada penelitian ini digunakan untuk membaca citra, konversi warna, aplikasi CLAHE (Contrast Limited Adaptive Histogram Equalization), *Sharpening* citra, dan super-resolusi
	g. Pillow	10.0.1	Pillow merupakan library untuk membuka, memanipulasi, dan menyimpan berbagai format file citra dalam *Python* .Pillow digunakan untuk mengonversi citra yang telah diproses menjadi format PIL untuk visualisasi.
	h. *Scikit-learn*	1.5.1	*Library* yang digunakan untuk untuk *Machine learning* dan data *mining*. Pada penelitian ini digunakan pada evaluasi model dan metrik, untuk menghitung metrik evaluasi seperti ROC-AUC dan *F1 score*.
2.	*Microsoft excel*	2016	Perangkat lunak untuk melakukan pengolahan data di tahap pengumpulan data dan *scrubbing* data.
3.	*Visual Studio Code*	1.92.0	*Code editor* yang akan digunakan selama proses

			pembuatan algoritma *CNN*.

3.2.2 Bahan

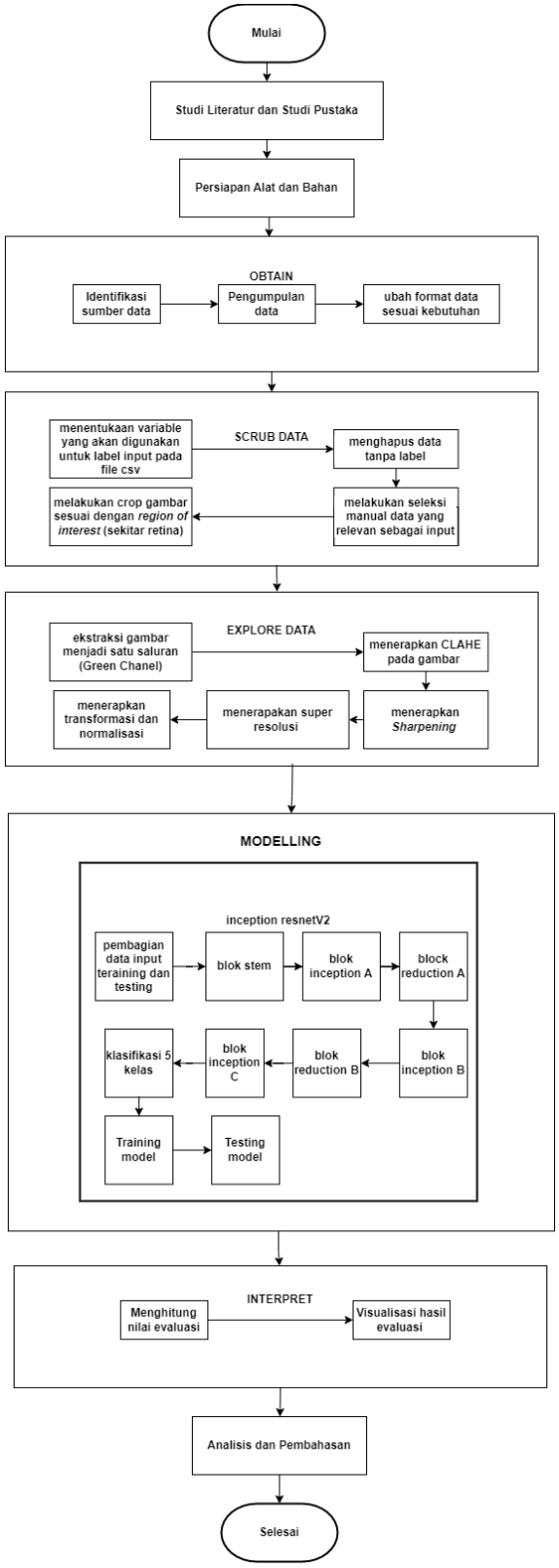
Bahan pada penelitian ini menggunakan data berupa citra fundus. Data yang digunakan yaitu dari database MESSIDOR-2. MESSIDOR-2 dataset merupakan penambahan citra dari MESSIDOR dataset. Total citra pada dataset MESSIDOR-2 yaitu 1748 citra dengan 690 citra dalam format JPG dan 1058 citra dalam format PNG.

3.3 Alur Penelitian

Alur kerja atau tahapan-tahapan dalam penelitian ini dimulai dengan melakukan studi pustaka dan studi literatur. Tujuan dari tahap ini adalah untuk mengumpulkan informasi dan referensi yang relevan dengan topik penelitian yang sedang dilakukan. Dengan cara ini, penulis dapat memperoleh pengetahuan yang mendukung pelaksanaan penelitian. Setelah itu, dilakukan persiapan alat dan bahan yang diperlukan untuk mendukung proses penelitian.

Penelitian ini menggunakan proses *data mining* dengan mengikuti kerangka kerja data science OSEMN. Kerangka ini menjelaskan langkah-langkah sistematis dan terstruktur dalam membangun sebuah model data science. Gambar 3.1 menunjukkan alur kerja penelitian, yang dimulai dari studi literatur dan studi pustaka hingga tahap interpretasi dalam framework OSEMN.

.



Gambar 3. 1.Alur Penelitian

Berdasarkan gambar 3.1 tahapan pada proses *data mining* menggunakan *framework*

OSEMN yang berisikan tahapan-tahapan pengerjaan model *data science* dengan lebih terperinci yang akan dijelaskan dibawah ini

3.3.1. *Obtain*

Pada tahap ini dilakukan pengumpulan data sesuai dengan tujuan, yaitu data data berupa citra fundus. Data yang digunakan yaitu dari database MESSIDOR-2. MESSIDOR-2 dataset merupakan penambahan citra dari MESSIDOR dataset. Messidor-Original terdiri dari semua pasang citra dari dataset Messidor asli, yaitu 529 pemeriksaan (1058 citra, disimpan dalam format PNG). Kemudian *Ophthalmology department of Brest University Hospital(France)* menambahkan data pasien antara Oktober 2009-September 2010 yang mana Messidor-Extension yang diambil tanpa pelebaran farmakologis dan dengan menggunakan kamera fundus *non-mydriatic Topcon TRC NW6* dengan bidang pandang 45 derajat. Hanya citra yang berpusat pada area macula yang dimasukkan dalam kumpulan data Messidor-Extension yaitu 345 pemeriksaan (690 citra, dalam format JPG). Total citra pada dataset MESSIDOR-2 yaitu dari 1748. Selain data citra, dilengkapi juga dengan data berupa file dengan format *Comma Separated Values* (CSV).

File CSV berisi informasi tentang citra mata dari dataset Messidor 2. Setiap citra diberi nilai berdasarkan tingkat keparahan penyakit mata (DR) dan keberadaan DME (penyakit mata lain). Nilai-nilai ini ditentukan oleh para ahli dan digunakan untuk melatih model komputer yang lebih akurat dalam mendiagnosis penyakit mata. File CSV berisi penilaian Tingkat keparahan DR dari *International Classification of Diabetic Retinopathy* (ICDR) yang dibagi menjadi 5 poin (kelas) dengan ciri – ciri keparahan merujuk pada Tabel 2. 1 Tipe Diabetic retinopathy dan ciri klinis, dengan tambahan 1 kelas yaitu kelas 0 yang merupakan data pasein tidak menderita DR. Terdapat pula penilaian *Diabetic Macular Edema*(DME) yang merupakan data pasien medapat rujukan, pada dataset Messidor 2 ini. Setiap baris dalam CSV mewakili satu citra, dengan 4 kolom berlabel:

1. *image_id*, kolom ini berisi nama file citra seperti yang disediakan dalam dataset citra Messidor 2.
2. *adjudicated_dr_grade*, adalah nilai ICDR yang terdiri dari 5 poin tingkatan keparahan DR yang dapat dilihat pada tabel 3.3.

Tabel 3. 3. Tingkat Keparahan DR Berdasarkan ICDR

No	Kelas	Keterangan	Jumlah Data
1	0	Tidak menderita DR (none)	1017
2	1	DR Ringan	270
3	2	DR Sedang	347
4	3	DR Berat	75
5	4	PDR	35

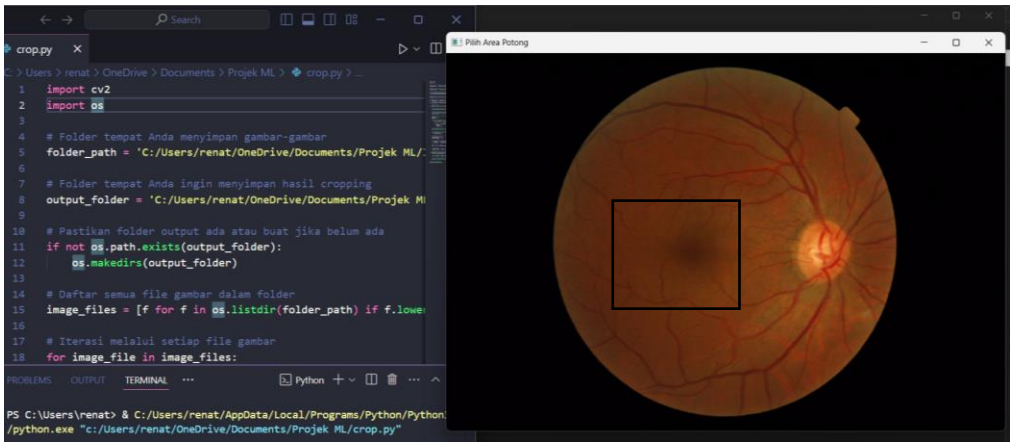
3. *adjudicated_dme*, menunjukan indikasi DME yang sudah menunjukan keadaan perlu dirujuk ke spesialis mata dikarenakan kondisi yang sudah cukup parah, yang didefinisikan oleh Eksudat Keras (endapan lemak yang terlihat pada retina dan sering menjadi tanda adanya DME) dalam jarak 1 Disc Diameter. Pada file CSV Label 0 pada kolom ini menunjukan Tidak ada DME yang menunjukan belum perlunya untuk dirujuk sedangkan label 1 menunjukan bahwa DME perlu dirujuk ke spesialis.
4. *adjudicated_gradable* adalah nilai dari kualitas citra. Label 0 berarti Tidak Dapat Dinilai, tidak ada nilai DR atau DME pada citra, sedangkan label 1 berarti Dapat Dinilai, baik DR maupun DME. Harap dicatat bahwa kolom adjudicated_dr_grade dan adjudicated_dme akan kosong untuk citra dengan adjudicated_gradable=0.

3.3.2. *Scrub Data*

Pada tahapan ini, dilakukan seleksi untuk menentukan variabel yang akan digunakan sebagai label pada citra *Input* sesuai tujuan penelitian yaitu berdasarkan tingkat keparahan DR. Berdasarkan *adjudicated_gradable* maka terdapat data yang tidak memiliki nilai DR sehingga data perlu diseleksi. Setelah melakukan seleksi tersebut selanjutnya kolom *adjudicated_gradable* dihapus bersamaan dengan kolom *adjudicated_dme* karena tidak termasuk dalam fokus pelnelitian dan berkemungkinan menimbulkan masalah pada proses *labelling* data. Seletah seleksi

variable, kemudian dilakukan seleksi manual berupa pengecekan tiap-tiap citra secara manual untuk menilai kelayakan citra bedasarkan tingkat kejelasan citra.

Setelah data *Input*an ditentukan, dilakukan *cropping* pada *region of interest (ROI)* yaitu daerah disekitar retina yang bertujuan agar model lebih mudah dalam mengenali citra. Proses *cropping* data dilakukan menggunakan bahasa pemrograman *Python* dengan *tools visual studio code* agar citra yang dihasilkan presisi dan sama ukurannya untuk tiap citra yaitu lebar 299 *pixel* dan tinggi 299 *pixel* . Contoh porses *cropping* yang dilakukan dapat dilihat pada gambar 3.2 berikut.



Gambar 3. 2 contoh proses *cropping* data pada ROI

3.3.3. *Explore Data*

Tahapan explorasi data dilakukan dengan dengan memperjelas pola atau tren dari dataset dilakukan pra-pemrosesan data dan *ekstraksi fitur* pada citra. Penggunaan 3 kanal RGB pada *input* dipertimbangkan untuk menguji apakah model lebih efisien tampa tambahan komputasi pada prapermrosesan citra *input*an dan apakan prapemrosesan citra *input*an lenoh lanjut diperlukan. Kanal hijau sering kali digunakan dalam pemrosesan gambar fundus mata karena memberikan kontras yang lebih baik untuk fitur-fitur penting dibandingkan dengan kanal lain. Sehingga dalam penelitian ini kanal tersebut pula yang diekstraksi kemudian di-*enhancement* sebelum dilakukan ekstraksi fitur. *Enhancement* citra dilakukan dengan meregangkan contrast menggunakan algoritma CLAHE.

CLAHE dipilih untuk digunakan karena teknik ini sangat efektif dalam

meningkatkan detail pada area dengan kontras rendah, seperti bayangan atau area yang overexposed. Dengan membatasi peningkatan kontras pada area homogen, CLAHE dapat mengurangi *noise* yang sering muncul pada teknik peningkatan kontras lainnya. Selain itu CLAHE juga menyesuaikan peningkatan kontras secara lokal untuk setiap blok citra, sehingga menghasilkan hasil yang lebih natural. Untuk visualisasi alur penerapan ekraksi fitur dan pra-pemrosesan citra dapat dilihat pada gambar 3.3.

Gambar 3. 3. Alur Pra-pemrosesan Citra

Namun demikian berdasarkan gambar 3.4 di atas, hasil citra dari hanya clahe berkemungkinan menghasilkan citra yang agak buram, sehingga pada penelitian ini dipertimbangkan untuk menambahkan proses pengolahan citra lainnya yaitu *Sharpening* dan/atau *Super Resolution. Sharpening Filter* **(***Filter* penguat tepi) digunakan untuk mengekstrak dan menonjolkan detail halus dari citra, serta meningkatkan detail yang buram. *Super-resolution* merujuk pada teknik yang digunakan untuk meningkatkan resolusi citra, sehingga menghasilkan citra yang lebih besar dan lebih jelas.

3.3.4. *Model*

Pada tahap *modeling* dilakukan pemodelan berdasarkan data yang telah dipersipakan pada tahap sebelumnya. Perancangan model menggunakan model *deep learning* yaitu arsitektur CNN dan model *machine learning* yaitu metode *InceptionResnetV2* . *CNN InceptionResNet-V2* memungkinkan ekstraksi fitur pada berbagai skala sehingga membantu model mengenali objek dengan lebih baik, terutama pada citra yang kompleks sehingga model ini dapat mengatasi masalah *vanishing gradient* yang sering terjadi pada jaringan yang sangat dalam tanpa

mengorbankan kinerja. Fokus utama perbedaan sistem pengenalan tingkatan DR menggunakan hanya model *CNN InceptionResNet-V2* untuk mengenali 5 kelas sekaligus dan disertai beberapa peningkatan pada *Input* citra (*prapemrosessan images*).

Metode tersebut diimplementasikan menggunakan bahasa pemrograman *Python* dengan *framework Pytorch*. Sebelum melakukan pelatihan model, dilakukan *splitting* data atau pembagian data menjadi dua bagian yakni data untuk pelatihan dan juga pengujian model dengan perbandingan 80:20. Selain itu juga dilakukan proses *hypertunning* untuk pemperoleh parameter model *InceptionResNet-V2* terbaik.

3.3.5. *Intepreter*

Tahapan interpretasi adalah tahap terakhir bertujuan untuk menginterpretasikan hasil pengujian model yang telah dibangun. Model yang diuji meliputi model dengan 3 pendekatan citra masuka. Pertama, citra dengan 3 kanal RGB, kemudian citra dengan pengambilan kanal hijau murni, dan terakhir citra dengan pendekatan pengambilan kanal hijau RGB tiruan. Pada citra *input*an dengan kanal hijau diteapkan juga praproses tambahan seperti CLAHE, *sharppenig,* dan *super resolusi.* Hasil pengujian model berupa visualisasi grafik kinerja model. Selain itu digunakan metrik evaluasi model untuk mengukur kesalahan pada prediksi yang dilakukan. Hasil evaluasi tersebut berupa nilai *sensitivitas (recall), spesifisitas, akurasi, Area Under the ROC Curve (AUC), F1 score* dan waktu komputasi. Dengan melakukan evaluasi ini diharapkan dapat memberikan citraan komprehensif tentang performa relatif antara model *CNN InceptionResNet-V2* dan gabungan antara model *CNN InceptionResNet-V2* dan *DenseNet* dalam pendeteksian penyakit *Diabetic retinophaty* berdasarkan citra fundus.

V. KESIMPULAN DAN SARAN

5.1 Kesimpulan

kesimpulan yang diperoleh dari penelitian pendeteksian penyakit *diabetic retinophaty* melalui citra fundus menggunakan model *cnn Inceptionresnet-v2* adalah sebagai berikut .

1. Model pendeteksian penyakit *diabetic retinophaty* melalui citra fundus menggunakan model *cnn Inceptionresnet-v2* untuk mendeteksi 5 kelas behasil dibangun. Model mencapai hasil yang sangat baik dengan akurasi pelatihan mencapai 100% dan 96% pada pengujian.
2. *Input* citra dengan Kanal Hijau yang juga disertai penambahan praproses citra CLAHE, *Sharpening* dan atau *Super Resolution* secara cukup signifikan mempengaruhi kinerja model dibandingkan dengan menggunakan 3 kanal RGB. Hal ini tercermin pada nilai akurasi pelatihan *input* dengan 3 kanal RGB hanya pada 86% sedangkan model yang menggunakan Kanal Hijau memiliki akurasi 93-96%.
3. Berdasarkan *confusion marix*, pengambilan Kanal Hijau dengan RGB tiruan lebih unggul pada keseimbangan pendeteksian, terutama saat model diuji dengan data yang belum pernah dilihat pada pelatihan.
4. Model yang menggunakan pendekatan pengambilan Kanal Hijau RGB tiruan disertai penambahan CLAHE dan *Sharpening* memberikan hasil Akurasi pelatihan 100% dan 96% pada pengujian dengan dengan *F1 score* 95.91% menjadi kinerja terbaik secara keseluruhan. Dibandingkan dengan kinerja model dengan prapemrosesan lain yang memiliki hasil akurasi pengujian 95% dengan

Sharpening dan *Super Resolution* ,94% dengan hanya *Super Resolution* , dan 94% dengan hanya CLAHE

5.2 Saran

Saran penelitian selanjutnya berdasarkan penelitian yang telah dilakukan adalah sebagai berikut.

1. Melakukan eksplorasi lebih luas pada penggunaan paramerter pada CLAHE, *Sharpening*, dan *Super Resolution* untuk lebih mengoptimalkan kinerja model dan beban komputasi.
2. Mempertimbangkan penggunaan penambahan pra-pemrosesan citra seperti pem*Filter*an Gabor untuk deteksi *pixel* frekuensi tertentu untuk memperjelas garis syaraf pada citra fundus sebelum dimasukan pada model.
3. Membuat deployment lewat aplikasi ataupun website dengan model yang terbaik pada penelitian ini.

DAFTAR PUSTAKA

[1] C. J. Flaxel, R. A. Adelman, S. T. Bailey, A. Fawzi, J. I. Lim, G. A. Vemulakonda, and G. Ying, "Diabetic Retinopathy Preferred Practice Pattern®," *Ophthalmology*, vol. 127, no. 1, pp. P66–P145, Jan. 2020, doi: 10.1016/j.ophtha.2019.09.025.

[2] WHO South-East Asia, "Strengthening diagnosis and treatment of diabetic retinopathy in the south-east asia region." 2020.

[3] M. D. Abramoff, M. K. Garvin, and M. Sonka, "Retinal Imaging and Image Analysis," *IEEE Rev. Biomed. Eng.*, vol. 3, pp. 169–208, 2010, doi: 10.1109/RBME.2010.2084567.

[4] M. S. Miri and A. Mahloojifar, "Retinal Image Analysis Using Curvelet Transform and Multistructure Elements Morphology by Reconstruction," *IEEE Trans. Biomed. Eng.*, vol. 58, no. 5, pp. 1183–1192, May 2011, doi: 10.1109/TBME.2010.2097599.

[5] R. Pires, H. F. Jelinek, J. Wainer, and A. Rocha, "Retinal Image Quality Analysis for Automatic Diabetic Retinopathy Detection," in *2012 25th SIBGRAPI Conference on Graphics, Patterns and Images*, Ouro Preto, Brazil: IEEE, Aug. 2012, pp. 229–236. doi: 10.1109/SIBGRAPI.2012.39.

[6] N. Patton, T. M. Aslam, T. MacGillivray, I. J. Deary, B. Dhillon, R. H. Eikelboom, K. Yogesan, and I. J. Constable, "Retinal image analysis: Concepts, applications and potential," *Progress in Retinal and Eye Research*, vol. 25, no. 1, pp. 99–127, Jan. 2006, doi: 10.1016/j.preteyeres.2005.07.001.

[7] H. A. Nugroho, T. Yulianti, N. A. Setiawan, and D. A. Dharmawan, "Contrast measurement for no-reference retinal image quality assessment," in *2014 6th International Conference on Information Technology and Electrical Engineering (ICITEE)*, Yogyakarta, Indonesia: IEEE, Oct. 2014, pp. 1–4. doi: 10.1109/ICITEED.2014.7007902.

[8] S. Marshall, *Diabetes Mellitus: A Practical Handbook*. Class Health Publishing, 2015.

[9] A. R. Shah and T. W. Gardner, "Diabetic retinopathy: research to clinical practice," *Clin Diabetes Endocrinol*, vol. 3, no. 1, p. 9, Dec. 2017, doi: 10.1186/s40842-017-0047-y.

[10] A. A. Alghadyan, "Diabetic retinopathy – An update," *Saudi Journal of Ophthalmology*, vol. 25, no. 2, pp. 99–111, Apr. 2011, doi:

10.1016/j.sjopt.2011.01.009.

[11] M. R. K. Mookiah, U. R. Acharya, C. K. Chua, C. M. Lim, E. Y. K. Ng, and A. Laude, "Computer-aided diagnosis of diabetic retinopathy: A review," *Computers in Biology and Medicine*, vol. 43, no. 12, pp. 2136–2155, Dec. 2013, doi: 10.1016/j.compbiomed.2013.10.007.

[12] American Optometric Association (AOA) Evidence-Based Optometry Committee, *Eye Care Of The Patient With Diabetes Mellitus Second Edition*, 2nd ed. American Optometric Association, 2019.

[13] S. Patel, M. Lohakare, S. Prajapati, S. Singh, and N. Patel, "DiaRet: A Browser-Based Application for the Grading of Diabetic Retinopathy with Integrated Gradients," in *2021 IEEE International Conference on Robotics, Automation and Artificial Intelligence (RAAI)*, Hong Kong, Hong Kong: IEEE, Apr. 2021, pp. 19–23. doi: 10.1109/RAAI52226.2021.9507938.

[14] N. eindert, "Automatic Detection of Diabetic Retinopathy in Digital Fundus Photographs," Thesis, University Medical CenterUtrecht, Uthrect, The Netherlands, 2006.

[15] Jamaaluddin and indah Sulistyowati, *Buku Ajar Mata Kuliah Kecerdasan Buatan (Artificial Intelligence)*. UMSIDA Press, 2021.

[16] O. G. Yalçın, "Image Classification in 10 Minutes with MNIST Dataset," MEDIUM. Accessed: Nov. 06, 2024. [Online]. Available: https://towardsdatascience.com/image-classification-in-10-minutes-with-mnist-dataset

[17] H. Kinsley and D. Kukieła, "Neural Networks from Scratch in Python".

[18] W. Setiawan, *Deep Learning menggunakan Convolutional Neural Network Teori dan Aplikasi*. Media Nusa Creative (MNC Publishing), 2021.

[19] A. Santoso and G. Ariyanto, "Implementasi Deep Learning berbasis Keras untuk Pengenalan Wajah," *emitor*, vol. 18, no. 1, pp. 15–21, Jun. 2018, doi: 10.23917/emitor.v18i01.6235.

[20] A. Waasiu, "Implementasi Arsitektur Inception Resnet-V2 Dalam Diagnosis Penyakit Parkinson Berbasis Android," Universitas Hasanuddin, Makasar, 2023.

[21] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016.

[22] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology (ICET)*, Antalya: IEEE, Aug. 2017, pp. 1–6. doi: 10.1109/ICEngTechnol.2017.8308186.

[23] M. F. Dzulqarnain, S. Suprapto, and F. Makhrus, "Improvement of Convolutional Neural Network Accuracy on Salak Classification Based Quality on Digital Image," *Indonesian J. Comput. Cybern. Syst.*, vol. 13, no. 2, p. 189, Apr. 2019, doi: 10.22146/ijccs.42036.

[24] Y. Zhang and X. Li, "Fast Convolutional Neural Networks with Fine-Grained FFTs," in *Proceedings of the ACM International Conference on Parallel Architectures and Compilation Techniques*, Virtual Event GA USA: ACM, Sep. 2020, pp. 255–265. doi: 10.1145/3410463.3414642.

[25] F. Sulianta, *Basic Data Mining from A to Z*. 2023.

[26] A. Yusuf, R. C. Wihandika, and C. Dewi, "Klasifikasi Emosi Berdasarkan Ciri Wajah Menggunakan Convolutional Neural Network".

[27] J. Karmeshu, Ed., *Entropy measures, maximum entropy principle, and emerging applications*. in Studies in fuzziness and soft computing, no. v. 119. Berlin Heidelberg New York Hong Kong London Milano Paris Tokyo: Springer, 2003.

[28] Golnaz. Hossein, "Beginner Tutorial: Image Classification Using Pytorch," MEDIUM. Accessed: Sep. 06, 2024. [Online]. Available: https://medium.com/@golnaz.hosseini/beginner-tutorial-image-classification-using-pytorch

[29] S. Hesaraki, "Feature Map," MEDIUM. Accessed: Nov. 07, 2024. [Online]. Available: https://medium.com/@saba99/feature-map-35ba7e6c689e

[30] S. Bianco, R. Cadene, L. Celona, and P. Napoletano, "Benchmark Analysis of Representative Deep Neural Network Architectures," *IEEE Access*, vol. 6, pp. 64270–64277, 2018, doi: 10.1109/ACCESS.2018.2877890.

[31] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," *AAAI*, vol. 31, no. 1, Feb. 2017, doi: 10.1609/aaai.v31i1.11231.

[32] S. Bhure, "Concatenation Operation in CNN," Concatenation Operation in CNN. Accessed: Dec. 30, 2024. [Online]. Available: https://iq.opengenus.org/concatenation-operation-in-cnn/

[33] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Jan. 29, 2017, *arXiv*: arXiv:1412.6980. Accessed: Jul. 29, 2024. [Online]. Available: http://arxiv.org/abs/1412.6980

[34] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks".

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," 2015, *arXiv*. doi: 10.48550/ARXIV.1502.01852.

[36] X. Chen, K. B. Letaief, and K. Huang, "On the View-and-Channel Aggregation Gain in Integrated Sensing and Edge AI," *IEEE J. Select. Areas Commun.*, vol. 42, no. 9, pp. 2292–2305, Sep. 2024, doi: 10.1109/JSAC.2024.3413988.

[37] A. Sharma, S. Shinde, I. I. Shaikh, M. Vyas, and S. Rani, "Machine Learning Approach for Detection of Diabetic Retinopathy with Improved Pre-Processing," in *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, Greater Noida, India: IEEE, Feb. 2021, pp. 517–522. doi: 10.1109/ICCCIS51004.2021.9397115.

[38] P. Joshi, *OpenCV with Python by example: build real-world computer vision applications and develop cool demos using OpenCV for Python*. Birmingham: Packt Publishing, 2015.

[39] K. Najarian and R. Splinter, Eds., *Biomedical signal and image processing*. Boca Raton: CRC/Taylor & Francis, 2006.

[40] B. C. Tom, N. P. Galatsanos, and A. K. Katsaggelos, "Reconstruction of a High Resolution Image from Multiple Low Resolution Images," in *Super-Resolution Imaging*, vol. 632, S. Chaudhuri, Ed., in The International Series in Engineering and Computer Science, vol. 632. , Boston: Kluwer Academic Publishers, 2002, pp. 73–105. doi: 10.1007/0-306-47004-7_4.

[41] N. Hotz, "OSEMN Data Science Life Cycle," Data Science Process Alliance. Accessed: Jun. 06, 2024. [Online]. Available: https://www.datascience-pm.com/osemn/

[42] A. Fernández Villán, *Mastering OpenCV 4 with Python: a practical guide covering topics from image processing, augmented reality to deep learning with OpenCV 4 and Python 3.7*. Birmingham, UK: Packt Publishing, 2019.

[43] K. Jolly, *Machine Learning with scikit-learn quick start guide: classification, regression, and clustering techniques in Python*. Birmingham, UK: Packt Publishing, 2018.

[44] A. R. JHA, *Mastering PyTorch- build powerful deep learning architectures using advanced pytorch features*. S.l.: Packt Publishing Limited, 2023.

[45] J. Pardede and D. A. L. Putra, "Implementasi DenseNet Untuk Mengidentifikasi Kanker Kulit Melanoma," *JuTISI*, vol. 6, no. 3, Dec. 2020, doi: 10.28932/jutisi.v6i3.2814.

[46] S. Wardani, Sawaluddin, and P. Sihombing, "Hybrid of Support Vector Machine Algorithm and K-Nearest Neighbor Algorithm to Optimize the Diagnosis of Eye Disease," in *2020 3rd International Conference on Mechanical, Electronics, Computer, and Industrial Technology (MECnIT)*, Medan, Indonesia: IEEE, Jun. 2020, pp. 321–326. doi: 10.1109/MECnIT48290.2020.9166599.

[47] L. Sarker, Md. M. Islam, T. Hannan, and Z. Ahmed, "COVID-DenseNet: A Deep Learning Architecture to Detect COVID-19 from Chest Radiology Images," May 09, 2020. doi: 10.20944/preprints202005.0151.v1.

[48] I. Odeh, M. Alkasassbeh, and M. Alauthman, "Diabetic Retinopathy Detection using Ensemble Machine Learning," in *2021 International Conference on Information Technology (ICIT)*, Amman, Jordan: IEEE, Jul. 2021, pp. 173–178. doi: 10.1109/ICIT52682.2021.9491645.

[49] V. N. Joshi, M. R. Gujar, S. R. Chaudhary, S. P. Paranjape, and J. Wagh, "Diabetic Retinopathy Detection using Convolutional Neural Networks," *IJRASET*, vol. 10, no. 5, pp. 2971–2975, May 2022, doi: 10.22214/ijraset.2022.43006.

[50] I. Bakti and M. Firdaus, "Arsitektur Convolutional Neural Network InceptionResNet-V2 Untuk Pengelompokan Pneumonia Chest X-Ray,"

Jurnal komputer dan teknologi, vol. 01, no. 02, Jan. 2023, doi: 10.58290/jukomtek. v1i2.66.

[51] R. Nandakumar, P. Saranya, V. Ponnusamy, S. Hazra, and A. Gupta, "Detection of Diabetic Retinopathy from Retinal Images Using DenseNet Models," *Computer Systems Science and Engineering*, vol. 45, no. 1, pp. 279–292, 2023, doi: 10.32604/csse.2023.028703.

[52] G. H. Vardhan, M. V. S. Jyoshna, P. K. Viswanath, S. Zubayr, and V. Sravanth, "Diabetic Retinopathy Detection Using InceptionResnet-V2 and Densenet121," *JIPIRS*, no. 42, pp. 30–40, Feb. 2024, doi: 10.55529/jipirs.42.30.40.

[53] Evolucare group, "Computer Vision experts in Image Acquisition / Processing / Artificial Intelligence," Messidor. Accessed: Jul. 07, 2024. [Online]. Available: https://www.adcis.net/en/third-party/messidor/