RANCANG BANGUN APLIKASI *MOBILE* KLASIFIKASI KANKER KULIT DENGAN PEMILIHAN MODEL *TRANSFER LEARNING*

SKRIPSI

AJRANA

H071181003

Pembimbing Utama	**:**	**Dr. Eng. Armin Lawi, S.Si., M.Eng.**
Pembimbing Pendamping	**:**	**A. Muh. Amil Siddik, S.Si., M.Si.**
Penguji	**:**	**1. Andi Muhammad Anwar, S.Si., M.Si.**
		2. Rozalina Amran, S.T., M.Eng.

PROGRAM STUDI SISTEM INFORMASI

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2022

RANCANG BANGUN APLIKASI *MOBILE* KLASIFIKASI KANKER KULIT DENGAN PEMILIHAN MODEL *TRANSFER LEARNING*

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer pada Program Studi Sistem Informasi Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin

AJRANA

H071181003

PROGRAM STUDI SISTEM INFORMASI

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN

ALAM UNIVERSITAS HASANUDDIN

AGUSTUS 2022

HALAMAN PERNYATAAN KEOTENTIKAN

Yang bertanda tangan dibawah ini

Nama : Ajrana

NIM : H071181003

Program Studi : Sistem Informasi

Jenjang : S1

Menyatakan dengan ini bahwa karya tulisan saya berjudul :

RANCANG BANGUN APLIKASI *MOBILE* KLASIFIKASI KANKER KULIT DENGAN PEMILIHAN MODEL *TRANSFER LEARNING*

Adalah karya tulisan saya sendiri, bukan merupakan pengambil alihan tulisan orang lain dan bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri.

Apabila dikemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan isi skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Makassar, 26 Agustus 2022

Ajrana

H071181003

RANCANG BANGUN APLIKASI MOBILE KLASIFIKASI KANKER KULIT DENGAN PEMILIHAN MODEL *TRANSFER LEARNING*

Disusun dan diajukan oleh :

AJRANA

H071181003

Telah dipertahankan dihadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana pada Program Studi Sistem Informasi Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin dan dinyatakan telah memenuhi syarat kelulusan.

Pembimbing Utama	Pembimbing Pertama
Dr. Eng. Armin Lawi, S.Si., M.Eng.	A. Muh Amil Siddik S. Si,. M.Si.
NIP.197204231995121001	NIP. 199102242018016001

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh	:
Nama	: Ajrana
NIM	: H071181003
Program Studi	: Sistem Informasi
Judul Skripsi	: Rancang Bangun Aplikasi *Mobile* Klasifikasi Kanker Kulit Dengan Pemilihan Model *Transfer Learning*.

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Komputer pada Program Studi Sistem Informasi Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin.

DEWAN PENGUJI

Tanda Tangan

1. Ketua: Dr. Eng. Armin Lawi, S.Si., M.Eng. (.....)
2. Sekretaris: A. Muh Amil Siddik, S.Si., M.Si. (.....)
3. Anggota: A. Muhammad Anwar, S.Si., M.Si. (.....)
4. Anggota: Rozalina Amran, S.T., M.Eng. (.....)

Ditetapkan di : Makassar
Tanggal : 26 Agustus 2022

KATA PENGANTAR

Segala puji bagi Allah *Subhanahu Wa ta'ala*, Tuhan atas langit dan bumi beserta segala isinya. Karena, berkat nikmat dan karunianya sehingga penulisan skripsi ini dapat terselesaikan. Shalawat serta salam semoga senantiasa tercurahakan kepada Baginda *Rasulullah* Muhammad *Shallallahu Alaihi Wasallam* dan kepada para keluarga serta sahabat beliau, yang senantiasa menjadi teladan yang baik.

Alhamdulillah, skripsi dengan judul "Rancang Bangun Aplikasi *Mobile* Klasifikasi Kanker Kulit Dengan Pemilihan Model *Transfer Learning*" yang disusun sebagai salah satu syarat akademik untuk meraih gelar Sarjana Komputer pada Program Studi Sistem Informasi Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin ini dapat dirampungkan. Tentunya, dalam penulisan skripsi ini, penulis mampu melewati berbagai hambatan dan masalah berkat bantuan moril dan materil, serta dorongan dari berbagai pihak. Oleh karena itu, penulis ingin menyampaikan ucapan terima kasih yang tak terhingga kepada orang tua penulis, Ibunda **DARJATIH** dan Ayahanda **BAKRIE**, sebagai tempat kembali setelah pergi, terima kasih atas kasih sayang, doa, dan nasihat yang tulus sebagai bekal kehidupan. Rasa terima kasih juga penulis tujukan kepada saudara(i) tercinta yang telah menjadi motivator, dan rival dalam membanggakan kedua orang tua, terima kasih atas dukungan yang penulis dapatkan selama ini.

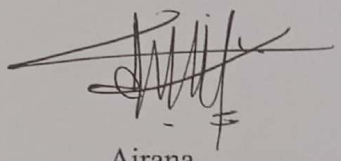
Penghargaan dan ucapan terima kasih dengan penuh ketulusan juga penulis ucapkan kepada:

1. Rektor Universitas Hasanuddin Makassar **Prof. Dr. Ir. Jamaluddin Jompa, M.Sc** dan seluruh Wakil Rektor dalam Lingkungan Universitas Hasanuddin.
2. Bapak Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam **Dr. Eng Amiruddin** dan para Wakil Dekan serta seluruh staf yang telah memberikan bantuan selama penulis mengikuti pendidikan di FMIPA Universitas Hasanuddin.
3. Bapak **Dr. Nurdin, S.Si. M.Si**, sebagai Ketua Departemen Matematika FMIPA Unhas. Penulis juga berterima kasih atas dedikasi dosen-dosen pengajar, serta staf Departemen atas ilmu dan bantuan yang bermanfaat.
4. Bapak **Dr. Muhammad Hasbi, M.Sc** sebagai Ketua Program Studi Sistem Informasi Universitas Hasanuddin.

5. Bapak **Dr. Eng. Armin Lawi, S.Si., M.Eng**., dan Bapak **A. Muh Amil Siddik S. Si,. M.Si** sebagai dosen pembimbing utama dan dosen pembimbing pertama atas ilmu yang beliau berikan selama proses perkuliahan, dan kesediaan beliau dalam membimbing, serta memotivasi penulis dalam penyusunan skripsi ini.
6. Bapak **A. Muhammad Anwar, S.Si., M.Si** dan ibu **Rozalina Amran, S.T., M.Eng** sebagai dosen penguji pertama dan penguji kedua atas ilmu yang beliau berikan selama proses perkuliahan, dan saran serta masukan yang telah beliau berikan dalam penyusunan skripsi ini.
7. Kepada saudara(i) ku **Cecilia**, **Islah**, **Fuad**, **Raynaldi**, **Nasrullah**, **Maxi** yang telah senantiasa membantu penulis dalam penyusunan skripsi. Juga kepada saudara **djihad** dan **khoir** yang telah menjadi *partner* tugas matakuliah mulai dari semester 3 sampai dengan semester 6. Semangat dan motivasi juga di berikan oleh saudara(i) ku **Sari**, **Aviva**, **Nyssa**, **Iksan** serta teman-teman **Sistem Informasi Unhas 2018** terimakasih atas kebersamaan, kepedulian, suka-duka, canda tawa yang telah kita lewati selama ini.
8. Kepada **kak Fitrah** dan **kak Khaiz** yang telah meluangkan waktu dalam membagi ilmu. Terimakasih atas segala ilmu yang telah diberikan.
9. Saudara(i) ku Sekampoeng Club (**Yusuf, Dandi, Siska**) terimakasih telah saling menguatkan, Semoga kesuksesan selalu kita dapatkan dalam setiap langkah-langkah kita.
10. Kepada teman penyusur pelosok **Amalia Wulan Purnama** serta kakak-kakak hebat **TAPUKAREN 38** dan **REFORMASI 32**, terimakasih atas pengalaman berharga yang diberikan kepada penulis.
11. Keluarga Besar **Sikola Inspirasi Alam**, **Sokola Kaki Langit** dan **Komunitas Oana Cendekia** yang telah mewadahi penulis dalam mengembangkan diri.
12. Semua pihak yang telah memberikan bantuan kepada penulis baik berupa materi dan non materi yang tidak dapat penulis sebutkan satu per satu, terima kasih untuk bantuan dan dukungannya.

Penulis menyadari bahwa skripsi ini masih jauh dari kesempurnaan karena keterbatasan penulis. Oleh karena itu, saran dan kritik demi penyempurnaan skripsi ini sangat penulis harapkan. Akhir kata, semoga skripsi ini membawa manfaat dan semoga Allah Subehanahu Wata'ala membalas semua kebaikan semua pihak yang telah membantu.

Makassar, Agustus 2022



Ajrana

PERNYATAAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Hasanuddin, saya yang bertanda tangan di bawah ini:

Nama	: Ajrana
NIM	: H071181003
Program Studi	: Sistem Informasi
Fakultas	: Matematika dan Ilmu Pengetahuan Alam
Jenis Karya	: Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Hasanuddin Hak Bebas Royalti Non Eksklusif (Non-exclusive Royalty Free Right) atas karya ilmiah saya yang berjudul:

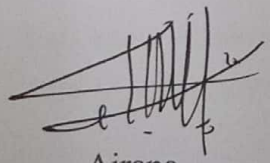
"Rancang Bangun Aplikasi Mobile Klasifikasi Kanker Kulit Dengan Pemilihan Model Transfer Learning"

beserta perangkat yang ada (jika diperlukan). Terkait dengan hal di atas, maka pihak universitas berhak menyimpan, mengalih-media/format-kan, mengelola dalam bentuk pangkalan data (database), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di Makassar pada tanggal 26 Agustus 2022

Yang menyatakan



Ajrana

ABSTRAK

Kulit merupakan lapisan tubuh manusia yang sangat luas dan berfungsi untuk menutupi seluruh permukaan pada tubuh manusia. Kulit yang tidak terawat akan menimbulkan berbagai penyakit dan gangguan pada kulit diantara yaitu Kanker kulit. Dalam mendiagnosis penyakit kanker kulit digunakan metode biopsi, Namun terdapat beberapa kekurangan biopsi diantaranya yaitu butuh persiapan yang panjang, waktu penyembuhan luka yang sedikit lama dan biaya yang mahal. Metode *Deep Learning* yang saat ini memiliki hasil paling signifikan dalam pengenalan citra adalah *Convolutional Neural Network* (CNN). Dalam penelitian dilakukan pengklasifikasian kanker kulit dengan metode CNN dengan data yang digunakan merupakan data 9 kelas kanker kulit yaitu *actinic keratosis, basal cell carcinoma, dermatofibroma, Healty Skin, melanoma, nevus, pigmented benign keratosis, seborrheic keratosis, squamous cell carcinoma, vascular lesion* serta 1 kelas kulit sehat (*healty skin*). Dalam membangun model klasifikasi penyakit kanker kulit digunakan kerangka kerja pemilihan model transfer learning. Dimana terdapat tiga model arsitektur yang digunakan yaitu *VGG16*, *DenseNet121* dan *NASNetMobile*. *hyperparameter* yang digunakan pada masing-masing model antara lain *learning rate* sebesar 0.0001, *batch size* sebesar 64, dan *epoch* sebanyak 100 kali. Dari ketiga model yang digunakan, model *VGG16* mendapat hasil akurasi tertinggi. hasil akurasi data *train* pada model arsitektur *VGG16* yaitu sebesar 98%, sedangkan hasil untuk data *test* sebesar 85%. Kemudian untuk *DenseNet121* menghasilkan nilai akurasi sebesar 99% untuk data *train* dan 82% untuk data *test*. Selanjutnya untuk model arsitektur *NASNetMobile* menghasilkan nilai akurasi pada data *train* sebesar 96% dan 68% untuk data *test*. Model klasifikasi yang di *deploy* menggunakan *tensorflow lite* pada aplikasi android yaitu *VGG16*.

Kata kunci: Kanker Kulit, *Convolutional Neural Network* (CNN), *Transfer Learning, VGG16, DenseNet121, NASNetMobile, deployment* Model.

ABSTRACT

Skin is a very broad layer of the human body and serves to cover the entire surface of the human body. Untreated skin will cause various diseases and disorders of the skin, including skin cancer. In diagnosing skin cancer, the method that used is biopsy. However, there are several disadvantages of a biopsy, including need for long preparation, rather longer recovery time and high cost. The Deep Learning that currently has the most significant results in image recognition is the Convolutional Neural Network (CNN). In this study, the CNN method was used with data used for 9 classes of skin cancer, namely actinic keratosis, basal cell carcinoma, dermatofibroma, healthy skin, melanoma, nevus, pigmented benign keratosis, seborrheic keratosis, squamous cell carcinoma, vascular lesion and 1 healthy skin class. In developing a skin cancer classification model, transfer learning model selection frameworks is used. Where there are three architectural models used, that is VGG16, DenseNet121 and NASNetMobile. The hyperparameters used in each model include a learning rate of 0.0001, a batch size of 64, and epochs of 100 times. By the three models used, the VGG16 got the highest accuracy results of the train data accuracy on the VGG16 are 98%, while the results for the test data are 85%. Then DenseNet121 produces an accuracy value of 99% for train data and 82% for test data. Furthermore, for the NASNetMobile data are train 96% and 68% for the test. The classification model deployed using tensorflow lite on the android application is VGG16.

Keywords: Skin Cancer, Convolutional Neural Network (CNN), Transfer Learning, VGG16, DenseNet121, NASNetMobile, deployment model.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Kulit merupakan lapisan tubuh manusia yang sangat luas dan berfungsi untuk menutupi seluruh permukaan pada tubuh manusia. Kulit merupakan bagian pertama yang dapat menerima sentuhan, rasa sakit dan pengaruh lainnya (Hendaria dkk., 2015). Oleh karena itu, mengingat pentingnya kulit sebagai pelindung organ tubuh, maka penting sekali untuk menjaga kesehatan kulit sejak usia dini. Kulit yang tidak terawat akan menimbulkan berbagai penyakit dan gangguan pada kulit diantara yaitu Kanker kulit.

Kanker kulit merupakan salah satu kanker yang umum didiagnosis di seluruh dunia, terutama pada orang-orang berkulit putih, insiden dan kematian terus meningkat selama dekade terakhir. Insiden kanker kulit di Amerika tercatat 4,9 juta kasus pada tahun 2007-2011. Di Indonesia sendiri, kanker kulit menempati urutan ketiga setelah kanker rahim dan kanker payudara. Kanker kulit dijumpai 5,9 – 7,8 % dari semua jenis kanker per tahun. Kanker kulit yang paling sering di jumpai di Indonesia adalah *karsinoma sel basal* (65,5%), lalu diikuti *karsinoma sel skuamosa* (23%), *melanoma maligna* (7,9%) dan kanker kulit lainnya. Jenis kanker kulit yang paling berbahaya adalah *melanoma*, dengan memiliki tingkat kematian yang tinggi, terutama jika tidak terdeteksi dini. Kanker kulit *non-melanoma* (NMS Cs), seperti *karsinoma sel basal* dan *karsinoma sel skuamosa* lebih umum tetapi metastasisnya kurang, dan hanya sebagian kecil yang mengarah ke kematian (Wilvestra dkk., 2018).

Terdapat berbagai faktor penyebab timbulnya kanker kulit yaitu diantaranya Faktor peningkatan radiasi sinar *ultraviolet*, faktor genetik, pola hidup yang tidak sehat, dan infeksi human *papillomavirus* (Hendaria dkk., 2015). Penentuan penyakit kulit tidak boleh dilakukan secara sembarangan, karena penyakit kulit bisa sangat berbahaya bila terjadi kesalahan dalam perawatan dan penanganannya (Nuraeni dkk., 2016). Maka melalui fakta tersebut, deteksi dini merupakan salah satu cara untuk mengatasi penyakit kanker kulit. Akan tetapi, pakar dermatologis mengatakan bahwa adanya kesulitan dalam membedakan antara luka bakar dan tahi lalat. Dokter dermatologis dapat mendiagnosis kanker kulit dengan melalui proses biopsi. Biopsi adalah pengambilan sejumlah kecil jaringan tubuh manusia untuk pemeriksaan

laboratorium yang bertujuan untuk mendeteksi adanya suatu penyakit (Wardhani, 2010). Namun, terdapat beberapa kekurangan biopsi diantaranya yaitu butuh persiapan yang panjang, waktu penyembuhan luka yang sedikit lama dan biaya yang mahal. Proses ini akan sulit dilakukan di daerah yang tidak memiliki fasilitas kesehatan, karena prosesnya memerlukan teknologi yang canggih. Masalah ini menimbulkan ketertarikan dalam mengklasifikasikan citra kanker kulit untuk memudahkan diagnosa secara klinis.

Metode *Deep Learning* yang saat ini memiliki hasil paling signifikan dalam pengenalan citra adalah *Convolutional Neural Network* (CNN). Hal tersebut dikarenakan CNN berusaha meniru sistem pengenalan citra pada visual *cortex* manusia sehingga memiliki kemampuan mengolah (Suartika E.P dkk., 2016)

Pada penelitian yang dilakukan oleh Fu'adah dkk., menggunakan algoritma CNN dalam mengklasifikasikan dataset lesi kanker kulit dan lesi tumor jinak, menghasilkan kinerja dengan akurasi 99%, *loss* 0,0346 dan nilai presisi, *Recall*, skor *F1-score* hampir 100%. Berdasarkan hasil performa, sistem menunjukkan bahwa model yang diusulkan adalah menjanjikan untuk digunakan sebagai alat untuk tenaga medis dalam menentukan diagnosis (Fu'adah dkk., 2020). Kemudian Pada tahun 2021, algoritma CNN digunakan dalam penelitian yang dilakukan oleh Hakim, dkk., menggunakan Dataset yang diperoleh dari *International Skin Imaging Collaboration* (ISIC) 2018 dengan jumlah 10015 gambar. Berdasarkan hasil report pengujian dan evaluasi diperoleh akurasi sebesar 75% (Luqman Hakim dkk., 2021). selanjutnya pada penelitian Hanin dkk., mengatakan bahwa Metode *Convolutional Neural Network* yang digunakan dalam proses klasifikasi penyakit kulit ini memberikan hasil yang maksimal (Hanin dkk., 2021).

Berdasarkan penelitian-penelitian di atas, algoritma CNN telah banyak digunakan oleh para peneliti dalam menganalisis suatu objek, sebab algoritma ini telah diklaim sebagai model terbaik dalam menyelesaikan permasalahan pengenalan objek.

Maka dari itu, penulis tertarik untuk melakukan penelitian dengan menggunakan model *transfer learning* CNN pada dataset Kanker kulit. Sehingga peneliti memutuskan untuk membuat penelitian yang berjudul "Rancang Bangun Aplikasi *Mobile* Klasifikasi Kanker Kulit Dengan Pemilihan Model *Transfer Learning*".

1.2 Rumusan Masalah

Berdasarkan uraian pada latar belakang masalah diatas, dapat dirumuskan masalah:

1. Bagaimana mengumpulkan dan mengakuisisi citra kanker kulit dari berbagai sumber di internet dan masyarakat dengan label 9 kelas penyakit dan 1 kelas kulit sehat.
2. Bagaimana membangun kinerja model klasifikasi penyakit kanker kulit dengan pemilihan model *transfer learning*.
3. Bagaimana mengevaluasi kinerja model klasifikasi penyakit kanker kulit dengan pemilihan model *transfer learning*.
4. Bagaimana men*deploy* model klasifikasi penyakit kanker kulit dengan pemilihan model *transfer learning* ke dalam aplikasi *mobile* berbasis *Android.*

1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah:

1. Dataset yang digunakan adalah Dataset Kanker Kulit.
2. Dataset diambil dengan dua cara yaitu yang pertama mengunduh gambar kanker kulit di 2 sumber pada internet. Dataset yang di peroleh dari internet pada halaman kaggle dan diperoleh dari *International Skin Imaging Collaboration* (ISIC). Kemudian sumber kedua diperoleh dengan mengumpulkan langsung dari masyarakat. Gambar yang diperoleh dari masyarakat merupakan gambar kulit sehat (*healty skin*).
3. Jenis kanker kulit yang akan diteliti ada 9 jenis yaitu *actinic keratosis, basal cell carcinoma, dermatofibroma, melanoma, nevus, pigmented benign keratosis, seborrheic keratosis, squamous cell carcinoma, vascular lesion* serta 1 jenis kelas kulit sehat (*healty skin*).
4. Model *Transfer Learning Convolutional Neural Network* yang digunakan yaitu *VGG16, NASNetMobile* dan *DenseNet121*.
5. Aplikasi *mobile* yang digunakan berupa *android.*

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah, maka tujuan dari penelitian ini adalah:

1. Membangun dataset citra penyakit kanker kulit dengan label yang memuat 9 kelas penyakit kanker kulit dan 1 kelas kulit sehat (*healty skin*).
2. Membangun model klasifikasi penyakit kanker kulit menggunakan kerangka kerja pemilihan model *transfer learning*.
3. Mengevaluasi kinerja model klasifikasi penyakit kanker kulit menggunakan kerangka kerja pemilihan model *transfer learning*.
4. Men*deploy* model yang telah dibuat menggunakan kerangka kerja pemilihan model *transfer learning* ke dalam aplikasi *mobile* berbasis *Android.*

1.5 Manfaat Penelitian

Berdasarkan tujuan penelitian diatas, maka manfaat dari penelitian ini adalah sebagai berikut:

1. Menghasilkan Dataset citra kanker kulit dengan label 9 kelas penyakit dan 1 kelas kulit sehat (*healty skin*).
2. Menjadi sumber informasi mengenai kinerja model klasifikasi penyakit kanker kulit menggunakan kerangka kerja pemilihan model *transfer learning*.
3. Menyediakan aplikasi *Mobile* berbasis *android* yang dapat mengenali Citra Kanker Kulit.

BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Relevan

Penelitian ini merujuk pada beberapa penelitian yang telah dilakukan sebelumnya. Berikut ini penelitian terdahulu yang berhubungan dengan skripsi ini antara lain:

Penelitian dengan judul "*Skin Cancer Detection Using VGG-16*" diambil pada jurnal *European Journal of Molecular & Clinical Medicine* yang diteliti oleh Kanneboina Manasa dan Dr.G. Vishnu Murthy pada tahun 2021 dengan tujuan menguji kinerja CNN dengan model *VGG16* dan *RESNET-50* dalam mengklasifikasikan kanker kulit (Manasa & Murthy, 2021). Dimana data yang digunakan diambil dari kaggle dengan jumlah 1800 citra benign dan 1497 citra *malignant*. Gambar lesi diubah ukurannya menjadi (224x224x3) RGB. Dengan *epoch* 100 dan menggunakan bobot *imagenet (Transfer learning)* didapatkan hasil akurasi sebesar 80% pada model *VGG16* dan 87% pada model *RESNET-50*.

Penelitian mengenai kanker kulit juga pernah dilakukan oleh Mustafa Çakmak dan Mehmet Emin Tenekecı pada tahun 2021 dengan judul "*Melanoma detection from dermoscopy images using Nasnet Mobile with Transfer Learning*" (Çakmak dkk., 2021). Pada penelitian ini menggunakan dataset skin lesion HAM10000 yang disediakan oleh ISIC 2018 dengan metode *Transfer Learning* menggunakan model *Nasnet Mobile*. Dalam penelitian tersebut, peneliti melakukan perbandingan proses *training* dan *testing* sebelum dan sesudah augmentasi data. Didapatkan tingkat akurasi yang diperoleh dengan model Nasnet *Mobile* sebesar 89,20% sebelum augmentasi data dan meningkat sebesar 97,90% setelah dilakukan augmentasi data.

Selanjutnya penelitian juga pernah dilakukan oleh Jasman Pardede dan Dwi Adi Lenggana Putra pada tahun 2020 dengan judul "*Implementasi DenseNet Untuk Mengidentifikasi Kanker Kulit Melanoma*" (Pardede & Putra, 2020). Pada penelitian ini telah mengimplementasikan CNN dengan arsitektur *DenseNet121* untuk mengidentifikasi kanker kulit *melanoma*. Metode yang digunakan mampu melakukan klasifikasi kanker kulit *melanoma* dengan nilai rata-rata *accuracy, precision, Recall,* dan *F-Measure* masing-masing adalah 0,94, 0,95, 0,92 dan 0,94. Dengan

preprocessing yang digunakan pada arsitektur *DenseNet121* mempengaruhi dari tingkat akurasi.

Penelitian dengan judul "*Aplikasi Mobile Deteksi Dini Kanker Kulit Berdasarkan Image Processing*" diambil pada Jurnal Litbang Edusaintech (JLE) yang diteliti oleh Fina Royana, Puput Yuniar Maulida, Rully Nurul Hasanah, Sondari Setia Rahayu dan Rasim pada tahun 2021 dengan tujuan mengembangkan aplikasi deteksi kanker kulit awal yang bekerja sama dengan dokter kulit (Royana dkk., 2021). Dataset yang digunakan pada penelitian ini diambil dari kaggle dengan 7 jenis kanker kulit. Dengan menggunakan tiga teknik pengelolaan seperti *machine learning* untuk membuat *pipeline* data, membangun model, dan mengonversi model ke *Tensorflow lite*. *Android* untuk menerapkan model *Tensorflow lite* dan membuat aplikasi. Memiliki koneksi *real-time* menggunakan *firebase*. *Cloud* untuk membuat *database* sederhana untuk layanan dokter dan diagnosis di *firebase*. Peneliti mendapatkan akurasi aplikasi sebesar 97%.

2.2 Kanker Kulit

Kanker kulit merupakan suatu penyakit yang disebabkan oleh perubahan sifat-sifat penyusun sel kulit yang normal menjadi ganas, dimana nantinya sel-sel akan terus membelah diri menjadi bentuk yang abnormal secara terus menerus tumbuh dan tidak terkendali. dilihat dari segi histopatologik memiliki struktur yang tidak teratur dengan diferensiasi sel dalam berbagai tingkatan pada *kromatin*, *nukleus*, dan *sitoplasma* (Hendaria dkk., 2015). Pada tabel 2.1 dapat dilihat klasifikasi terhadap jenis kanker kulit beserta deskripsinya.

Tabel 2. 1 Jenis dan Deskripsi Kanker Kulit

Penyakit	Deskripsi
Actinic keratosis	Bercak kasar dan bersisik pada kulit yang disebabkan oleh paparan sinar matahari selama bertahun-tahun.
Basal cell carcinoma	Benjolan lunak putih atau bercak bersisik coklat di bagian tubuh yang terkena matahari, seperti wajah atau leher.

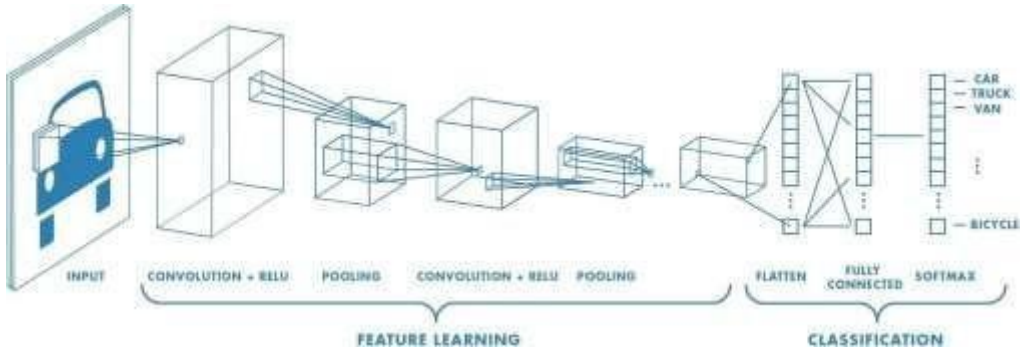
Dermatofibroma	Benjolan bundar yang tumbuh dari bawah kulit yang dapat berwarna merah, merah muda, coklat, abu-abu atau ungu.
Healty skin	Merupakan jenis kulit sehat yang dimana tidak ada tanda-tanda adanya sel kanker.
Melanoma	Bintik hitam pada kulit yang nampak seperti tahi lalat ataupun kotoran biasa.
Nevus	Bintik hitam atau cokelat menonjol yang biasa disebut tahi lalat, dapat ditemukan pada bagian tubuh manapun.
Pigmented benign keratosis	Benjolan seperti kutil pada permukaan kulit yang sering pada bagian wajah, dada, bahu serta punggung.
Seborrheic keratosis	Wujudnya berupa sisik lembut yang sedikit menonjol dengan warna coklat, hitam atau coklat muda.
Squamous cell carcinoma	Benjolan besar pada kulit yang bisa mengeras atau berdarah pada bagian tubuh yang terkena sinar matahari.
Vascular lesion	Bercak yang dikenal dengan tanda lahir dengan warna merah yang kemudian warnanya menjadi gelap seiring waktu.

2.3 *Convolutional Neural Network*

Convolutional Neural Network (CNN) adalah salah satu algoritma paling populer digunakan untuk *Deep Learning*, sebuah machine learning yang model pembelajarannya dikhususkan untuk melakukan klasifikasi langsung pada media dua dimensi seperti gambar, video, teks atau suara. Algoritma CNN akan sangat berguna khususnya ketika digunakan untuk mencari pola pada suatu gambar kemudian mengenali objek pada gambar tersebut. Bukan hanya pada objek atau benda saja, CNN ini sebenarnya juga bisa digunakan untuk mengenali wajah yang selama ini perlu

segmentasi untuk meningkatkan akurasinya. Penelitian awal yang mendasari CNN ini pertama kali dilakukan oleh Hubel dan Wiesel mengenai visual *cortex* pada indera penglihatan kucing. Pada dasarnya klasifikasi citra menggunakan MLP sudah bisa dilakukan, akan tetapi ketika digunakan untuk melakukan klasifikasi data dalam jumlah banyak, akurasi yang didapatkannya pun menurun. Oleh karena itu, algoritma CNN ini dikembangkan karena algoritma ini mampu untuk mempelajari langsung data yang ada pada gambar, kemudian menggunakan pola yang didapatkan untuk mengklasifikasi.

Berbeda dengan arsitektur MLP, arsitektur CNN ini sebenarnya lebih kompleks dan memiliki proses yang cukup panjang sebelum masuk tahap klasifikasi. Namun secara garis besar ada 2 tahapan pemrosesan yang dilakukan oleh algoritma CNN ini, yaitu tahap *feature learning* dan tahap *classification*. Tahap *feature learning* merupakan tahap dimana gambar yang dimasukkan akan diekstraksi untuk dipelajari *value* dari gambar tersebut. Proses inilah yang membedakan algoritma CNN dengan MLP. Jika MLP menggunakan satu proses ekstraksi dalam sekali input untuk melakukan klasifikasi. Sedangkan CNN ini, untuk melakukan klasifikasi bisa menggunakan banyak sekali ekstraksi dalam sekali input. Banyaknya ekstraksi ini disimpan dalam bentuk kedalaman gambar (*depth*). Proses *feature learning* ini sangat bergantung pada kedalaman suatu gambar. Semakin dalam suatu gambar maka semakin banyak ekstraksi yang didapatkan sehingga pola yang didapat juga semakin jelas terbentuk (Li dkk., 2018). *Value* inilah yang nantinya akan dikonversi menjadi vektor dan kemudian masuk pada tahap klasifikasi. Pada tahap klasifikasi ini, model *neural network* akan digunakan untuk melakukan klasifikasi objek berdasarkan kelasnya. Ilustrasi arsitektur dari *Convolutional Neural Network* dapat dilihat pada gambar 2.1.



Gambar 2. 1 Arsitektur *Convolutional Neural Network*

(Sumber gambar: medium.com)

Pada gambar 2.1 merupakan gambar arsitektur dari *Convolutional Neural Network*. Secara umum tahapan klasifikasi CNN dibagi menjadi 2 bagian besar yaitu *feature learning* dan *classification*. Berikut adalah penjelasan tahapan dari arsitektur CNN tersebut.

2.3.1 Convolutional Layer

Konvolusi merupakan salah satu tahap pada arsitektur CNN. Konvolusi merupakan suatu istilah matematis yang berarti mengaplikasikan sebuah fungsi pada *output* fungsi lain secara berulang. Dalam pengolahan citra, konvolusi berarti mengaplikasikan sebuah *kernel* pada citra. *Kernel* adalah sebuah matriks kecil dengan tinggi dan lebarnya lebih kecil dari matriks citra yang akan di konvolusi. *Kernel* biasanya juga dikenal dengan istilah *filter* atau *convolution mask* (Pokhrel, 2019).

Dalam *machine learning*, *input* citra berbentuk *array* dua dimensi dan *kernel* merupakan parameter berbentuk *array* multidimensi yang disesuaikan dengan model algoritma. Konvolusi dapat digunakan pada lebih dari satu dimensi. Sebagai contoh jika menggunakan gambar dua dimensi I sebagai *input*, maka *kernel* K juga berbentuk dua dimensi:

$$S(i,j) = (I \times K)(i,j) = \sum_a \sum_b I(a,b)K(i-a, j-b) \qquad (2.1)$$

Keterangan:

$S\,(i,j)$ = Fungsi hasil konvolusi

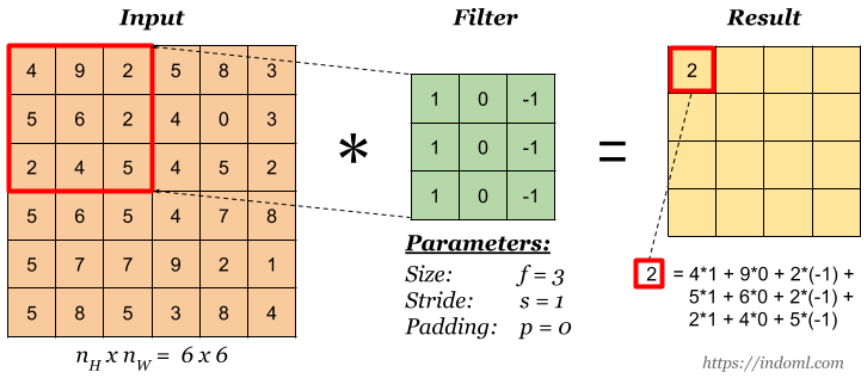
I = *Input*

K = *Kernel* atau *Filter*

(i, j) = *Pixel Input*

(a, b) = *Pixel Kernel*

Tujuan dilakukannya konvolusi pada data citra adalah untuk mengekstraksi fitur dari citra input. Konvolusi akan menghasilkan transformasi linear dari data input sesuai informasi spasial pada data. Bobot pada *layer* tersebut menspesifikasikan kernel konvolusi yang digunakan, sehingga kernel konvolusi dapat dilatih berdasarkan input pada CNN (Suartika E.P dkk., 2016). Pada gambar 2.2 diilustrasikan sebuah matriks input berukuran 6×6.



Gambar 2. 2 Contoh Bentuk Konvolusi

(Sumber gambar: indoml.com)

Gambar 2.2 merupakan ilustrasi sebuah matriks dengan input berukuran 6×6 dimana dilakukan konvolusi dengan *kernel/filter* berukuran 3×3, dengan *stride* atau perpindahannya sebanyak 1, dan *zero padding*. Hasil dari konvolusi tersebut ditunjukkan pada matriks *result* atau biasa disebut sebagai *feature map* (Bayat dkk., 2017).

2.3.2 Stride & Padding

Stride adalah parameter yang menentukan berapa jumlah pergeseran *filter*. Jika nilai *stride* adalah 1, maka konvolusi *filter* akan bergeser sebanyak 1 *pixels* secara *horizontal* kemudian akan bergeser secara *vertikal*. Semakin kecil *stride* maka akan semakin detail informasi yang didapatkan dari sebuah input, namun membutuhkan komputasi yang lebih jika dibandingkan dengan *stride* yang besar dan perlu kita perhatikan bahwa dengan menggunakan *stride* yang kecil tidak selalu akan mendapatkan performa yang bagus.

Padding atau *zero padding* adalah parameter yang menentukan jumlah *pixel*s (berisi nilai 0) yang akan ditambahkan di setiap sisi dari input. Hal ini digunakan dengan tujuan untuk memanipulasi dimensi *output* dari *konvolusi layer* (*feature map*), Dikarenakan size filter yang tidak selalu lebih kecil dari input *size input*, suatu *padding* p diberikan kepada input citra agar *size* dari input lebih besar atau sama dengan *size* dari *filter*. Untuk suatu input citra berukuran $n \times n$, padding p akan merubah *size input* menjadi $(n + 2p) \times (n + 2p)$.

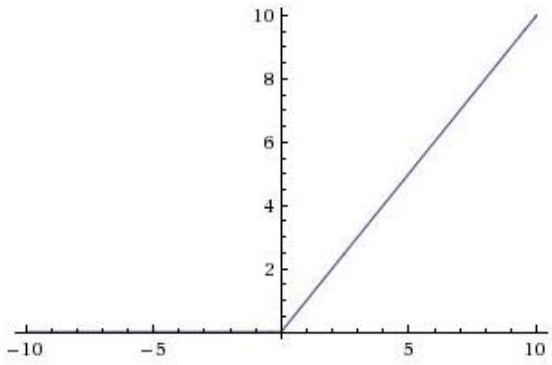
2.3.3 Activation layer

Fungsi aktivasi biasa disebut sebagai lapisan pemetaan non-linear. Fungsi aktivasi biasa digunakan untuk meningkatkan kemampuan *klasifikasi network* (Chen dkk., 2018). Salah satu peranan dari fungsi aktivasi adalah untuk memberikan kemampuan network agar dapat melakukan tugas non-linear. Tanpa fungsi aktivasi, *neural network* hanyalah kombinasi operasi linear yang hanya melakukan tugas-tugas yang linear pula. Padahal kebanyakan kasus nyata di lapangan merupakan kasus non-linear (Santosa & Umam, 2018).

Rectified linear unit (*ReLu*) merupakan salah satu fungsi aktivasi yang sering digunakan pada *Convolutional Neural Network* (Chen dkk., 2018). Fungsi aktivasi *ReLu* menjalankan operasi nilai ambang (*treshold*) untuk setiap elemen input dimana jika nilai lebih kecil dari nol akan diset menjadi nol. Persamaan 2.2 merupakan bentuk fungsi *ReLu*.

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{2.2}$$

Pada fungsi aktivasi *ReLu*, semua nilai x *negative* akan dipetakan ke 0, seperti pada gambar 2.3.



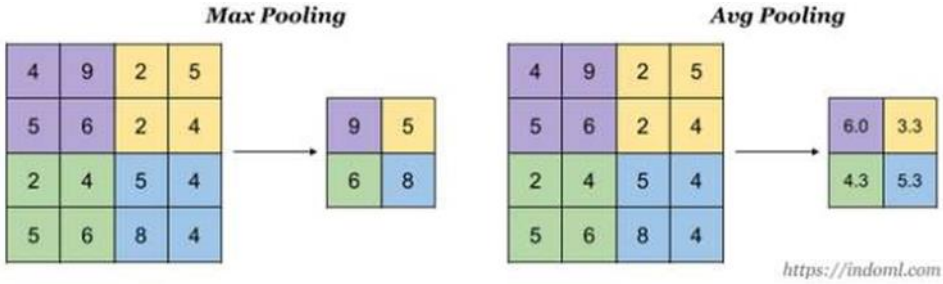
Gambar 2. 3 Grafik Fungsi *ReLu*

(Sumber gambar: medium.com)

Pada fungsi ini input dari neuron-neuron berupa bilangan negatif, maka fungsi ini akan menterjemahkan nilai tersebut kedalam nilai 0 dan jika input bernilai positif maka *output* dari neuron adalah nilai aktivasi itu sendiri. Fungsi aktivasi *ReLu* memiliki kelebihan yaitu dapat mempercepat proses konfigurasi yang dilakukan dengan *Stochastic Gradient Descent* (SGD) jika dibandingkan dengan fungsi *sigmoid* dan *tanh*.

2.3.4 Pooling layer

Pooling layer atau *subsampling* diletakkan diantara *convolutional layer* dan *ReLu layer* dengan tujuan untuk mengurangi jumlah parameter perhitungan, seperti lebar dan tinggi citra, tetapi bukan kedalaman citra (Simonyan & Zisserman, 2014). Langkah melakukan *pooling* adalah dengan membagi hasil keluaran *convolutional layer* menjadi beberapa *grid* berdasarkan penentuan jumlah *stride* dan jenis *pooling* yang digunakan. Terdapat dua jenis *pooling* yang umum digunakan pada *pooling layer*, yaitu *Max Pooling* dan *Average Pooling*. Pada *max pooling*, nilai yang diambil merupakan nilai maksimal dari setiap *grid*. Sedangkan pada *average pooling*, nilai yang diambil merupakan nilai rata-rata dari setiap *grid*. Ilustrasi perhitungan pada *pooling layer* dapat dilihat pada gambar 2.4.

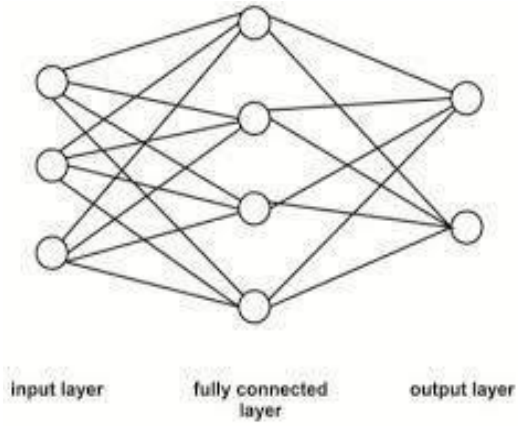


Gambar 2. 4 Ilustrasi Perhitungan pada *Pooling layer*

(Sumber gambar: indoml.com)

2.3.5 Fully connected layer

Fully connected layer adalah lapisan yang digunakan untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear. Gambar 2.5 menampilkan proses yang ada dalam *fully connected layer*.



Gambar 2. 5 Ilustrasi proses *fully connected layer*

(Sumber gambar: ums.ac.id)

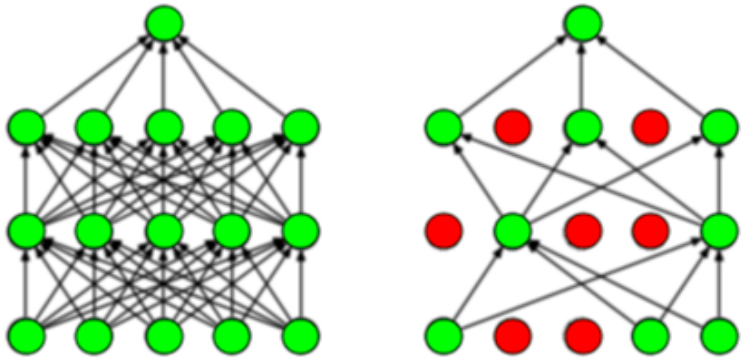
Pada gambar 2.5 terdapat *input layer* yang mana *input layer* tersebut merupakan hasil dari proses *flattening*. Proses *flattening* inilah yang menghasilkan sebuah vektor yang akan digunakan sebagai input dari *Fully connected layer*. Kemudian pada bagian *Fully connected layer* memiliki beberapa *layer-layer* seperti *Dense layer* dan *Dropout layer*. Pada bagian *output layer* merupakan tahapan terakhir yang berfungsi untuk mendefinisikan jenis dari sebuah kelas.

2.3.6 Dropout Regularization

Regularization merupakan teknik yang digunakan untuk mengurangi *overfitting*, yakni kondisi dimana sistem jaringan syaraf tiruan mampu belajar dengan baik dengan

data pelatihan, namun tidak bisa menggeneralisasi pada data *test*. Terdapat beberapa teknik dalam regularisasi, misalnya L2 *regularization* dan *dropout*.

Dropout merupakan salah satu usaha untuk mencegah terjadinya *overfitting* dan juga mempercepat proses *learning* (Abhirawa dkk., 2017). *Overfitting* adalah kondisi dimana hampir semua data yang telah melalui proses *training* mencapai persentase yang baik, tetapi terjadi ketidaksesuaian pada proses prediksi. Dalam sistem kerjanya, *dropout* menghilangkan sementara suatu *neuron* yang berupa *Hidden Layer* maupun *Visible Layer* yang berada di dalam jaringan. Gambar 2.6 adalah contoh *neural network* sebelum dan sesudah adanya proses *dropout.*



(a) (b)

Gambar 2. 6 Sebelum *Dropout* (a) dan Setelah *Dropout* (b)

(Sumber gambar: Medium.com)

Pada gambar 2.6 diatas jaringan syaraf sebelum dilakukan *dropout*(a) merupakan jaringan syaraf biasa dengan 2 lapisan tersembunyi. Sedangkan pada bagian (b) jaringan syaraf sudah diaplikasikan teknik *dropout* dimana terdapat beberapa neuron aktivasi yang tidak dipakai lagi. Teknik ini akan berdampak pada performa model dalam melatih serta mengurangi *overfitting*.

2.4 *Transfer Learning*

Transfer learning adalah suatu teknik atau metode yang memanfaatkan model yang sudah dilatih terhadap suatu dataset untuk menyelesaikan permasalahan lain yang serupa dengan cara menggunakannya sebagai *starting point*, memodifikasi dan meng-update parameternya sehingga sesuai dengan dataset yang baru. Daripada merancang model CNN baru dengan parameter acak, parameter model CNN yang telah dilatih sebelumnya dapat digunakan kembali untuk dataset yang baru. Pada *transfer learning*,

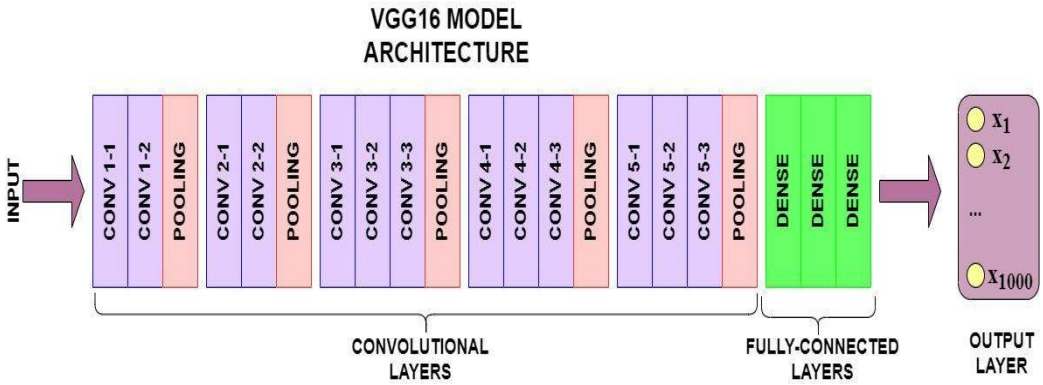
parameter di beberapa *layer* dibekukan dan parameter lain dilatih kembali untuk mempelajari dataset baru. Proses pelatihan kembali ini disebut *fine tuning* (Akçay dkk., 2016).

2.5 Beberapa arsitektur model CNN

Terdapat banyak model arsitektur *Convolutional Neural Network* yang dapat digunakan dalam mengklasifikasi gambar, penelitian ini akan mencoba 3 model berikut adalah penjelasan dari masing-masing model yang akan digunakan pada penelitian ini.

2.5.1 *VGG16*

VGG16 adalah Arsitektur *Convolutional Neural Network* (CNN) yang banyak digunakan untuk *ImageNet*, *VGG* adalah singkatan dari *Visual Geometry Group*, dan '16' menyiratkan bahwa arsitektur ini memiliki 16 lapisan (Simonyan & Zisserman, 2014). Arsitektur model *VGG16* dapat dilihat pada gambar 2.7.



Gambar 2. 7 Arsitektur Model *VGG16*

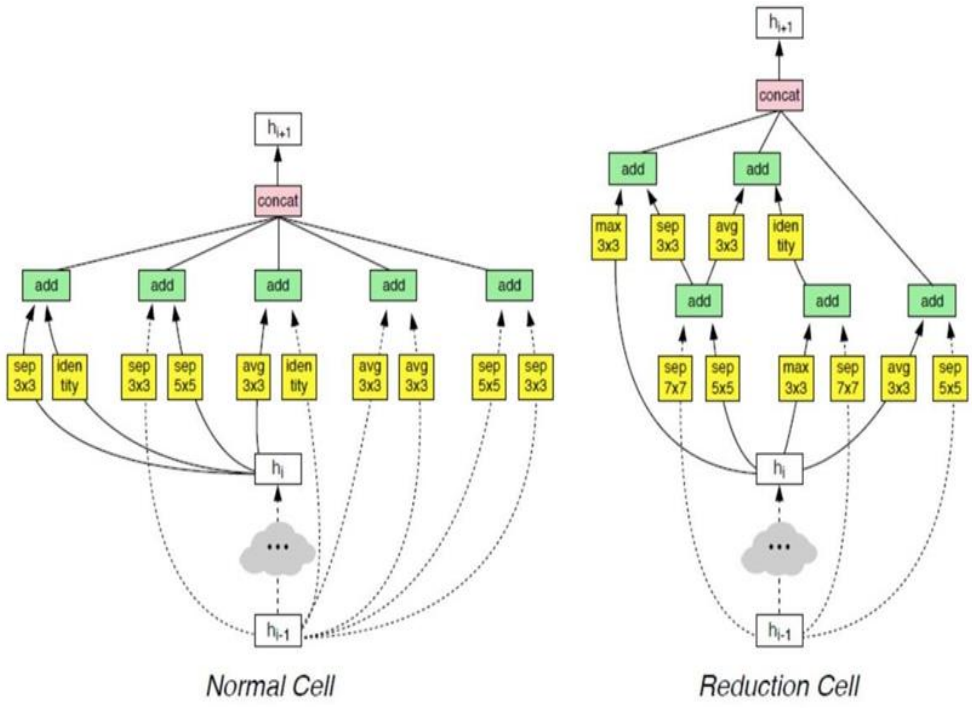
(Sumber gambar: learndatasci.com)

VGG16 merupakan model CNN yang memanfaatkan *convolutional layer* dengan spesifikasi *convolutional filter* yang kecil (3×3). Dengan ukuran *convolutional filter* tersebut, kedalaman *neural network* dapat ditambah dengan lebih banyak lagi *convolutional layer*. Hasilnya, model CNN menjadi lebih akurat dari pada model-model CNN sebelumnya. Model *VGG16* mempunyai 16-*layer* yang terdiri dari 13 *convolutional layer* dan 3 *fully connected layer*. Seperti pada gambar 2.7 terdapat 13

convolutional layer dan 3 *fully connected layer*. Terdapat 5 *pooling layer* yang bertipe *max pooling*. Bentuk inputnya yaitu 224×224×3 (Afif dkk., 2020).

2.5.2 *NASNetMobile*

NASNet adalah sebuah model berdasarkan sebuah penelitian dengan cara mencari arsitektur blok terbaik pada sebuah *dataset* kecil, kemudian menyalin arsitektur terbaik yang ditemukan untuk selanjutnya digunakan pada *dataset* yang lebih besar yakni *ImageNet*. Blok-blok tersebut disusun sedemikian rupa membentuk berbagai variasi arsitektur *NASNet*, variasi yang kecil adalah *NASNetMobile* atau *NASNet-A*. *NASNetMobile* merupakan arsitektur *NASNet* yang lebih kecil dengan jumlah parameter yang menyerupai *MobileNet* namun dengan performa akurasi yang lebih baik. Arsitektur *NASNet* terdiri dari 2 blok utama atau sering disebut *cell* yakni *normal cell* dan *reduction cell* susunan *layer* yang tepat untuk kedua *cell* tersebut dicari menggunakan *recurrent neural network*. Gambar 2.8 di bawah ini adalah gambar susunan *layer* normal *cell* dan *reduction cell* tersebut untuk arsitektur *NASNet A*.



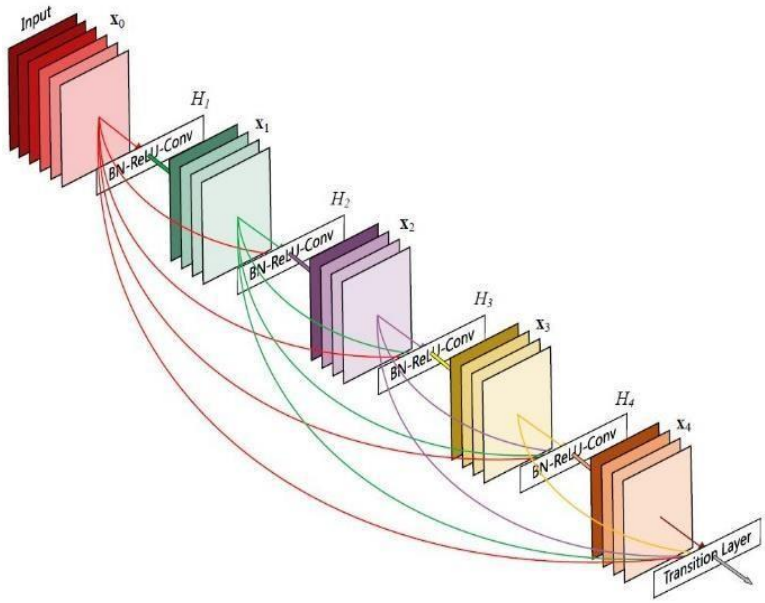
Gambar 2. 8 Cell pada *NASNetMobile*

(Sumber gambar: medium.com)

NASNetMobile atau *NASNet-A* memiliki arti bahwa jumlah *normal cell* diulang sebanyak 4 kali yang selanjutnya diikuti oleh *reduction cell*. Berdasarkan pustaka keras *NASNetMobile* memiliki jumlah total *layer* sebanyak 765-*layer* yang terbagi ke dalam 12 *normal cell* dan 3 *reduction cell* (Zoph dkk., 2017).

2.5.3 *DenseNet121*

*Dense*Net merupakan arsitektur model dengan karakteristik khusus yang disebut dengan *dense block* dimana pada blok tersebut setiap *layer* terhubung dengan semua *layer* secara langsung. Sebuah *layer* mengambil *input* dari *output* semua *layer* sebelumnya dan memberikan *output* untuk semua *layer* setelahnya hal tersebut memungkinkan jaringan menjadi lebih ramping. Hal ini berbeda dengan *layer convolution* tradisional dimana sebuah *layer* mengambil *input* dari *layer* sebelumnya dan memberikan *output* untuk *layer* setelahnya (Huang dkk., 2017). Gambar 2.9 di bawah ini menunjukan arsitektur *layer* pada sebuah *dense block.*



Gambar 2. 9 *Dense Block*

(Sumber gambar: pytorch.org)

Salah satu jenis *DenseNet* yang paling kecil untuk *ImageNet* adalah *DenseNet121*, variasi model *DenseNet* tersebut memiliki jumlah parameter yang kecil serupa dengan *MobileNet*. Berdasarkan pustaka keras, *DenseNet121* terdiri dari total 427-*layer* yang terbagi ke dalam 58 blok.

2.6 Ukuran kinerja model

2.6.1 *Confusion Matrix*

Evaluasi kinerja pada suatu sistem klasifikasi merupakan sesuatu yang penting untuk mengetahui seberapa baik sistem yang dibangun dalam mengklasifikasikan data. Berdasarkan jumlah kelas pada keluaran hasil sistem klasifikasi, jenis klasifikasi dikelompokkan menjadi empat jenis yaitu klasifikasi biner, *multi-case*, *multi-label,* dan *hierarchical.* Pada pengukuran kinerja sistem klasifikasi umumnya digunakan *Confusion Matrix* untuk membandingkan hasil klasifikasi oleh sistem dengan hasil klasifikasi sesungguhnya. Berikut tabel *Confusion Matrix multiclass* yang dapat dilihat pada tabel 2.2.

Tabel 2. 2 *Confusion Matrix multiclass*

	predicted									
Actual	*Class 1*	*Class 2*	*Class 3*	*Class 4*	*Class 5*	*Class 6*	*Class 7*	*Class 8*	*Class 9*	*Class 10*
Class 1	$T_{(1,1)}$	$F_{(1,2)}$	$F_{(1,3)}$	$F_{(1,4)}$	$F_{(1,5)}$	$F_{(1,6)}$	$F_{(1,7)}$	$F_{(1,8)}$	$F_{(1,9)}$	$F_{(1,10)}$
Class 2	$F_{(2,1)}$	$T_{(2,2)}$	$F_{(2,3)}$	$F_{(2,4)}$	$F_{(2,5)}$	$F_{(2,6)}$	$F_{(2,7)}$	$F_{(2,8)}$	$F_{(2,9)}$	$F_{(2,10)}$
Class 3	$F_{(3,1)}$	$F_{(3,2)}$	$T_{(3,3)}$	$F_{(3,4)}$	$F_{(3,5)}$	$F_{(3,6)}$	$F_{(3,7)}$	$F_{(3,8)}$	$F_{(3,9)}$	$F_{(3,10)}$
Class 4	$F_{(4,1)}$	$F_{(4,2)}$	$F_{(4,3)}$	$T_{(4,4)}$	$F_{(4,5)}$	$F_{(4,6)}$	$F_{(4,7)}$	$F_{(4,8)}$	$F_{(4,9)}$	$F_{(4,10)}$
Class 5	$F_{(5,1)}$	$F_{(5,2)}$	$F_{(5,3)}$	$F_{(5,4)}$	$T_{(5,5)}$	$F_{(5,6)}$	$F_{(5,7)}$	$F_{(5,8)}$	$F_{(5,9)}$	$F_{(5,10)}$
Class 6	$F_{(6,1)}$	$F_{(6,2)}$	$F_{(6,3)}$	$F_{(6,4)}$	$F_{(6,5)}$	$T_{(6,6)}$	$F_{(6,7)}$	$F_{(6,8)}$	$F_{(6,9)}$	$F_{(6,10)}$
Class 7	$F_{(7,1)}$	$F_{(7,2)}$	$F_{(7,3)}$	$F_{(7,4)}$	$F_{(7,5)}$	$F_{(7,6)}$	$T_{(7,7)}$	$F_{(7,8)}$	$F_{(7,9)}$	$F_{(7,10)}$
Class 8	$F_{(8,1)}$	$F_{(8,2)}$	$F_{(8,3)}$	$F_{(8,4)}$	$F_{(8,5)}$	$F_{(8,6)}$	$F_{(8,7)}$	$T_{(8,8)}$	$F_{(8,9)}$	$F_{(8,10)}$
Class 9	$F_{(9,1)}$	$F_{(9,2)}$	$F_{(9,3)}$	$F_{(9,4)}$	$F_{(9,5)}$	$F_{(9,6)}$	$F_{(9,7)}$	$F_{(9,8)}$	$T_{(9,9)}$	$F_{(9,10)}$
Class 10	$F_{(10,1)}$	$F_{(10,2)}$	$F_{(10,3)}$	$F_{(10,4)}$	$F_{(10,5)}$	$F_{(10,6)}$	$F_{(10,7)}$	$F_{(10,8)}$	$F_{(10,9)}$	$F_{(10,10)}$

Dengan tabel 2.2 tersebut bisa diketahui parameter *Accuracy*, *Precision*, *Recall* dan *F1-Score*. Berikut adalah penjelasan dan rumus untuk masing-masing parameter tersebut:

a. *Accuracy* merupakan rasio prediksi positif dan negatif dengan keseluruhan data. Berdasarkan tabel di atas maka akurasi dapat dirumuskan:

$$Accuracy = \frac{\sum_{i=1}^{c} T_i}{\mathrm{N}} \quad (2.3)$$

Keterangan:

i = 1, 2, 3, …10

c = Banyaknya *class*

N = Banyaknya dataset

T_i = Jumlah data yang terklasifikasi *True* untuk kelas ke-i.

b. *Precision* merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positif. Presisi dapat diwakili dengan rumus:

$$Precision(i) = \frac{T_i}{T_i + \sum_{j=1}^{c} F_{ij}} \quad (2.4)$$

Keterangan:

i = 1, 2, 3, …10

j = 1, 2, 3, …10

c = Banyaknya *class*

F = Jumlah data yang terklasifikasi *False* pada masing-masing *class*

i = menyatakan baris pada masing-masing *class*

j = menyatakan kolom pada masing-masing *class*

c. *Recall* atau bisa disebut juga dengan *sensitivity* merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif. *Recall* dapat dihitung dengan rumus:

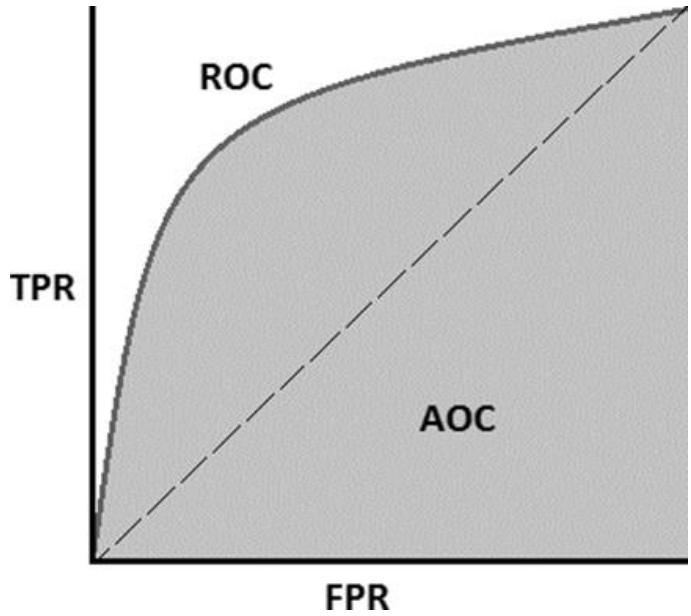
$$Recall(j) = \frac{T_i}{T_i + \sum_{i=1}^{c} F_{ij}} \quad (2.5)$$

d. *F1-score* adalah kombinasi rata-rata dari precision dan *Recall*. *F1-Score* bisa dihitung dengan menggunakan rumus:

$$F1\text{-}Score = \frac{2 \times Recall \times Precision}{Recall + Precision} \quad (2.6)$$

2.6.2 Kurva AUC-ROC

Kurva AUC - ROC adalah pengukuran kinerja untuk masalah klasifikasi pada berbagai pengaturan ambang batas. *Receiver Operator Characteristic* (ROC) adalah kurva probabilitasdan *Area Under the Curve* (AUC) adalah ukuran yang digunakan sebagai ringkasan dari kurva ROC. Ini memperlihatkan seberapa besar model mampu membedakan antar kelas. Semakin tinggi AUC, semakin baik model dalam memprediksi 0 sebagai 0 dan 1 sebagai 1. Dengan analogi, semakin tinggi AUC, semakin baik model dalam membedakan antara penyakit satu dengan yang lainnya. Berikut adalah contoh dari kurva ROC dan AUC dapat dilihat pada gambar 2.10.



Gambar 2. 10 Kurva ROC dan AUC

(Sumber gambar: towardsdatascience.com)

Kurva ROC diplotkan dengan *True Positive Rate* (TPR) terhadap *False Positive Rate* (FPR) dimana TPR berada pada sumbu y dan FPR pada sumbu x seperti yang terlihat pada Gambar 2.10.

2.7 *Overfitting* dan *Underfitting*

Overfitting terjadi karena model yang dibuat terlalu fokus pada *training* dataset tertentu, hingga tidak bisa melakukan prediksi dengan tepat jika diberikan dataset *testing*. *Overfitting* biasanya akan menangkap data *noise* yang seharusnya diabaikan. *Overfitting* model akan memiliki akurasi data *training* tinggi tetapi akurasi data *testing* rendah. Sedangkan *Underfitting* Terjadi ketika model tidak bisa melihat logika dibelakang data, hingga tidak bisa melakukan prediksi dengan tepat, baik untuk dataset *training* maupun dataset *testing*. *Underfitting* model akan memiliki *loss* tinggi dan akurasi rendah (Afif dkk., 2020).

2.8 *Library* yang digunakan

Dalam membangun model arsitektur digunakan bahasa *python* dengan dukungan *library* atau *framework* yang memadai. Berikut merupakan *library python* yang digunakan dalam membangun arsitektur model CNN (Afif dkk., 2020).

2.8.1 Tensorflow

Tensorflow merupakan *library* yang biasa digunakan untuk *Deep Learning* dalam pengolahan data seperti klasifikasi dan lain-lain. *Library* ini dikembangkan dan dikelola oleh Google. Dibangun untuk berjalan pada CPU, GPU atau bahkan OS *mobile*. *Tensorflow* memiliki tingkat fleksibilitas yang tinggi, artinya banyak hal yang dapat dimodifikasi sesuai kebutuhan.

2.8.2 Keras

Keras adalah *library* berbasis sumber terbuka yang dirancang untuk menyederhanakan model dari kerangka *Deep Learning*. Keras dapat dijalankan di atas *framework* (kerangka kerja) kecerdasan buatan seperti *Tensorflow*. Contoh pengguannya di CNN yaitu: *one-hot-encode labels* citra, membangun arsitektur CNN, mem-plot model CNN, men-*load* dataset citra kedalam arsitektur CNN, melakukan *training* dan *testing* model CNN sehingga menghasilkan bahan evaluasi berupa *accuracy* dan *loss*, serta menyimpan (*save*) dan mengakses (*load*) model CNN dalam format h5.

2.8.3 cv2 (*OpenCV*)

cv2 digunakan untuk tujuan pengolahan citra digital (digital *image processing*). Contoh pengguannya di CNN yaitu membaca data citra (jpg, png, dll), mengubah data citra menjadi sebuah array agar bisa diolah, mengkonversi citra (misal dari RGB ke *grayscale*), dan *resize* citra (misal dari 500x500 piksel diubah menjadi 224x224 piksel).

2.8.4 Sklearn

Sklearn digunakan sebagal *tools* ML dan model statistik berupa *classification*, *regression* dan lain-lain. Contoh pengguannya di CNN dapat berupa membuat *Confusion Matrix* dan *Classification Report*.

2.8.5 Numpy

Numpy digunakan untuk memproses/mengolah beragam keperluan array maupun matriks. Contoh pengguannya di CNN yaitu *reshape* dimensi citra, menghitung rata-rata maupun standar deviasi akurasi model CNN, dan pengolahan berbasis array/matriks lainnya.

2.8.6 Matplotlib

Matplotlib digunakan untuk membuat visualisasi data. Contoh pengguannya di CNN yaitu: mem-plot dataset beserta labelnya dan mem-plot hasil *training* maupun *testing* model CNN kedalam sebuah grafik.

2.9 Rancang bangun aplikasi *mobile (Android)*

2.9.1 *Android*

Android merupakan sebuah sistem operasi untuk perangkat *mobile*. *Android* menyediakan *platform* terbuka bagi para pengembang buat membangun aplikasi mereka. *Android*, Inc. ialah awal dikembangkanya *Android*, dengan dukungan finansial dari *Google*, yang kemudian membelinya di tahun 2005. Sistem operasi Android ini dirilis secara resmi di tahun 2007, bersamaan dengan didirikannya *Open Handsdkkliance*, *konsorsium* berasal perusahaan-perusahaan perangkat keras, perangkat lunak, dan telekomunikasi yg bertujuan buat memajukan standar terbuka perangkat seluler. Ponsel *Android* pertama mulai dijual di bulan Oktober 2008.

2.9.2 *Tensorflow lite*

Tensorflow lite adalah kerangka belajar dalam yang ringan untuk perangkat *mobile* dan *embedded*. *Tensorflow lite* mengkompres sebuah model *tensorflow* ke model. *Tflite* yang memiliki ukuran biner kecil. Hal ini memungkinkan mesin belajar diperangkat dan menggunakan akselerasi perangkat keras untuk meningkatkan kinerja (Istiqamah, 2020).

Tensorflow dapat melatih dan menjalankan jaringan saraf yang mendalam untuk klasifikasi digit tulisan tangan, pengenalan gambar, kata yang disematkan, dll. *Tensorflow* mendukung prediksi produksi pada skala besar, dengan model yang sama digunakan untuk pelatihan (Istiqamah, 2020).

BAB III
METODE PENELITIAN

3.1 Waktu dan Tempat

Penelitian ini dilaksanakan dari bulan Maret 2022 sampai dengan bulan Juni 2022. Lokasi penelitian dilakukan di Laboratorium Rekayasa Perangkat Lunak, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Hasanuddin.

3.2 Tahap Penelitian

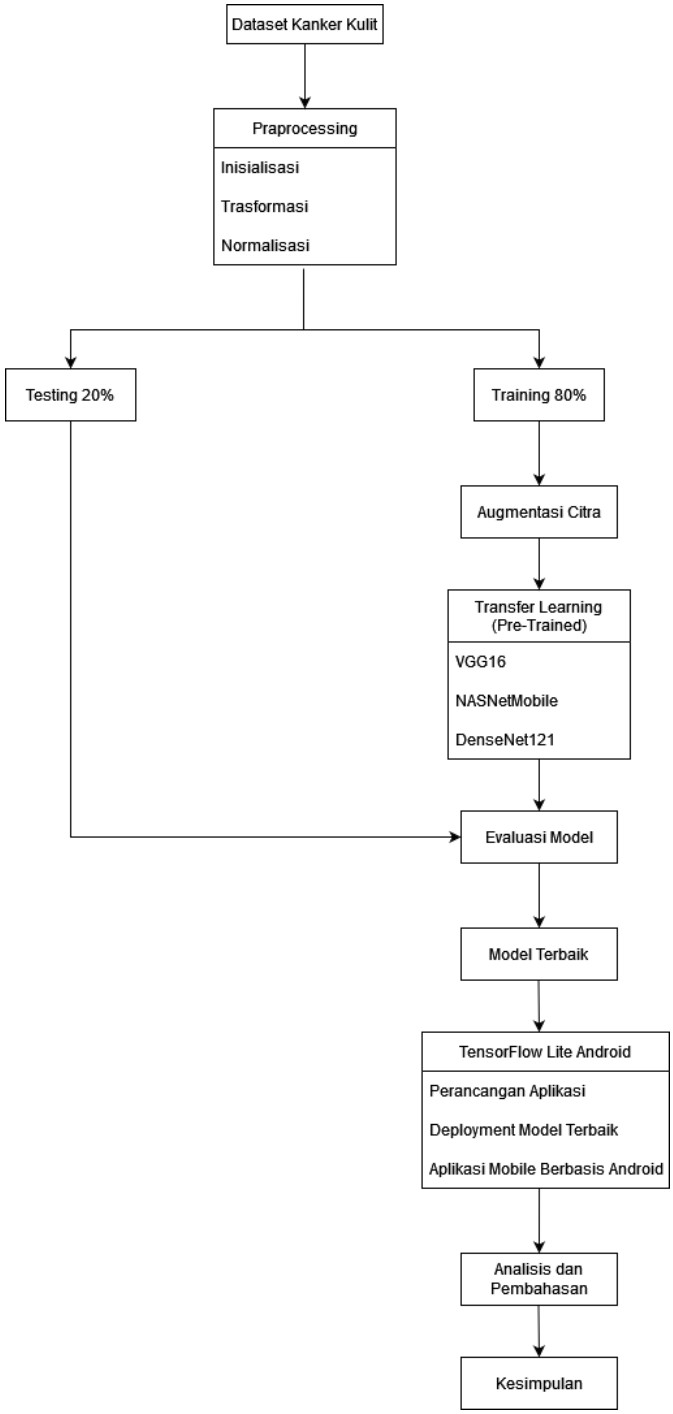
Tahap pada penelitian ini terdiri atas tahapan pra-penelitian dan tahapan penelitian. Tahap pra penelitian merupakan tahap persiapan sebelum meneliti. Sedangkan, tahap penelitian merupakan proses inti untuk mencapai tujuan penelitian.

3.2.1 Tahapan Pra-Penelitian

Pada tahap pra-penelitian, peneliti menentukan tema penelitian, masalah yang akan diteliti, mengumpulkan sumber referensi atau literatur seperti jurnal dan buku yang mendukung dalam penelitian, dan menentukan metode yang digunakan beserta batasan masalahnya. Kemudian peneliti mencari data yang sesuai dengan tema penelitian sebagai objek penelitian.

3.2.2 Tahapan Penelitian

Alur atau tahapan yang dilakukan pada penelitian ini digambarkan melalui gambar 3.1.

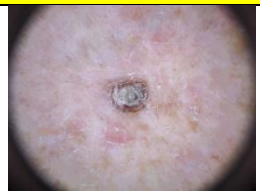


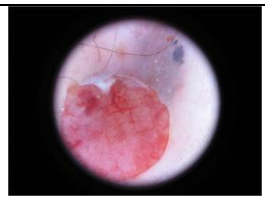
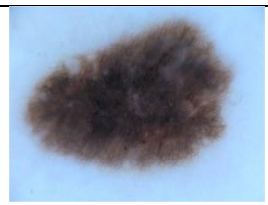
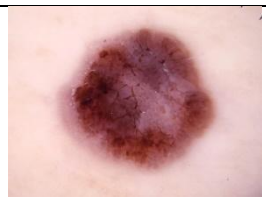
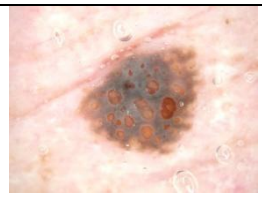
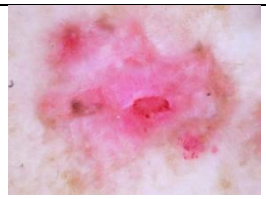
Gambar 3. 1 Alur Penelitian

1. Deskripsi Data

Data yang digunakan yaitu dataset berupa citra kanker kulit. Dataset diambil dengan dua cara yaitu yang pertama mengunduh gambar kanker kulit di 2 sumber pada internet. Dataset di peroleh dari internet pada halaman kaggle dan diperoleh dari halaman *International Skin Imaging Collaboration* (ISIC). Kemudian sumber kedua diperoleh dengan mengumpulkan langsung dari masyarakat. Gambar yang diperoleh dari masyarakat merupakan gambar kulit sehat (*healty skin*). Data terdiri dari 10 kelas yaitu *actinic keratosis, basal cell carcinoma, dermatofibroma, melanoma, nevus, pigmented benign keratosis, seborrheic keratosis, squamous cell carcinoma* dan *vascular lesion.* Format citra kanker kulit tersebut adalah .jpg yang di ambil dari berbagai jenis kamera dengan total 3000 gambar. Berikut 9 kelas kanker kulit dan 1 kelas kulit sehat (*healty skin*) dapat dilihat pada tabel 3.1.

Tabel 3. 1 Dataset Kanker Kulit

No	Gambar	Kelas Kanker Kulit
1.		*Actinic keratosis* (300)
2.		*Basal cell carcinoma* (300)
3.		*Dermatofibroma* (300)
4.		*Healty skin (300)*

5.		*Melanoma* (300)
6.		*Nevus* (300)
7.		*Pigmented benign keratosis* (300)
8.		*Seborrheic keratosis* (300)
9.		*Squamous cell carcinoma* (300)
10.		*Vascular lesion* (300)

2. Preprocessing

Data citra kanker kulit dimasukkan ke dalam sistem. Kemudian dilakukan inisialisasi berupa jumlah *epoch*, *learning rate*, *batch size*, *image size directory* dan *input size*. Dimana untuk *epoch* yang digunakan sebanyak 100, *learning rate* 0,0001, *batch size* 64. Untuk ukuran awal dari citra yang dimiliki sangat beragam, sehingga dilakukan *resize* dengan ukuran 224×224×3 piksel untuk masing-masing model arsitektur yang digunakan. Setelah itu data citra dikonversi menjadi *array* dan dinormalisasi dari rentang 0-255 menjadi rentang 0-1.

3. Pembagian Data

Setelah preprocessing, data akan dibagi menjadi data *training* dan data *testing* dengan perbanding 80% dan 20%, dengan jumlah citra pada data *training* sebanyak 2.400 dan pada data *testing* sebanyak 600 citra.

4. Augmentasi Citra

Setelah data dibagi menjadi 20% untuk *testing* dan 80% untuk *training*, pada data *training* sebanyak 80% tersebut dilakukan proses augmentasi data untuk mengurangi *overfitting* dengan menghasilkan data yang mengandung *noise* yang menyebabkan model tidak condong terhadap data yang terlalu ideal.

5. Pelatihan Model

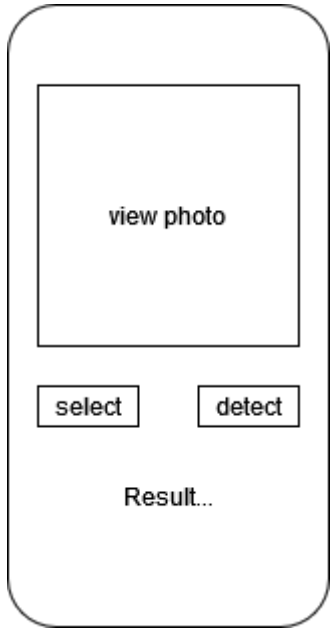
Setelah data sudah melalui *preprocessing*, maka selanjutnya masuk ke tahap pelatihan model CNN yang terdiri dari proses ekstraksi fitur dan klasifikasi. Penelitian ini menggunakan algoritma *Convolutional Neural Network* (CNN) dengan arsitektur *VGG16*, *NASNetMobile*, dan *DenseNet121*.

6. Evaluasi Model

Pada tahap ini, evaluasi model akan dilakukan dengan data *testing*. *Confusion Matrix* digunakan untuk menghitung tingkat akurasi, presisi, *Recall*, serta *F1-score* dalam mengklasifikasikan kanker kulit serta melihat seberapa baik model dalam mengklasifikasikan kanker kulit menggunakan kurva AUC-ROC.

7. Perancangan Aplikasi

Setelah dilakukan evaluasi model dari model terbaik, selanjutnya dilakukan perancangan Aplikasi menggunakan *Android Studio*. Berikut rancangan aplikasi yang dapat dilihat pada gambar 3.2.



Gambar 3. 2 Rancangan Aplikasi

Pada gambar 3.2 Rancangan Aplikasi, terdapat beberapa fitur diantaranya sebagai berikut:

a. Fitur *Select*: berfungsi untuk memasukkan gambar kanker kulit yang berada pada galeri *smartphone.*
b. Fitur *View photo*: berfungsi untuk menampilkan gambar kanker kulit.
c. Fitur *Detect*: berfungsi untuk mendeteksi gambar kanker kulit yang dimasukkan pada aplikasi.
d. Fitur *result*: berfungsi untuk menampilkan hasil deteksi gambar kanker kulit.

8. *Deploy* Model

Setelah dilakukan perancangan aplikasi, maka selanjutnya di lakukan *convert* model untuk di *deploy* ke dalam aplikasi *mobile* yang di rancang.

9. Analisis dan Pembahasan

Setelah melakukan *deploy* model dan perancangan aplikasi, selanjutnya adalah analisis dan pembahasan. Pada tahap ini peneliti akan melakukan analisis serta pembahasan terhadap hasil dari model arsitektur yang digunakan yaitu *VGG16, NASNetMobile* dan *DenseNet121* serta aplikasi *mobile* yang telah dirancang.

10. Kesimpulan

Setelah melalui analisis dan pembahasan, selanjutnya adalah membuat kesimpulan terhadap analisis serta pembahasan yang didapatkan.

3.3 Instrumen Penelitian

Perangkat keras yang digunakan dalam penelitian ini yaitu laptop dengan *processor* AMD A4-5000 APU *with Radeon* (TM) HD *Graphics*. RAM 8 GB dengan system operasi Windows 11. Perangkat lunak yang digunakan untuk membangun dan menguji model kami menggunakan *Google Colab* akselerator *hardware* GPU dan *library* yang di gunakan yaitu keras dengan basis *tensorflow*-GPU. Dengan penggunaan google colab ini di harapkan waktu komputasi akan lebih efisien di karenakan *google colab* yang memberikan fasilitas RAM, memori penyimpanan, dan GPU yang disediakan oleh *google*.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Deskripsi Data

Data yang digunakan yaitu dataset berupa citra kanker kulit. Dataset tersebut merupakan gabungan dari Dataset yang diup*load* oleh pattnaik satyajik pada halaman kaggle (https://www.kaggle.com/pattnaiksatyajit/skin-cancer) dengan dataset yang diperoleh dari *International Skin Imaging Collaboration* (ISIC). Data terdiri dari 10 kelas yaitu *actinic keratosis*, *basal cell carcinoma*, *dermatofibroma*, *melanoma*, *nevus*, *pigmented benign keratosis*, *seborrheic keratosis*, *squamous cell carcinoma* dan *vascular lesion*. Format citra kanker kulit tersebut adalah .jpg yang diambil dari berbagai jenis kamera dengan total keseluruhan 3000 gambar. Berikut tabel 4.1 yang memaparkan jenis dan jumlah dataset kanker kulit.

Tabel 4. 1 Dataset kanker kulit

No.	Jenis Penyakit	Jumlah Citra
1.	*actinic keratosis*	300
2.	*basal cell carcinoma*	300
3.	*dermatofibroma*	300
4.	*melanoma*	300
5.	*nevus*	300
6.	*pigmented benign keratosis*	300
7.	*seborrheic keratosis*	300
8.	*squamous cell carcinoma*	300
9.	*vascular lesion*	300
10.	*Healty skin*	300

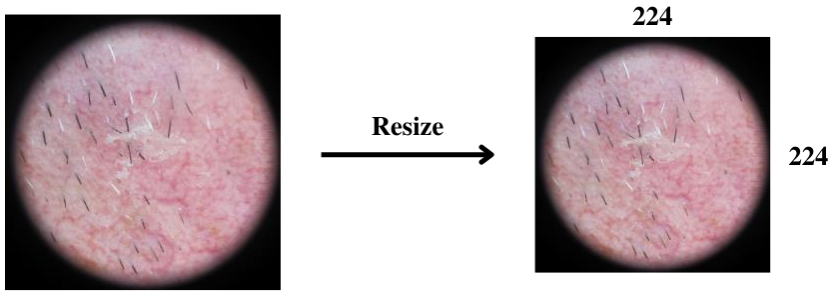
4.2 Preprocessing

Preprocessing adalah tahapan awal yang dilakukan sebelum mengolah citra. pada tahap awal dilakukan inisialisasi seperti yang dapat dilihat pada tabel 4.2 berikut.

Tabel 4. 2 Inisialisasi *hyperparameter*

Model	***Epoch***	**Learning rate**	**Batch size**	**Input size**
VGG16	100	0.0001	64	224×224×3
NASNetMobile	100	0.0001	64	224×224×3
DenseNet121	100	0.0001	64	224×224×3

Epoch menyatakan satu putaran penuh pelatihan (*training*) terhadap seluruh dataset. Pada penelitian digunakan 100 *epoch* untuk pelatihan. *Learning rate* merupakan parameter yang mengontrol seberapa cepat atau lambat model mempelajari masalah pada saat pelatihan (*training*). Dapat dilihat pada tabel 4.2 *learning rate* yang digunakan yaitu 0.0001. kemudian *batch size* berperan untuk memecah dataset kedalam beberapa bagian kecil. *Batch size* yang digunakan pada penelitian ini yaitu 64. Kemudian Pemilihan *input size* untuk model *VGG16* dipilih berdasarkan pada paper yang ditulis oleh Kanneboina Manasa dan Dr.G.Vishnu Murthy pada tahun 2021, sedangkan model *NASNetMobile* dipilih berdasarkanpada paper yang ditulis oleh Elia Cano dkk., pada tahun 2021 dan pada model *DenseNet121* dipilih berdasarkan pada paper yang ditulis oleh Jasman Pardede dan Dwi Adi Lenggana Putra pada tahun 2020. Gambar 4.1 merupakan ilustrasi hasil *resize* dengan input 224×224.



Gambar 4. 1 Ilustrasi pada arsitektur *VGG16*, *DenseNet121* dan *NASNetMobile*

Setelah dilakukan *resize* data, kemudian dilakukan normalisasi. Tujuan dari normalisasi untuk menggunakan seluruh *range* nilai *grayscale* agar diperoleh gambar yang lebih tajam dan juga jelas jika mata manusia melihatnya. Selain itu normalisasi juga bertujuan untuk mempercepat kinerja pada saat *training* data.

4.3 *Splitting* Data

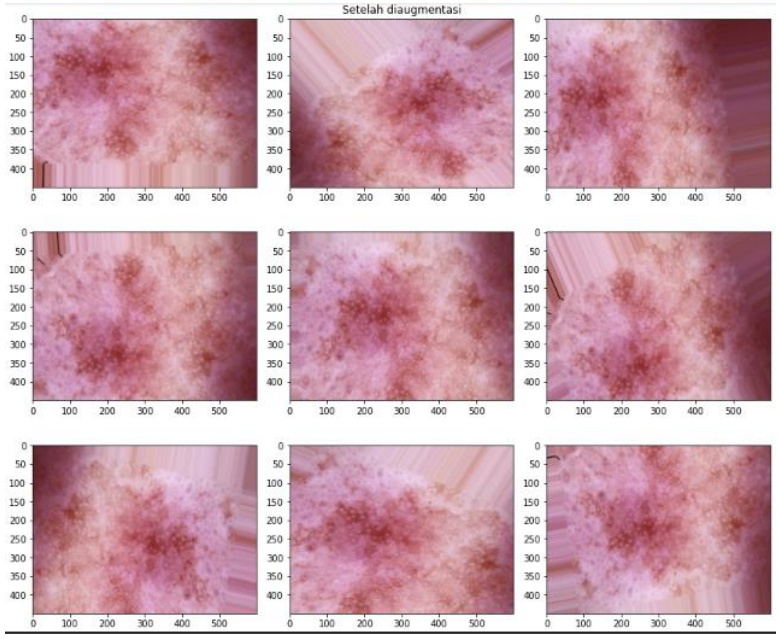
Terdapat 3000 data citra kanker kulit yang digunakan dalam penelitian ini dengan perbandingan 80% untuk *training* dan 20% untuk *testing* dengan jumlah citra untuk *training* sebanyak 2400 dan jumlah citra untuk *testing* sebanyak 600 citra. Pembagian data yang digunakan dalam penelitian ini dapat dilihat pada tabel 4.3.

Tabel 4. 3 Pembagian dataset

Kelompok Data	**Jumlah Data**
Training	2400
Testing	600

4.4 Augmentasi Citra

Setelah dilakukan *resize* dan *Splitting* data, maka selanjutnya data citra pada data *training* akan dilakukan augmentasi. Augmentasi citra merupakan teknik yang dilakukan oleh setiap algoritma *machine learning* dalam aplikasi pada sebuah klasifikasi, baik itu klasifikasi pada teks maupun pada gambar. Augmentasi adalah suatu proses dalam pengolahan data gambar, augmentasi merupakan proses mengubah atau memodifikasi gambar sedemikian rupa sehingga komputer akan mendeteksi bahwa gambar yang diubah adalah gambar yang berbeda, namun manusia masih dapat mengetahui bahwa gambar yang diubah tersebut adalah gambar yang sama. Augmentasi dapat meningkatkan akurasi dari model CNN yang dilatih karena dengan augmentasi model mendapatkan data-data tambahan yang dapat berguna untuk membuat model yang dapat melakukan generalisasi dengan lebih baik. Pada gambar 4.2 menunjukkan bentuk citra yang telah dilakukan augmentasi.



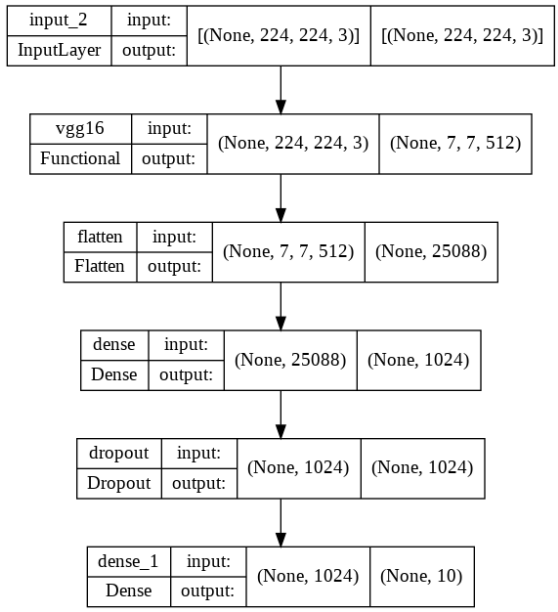
Gambar 4. 2 Citra hasil augmetasi

Pada gambar 4.2 merupakan hasil augmentasi citra dengan 8 perlakuan diantaranya yaitu *rotation_range*, *width_shift_range*, *height_shift_range*, *shear_range*, *zoom_range*, *horizontal_flip*, *vertical_flip*, *fill_mode* dan terdapat 1 gambar yang merupakan gabungan dari semua perlakuan.

4.5 Implementasi Arsitektur *Convolutional Neural Network* (CNN)

Pada penelitian ini digunakan 3 arsitektur CNN, yaitu arsitektur *VGG16*, *DenseNet121* dan *NASNetMobile*. Dimana pada masing-masing model arsitektur diterapkan metode *transfer leaning*. Berikut merupakan arsitektur yang digunakan.

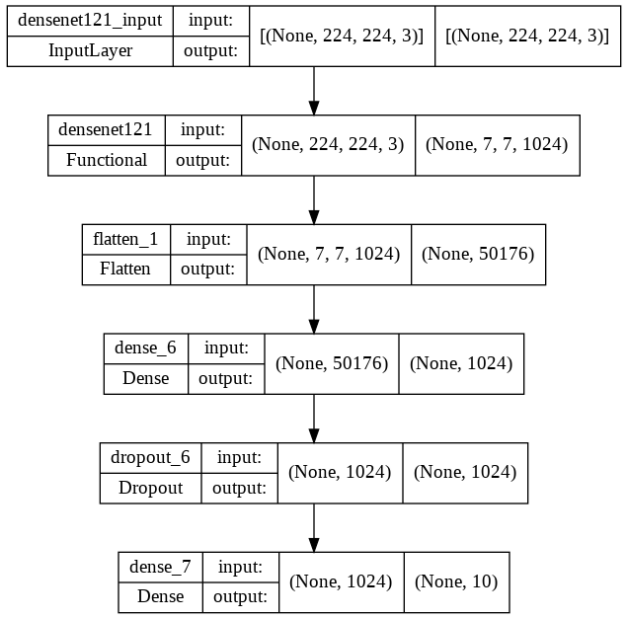
4.5.1 Arsitektur *VGG16*



Gambar 4. 3 Arsitektur *VGG16*

Pada gambar 4.3 merupakan ringkasan arsitektur dari model *VGG16*, dengan input *layer* berupa gambar RGB dengan ukuran 224×224×3 *pixel*. lalu masuk pada tahap penggunaan model *VGG16*, pada model *VGG16* terdapat beberapa *layer* yang digunakan model tersebut diantaranya yaitu *layer convolution* dan *pooling layer*. Setelah itu pada tahap ini *feature* yang dihasilkan masih berbentuk 3 dimensi, sehingga dilakukan *flatten* atau *reshape feature* menjadi *array* 1 dimensi. Kemudian ditambahkan *dense layer* sebanyak 1024 *units* dengan aktivasi *ReLu*, *dropout* sebanyak 0,5 dan *output layer* sebesar 10 dengan aktivasi *softmax*.

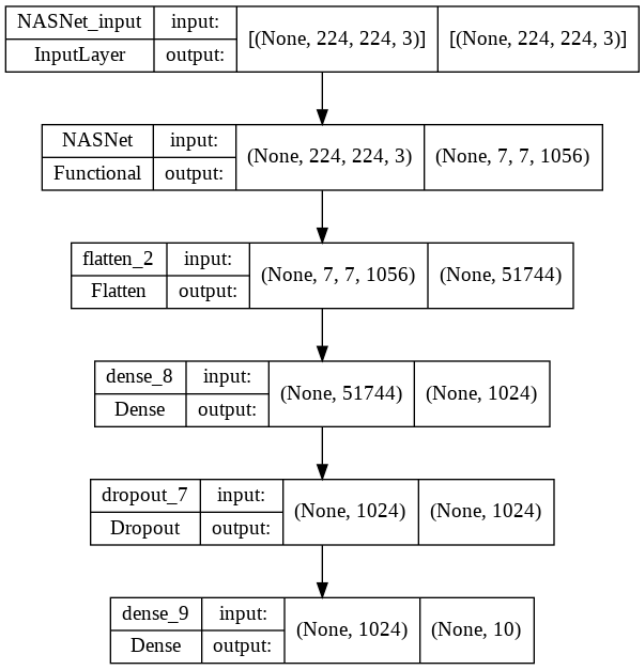
4.5.2 Arsitektur *DenseNet121*



Gambar 4. 4 Arsitektur *DenseNet121*

Pada gambar 4.4 merupakan ringkasan arsitektur dari model *DenseNet121*, dengan input *layer* berupa gambar RGB dengan ukuran 224×224×3 *pixel*. lalu masuk pada tahap penggunaan model *DenseNet121*, pada model *DenseNet121* terdapat *layer convolution, pooling layer*, *BatchNormalization* dan *concatenate*. setelah itu Pada tahap ini *feature* yang dihasilkan masih berbentuk 3 dimensi, sehingga untuk merubah ke bentuk *array* 1 dimensi digunakan *flatten*. Kemudian ditambahkan *dense layer* sebanyak 1024 *units* dengan aktivasi *ReLu*, *dropout* sebanyak 0,5 dan *output layer* sebesar 10 dengan aktivasi *softmax*.

4.5.3 Arsitektur *NASNetMobile*



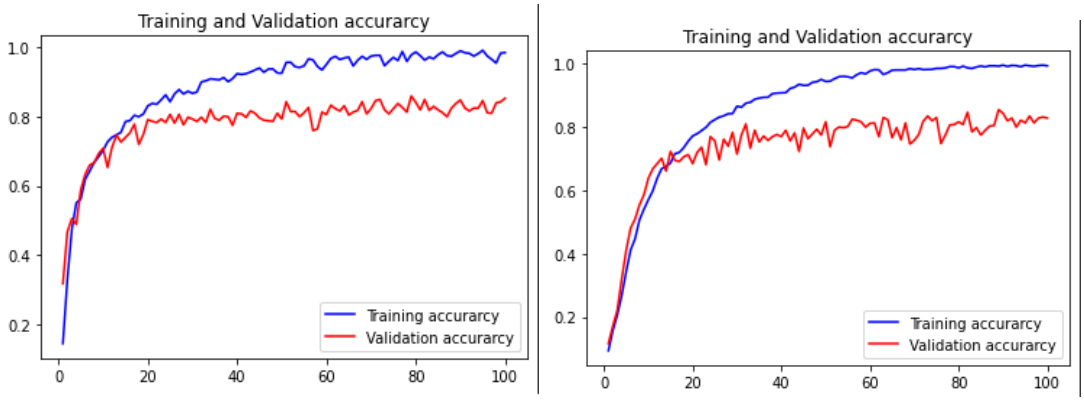
Gambar 4. 5 Arsitektur *NASNetMobile*

Pada gambar 4.5 merupakan ringkasan arsitektur dari model *NASNetMobile*, dengan input *layer* berupa gambar RGB dengan ukuran 224×224×3 *pixel*. lalu masuk pada tahap penggunaan model *NASNetMobile*, pada model *NASNetMobile* terdapat *layer convolution, pooling layer, BatchNormalization, SeparableConv2D* dan *concatenate*. setelah itu Pada tahap ini *feature* yang dihasilkan masih berbentuk 3 dimensi, sehingga untuk merubah ke bentuk *array* 1 dimensi digunakan *flatten*. Kemudian ditambahkan *dense layer* sebanyak 1024 *units* dengan aktivasi *ReLu, dropout* sebanyak 0,5 dan *output layer* sebesar 10 dengan aktivasi *softmax*.

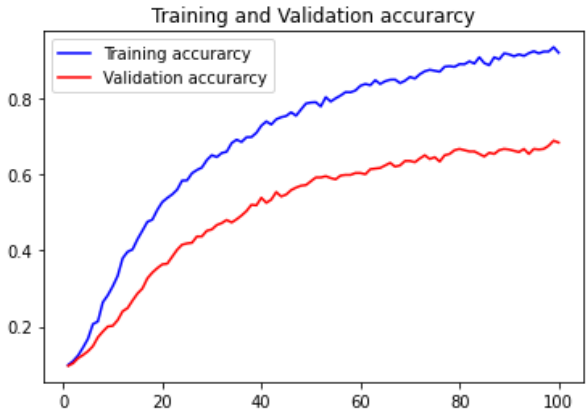
4.6 Evaluasi Model

4.6.1 Akurasi

Kurva akurasi data *train* dan data *test* pada masing-masing model dapat dilihat pada gambar 4.6. Dimana untuk masing-masing model *VGG16, DenseNet121, NASNetMobile* di *training* dengan 100 *epoch*. Pada model *VGG16* mencapai akurasi sebesar 98% dan validasi akurasi sebesar 85%, untuk model *DenseNet121* mencapai akurasi sebesar 99% dan validasi akurasi sebesar 82%, serta untuk model *NASNetMobile* mencapai akurasi sebesar 96% dan validasi akurasi sebesar 68%.



(a)*VGG16* (b) *DenseNet121*

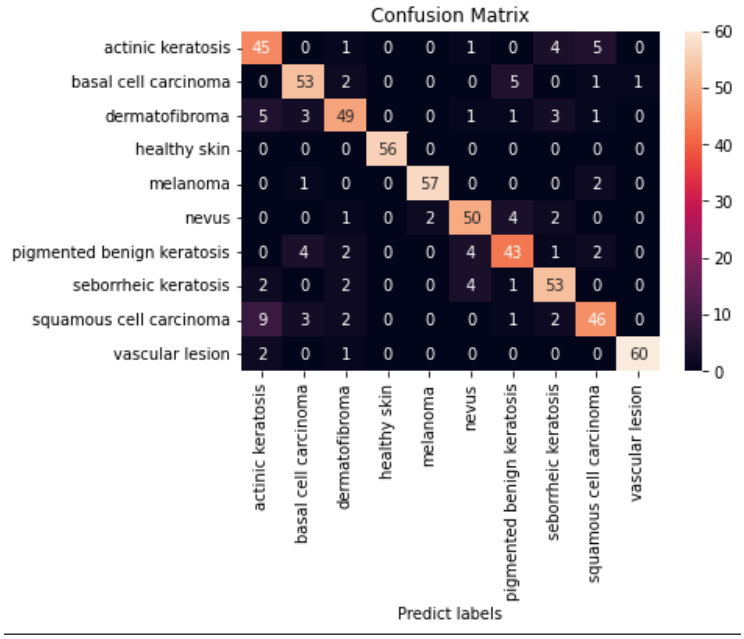


(c) *NASNetMobile*

Gambar 4. 6 kurva akurasi dari arsitektur *VGG16*, *DenseNet121* dan *NASNetMobile*

4.6.2 Confusion Matrix

Confusion Matrix digunakan untuk menampilkan pengujian model hasil perbandingan data aktual dan data prediksi pada dataset kanker kulit dengan 10 kelas jenis penyakit. Gambar 4.7 merupakan gambar confusion matriks dari arsitektur *VGG16*. Dapat dilihat bahwa terjadi *miss classification* dengan total 11 gambar pada *class actinic keratosis*, 9 gambar pada *class basal cell carcinoma*, 14 gambar pada *class dermatofibroma*, 3 gambar pada *class melanoma*, 9 gambar pada *class nevus*, 13 gambar pada *class pigmented benign keratosis*, 7 gambar pada *class seborrheic keratosis*, 17 gambar pada *class squamous cell carcinoma*, 3 gambar pada *class vascular lesion*.



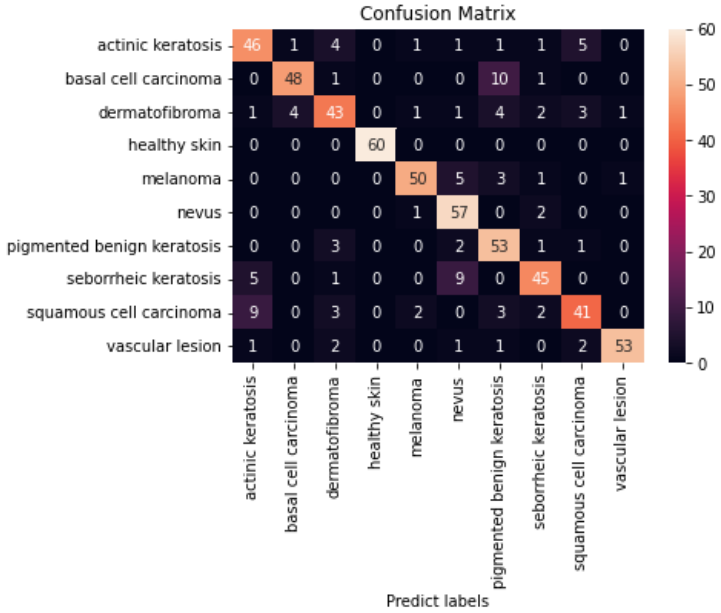
Gambar 4. 7 *Confusion Matrix VGG16*

Tabel 4.4 merupakan hasil evaluasi kinerja model yang berfokus pada tiap kelas berupa *precision*, *Recall* dan *f1-score* dari model *VGG16*.

Tabel 4. 4 Hasil evaluasi kinerja model arsitektur *VGG16*

Kelas	Precision	*Recall*	F1-score
Actinic keratosis	0.71	0.80	0.76
Basal cell carcinoma	0.83	0.85	0.84
Dermatofibroma	0.82	0.78	0.80
Healty Skin	1.00	1.00	1.00
Melanoma	0.97	0.95	0.96
Nevus	0.83	0.85	0.84
Pigmented benign keratosis	0.78	0.77	0.77
Seborrheic keratosis	0.82	0.85	0.83
Squamous cell carcinoma	0.81	0.73	0.77
Vascular lesion	0.98	0.95	0.97

Kemudian *Confusion Matrix* Pada model *DenseNet121* yang dapat dilihat pada gambar 4.8 terjadi beberapa *miss classification* pada masing-masing *class* penyakit kanker kulit. Terjadi *miss classification* dengan total 14 gambar pada *class actinic keratosis*, 12 gambar pada *class basal cell carcinoma*, 17 gambar pada *class dermatofibroma*, 10 gambar pada *class melanoma*, 3 gambar pada *class nevus*, 7 gambar pada *class pigmented benign keratosis*, 15 gambar pada *class seborrheic keratosis*, 19 gambar pada *class squamous cell carcinoma* dan 7 gambar pada *class vascular lesion*.



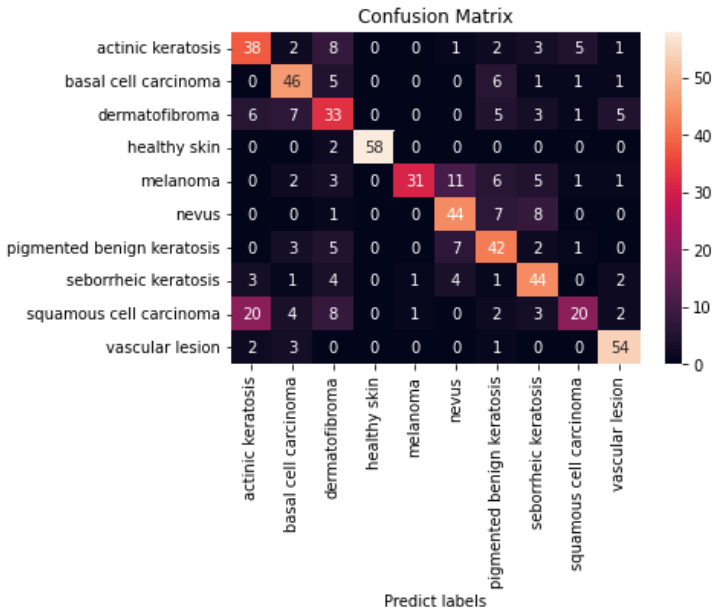
Gambar 4. 8 *DenseNet121*

Tabel 4.5 merupakan hasil evaluasi kinerja model yang berfokus pada tiap kelas berupa *precision*, *Recall* dan *f1-score* dari model *DenseNet121*.

Tabel 4. 5 Hasil evaluasi kinerja model arsitektur *DenseNet121*

Kelas	Precision	*Recall*	F1-score
Actinic keratosis	0.74	0.77	0.75
Basal cell carcinoma	0.91	0.80	0.85
Dermatofibroma	0.75	0.72	0.74
Healty Skin	1.00	1.00	1.00
Melanoma	0.91	0.83	0.87
Nevus	0.75	0.95	0.84
Pigmented benign keratosis	0.71	0.88	0.79
Seborrheic keratosis	0.82	0.75	0.78
Squamous cell carcinoma	0.79	0.68	0.73
Vascular lesion	0.96	0.88	0.92

Confusion Matrix Pada model *NASNetMobile* dapat dilihat pada gambar 4.9, terjadi beberapa *miss classification* pada masing-masing *class* penyakit kanker kulit. Dapat dilihat bahwa terjadi *miss classification* dengan total 22 gambar pada *class actinic keratosis*, 14 gambar pada *class basal cell carcinoma*, 27 gambar pada *class dermatofibroma*, 2 gambar pada *class healty skin*, 29 gambar pada *class melanoma*, 16 gambar pada *class nevus*, 18 gambar pada *class pigmented benign keratosis*, 16 gambar pada *class seborrheic keratosis*, 40 gambar pada *class squamous cell carcinoma*, 6 gambar pada *class vascular lesion*.



Gambar 4. 9 *NASNetMobile*

Tabel 4.6 merupakan hasil evaluasi kinerja model yang berfokus pada tiap kelas berupa *precision*, *Recall* dan *f1-score* dari model *NASNetMobile*.

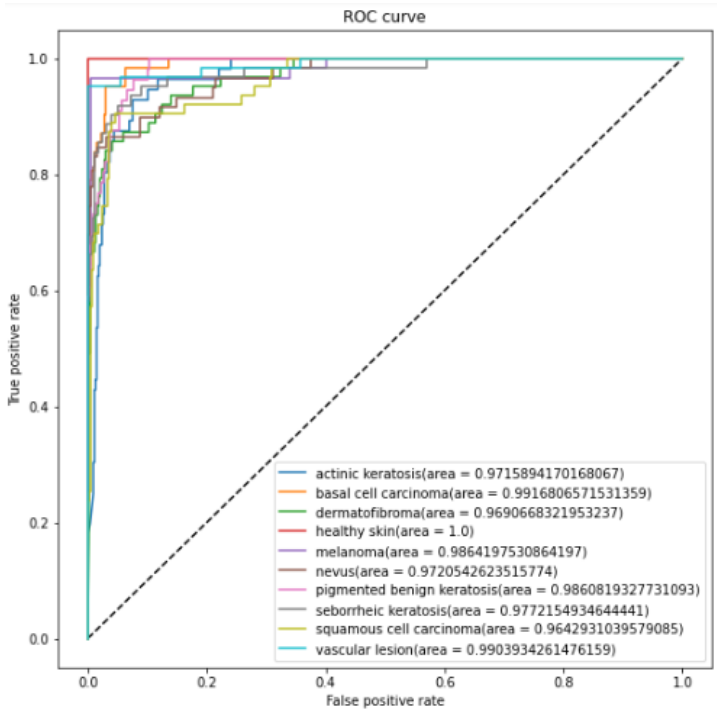
Tabel 4. 6 Hasil evaluasi kinerja model arsitektur *NASNetMobile*

Kelas	Precision	*Recall*	F1-score
Actinic keratosis	0.55	0.63	0.59
Basal cell carcinoma	0.68	0.77	0.72
Dermatofibroma	0.48	0.55	0.51
Healty Skin	1.00	0.97	0.98
Melanoma	0.94	0.52	0.67
Nevus	0.66	0.73	0.69
Pigmented benign keratosis	0.58	0.70	0.64
Seborrheic keratosis	0.64	0.73	0.68
Squamous cell carcinoma	0.69	0.33	0.45
Vascular lesion	0.82	0.90	0.86

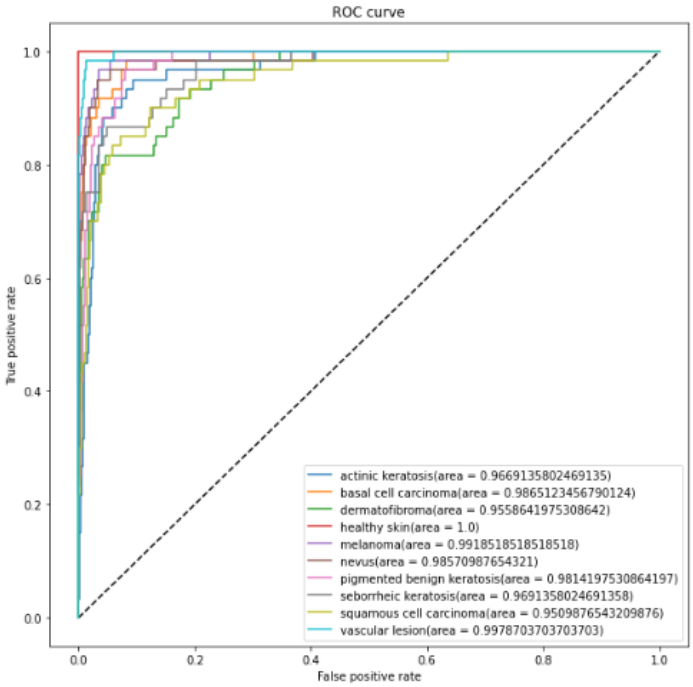
4.6.3 Kurva ROC

Hasil evaluasi berupa Kurva *ROC* pada masing-masing model *VGG16*, *DenseNet121* dan *NASNetMobile* dapat dilihat pada gambar 4.10. Kurva ROC pada model *VGG16* menunjukan hasil yang baik yaitu berada pada rentang nilai terendah 0.96 untuk kelas *Squamous cell carcinoma* dan nilai tertinggi dengan nilai 1.0 pada kelas Healty Skin. Kemudian pada model *DenseNet121* memiliki nilai terendah sebesar 0.95 pada kelas *Dermatofibroma* dan *Squamous cell carcinoma*, dan nilai tertinggi sebesar 1.0 pada kelas *Healty Skin*. Sedangkan pada model *NASNetMobile*

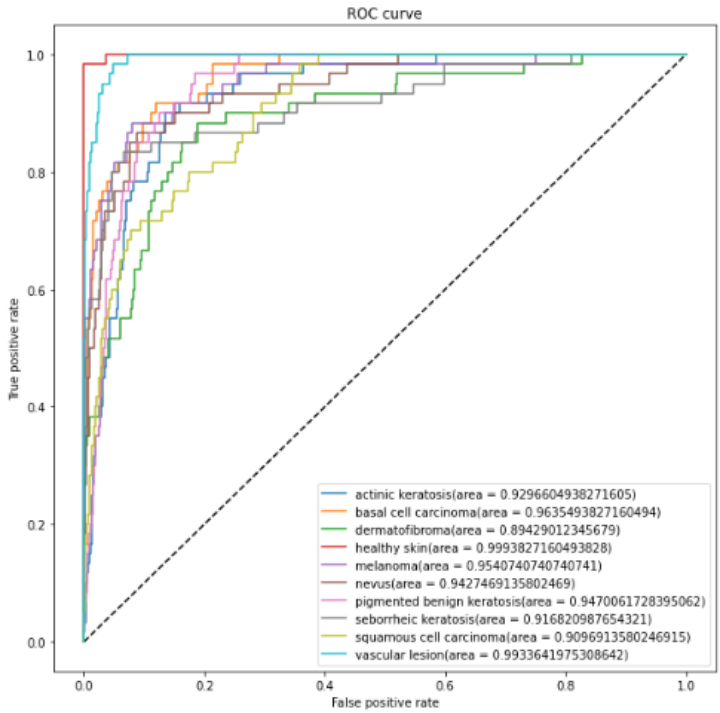
memiliki nilai terendah sebesar 0.89 pada kelas *Dermatofibroma,* serta nilai tertinggi sebesar 0.99 pada kelas *Healty Skin* dan *Vascular lesion.*



(a) ROC VGG16



(b) *ROC DenseNet12*



(c) *ROC NASNetMobile*

Gambar 4. 10 *ROC* pada model *VGG16*, *DenseNet121* dan *NASNetMobile*

4.7 Model terbaik

Model terbaik dilihat dari hasil eksperimen dari ketiga arsitektur model yang digunakan yaitu *VGG16*, *DenseNet121* dan *NASNetMobile*. Model tersebut menghasilkan kinerja akurasi yang dapat dilihat Pada tabel 4.7 berikut.

Tabel 4. 7 Evaluasi akurasi pada masing-masing model.

Model	Akurasi	Validasi Akurasi
VGG16	98%	85%
DenseNet121	99%	82%
NASNetMobile	96%	68%

Pada Tabel 4.7 dapat dilihat bahwa akurasi untuk data *train* pada model *DenseNet121* lebih besar dibanding akurasi pada model *VGG16* dan *NASNetMobile*. Akan tetapi, validasi akurasi data *test* pada model *VGG16* cenderung lebih besar dibandingkan dengan model *DenseNet121* dan *NASNetMobile*. akurasi menandakan bahwa seberapa bagus model mempelajari data, sedangkan validasi akurasi menandakan bahwa seberapa bisa model mengenali data baru.

Sementara itu, pada gambar 4.7, gambar 4.8 dan gambar 4.9 dapat dilihat confusion matrix dari masing model. Jumlah total *miss classification* pada masing-masing model dapat dilihat pada tabel 4.8 berikut.

Tabel 4. 8 Jumlah total *miss classification* pada masing-masing model

Model	Total *miss classification*
VGG16	86 gambar
DenseNet121	104 gambar
NASNetMobile	190 gambar

Pada tabel 4.8 dapat dilihat bahwa model *VGG16* merupakan model dengan total *miss classification* terkecil dibanding kedua model yang digunakan. Nilai kurva ROC pada masing-masing model dapat dilihat pada tabel 4.9 berikut.

Tabel 4. 9 Nilai kurva ROC

Kelas Kanker Kulit	Kurva ROC Model		
	VGG16	*DenseNet121*	*NASNetMobile*
actinic keratosis	0.97	0.96	0.92
basal cell carcinoma	0.99	0.98	0.96
dermatofibroma	0.96	0.95	0.89
melanoma	0.98	0.99	0.95
nevus	0.97	0.98	0.94
pigmented benign keratosis	0.98	0.98	0.94
seborrheic keratosis	0.97	0.96	0.91
squamous cell carcinoma	0.96	0.95	0.90
vascular lesion	0.99	0.99	0.99
Healty skin	1.0	1.0	0.99

Setelah melihat hasil evaluasi kinerja dari ketiga model tersebut, maka dapat disimpulkan bahwa model *VGG16* merupakan model terbaik yang didapatkan dari eksperimen yang telah dilakukan.

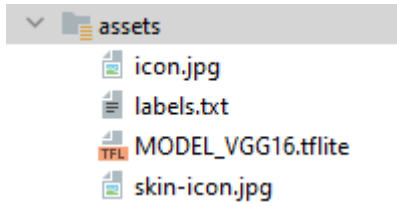
4.8 *Deploy* ke Aplikasi Android

Hasil evaluasi dari ketiga model yang digunakan yaitu model arsitektur *VGG16*, *DenseNet121* dan *NASNetMobile* menujukan bahwa model arsitektur *VGG16* mempunyai hasil yang lebih baik dibanding dengan kedua model lainnya. Model *VGG16* akan di *deploy* ke android menggunakan *tensorflowlite* agar dapat digunakan pada aplikasi android. Adapun tahapan dari *deployment* model *VGG16* yaitu pertama dengan meng*convert* model *Deep Learning* ke dalam *Tensorflow lite* yang nantinya model akan berformat tflite. Gambar 4.11 merupakan implemetasi dari proses konversi model.

```
from tensorflow import lite
converter = lite.TFLiteConverter.from_keras_model(model)
tfmodel = converter.convert()
open('VGG16_10.tflite', 'wb').write(tfmodel)
```

Gambar 4. 11 Implemetasi konversi model ke dalam *tensorflow lite*

Setelah dilakukan konversi model kedalam bentuk tflite, selanjutnya yaitu memasukkan model *VGG16*_10.tflite dan *labels*.txt kedalam folder *assets* pada *project* aplikasi android. Gambar 4.12 merupakan tempat menyimpan model beserta *labels*.



Gambar 4. 12 folder *assets* pada *project* android

Pada gambar 4.12 terdapat file *labels.txt* yang mana file *labels* tersebut berisi nama dari 10 jenis kanker kulit. Kemudian langkah selanjutnya yaitu menambahkan dependensi pada *build.gradle(:app)* dalam *project* android. Gambar 4.13 merupakan Implementasi dependensi *tensorflow*.

```
dependencies {
    implementation 'org.tensorflow:tensorflow-lite:1.14.0'
}
```

Gambar 4. 13 implementasi dependensi *tensorflow*

Penambahan dependensi ini dilakukan untuk memudahkan dalam menyertakan modul *library*. Selanjutnya dilakukan pengkodisian *noCompress* untuk memastikan bahwa model tidak dikompresi dengan menyetel *aaptOptions* pada *build.gradle(:app)*. Berikut adalah gambar 4.14 potongan kode *noCompress*.

```
aaptOptions {
    noCompress "tflite"
}
```

Gambar 4. 14 implementasi kode *noCompress*

Model tidak dikompresi untuk menghindari kegagalan dalam proses kompresi model. Selanjutnya pada halaman *classifier.kt* digunakan untuk memuat model dan *labels*. Berikut gambar 4.15 yang merupakan implementasi untuk memuat model dan *labels*.

```
// Load file tflite
private fun loadModelFile(assetManager: AssetManager, modelPath: String): MappedByteBuffer {
    val fileDescriptor = assetManager.openFd(modelPath)
    val inputStream = FileInputStream(fileDescriptor.fileDescriptor)
    val fileChannel = inputStream.channel
    val startOffset = fileDescriptor.startOffset
    val declaredLength = fileDescriptor.declaredLength
    return fileChannel.map(FileChannel.MapMode.READ_ONLY, startOffset, declaredLength)
}

// Load file label.txt
private fun loadLabelList(assetManager: AssetManager, labelPath: String): List<String> {
    return assetManager.open(labelPath).bufferedReader().useLines { it.toList() }

}
```

Gambar 4. 15 implementasi untuk memuat model dan *labels*

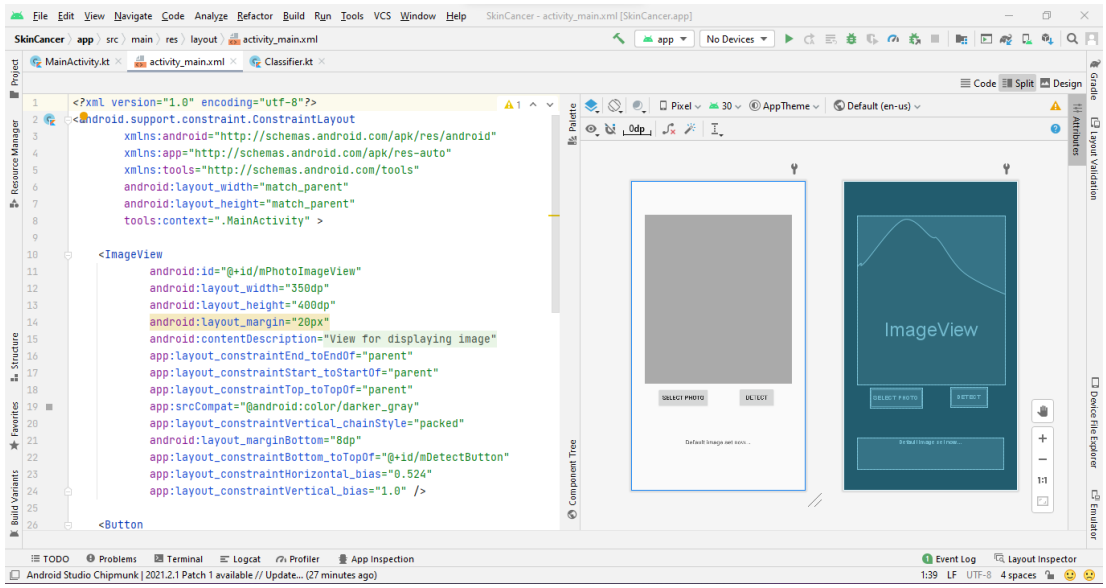
Implementasi kode pada gambar 4.15 digunakan untuk membaca file model dan label yang ada pada direktori *assest* agar dapat digunakan untuk prediksi data gambar. Model yang telah dilatih dimuat menggunakan *class interpreter* dari *package org.tensorflow.lite* yang dimana akan dilakukan konversi untuk gambar yang berupa *bitmap* ke dalam bentuk *bytebuffer*. Implementasi kode dapat dilihat pada gambar 4.16.

```
fun recognizeImage(bitmap: Bitmap): List<Classifier.Recognition> {
    val scaledBitmap = Bitmap.createScaledBitmap(bitmap, INPUT_SIZE, INPUT_SIZE, filter: false)
    val byteBuffer = convertBitmapToByteBuffer(scaledBitmap)
    val result = Array( size: 1) { FloatArray(LABEL_LIST.size) }
    INTERPRETER.run(byteBuffer, result)
    return getSortedResult(result)
}
```

Gambar 4. 16 implementasi *convert bitmap* ke *bytebuffer*

Method yang digunakan dalam mengkonversi *bitmap* kedalam *bytebuffer* adalah *convertBitmapToBytebuffer* dimana ini berfungsi untuk mengubah bit kedalam bentuk byte agar gambar yang nantinya di masukkan sebagai argumen akan terdeteksi sesuai dengan probabilitas label yang tetapkan.

Tampilan aplikasi yang di rancang pada android studio di atur pada halaman *activity_main.xml* yang yang dapat dilihat pada gambar 4.17 berikut.



Gambar 4. 17 halaman activity_main.xml

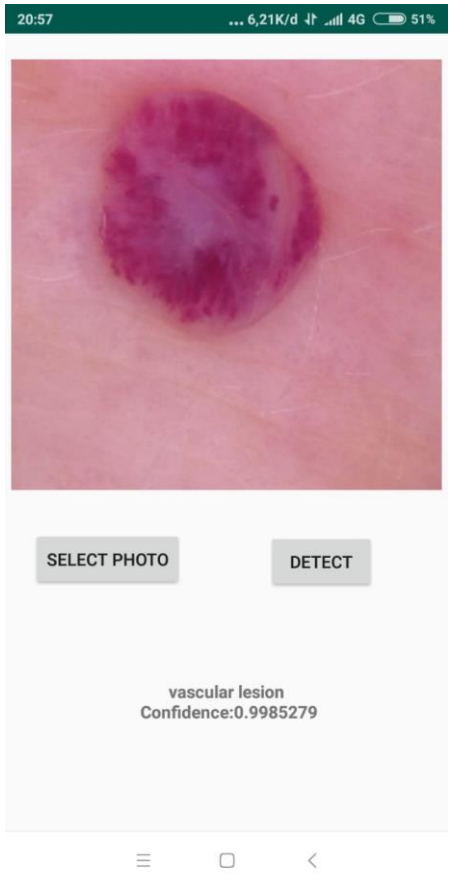
Pada gambar 4.17 tersebut *ImageView* yang berfungsi untuk menampilkan gambar yang dimasukkan nantinya. Kemudian terdapat dua *button* yaitu *button select* photo yang akan mengambil gambar pada galeri dan button *detect* yang akan mendeteksi gambar yang telah di masukkan.

4.9 Visualisasi aplikasi Android

Aplikasi android yang dirancang hanya memiliki 1 layout. Gambar 4.18 merupakan tampilan awal dari aplikasi android klasifikasi kanker kulit.

Gambar 4. 18 tampilan awal aplikasi klasifikasi kanker kulit

Pada gambar 4.18 terdapat 2 tombol, yaitu tombol *select* dan tombol *detect*. Kemudian terdapat hasil yang menunjukkan jenis kanker beserta skor yang dihasikan. Gambar 4.19 merupakan gambar hasil setelah dilakukan pengklasifikasian jenis kanker kulit menggunakan aplikasi.



Gambar 4. 19 Aplikasi klasifikasi kanker kulit

Pada gambar 4.19 telah dilakukan uji coba aplikasi dengan mendeteksi gambar dari jenis kanker kulit yaitu *vascular lesion* dengan hasil ketepatan gambar sebesar 99%.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan maka dapat ditarik kesimpulan sebagai berikut:

1. Data yang digunakan dalam penelitian ini merupakan data 9 kelas kanker kulit yaitu *actinic keratosis, basal cell carcinoma, dermatofibroma, Healty Skin, melanoma, nevus, pigmented benign keratosis, seborrheic keratosis, squamous cell carcinoma, vascular lesion* serta 1 kelas kulit sehat (*healty skin*).
2. Dalam membangun model klasifikasi penyakit kanker kulit digunakan kerangka kerja pemilihan model transfer learning. Dimana terdapat tiga model arsitektur yang digunakan yaitu *VGG16, DenseNet121* dan *NASNetMobile*. *hyperparameter* yang digunakan pada masing-masing model antara lain *learning rate* sebesar 0.0001, *batch size* sebesar 64, dan *epoch* sebanyak 100 kali.
3. Dari ketiga model yang digunakan, model *VGG16* mendapat hasil akurasi tertinggi. hasil akurasi data *train* pada model arsitektur *VGG16* yaitu sebesar 98%, sedangkan hasil untuk data *test* sebesar 85%. Kemudian untuk *DenseNet121* menghasilkan nilai akurasi sebesar 99% untuk data *train* dan 82% untuk data *test*. Selanjutnya untuk model arsitektur *NASNetMobile* menghasilkan nilai akurasi pada data *train* sebesar 96% dan 68% untuk data *test*. Dari ketiga model yang digunakan mengalami *overfitting*.
4. Model klasifikasi yang di *deploy* menggunakan *tensorflow lite* pada aplikasi android yaitu *VGG16*. Model ini dilatih menggunakan dataset kanker kulit yang berarti model ini tidak dapat mengenali objek diluar dataset tersebut.

5.2 Saran

Setelah melakukan penelitian, peneliti menyarankan sebagai berikut:

1. Peneliti selanjutnya disarankan melakukan *ensamble learning*.
2. Pada penelitian selanjutnya disarankan menggunakan model arsitektur lain seperti *Inception-v4, ResNet-18, ResNet-34, ResNet-50, Inception ResNet-v2*.

DAFTAR PUSTAKA

Abhirawa, H., Jondri, M. S., & Arifianto, A. (2017). Pengenalan Wajah Menggunakan *Convolutional Neural Network* Face Recognition Using *Convolutional Neural Network*. *E-Proceeding of Engineering*, *4*, 4907–4916.

Afif, M., Fawwaz, A., Ramadhani, K. N., & Sthevanie, F. (2020). Klasifikasi Ras pada Kucing menggunakan Algoritma *Convolutional Neural Network* (CNN). *EProceedings of Engineering*, *8*(1).

Akçay, S., Kundegorski, M. E., Devereux, M., & Breckon, T. P. (2016). Transfer Learning Using *Convolutional Neural Network*s for Object *Classification* Within X-RAY Baggage Security Imagery. *In 2016 IEEE International Conference on Image Processing (ICIP)*, 1057–1061.

Bayat, O., Aljawarneh, S., Carlak, H. F., International Association of Researchers, Institute of Electrical and Electronics Engineers, & Akdeniz Üniversitesi. (2017). Understanding of a *Convolutional Neural Network*. *Proceedings of 2017 International Conference on Engineering & Technology (ICET'2017)*, 21–23.

Chen, W., Sun, Q., Wang, J., Dong, J. J., & Xu, C. (2018). A Novel Model Based on AdaBoost and Deep CNN for Vehicle *Classification*. *IEEE Access*, *6*, 60445–60455. https://doi.org/10.1109/ACCESS.2018.2875525

Dutt, A., & Dutt, A. (2017). Handwritten Digit Recognition Using *Deep Learning*. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, *6*(7), 990–997.

Fu'adah, Y. N., Pratiwi, N. C., Pramudito, M. A., & Ibrahim, N. (2020). *Convolutional Neural Network* (CNN) for Automatic Skin Cancer *Classification* System. *IOP Conference Series: Materials Science and Engineering*, *982*(1). https://doi.org/10.1088/1757-899X/982/1/012005

Hanin, M. A., Patmasari, R., Yunendah, R., & Fu'adah, N. (2021). Sistem Klasifikasi Penyakit Kulit Menggunakan *Convolutional Neural Network* (CNN) Skin DIsease *Classification* System Using *Convolutional Neural Network* (CNN). *EProceedings of Engineering*, , *8*(1), 273–281.

Hendaria, M. P., Asmarajaya, A., & Maliawan, S. (2015). Kanker Kulit. *Fakultas Kedokteran Universitas Udayana.*, 1–17.

Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2017). *Dense*ly Connected Convolutional Networks. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*, 4700–4708. https://github.com/liuzhuang13/*Dense*Net.

Istiqamah, A. M. (2020). *Klasifikasi Citra Menggunakan Convolutional Neural Network Dengan Arsitektur Inception V4 Berbasis Android Pada Dataset Flower Recognition*.

Li, H., Wang, P., You, M., & Shen, C. (2018). Reading car license plates using deep neural networks. *Image and Vision Computing*, *72*, 14–23. https://doi.org/10.1016/j.imavis.2018.02.002

Luqman Hakim, Sari, Z., & Handhajani, H. (2021). Klasifikasi Citra Pigmen Kanker Kulit Menggunakan *Convolutional Neural Network*. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *5*(2), 379–385. https://doi.org/10.29207/resti.v5i2.3001

Manasa, K., & Murthy, D. G. V. (2021). Skin Cancer Detection Using VGG-16. *European Journal of Molecular & Clinical Medicine*, *8*(1), 1419–1426.

Nuraeni, fitri, Yoga Handoko, A., & Nirwani Yusup, E. (2016). Aplikasi pakar untuk diagnosa penyakit kulit menggunakan metode forward chaining di al arif skin care kabupaten ciamis. *Seminar Nasional Teknologi Informasi Dan Multimedia*, 55–60.

Pardede, J., & Putra, D. A. L. (2020). Implementasi *Dense*Net Untuk Mengidentifikasi Kanker Kulit *Melanoma*. *Jurnal Teknik Informatika Dan Sistem Informasi*, *6*(3). https://doi.org/10.28932/jutisi.v6i3.2814

Royana, F., Yuniar Maulida, P., Nurul Hasanah, R., & Setia Rahayu, S. (2021). Aplikasi *Mobile* Deteksi Dini Kanker Kulit Berdasarkan Image Processing. *Jurnal Litbang Edusaintech*, *2*(2), 100–106. http://journal.pwmjateng.com/index.php/jle

Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv Preprint ArXiv*, 1409–1556. http://arxiv.org/abs/1409.1556

Suartika E.P, I. W., Wijaya, A. Y., & Soelaima, R. (2016). Klasifikasi Citra Menggunakan *Convolutional Neural Network* (Cnn) pada Caltech 101. *JURNAL TEKNIK ITS*, *5*, A65–A69.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2014). Going Deeper with Convolutions. *Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1–9. http://arxiv.org/abs/1409.4842

Wardhani, S. R. (2010). Biopsi dalam Bidang Dermatologi. *Maranatha Journal of Medicine and Health*, *5*(1), 14–23.

Wilvestra, S., Lestari, S., & Asri, E. (2018). Studi Retrospektif Kanker Kulit di Poliklinik Ilmu Kesehatan Kulit dan Kelamin RS Dr. M. Djamil Padang Periode Tahun 2015-2017. In *Jurnal Kesehatan Andalas* (Vol. 7). http://jurnal.

Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. v. (2017). *Learning Transferable Architectures for Scalable Image Recognition*. http://arxiv.org/abs/1707.07012

LAMPIRAN

1. Lampiran Code program

Berikut merupakan *Souce Code* yang digunakan pada pengimplementasian arsitektur CNN dalam mengklasifikasikan kanker kulit.

ln [1]: *import library*

```
import Numpy as np
import matplotlib.pyplot as plt
import cv2
import os
from os import listdir

import keras
from keras.preprocessing import image
from keras import backend as K
from keras.layers import Input
from keras.optimizers import Adam

from keras.preprocessing.image import ImageDataGenerator
from keras.preprocessing import image
from keras.preprocessing.image import img_to_array
from Sklearn.preprocessing import MultiLabelBinarizer
from Sklearn.preprocessing import LabelBinarizer
from Sklearn.model_selection import train_test_split

import tensorflow as tf
from tensorflow.keras import optimizers
from tensorflow.keras.models import Sequential, Model
from keras.layers import Conv2D, MaxPooling2D, Dense, Input,
Activation, Dropout, GlobalAveragePooling2D, \
    BatchNormalization, concatenate, AveragePooling2D, Flatten
```

Ln [2]: directory dataset

```
%cd "/content/drive/MyDrive/SkinCancer_10/cancer/actinic keratosis"
# read an image
img = cv2.imread('ISIC_0053826.jpg')
# show image format
print(img)
```

ln [3]: inisialisasi *hyperparameter*

```
EPOCHS = 100
INIT_LR = 1e-4
default_image_size = tuple((224, 224))
directory_root = "/content/drive/MyDrive/SKIN_CANCER"
width = 224
height = 224
depth = 3
```

ln [4]: Konversi Citra Menjadi Array

```
def convert_image_to_array(image_dir):
    try:
        image = cv2.imread(image_dir)
        if image is not None :
            image = cv2.resize(image, default_image_size)
            return img_to_array(image)
        else :
            return np.array([])
    except Exception as e:
        print("Error :", e)
        return None
```

```
image_list, label_list = [], []
try:
    print("[INFO] Loading images ...")
    root_dir = listdir(directory_root)
    for directory in root_dir :
        # remove .DS_Store from list
        if directory == ".DS_Store" :
            root_dir.remove(directory)

    for Skin_folder in root_dir :

        Skin_cancer_folder_list =
listdir(f"{directory_root}/{Skin_folder}")

        for cancer_folder in Skin_cancer_folder_list :
            # remove .DS_Store from list
            if cancer_folder == ".DS_Store" :
                Skin_cancer_folder_list.remove(cancer_folder)

        for Skin_cancer_folder in Skin_cancer_folder_list:
            print(f"[INFO] Processing {Skin_cancer_folder} ...")
            Skin_cancer_folder_list =
listdir(f"{directory_root}/{Skin_folder}/{Skin_cancer_folder}/")

            for single_plant_disease_image in
Skin_cancer_folder_list :
                if single_plant_disease_image == ".DS_Store" :

Skin_cancer_folder_list.remove(single_plant_disease_image)

            for image in Skin_cancer_folder_list[:300]:
                image_directory =
f"{directory_root}/{Skin_folder}/{Skin_cancer_folder}/{image}"
                if image_directory.endswith(".jpg") == True or
image_directory.endswith(".JPG") == True:

image_list.append(convert_image_to_array(image_directory))
                    label_list.append(Skin_cancer_folder)
    print("[INFO] Image loading completed")
except Exception as e:
    print(f"Error : {e}")
```

ln [6]: pelabelan citra

```
label_binarizer = LabelBinarizer()
image_labels = label_binarizer.fit_transform(label_list)
n_classes = len(label_binarizer.classes_)
labels = os.listdir(directory_root)
print("Consist of " + str(n_classes) + " Classes")
```

ln [7]: normalisasi data citra

```
print("Normalizing data ..")
np_image_list = np.array(image_list, dtype=np.float16) / 225.0
```

ln [8]: *Splitting* data

```
x_train, x_test, y_train, y_test =
train_test_split(np_image_list,image_labels, test_size=0.2)
len(x_test), len(x_train)
```

ln [9]: augmetasi citra

```
aug = ImageDataGenerator(
    rotation_range=25,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.2,
    zoom_range=0.2,
    vertical_flip=True,
    horizontal_flip=True,
    fill_mode="nearest")
```

ln [10]: arsitektur model

```
#Arsitektur VGG16
import tensorflow as tf
from tensorflow.keras.applications.VGG16 import VGG16
model = tf.keras.Sequential([
    VGG16(
        include_top=False,
        weights='imagenet',
        input_shape=(224, 224, 3)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(1024, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(10, activation='softmax')
])
```

```
#Arsitektur DenseNet121
import tensorflow as tf
from tensorflow.keras.applications.densenet import DenseNet121
model = tf.keras.Sequential([
    DenseNet121(
        include_top=False,
        weights='imagenet',
        input_shape=(224, 224, 3)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(1024, activation='relu'),
    tf.keras.layers.Dropout(0.5),
```

```
    tf.keras.layers.Dense(10, activation='softmax')
])
```

```
#Arsitektur NASNetMobile
import tensorflow as tf
from tensorflow.keras.applications.nasnet import NASNetMobile
model = tf.keras.Sequential([
    NASNetMobile(
        include_top=False,
        weights='imagenet',
        input_shape=(224, 224, 3)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(1024, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(10, activation='softmax')
])
```

```
optimizer = Adam(lr=0.0001)
model.compile(loss='categorical_crossentropy', optimizer=optimizer,
metrics=['accuracy'])
model.summary()
```

ln [12]: *training* model

```
%%time
history = model.fit(
    aug.flow(x_train, y_train, batch_size=64),
    validation_data=(x_test, y_test),
    steps_per_epoch=len(x_train) // 64,
    epochs=100,
    verbose=1)
```

```
scores = model.evaluate(x_test, y_test)
print(f"Test Accuracy: {scores[1]*100}")
```

ln [14]: menampilkan plot *accuracy*

```
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(acc) + 1)
#Train and validation accuracy
plt.plot(epochs, acc, 'b', label='Training accurarcy')
plt.plot(epochs, val_acc, 'r', label='Validation accurarcy')
plt.title('Training and Validation accurarcy')
plt.legend()

plt.figure()
#Train and validation loss
plt.plot(epochs, loss, 'b', label='Training loss')
plt.plot(epochs, val_loss, 'r', label='Validation loss')
plt.title('Training and Validation loss')
plt.legend()
plt.show()
```

ln [15]: menampilkan *Confusion Matrix*

```
import seaborn as sns

ax = plt.subplot()
sns.heatmap(model_densenet, annot= True, ax = ax)

ax.set_xlabel('Predict labels')
ax.set_title('Confusion Matrix')
ax.xaxis.set_ticklabels(label_binarizer.classes_, rotation=90)
ax.yaxis.set_ticklabels(label_binarizer.classes_, rotation=0)
```

ln [16]: menampilkan kurva ROC

```
plt.figure(1, figsize=(10, 10))
plt.plot([0, 1], [0, 1], 'k--')
for i in range(len(label_binarizer.classes_)):
   fpr, tpr, thresholds = roc_curve(y_test[:, i], model_pred[:, i])
   individual_auc = auc(fpr, tpr)
   plt.plot(fpr, tpr, label= (label_binarizer.classes_[i] + '(area =
{})'.format(individual_auc)))

plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC curve')
plt.legend(loc='best')
plt.show()
```

ln [17]: menampilakan *precision, Recall, F1-score*

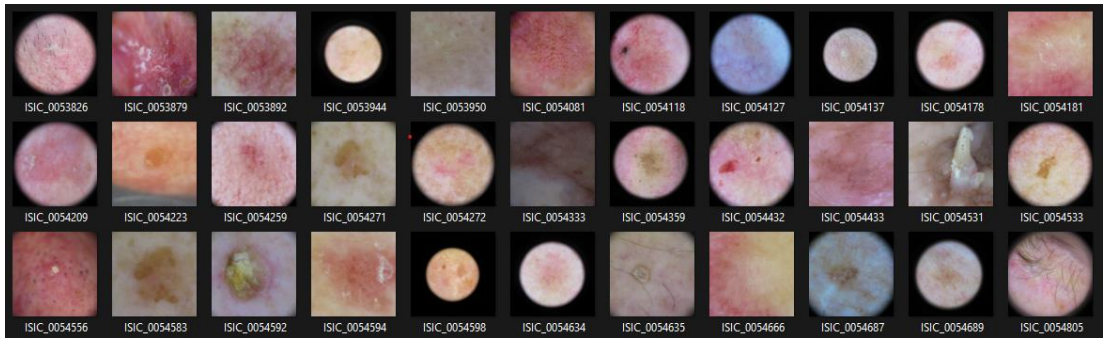
```
from Sklearn.metrics import classification_report
print (classification_report(y_test,y_pred))
```

source code Android

```
https://github.com/Ajrana/AppSkinCancer
```

2. Lampiran Gambar

Actinic keratosis

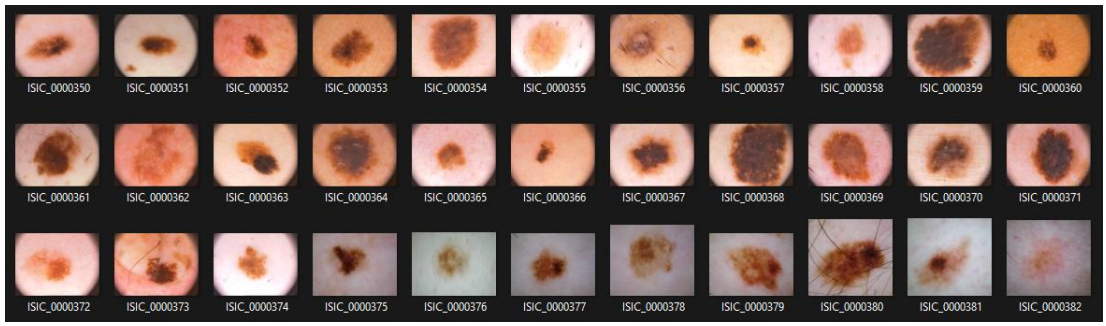


basal cell carcinoma

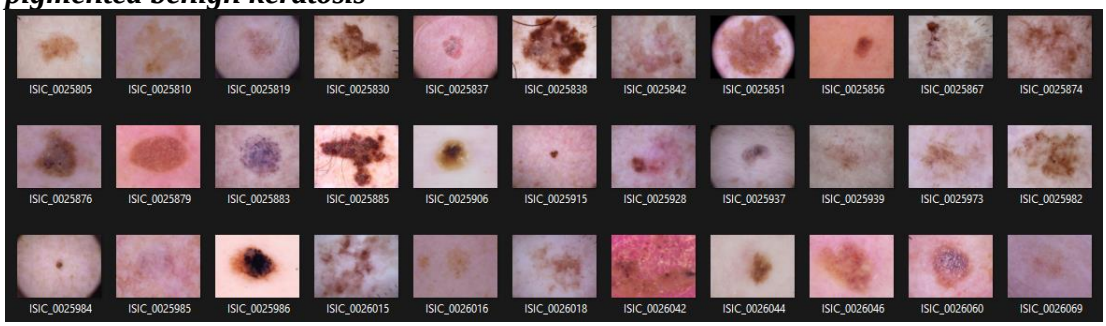
Dermatofibroma

melanoma

Nevus

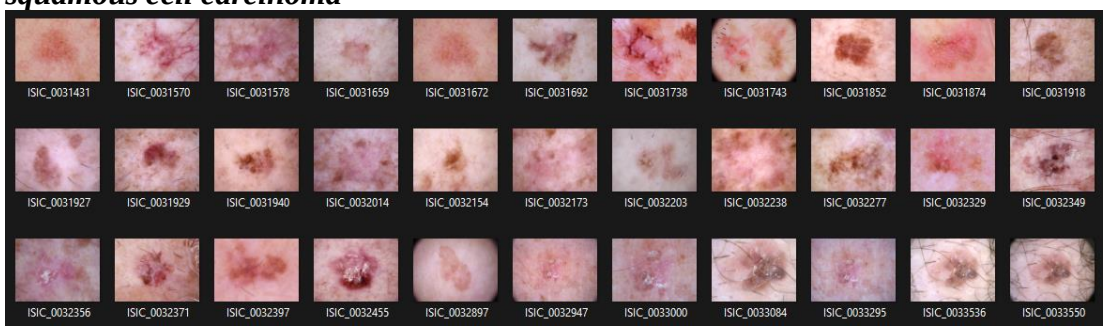


pigmented benign keratosis



seborrheic keratosis

squamous cell carcinoma



vascular lesion