PERANCANGAN APLIKASI PENDETEKSI PENYAKIT DAUN PADI MENGGUNAKAN ARSITEKTUR *MOBILENETV2*

SKRIPSI

MAXI MILIAN CRISTOPORUS NUEL JAYA

H071181314

PROGRAM STUDI SISTEM INFORMASI

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2022

PERANCANGAN APLIKASI PENDETEKSI PENYAKIT DAUN PADI MENGGUNAKAN ARSITEKTUR *MOBILENETV2*

SKRIPSI

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

MAXI MILIAN CRISTOPORUS NUEL JAYA

H071181314

PROGRAM STUDI SISTEM INFORMASI

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2022

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama : Maxi Milian Cristoporus Nuel Jaya

NIM : H071181314

Program Studi : Sistem Informasi

Jenjang : S1

Menyatakan dengan ini bahwa karya tulisan saya dengan judul:

Perancangan Aplikasi Pendeteksi Penyakit Daun Padi Menggunakan Arsitektur *MobileNetV2*

adalah benar hasil karya saya sendiri, bukan hasil plagiat dan belum pernah dipublikasikan dalam bentuk apapun.

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

Makassar, 28 Oktober 2022

Yang menyatakan,

Maxi Milian Cristoporus Nuel Jaya

H071181314

PERANCANGAN APLIKASI PENDETEKSI PENYAKIT DAUN PADI MENGGUNAKAN ARSITEKTUR MOBILENETV2

Disusun dan diajukan oleh:

Maxi Milian Cristoporus Nuel Jaya

H071181314

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.

Menyetujui,

Pembimbing Utama	**Pembimbing Pertama**
Dr.Eng. Armin Lawi, S.Si., M.Eng.	**A. Muh Amil Siddik, S,Si., M.Si**
NIP. 197204231995121001	NIP. 199110032019031

Ketua Program Studi

Dr. Hendra, S.Si., M.Kom.

NIP. 197601022002121001

Pada tanggal 28 Oktober 2022

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh :

Nama : Maxi Milian Cristoporus Nuel Jaya
NIM : H071181314
Program Studi : Sistem Informasi
Judul Skripsi : Perancangan Aplikasi Pendeteksi Penyakit Daun Padi Menggunakan Arsitektur MobileNetV2

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
Ketua	: Dr.Eng. Armin Lawi, S.Si., M.Eng.	(.....)
Sekretaris	: A. Muh. Amil Siddik, S.Si., M.Si.	(.....)
Anggota	: Ir. Eliyah Acantha Manapa Sampetoding, S.Kom., M.Kom.	(.....)
Anggota	: Muhammad Sadno, S.Si., M.Si.	(.....)

Ditetapkan di : Makassar
Tanggal : 28 Oktober 2022

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN UNIVERSITAS HASANUDDIN DEPARTEMEN MATEMATIKA FAKULTAS MIPA

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya penulis dapat menyelesaikan skripsi ini. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Komputer pada Program Studi Sistem Informasi Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin.

Penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada kedua orang tua tercinta, Ayahanda **Yohannes Firdaus Jaya**, dan Ibunda **Tita Anna Tangdilintin**, yang dengan penuh kesabaran dalam mengasuh dan mendidik penulis, yang senantiasa mencurahkan kasih saying yang tak pernah putus serta memberikan dukungan doa yang tulus sehingga penulis dapat menyelesaikan skripsi ini. Ucapan terima kasih juga kepada saudara-saudari tercinta, **Richy Epifanius Jaya** dan **Priyankan Clara Beatrix Tibe** serta seluruh keluarga yang senantiasa memberikan dukungan dan doa bagi penulis dalam menyelesaikan skripsi ini.

Penulisan skripsi ini dapat terselesaikan berkat bantuan dan motivasi dari berbagai pihak. Oleh karena itu, penulis menyampaikan ucapan terima kasih dan penghargaan yang tulus kepada:

1. Rektor Universitas Hasanuddin Makassar **Prof. Dr. Ir. Jamaluddin Jompa, M.Sc**.
2. Bapak Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam **Dr. Eng. Amiruddin**, **M.Si** dan para Wakil Dekan serta seluruh staf yang telah memberikan bantuan selama penulis mengikuti Pendidikan di FMIPA Universitas Hasanuddin.
3. Bapak **Prof. Dr. Nurdin, S.Si., M.Si**, selaku 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. Hendra, S.Si., M.Kom.** 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**, selaku dosen pembimbing yang telah menyediakan waktu, tenaga, dan pikiran untuk mengarahkan saya dalam penyusunan skripsi ini.
6. Bapak **Ir. Eliyah Acantha Manapa Sampetoding, S.Kom., M.Kom.** dan Bapak **Muhammad Sadno, S.Si., M.Si.** selaku dosen penguji terima kasih atas ilmu yang diberikan selama proses perkuliahan serta saran dan masukan yang diberikan dalam proses penyusunan skripsi ini.
7. Teman-teman seperjuangan **Sistem Informasi 2018** tercinta yang selalu menemani, menguatkan dan menyemangati selama masa perkuliahan, serta terima kasih terkhusus kepada **Islah**, **Cecil**, dan **Ajrana** yang turut membantu dan memudahkan penulis dalam penyusunan skripsi ini.
8. Kepada kanda-kanda ku **Islah**, **Ulil**, **Rifky**, **Ramdan**, **Luthfi** terima kasih atas kebersamaan, kepedulian, dan canda tawa yang telah kita lewati selama ini, semoga kesuksesan selalu kita dapatkan dalam setiap langkah-langkah kita.
9. 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 sempurna dikarenakan terbatasnya pengalaman dan pengetahuan yang dimiliki penulis. Oleh karena itu, penulis mengharapkan segala bentuk saran serta masukan bahkan kritik yang membangun dari berbagai pihak. Semoga skripsi ini dapat bermanfaat bagi yang membacanya, terutama untuk pengembangan ilmu pengetahuan..

Makassar, 28 Oktober 2022

Maxi Milian Cristoporus Nuel Jaya

PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Hasanuddin, saya yang bertanda tangan dibawah ini:

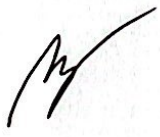
Nama : Maxi Milian Cristoporus Nuel Jaya
NIM : H071181314
Program Studi : Sistem Informasi
Departemen : Matematika
Fakultas : Matematika dan Ilmu Pengetahuan Alam
Jenis karya : Skripsi

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:

Perancangan Aplikasi Pendeteksi Penyakit Daun Padi Menggunakan Arsitektur *MobileNetV2*

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 pemilih Hak Cipta.

Yang Menyatakan



Maxi Milian Cristoporus Nuel Jaya

ABSTRAK

Padi adalah tanaman penghasil beras yang menjadi sumber bahan pokok di Indonesia. Berdasarkan hasil Survei KSA, pada tahun 2021, luas panen padi mengalami penurunan sebanyak 245,47 ribu hektar (2,30 persen) dibandingkan tahun 2020. Tingkat keberhasilan jumlah padi yang dipanen menjadi hal yang sangat berpengaruh. Serangan organisme penggangu tanaman yang sering menyerang tanaman padi khususnya pada bagian daun dapat menyebabkan gagal panen. Keterlambatan proses diagnosis secara manual menyebabkan penyakit yang ada pada tanaman padi mencapai tahap kritis karena minimnya pengetahuan petani dan menganggap gejala tersebut sudah biasa tejadi pada masa tanam, sehingga menimbulkan terjadinya gagal panen. Salah satu metode yang dapat digunakan untuk mengklasifikasikan data berbentuk citra adalah *Convolutional Neural Network* (CNN). Penelitian ini menggunakan salah satu arsitektur CNN yaitu *MobileNetV2* untuk mendeteksi penyakit daun padi melalui input berupa citra. Jenis penyakit daun padi yang digunakan pada penelitian ini yaitu *Bacterial Leaf Blight*, *Brown Spot*, dan *Leaf Smut*. Dengan melakukan *training* sebanyak 50 *epochs* model menghasilkan kinerja terbaik dengan 100% *training accuracy* dan 95,83% *validation accuracy*. Kemudian untuk *precision*, *recall*, dan *f1-score* sebanyak 77% nilai yang didapatkan berada diatas 0,93. Arsitektur *MobileNetV2* selanjutnya dideploy menggunakan *TensorFlow Lite* pada aplikasi android bernama PadiKu.

Kata kunci: *Convolutional Neural Network* (CNN), *MobileNetV2*, penyakit daun padi, *transfer learning*, aplikasi android.

ABSTRACT

Rice is a rice-producing plant which is a source of staple food in Indonesia. Based on the results of the KSA Survey, in 2021, the rice harvested area has decreased by 245.47 thousand hectares (2.30 percent) compared to 2020. The success rate of the amount of rice harvested is a very influential thing. The attack of plant-disturbing organisms that often attack rice plants, especially on the leaves can cause crop failure. The delay in the manual diagnosis process causes the disease in rice plants to reach a critical stage due to the lack of knowledge of farmers and assume these symptoms are common during the planting period, resulting in crop failure. One method that can be used to classify data in the form of images is Convolutional Neural Network (CNN). This study uses one of the CNN architectures, namely MobileNetV2 to detect rice leaf disease through image input. The types of rice leaf diseases used in this study were Bacterial Leaf Blight, Brown Spot, and Leaf Smut. By training as many as 50 epochs the model produces the best performance with 100% training accuracy and 95.83% validation accuracy. Then for precision, recall, and f1-score as much as 77% the value obtained is above 0.93. The MobileNetV2 architecture was then deployed using TensorFlow Lite on an android application called PadiKu.

Keywords: Convolutional Neural Networks (CNN), MobileNetV2, rice leaf disease, transfer learning, android application.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Padi adalah tanaman penghasil beras yang menjadi sumber bahan pokok di Indonesia. Berdasarkan hasil Survei KSA, pada tahun 2021, luas panen padi mencapai sekitar 10,41 juta hektar atau mengalami penurunan sebanyak 245,47 ribu hektar (2,30 persen) dibandingkan tahun 2020. Sementara itu, produksi padi tahun 2021 yaitu sebesar 54,42 juta ton GKG. Jika dikonversikan menjadi beras, produksi beras tahun 2021 mencapai sekitar 31,36 juta ton, atau turun sebesar 140,73 ribu ton (0,45 persen) dibandingkan dengan produksi beras tahun 2020 (BPS, 2022).

Tingkat keberhasilan jumlah padi yang dipanen menjadi hal yang sangat berpengaruh. Serangan organisme penggangu tanaman (OPT) yang sering menyerang tanaman padi khususnya pada bagian daun dapat menyebabkan gagal panen. Keterlambatan proses diagnosis secara manual menyebabkan penyakit yang ada pada tanaman padi mencapai tahap kritis karena minimnya pengetahuan petani dan menganggap gejala tersebut sudah biasa tejadi pada masa tanam, sehingga menimbulkan terjadinya gagal panen. Kontrol yang tepat bukan hanya ketika serangan sudah terjadi, tetapi yang paling penting adalah tindakan pencegahan (Alidrus dkk., 2021).

Pada tahun 2019, Kementerian ATR/BPN menetapkan luas lahan baku sawah nasional 2019 berdasarkan Keputusan Menteri ATR/Kepala BPN No. 686/SK-PG.03.03/XII/2019, tanggal 17 Desember 2019, tentang Penetapan Luas Lahan Baku Sawah Nasional Tahun 2019 yaitu sebesar 7.463.948 hektar (BPS, 2020). Karena luas lahan sawah di Indonesia yang mencapai 7.463.948 hektar, diharapkan penyakit daun padi dapat diidentifikasi secara cepat, akurat, dan tidak menghabiskan waktu yang lama.

Salah satu metode yang dapat digunakan untuk melakukan klasifikasi adalah dengan menggunakan *Artificial Neural Network* (Suyanto, 2018). *Artificial Neural Network* (ANN) adalah sistem pemrosesan komputasi yang terinspirasi oleh cara sistem saraf biologis (seperti otak manusia) beroperasi (O'Shea & Nash, 2015). Salah satu kelas *deep feedforward artificial neural network* yang banyak diaplikasikan pada analisis citra adalah *Convolutional Neural Networks* atau biasa disingkat CNN (Suyanto, 2018).

Untuk mengimplementasikan CNN terhadap metode klasifikasi, diperlukan data citra yang digunakan sebagai *input*. Untuk beberapa kasus, jumlah data yang tersedia sedikit dan terbatas. Akibatnya model yang dihasilkan memiliki kinerja jauh di bawah ekspektasi dan juga memberikan akurasi yang rendah sehingga model tersebut tidak dapat dikatakan model yang baik (Barman dkk., 2019). *Transfer learning* merupakan teknik *deep learning* di mana model dilatih untuk belajar dan menyimpan pengetahuan dari satu masalah dan menggunakan model yang sama untuk masalah lain yang sejenis. Selanjutnya model CNN yang sudah dilatih dari kumpulan data besar diimplementasikan pada klasifikasi citra yang lain (Rajayogi dkk., 2019). Untuk itu mengatasi masalah dataset yang terbatas dapat digunakan metode *Transfer learning*.

Maka dari itu, penulis tertarik untuk melakukan penelitian dengan menggunakan model *transfer learning* CNN pada dataset penyakit daun padi. Sehingga peneliti memutuskan untuk membuat penelitian yang berjudul "Perancangan Aplikasi Pendeteksi Penyakit Daun Padi Menggunakan Arsitektur *MobileNetV2*".

1.2 Rumusan Masalah

Adapun rumusan masalah dalam penelitian ini adalah:

1. Bagaimana membangun dan mengevaluasi kinerja model klasifikasi penyakit daun padi dengan arsitektur *MobileNetV2*?

2. Bagaimana membanguan dan menguji aplikasi android menggunakan metode *Black Box Testing* pada aplikasi *mobile* yang menggunakan model klasifikasi penyakit daun padi?

1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah:

1. *Dataset* yang digunakan pada penelitian ini adalah *Dataset Rice Leaf Disease* (archive.ics.uci.edu/ml/datasets/Rice+Leaf+Diseases).
2. Arsitektur *transfer learning* yang digunakan adalah *MobileNetV2*.
3. Penyakit daun padi yang diteliti adalah *Brown spot*, *Leaf smut*, dan *Bacterial blight*.
4. Input citra penyakit daun padi pada aplikasi adalah penyakit *Brown spot*, *Leaf smut*, dan *Bacterial blight.*
5. Aplikasi *mobile* yang digunakan berupa *android*.

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah, maka tujuan dari penelitian ini adalah:

1. Menganalisis kinerja yang dihasilkan oleh arsitektur *MobileNetV2* untuk klasifikasi data citra penyakit daun padi dengan data citra yang terbatas.
2. Membangun dan menguji aplikasi android menggunakan metode *Black Box Testing* pada aplikasi *mobile* yang menggunakan model klasifikasi penyakit daun padi.

1.5 Manfaat Penelitian

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

1. Menjadi sumber informasi mengenai kinerja model klasifikasi penyakit daun padi menggunakan arsitektur *MobileNetV2*.
2. Menyediakan aplikasi *mobile* berbasis *android* yang dapat mengenali citra penyakit daun padi.

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 "*Identification and Recognition of Rice Diseases and Pests Using Convolutional Neural Network*s" yang diteliti oleh Rahman pada tahun 2020 dengan tujuan membandingkan kinerja CNN dengan model *VGG16*, *InceptionV3*, *MobileNetV2*, *NasNet Mobile*, *SqueezeNet* dan *SimpleCNN*. Dimana data yang digunakan diambil secara langsung dengan jumlah 1426 citra dan total 9 kelas. Kemudian dilakukan teknik *fine tuning* untuk melatih kembali model *Transfer Learning*. Hasil akurasi yang didapatkan sebesar 97,12% pada model *VGG16*, 96,37% pada model *InceptionV3*, 96,12% pada model *MobileNetV2*, 96,95% pada model *NasNet Mobile*, 92,5% pada model *SqueezeNet*, dan 94,33% pada model *SimpleCNN* (Rahman dkk., 2018).

Penelitian penyakit daun padi dengan menggunakan aristektur CNN juga pernah dilakukan oleh Hossain pada tahun 2020 dengan judul "*Rice Leaf Diseases Recognition Using Convolutional Neural Networks*". Penelitian ini menggunakan dataset dari *International Rice Research Institute* (IRRI) dan *Bangladesh Rice Research Institute* (BRRI) dengan jumlah 323 citra untuk 5 kelas yaitu *Blast*, *Bacterial leaf blight*, *Brownspot*, *Sheath blight*, dan *Tungro*. Kemudian menggunakan arsitektur *custom* CNN dan teknik augmentasi didapatkan hasil akurasi tertinggi 99,78% dan 97,35% untuk masing-masing *training* dan *testing* (Hossain dkk., 2020).

Selanjutnya penelitian juga pernah dilakukan oleh Zhou pada tahun 2019 dengan judul "*Rapid Detection of Rice Disease Based on FCM-KM and Faster R-CNN Fusion*" diambil pada jurnal IEEE. metode untuk mendeteksi penyakit padi secara cepat

berdasarkan fusi FCM-KM dan *Faster* R-CNN digunakan untuk mengatasi berbagai masalah pada citra penyakit padi, seperti *noise*, tepi citra kabur, interferensi latar belakang yang besar dan akurasi deteksi yang rendah. data citra penyakit padi dikumpulkan dari ladang percobaan beras standar tinggi di *Hunan Rice Research Institute* dengan jumlah 3010 citra dan 3 kelas. Akurasi dan waktu yang diperlukan untuk mendeteksi penyakit *rice blast*, *bacterial blight* dan *blight* masing-masing adalah 96,71%/0,65 detik, 97,53%/0,82 detik dan 98,26%/0,53 detik, yang menunjukkan bahwa metode ini mampu mendeteksi penyakit padi dan memperbaiki akurasi identifikasi algoritma *Faster* R-CNN, sekaligus mengurangi waktu yang dibutuhkan untuk identifikasi (Zhou dkk., 2019).

Penelitian yang dilakukan Liu dan Wang pada tahun 2020 dengan judul "*Early recognition of tomato gray leaf spot disease based on MobileNetV2-YOLOV3 model*" menggunakan salah satu arsitektur CNN dan juga memanfaatkan metode *transfer learning*. 2166 citra asli dikumpulkan, termasuk hari berawan, cerah dan hujan, agar mencakup semua kondisi pencahayaan. Peneliti menggunakan arsitektur *MobileNetV2-YOLOv3* yang menggunakan arsitektur *MobileNetV2* sebagai tulang punggungnya, metode pra-pelatihan yang menggabungkan pelatihan *mixup* dan *transfer learning* digunakan untuk meningkatkan kemampuan generalisasi model. Dalam himpunan data pengujian gambar yang ditangkap di bawah latar belakang cahaya yang cukup tanpa naungan daun, skor F1 dan nilai AP adalah 94,13% dan 92,53%, dan nilai IOU rata-rata adalah 89,92%. Di semua set pengujian, skor F1 dan nilai AP adalah 93,24% dan 91,32%, dan nilai IOU rata-rata adalah 86,98%. Kecepatan deteksi objek dapat mencapai 246 *frame*/s pada GPU, kecepatan ekstrapolasi atau gambar 416 × 416 tunggal adalah 16,9 ms, kecepatan deteksi pada CPU dapat mencapai 22 *frame*/s, kecepatan ekstrapolasi adalah 80,9 ms dan memori yang digunakan oleh model adalah 28 MB (Liu & Wang, 2020).

2.2 Penyakit Daun Padi

Tanaman padi dipengaruhi oleh berbagai jenis penyakit. Penyakit padi dapat dikelompokkan di bawah varietas yang berbeda seperti penyakit yang disebabkan oleh bakteri, penyakit yang disebabkan oleh virus dan gangguan jamur (Premi dkk., 2019).

2.2.1 *Bacterial Leaf Blight*

Bacterial Leaf Blight disebabkan oleh bakteri *Xanthomonas oryzae pv. oryzicola*. Tanaman yang terinfeksi menunjukkan gejala pencoklatan dan pengeringan daun. Dalam kondisi yang parah, ini dapat mengakibatkan penurunan berat butir karena hilangnya ruang fotosintesis. Mikroorganisme ini tinggal di daerah dengan suhu panas dan kelembaban tinggi. Penyakit ini ditularkan melalui biji dan tunggul yang terinfeksi ke musim tanam berikutnya. Ini akan terjadi di daerah mana pun *X. oryzae pv*. Mikroorganisme *oryzicola* yang berhasil menginfeksi daun padi, di dalam air, atau di dalam puing-puing yang tersisa saat panen (Premi dkk., 2019).

Gambar 2. 1 Daun padi yang terinfeksi penyakit *Bacterial Leaf Blight* (archive.ics.uci.edu/ml/datasets/Rice+Leaf+Diseases)

2.2.2 *Brown Spot*

Brown Spot secara historis sebagian besar sering diabaikan sebagai salah satu penyakit padi yang paling umum dan paling merusak. *Brown Spot* merupakan penyakit jamur yang menyerang koleoptil, daun, pelepah daun, cabang daun, *glumes*, dan bulir. Gejala penyakit yang paling terlihat adalah banyaknya bintik-bintik besar pada daun yang dapat membunuh seluruh daun. Ketika infeksi terjadi pada biji, biji-bijian yang

tidak terisi atau biji berbintik atau berubah warna terbentuk. Penyakit ini dapat berkembang di daerah dengan kelembaban relatif tinggi (86-100%) dan suhu antara 16 dan 36°C. Ini umum terjadi di tanah yang tidak tergenang dan kekurangan nutrisi, atau di tanah yang banyak mengandung zat beracun. Agar infeksi terjadi, daun harus basah selama 8−24 jam. Jamur dapat bertahan hidup di dalam biji selama lebih dari empat tahun dan dapat menyebar dari tanaman ke tanaman melalui udara. Sumber utama penyakit *Brown Spot* di lahan meliputi:

1. Benih yang terinfeksi, yang menimbulkan bibit yang baru tumbuh terinfeksi
2. *Volunteer rice*
3. Puing-puing padi yang terinfeksi
4. Rumput liar

Brown Spot dapat terjadi pada semua tahap tanaman, tetapi infeksi yang paling kritis adalah selama bibit maksimum hingga tahap pematangan tanaman padi (IRRI, 2019).

Gambar 2. 2 Daun Padi yang terinfeksi penyakit *Brown Spot*
(archive.ics.uci.edu/ml/datasets/Rice+Leaf+Diseases)

2.2.3 *Leaf Smut*

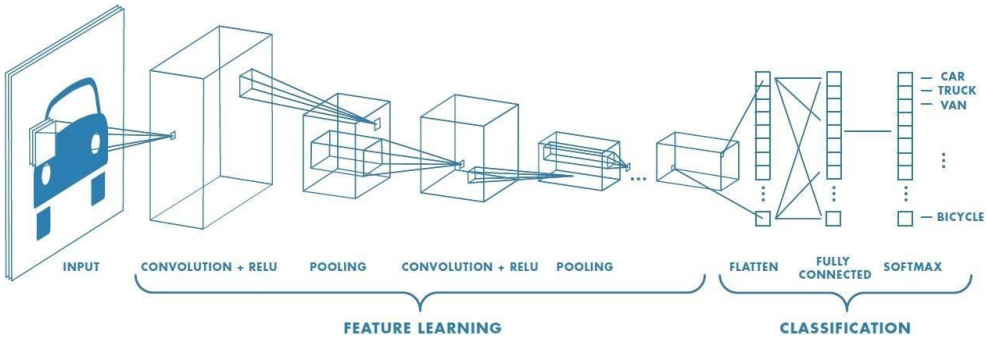
Penyakit *Leaf Smut*, yang disebabkan oleh jamur *Entyloma oryzae*, merupakan penyakit padi yang tersebar luas. Jamur menghasilkan bintik-bintik hitam yang agak menonjol, bersudut, di kedua sisi daun. Meski jarang, jamur ini juga bisa menghasilkan bintik-bintik pada pelepah daun. Bintik hitam memiliki panjang sekitar 0,5 hingga 5,0 milimeter dan lebar 0,5 hingga 1,5 milimeter. Banyak bintik dapat ditemukan pada daun yang sama, tetapi bintik-bintik itu tetap berbeda satu sama lain. Epidermis pecah saat basah dan melepaskan spora hitam. Daun yang terinfeksi berat menguning, dan

ujung daun mati dan berubah menjadi abu-abu. Jamur disebarkan oleh spora di udara dan menempel pada puing-puing daun yang sakit di tanah. Penyakit *Leaf Smut* terjadi di akhir musim tanam dan menyebabkan sedikit kerugian. Penyakit ini sangat mudah terjadi pada lahan dengan tingkat nitrogen yang tinggi. Tindakan pengendalian tidak dianjurkan (Groth & Hollier, 2020).

Gambar 2. 3 Daun Padi yang terinfeksi penyakit *Leaf Smut*
(archive.ics.uci.edu/ml/datasets/Rice+Leaf+Diseases)

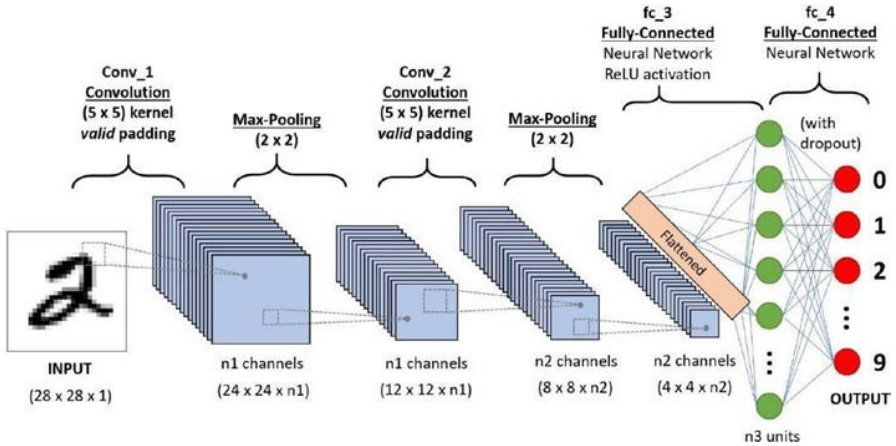
2.3 *Convolutional Neural Network*

Convolutional Network, juga dikenal sebagai *Convolutional Neural Networks*, atau CNN, adalah jenis jaringan saraf tiruan khusus untuk memproses data yang memiliki topologi seperti *grid*. Contohnya termasuk data deret waktu, yang dapat dianggap sebagai data 1-D yang mengambil sampel pada interval waktu reguler, dan data gambar, yang dapat dianggap sebagai data 2-D. Jaringan konvolusi sangat berhasil dalam aplikasi praktis. Nama "*Convolutional Neural Networks*" menunjukkan bahwa jaringan menggunakan operasi matematika yang disebut konvolusi. Konvolusi adalah jenis khusus dari operasi linier. CNN hanyalah jaringan saraf tiruan yang menggunakan konvolusi sebagai pengganti perkalian matriks umum di setidaknya satu lapisan jaringan (Goodfellow dkk., 2016).



Gambar 2. 4 Arsitektur *Convolutional Neural Network*

Convolutional Neural Network (CNN) didasarkan pada konvolusi gambar, dan mendeteksi fitur berdasarkan filter yang dipelajari oleh CNN melalui pelatihan. Misalnya, ketika filter yang diterapkan tidak diketahui, seperti filter untuk mendeteksi tepi atau untuk menghilangkan *Gaussian noise*, tetapi melalui pelatihan CNN, algoritma mempelajari filter pemrosesan gambar sendiri yang mungkin sangat berbeda dari filter pemrosesan gambar biasa. Untuk *Supervised Learning*, filter dipelajari sedemikian rupa sehingga fungsi bobot keseluruhan dikurangi sebanyak mungkin. Umumnya, lapisan konvolusi pertama belajar untuk mendeteksi tepi, sedangkan yang kedua dapat belajar untuk mendeteksi bentuk yang lebih kompleks yang dapat dibentuk dengan menggabungkan tepi yang berbeda, seperti lingkaran dan persegi panjang, dan sebagainya. Lapisan ketiga dan seterusnya mempelajari fitur yang jauh lebih rumit berdasarkan fitur yang dihasilkan di lapisan sebelumnya (El-Amir & Hamdy, 2020).



Gambar 2. 5 Arsitektur CNN digunakan untuk mengklasifikasi citra tulisan tangan

Mengapa Convolutional Neural Networks lebih banyak digunakan daripada jaringan saraf tiruan klasik lainnya dalam konteks *computer vision*?

1. Salah satu alasan utama untuk mempertimbangkan CNN dalam kasus tersebut adalah fitur pembagian bobot CNN, yang mengurangi jumlah parameter yang dapat dilatih dalam jaringan, yang membantu model untuk menghindari *overfitting* dan juga untuk meningkatkan generalisasi.
2. Pada CNN, lapisan klasifikasi dan lapisan ekstraksi fitur belajar bersama-sama, yang membuat *output* model lebih terorganisir dan membuat *output* lebih bergantung pada fitur yang diekstraksi.
3. Implementasi jaringan besar lebih sulit dengan menggunakan jenis jaringan saraf tiruan lain dibandingkan menggunakan CNN (Ghosh dkk., 2019).

2.3.1 *Convolutional Layer*

Konvolusi adalah proses sederhana penerapan matriks (juga disebut kernel atau filter) ke sebuah gambar sehingga dapat mengecilkannya, atau menambahkan beberapa lapisan *padding* agar ukurannya tetap sama. Konvolusi juga digunakan untuk mengekstrak fitur tertentu dari suatu gambar, seperti bentuk, tepi, dan sebagainya. Konvolusi banyak digunakan dalam pemrosesan citra, terutama dalam CNN dan deteksi wajah (Singh, 2019).

Lapisan konvolusi adalah komponen terpenting dari arsitektur CNN. Lapisan ini berisi satu set kernel konvolusi (juga disebut filter), yang dikonvolusikan dengan gambar input (matrix N-dimensi) untuk menghasilkan peta fitur output. Kernel dapat digambarkan sebagai tabel nilai atau angka diskrit, di mana setiap nilai dikenal sebagai berat kernel ini. Selama awal proses pelatihan model CNN, semua bobot kernel ditetapkan dengan angka acak (pendekatan yang berbeda juga tersedia untuk inisialisasi bobot). Kemudian, dengan setiap periode pelatihan, bobot disetel dan kernel dipelajari untuk mengekstraksi fitur yang bermakna (Ghosh dkk., 2019).

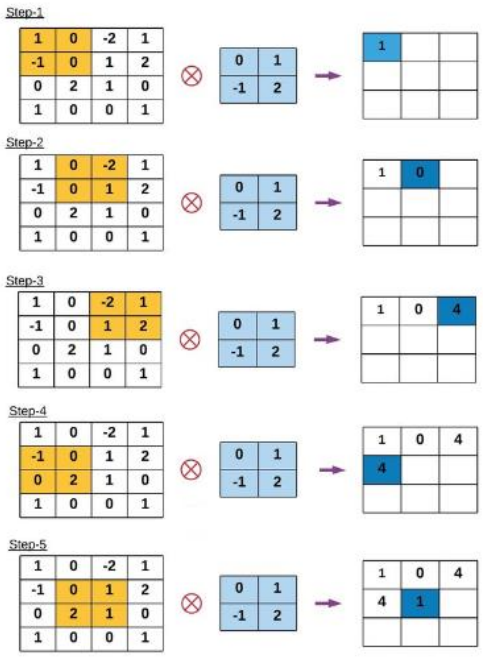
0	1
-1	2

Gambar 2. 6 Sebuah kernal dengan ukuran 2 × 2

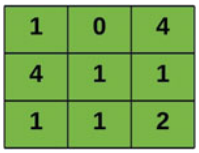
1	0	-2	1
-1	0	1	2
0	2	1	0
1	0	0	1

Gambar 2. 7 Sebuah citra *gray-scale* dengan ukuran 4 × 4

Dalam operasi konvolusi, kernel 2 × 2 Gambar 2.6 digeser di atas semua citra 4 × 4 Gambar 2.7 baik secara horizontal maupun vertikal dan dilakukan *dot product* antara kernel dan gambar input dengan mengalikan nilai yang sesuai antara kernel dan gambar kemudian jumlahkan semua nilai untuk menghasilkan satu nilai skalar di peta fitur keluaran. Proses ini berlanjut sampai kernel tidak bisa lagi bergeser lagi (Ghosh dkk., 2019).



Gambar 2. 8 Ilustrasi 5 langkah pertama dari operasi konvolusi



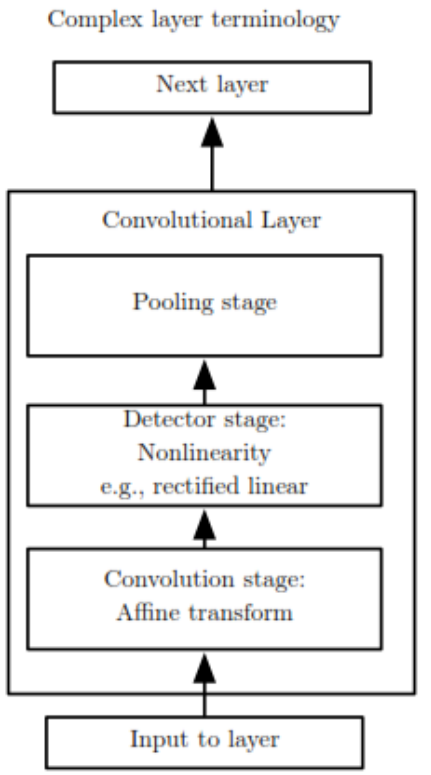
Gambar 2. 9 Peta fitur akhir setelah operasi konvolusi

Kelebihan utama digunakannya lapisan konvolusi adalah:

1. ***Sparse Connectivity***: Dalam *fully connected layer*, setiap neuron dari satu lapisan terhubung dengan setiap neuron di lapisan berikutnya tetapi di CNN sejumlah kecil bobot hadir di antara dua lapisan. Akibatnya, jumlah koneksi atau bobot yang dibutuhkan lebih kecil, dan jumlah memori untuk menyimpan bobot tersebut juga kecil, sehingga dapat menghemat memori. Juga, operasi *dot*(.) secara komputasi lebih ringan daripada perkalian matriks.
2. ***Weight Sharing***: Dalam CNN, tidak ada bobot khusus yang ada di antara dua neuron dari lapisan yang berdekatan tetapi semua bobot bekerja dengan setiap piksel dari matriks input. *Fully connected layer* mempelajari bobot baru untuk setiap neuron, sedangkan CNN mempelajari satu set bobot untuk semua input dan dengan ini CNN mengurangi waktu pelatihan secara drastis (Ghosh dkk., 2019).

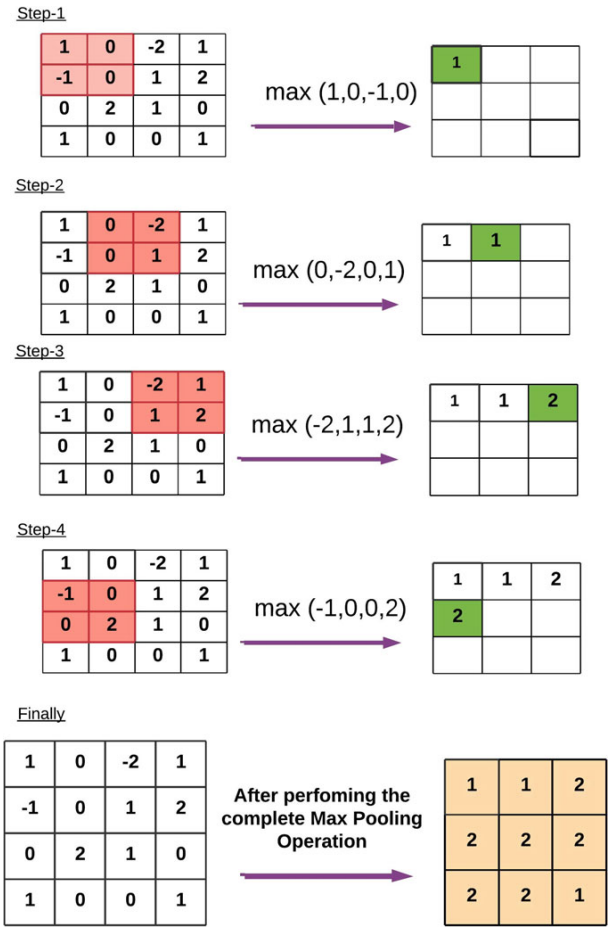
2.3.2 *Pooling*

Sebuah lapisan (*layer*) khas dari CNN terdiri dari tiga tahap yang dapat dilihat pada Gambar 2.10. Pada tahap pertama, lapisan melakukan beberapa konvolusi secara paralel untuk menghasilkan satu set aktivasi linier. Pada tahap kedua, setiap aktivasi linier dijalankan melalui fungsi aktivasi nonlinier, seperti *rectified linear activation function* (*ReLU*). Tahap ini juga biasa disebut tahap detektor. Pada tahap ketiga, fungsi *pooling* digunalan untuk memodifikasi output dari layer lebih lanjut (Goodfellow dkk., 2016).



Gambar 2. 10 Tipikal komponen pada CNN

Lapisan *pooling* digunakan untuk sub-sampel peta fitur (digunakan setelah operasi konvolusi), yaitu mengambil peta fitur ukuran lebih besar dan mengecilkannya ke peta fitur berukuran lebih rendah. Saat mengecilkan peta fitur, fitur (informasi) yang paling dominan tetap dipertahankan di setiap langkah. Operasi *pooling* dilakukan dengan menentukan ukuran *pooled region* dan langkah operasi, mirip dengan operasi konvolusi. Ada berbagai jenis teknik *pooling* yang digunakan dalam berbagai lapisan *pooling* seperti *max pooling*, *min pooling*, *average pooling*, *gated pooling*, *tree pooling*, dll. *Max Pooling* adalah yang paling populer dan paling banyak digunakan (Ghosh dkk., 2019).



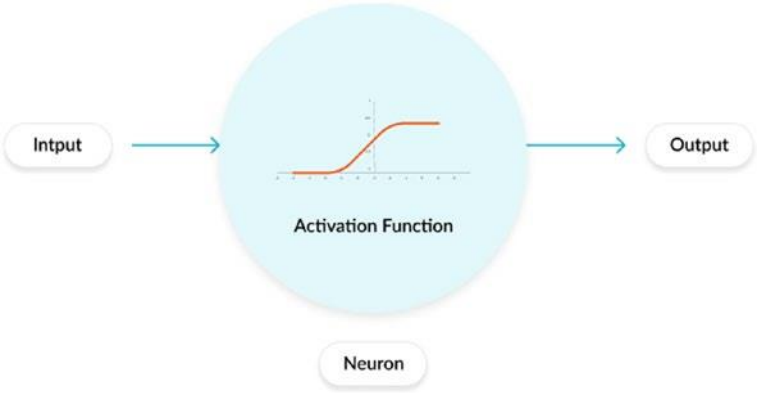
Gambar 2. 11 Ilustrasi langkah awal operasi *max pooling* pada citra 4 × 4

Kelemahan utama dari *pooling layer* adalah terkadang menurunkan kinerja CNN secara keseluruhan. Alasan di balik ini adalah bahwa *pooling layer* membantu CNN untuk menemukan apakah fitur tertentu ada dalam gambar masukan yang diberikan atau tidak, tanpa memperhatikan posisi yang benar dari fitur tersebut (Ghosh dkk., 2019).

2.3.3 *Activation Function*

Fungsi aktivasi pada jaringan saraf tiruan adalah "gerbang" matematis. Konsep utama fungsi aktivasi adalah sebagai *perceptron*, dan fungsi aktivasi pada jaringan saraf tiruan adalah komponen penting dari *Deep Learning*.

Fungsi aktivasi menentukan output dari model *deep learning*; akurasinya; dan juga efisiensi komputasi dari pelatihan model, yang dapat menentukan kinerja jaringan saraf tiruan skala besar. Fungsi aktivasi merupakan bagian yang sangat penting dalam arsitektur jaringan saraf tiruan, karena fungsi aktivasi memiliki pengaruh besar pada kemampuan jaringan saraf untuk melakukan konvergensi dan kecepatan konvergensi. Dalam beberapa kasus, fungsi aktivasi mungkin mencegah jaringan saraf dari konvergen di tempat pertama. Tujuan utamanya adalah untuk mengubah sinyal *input* dari sebuah *node* dalam JST menjadi sinyal *output* (El-Amir & Hamdy, 2020).



Gambar 2. 12 Penempatan fungsi aktivasi

Fungsi aktivasi terletak di antara *input* yang memberi masukan untuk *neuron* dan outputnya menuju ke lapisan berikutnya (Gambar 2.12). Ini bisa sesederhana fungsi yang dapat menghidupkan dan mematikan output neuron, tergantung pada aturan atau ambang batas. Atau dapat berupa transformasi yang memetakan sinyal input menjadi sinyal *output* yang diperlukan agar jaringan saraf dapat berfungsi.

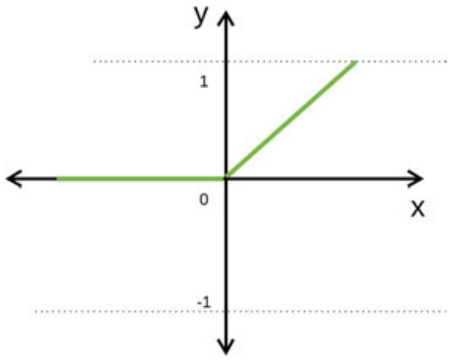
Gambar 2. 13 Langkah-langkah cara kerja fungsi aktivasi di *neuron*

Pada Gambar 2.13 terdapat visualisasi proses kerja fungsi aktivasi, yang dimulai dari *input*: kalikan data *input* dengan bobot, tambahkan bias, melalui fungsi aktivasi, lalu menjadi masukan untuk lapisan berikutnya.

Dalam arsitektur CNN, setelah setiap lapisan yang dapat dipelajari (lapisan dengan bobot, yaitu lapisan *convolutional* dan *Fully Connected*) lapisan aktivasi non-linear digunakan. Perilaku non-linier dari lapisan-lapisan tersebut memungkinkan model CNN untuk mempelajari hal-hal yang lebih kompleks dan mengelola untuk memetakan *input* ke *output* secara nonlinier. Fitur penting dari fungsi aktivasi adalah fungsi tersebut harus dapat dibedakan agar model dapat melihat kesalahan propagasi balik untuk melatih model. Fungsi aktivasi yang paling umum digunakan dalam jaringan saraf tiruan (termasuk CNN) dijelaskan di bawah ini (Ghosh dkk., 2019).

1. **ReLU,** *Rectifier Linear Unit* (ReLU) adalah fungsi aktivasi yang paling umum digunakan dalam *Convolutional Neural Networks*. Ini digunakan untuk mengubah semua nilai *input* menjadi bilangan positif. Kelebihan dari ReLU adalah membutuhkan beban komputasi yang sangat minim dibandingkan dengan yang lain. Representasi matematis dari ReLU adalah:

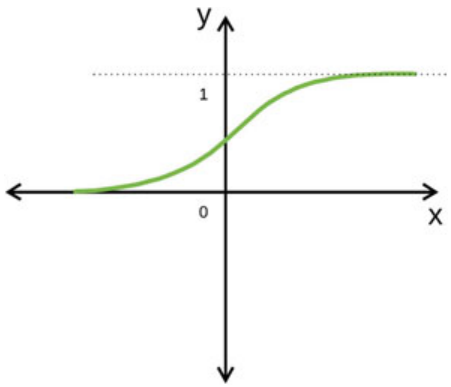
$$f(x)_{ReLU} = \boldsymbol{max}(0, x) \tag{1}$$



Gambar 2. 14 Fungsi ReLU

2. ***Sigmoid***, Fungsi aktivasi *sigmoid* mengambil bilangan real sebagai *input* dan mengikat *output*nya dalam kisaran [0,1]. Kurva fungsi sigmoid berbentuk ‘S’. Representasi matematis dari *sigmoid* adalah:

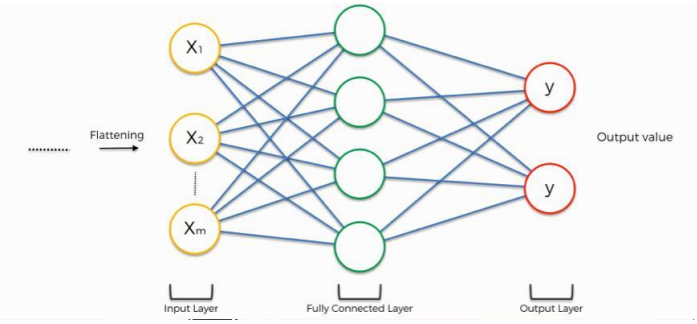
$$f(x)_{sigm} = \frac{1}{1 + e^{-x}} \tag{2}$$



Gambar 2. 15 Fungsi *Sigmoid*

2.3.4 *Fully-Connected Layer*

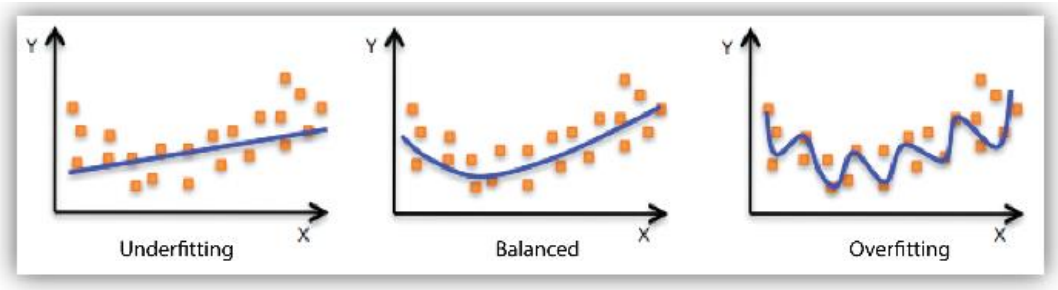
Bagian (atau lapisan) terakhir dari setiap arsitektur CNN (digunakan untuk klasifikasi) terdiri dari lapisan-lapisan yang terhubung penuh (*Fully Connected Layer*), di mana setiap neuron di dalam suatu lapisan terhubung dengan setiap neuron dari lapisan sebelumnya. Lapisan terakhir dari lapisan *Fully-Connected* digunakan sebagai lapisan keluaran (*classifier*) dari arsitektur CNN. *Fully-Connected Layers* adalah jenis *Artificial Neural Network* (ANN) *feed-forward* dan mengikuti prinsip jaringan saraf *Multi Layer Perceptron* (MLP) tradisional. Lapisan FC mengambil input dari *convolutional* atau *pooling* layer akhir, yang berupa sekumpulan metrik (peta fitur) dan metrik tersebut diperhatikan untuk membuat vektor dan vektor ini kemudian diumpankan ke lapisan FC untuk menghasilkan output akhir dari CNN seperti yang ditunjukkan pada Gambar 2.16 (Ghosh dkk., 2019).



Gambar 2. 16 Ilustrasi *Fully Connected Layer*

2.3.5 *Dropout Regularization*

Overfitting terjadi di mana jaringan mungkin menjadi terlalu terspesialisasi dalam jenis data input tertentu dan tidak memperhatikan yang lain. Salah satu teknik untuk membantu mengatasi hal ini adalah penggunaan regularisasi *Dropout*.



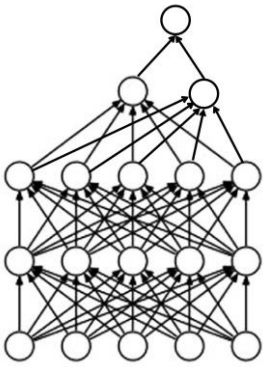
Gambar 2. 17 Grafik *Underfitting*, *Balanced*, *Overfitting*

Model dikatakan *Underfitting* dengan data pelatihan saat model berperforma buruk pada data pelatihan. Ini karena model tidak dapat menangkap hubungan antara contoh input (sering disebut X) dan nilai target (sering disebut Y). Model dikatakan *Overfitting* dengan data latihan saat model berperforma baik pada data pelatihan tetapi tidak berkinerja baik pada data evaluasi. Ini karena model menghafal data yang telah dilihatnya dan tidak dapat menggeneralisasi ke contoh yang tidak terlihat (AWS, 2019).

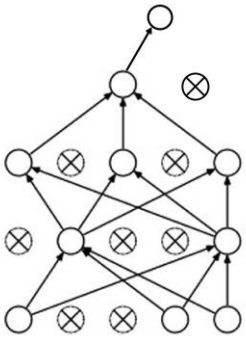
Ketika jaringan saraf sedang dilatih, setiap neuron individu akan memiliki efek pada neuron di lapisan berikutnya. Seiring waktu, terutama di jaringan yang lebih besar, beberapa neuron dapat menjadi terlalu terspesialisasi dan itu berpengaruh sampai lapisan klasifikasi, yang berpotensi menyebabkan jaringan secara keseluruhan

menjadi terlalu terspesialisasi dan menyebabkan *overfitting*. Selain itu, *neuron* tetangga dapat berakhir dengan bobot dan bias yang sama, dan jika tidak dipantau, ini dapat menyebabkan model keseluruhan menjadi terlalu terspesialisasi pada fitur yang diaktifkan oleh *neuron* tersebut (Moroney, 2021).

Dropout adalah salah satu pendekatan yang paling sering digunakan untuk regularisasi. Di sini *neuron* secara acak di nonaktifkan dari jaringan setiap periode pelatihan. Dengan menonaktifkan unit (neuron) model mencoba mendistribusikan daya seleksi fitur ke semua neuron secara merata dan memaksa model untuk mempelajari beberapa fitur independen. Menonaktifkan unit atau neuron berarti, unit yang dinonaktifkan tidak akan mengambil bagian dalam propagasi maju atau propagasi mundur selama proses pelatihan. Namun dalam hal proses pengujian, jaringan skala penuh digunakan untuk melakukan prediksi (Ghosh dkk., 2019).



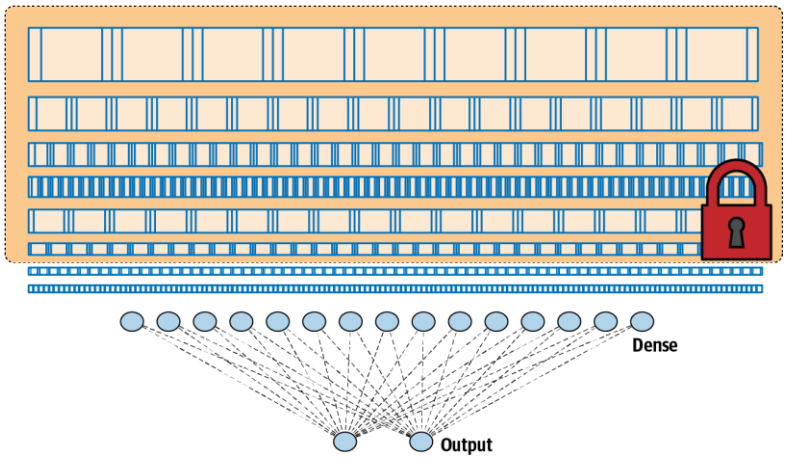
Gambar 2. 18 *Neural network* normal



Gambar 2. 19 Ilustrasi *dropout* pada *neural network*

2.4 *Transfer Learning*

Kinerja model dapat ditingkatkan lebih jauh lagi dengan menggunakan metode yang disebut *transfer learning*. Gagasan di balik *transfer learning* sederhana: alih-alih mempelajari serangkaian filter dari awal, dapat digunakan satu set filter yang mempelajari dataset yang jauh lebih besar, dengan lebih banyak fitur daripada membangun model dari awal. Nilai bobot ini dapat digunakan pada jaringan baru dan kemudian melatih model dengan dataset menggunakan filter yang telah dipelajari sebelumnya. Misalnya, dataset Kucing atau Anjing yang hanya memiliki dua kelas. Model yang sudah ada dan telah dilatih untuk seribu kelas dapat digunakan, tetapi pada titik tertentu beberapa jaringan yang sudah ada sebelumnya harus dibuang dan menambahkan lapisan yang memungkinkan model untuk klasifikasi dua kelas (Moroney, 2021).



Gambar 2. 20 Ilustrasi *Transfer Learning*

Transfer learning mengambil lapisan yang telah dipelajari sebelumnya dari model lain, membekukan atau menguncinya sehingga tidak dapat dilatih, dan kemudian meletakkannya di atas model baru, seperti pada Gambar 2.19.

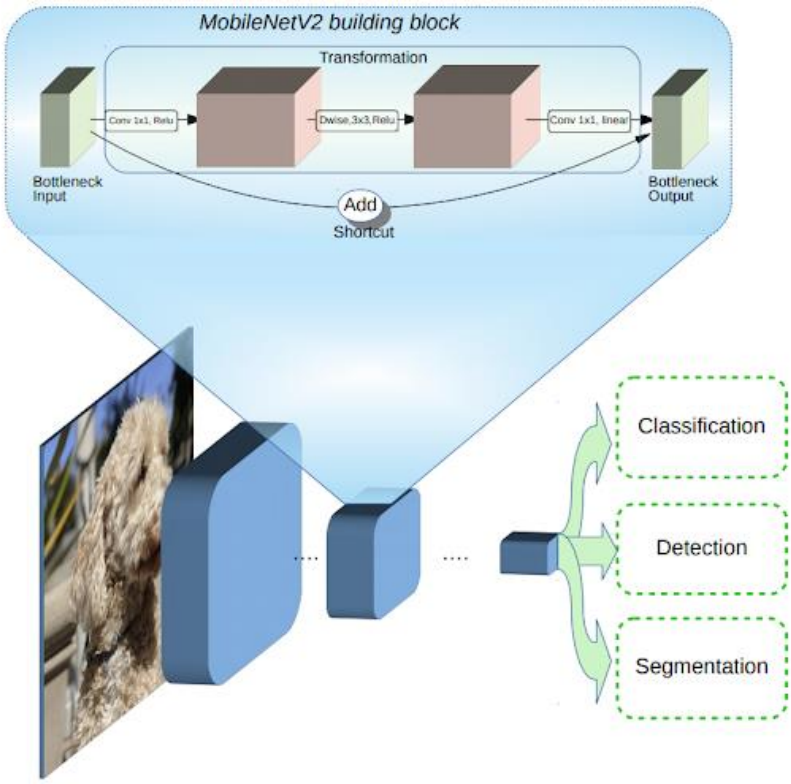
2.5 Arsitektur *Convolutional Neural Network*

Terdapat beberapa arsitektur pada *Convolutional Neural Network* (CNN) untuk pengklasifikasian citra. Pada tahun 2012, arsitektur CNN *AlexNet* berhasil menjadi

pemenang *ImageNet Competition* yaitu kompetisi untuk klasifikasi dan deteksi citra yang terdiri dari jutaan citra dengan puluhan ribu kelas (Wahyudi, 2019). Dalam penelitian ini arsitektur yang digunakan adalah *MobileNetV2*. Berikut penjelasan arsitektur yang digunakan pada penelitian ini.

2.5.1 *MobileNetV2*

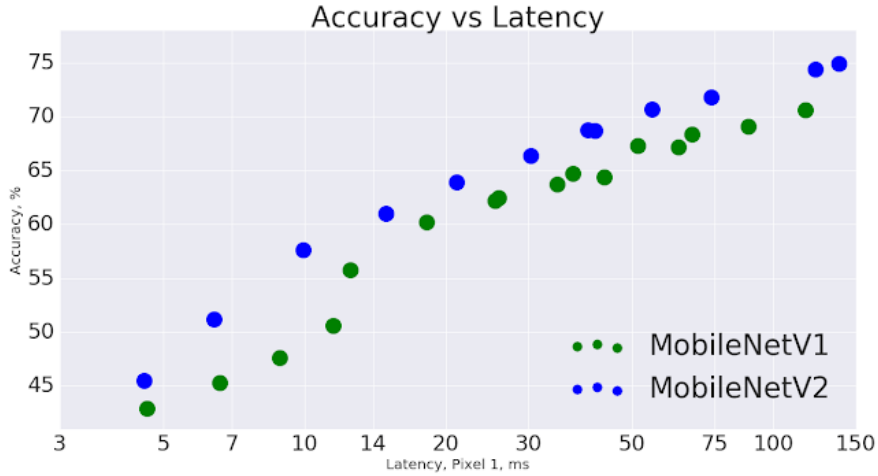
MobileNetV2 adalah peningkatan yang signifikan atas *MobileNetV1* dan mendorong *state of the art* untuk pengenalan visual *mobile* termasuk klasifikasi, deteksi objek dan segmentasi semantik. *MobileNetV2* dibangun di atas ide-ide dari *MobileNetV1*, menggunakan konvolusi yang dapat dipisahkan secara mendalam sebagai blok lapisan yang efisien. Namun, V2 memperkenalkan dua fitur baru pada arsitektur: *Bottleneck* linier antara lapisan, dan *shortcut* koneksi antara *Bottleneck* (Sandler dkk., 2018). Struktur dasar ditunjukkan di bawah ini.



Gambar 2. 21 Gambaran umum arsitektur *MobileNetV2*

Pada gambar diatas *Bottleneck* menyandikan *input* dan *output* antara model sementara lapisan dalam mengenkapsulasi kemampuan model untuk mengubah dari konsep tingkat rendah seperti piksel ke deskriptor tingkat yang lebih tinggi seperti kategori gambar. Terakhir, seperti koneksi residual tradisional, *shortcut* memungkinkan pelatihan yang lebih cepat dan akurasi yang lebih baik (Sandler dkk., 2018).

Secara keseluruhan, model *MobileNetV2* lebih cepat untuk akurasi yang sama di seluruh spektrum latensi. Secara khusus, model baru menggunakan operasi 2× lebih sedikit, membutuhkan parameter 30% lebih sedikit, dan sekitar 30-40% lebih cepat pada ponsel *Google Pixel* daripada model *MobileNetV1*, semuanya sambil mencapai akurasi yang lebih tinggi (Sandler & Andrew Howard, 2018).



Gambar 2. 22 Perbandingan akurasi dan latensi *MobileNetV2* dan *MobileNetV1*

2.6 Evaluasi Kinerja Model

2.6.1 *Confusion Matrix*

Confusion matrix merupakan pengukuran performa yang digunakan pada masalah klasifikasi, *confusion matrix* dapat berupa keluaran visualisasi yang menunjukan hasi prediksi dari model yang benar maupun salah.

Tabel 2. 1 *Confusion Matrix*

		Aktual		
Prediksi		Kelas 1	Kelas 2	Kelas 3
	Kelas 1	T_{11}	F_{12}	F_{13}
	Kelas 2	F_{21}	T_{22}	F_{23}
	Kelas 3	F_{31}	F_{31}	T_{33}

2.6.2 *Recall*

Recall merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif. Berikut rumus *recall* yang digunakan.

$$Recall(j) = \frac{T_i}{T_i + \sum_{i=1}^{C} F_{ij}} \tag{3}$$

2.6.3 *Precision*

Precision merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positf.

$$Precision(i) = \frac{T_i}{T_i + \sum_{j=1}^{C} F_{ij}} \tag{4}$$

2.6.4 *Accuracy*

Accuracy Merupakan rasio prediksi benar (positif dan negatif) dengan keseluruhan data. Biasanya akurasi suatu model ditentukan dalam bentuk persen.

$$Accuracy = \frac{\sum_{i=1}^{C} T_i}{N} \tag{5}$$

2.6.5 *F1-score*

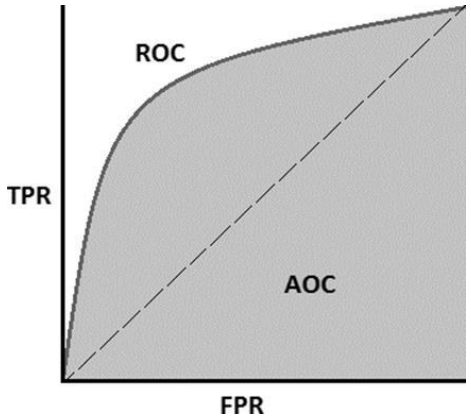
F1-score merupakan rata-rata harmonik dari *precision* dan *recall*. Berikut rumus *F1-score* yang digunakan.

$$F1 - score = \frac{2 \times Recall \times Precision}{Recall + Precision} \quad (6)$$

2.6.6 Kurva AUC-ROC

Kurva AUC-ROC merupakan pengukuran klasifikasi dengan menggunakan ambang batas. Kurva AUC-ROC divisualisasikan dalam bentuk kurva yang memiliki garis *baseline*. Jika kurva mendekati garis *baseline* maka dapat dikatakan bahwa performa model tidak baik, tetapi jika kurva berada jauh di atas garis baseline maka performa model dapat dikatakan baik. Kurva AUC-ROC digunakan untuk melihat seberapa baik model membedakan tiap kelas.

Kurva ROC (*Receiver Operating Characteristic*) menunjukkan kinerja model klasifikasi. Kurva ini memplot dua parameter *True Positive Rate*, dan *False Positive Rate*. AUC adalah singkatan dari "*Area under the* ROC *Curve*" Artinya, AUC mengukur seluruh area dua dimensi di bawah seluruh kurva ROC. Semakin tinggi nilai AUC semakin baik pula kinerja model dalam klasifikasi.



Gambar 2. 23 Kurva AUC-ROC

2.7 Rancang Bangun Aplikasi *Mobile* Android

2.7.1 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 Handset Alliance*, 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.

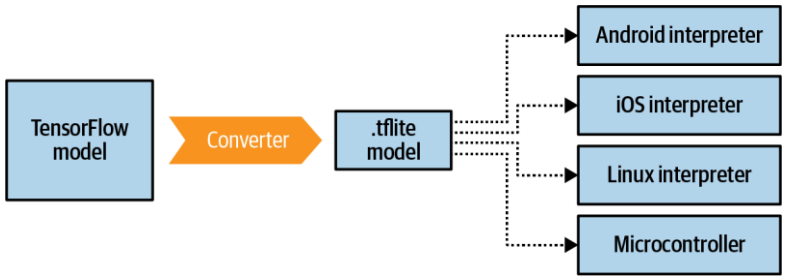
2.7.2 *TensorFlow Lite*

TensorFlow Lite adalah rangkaian alat yang melengkapi *TensorFlow*, mencapai dua tujuan utama. Yang pertama adalah membuat model *mobile-friendly*. Ini sering kali melibatkan pengurangan ukuran dan kerumitannya, dengan dampak sesedikit mungkin pada akurasinya, untuk membuatnya bekerja lebih baik di lingkungan dengan baterai terbatas seperti perangkat seluler. Yang kedua adalah menyediakan *runtime* untuk *platform* seluler yang berbeda, termasuk *Android*, *iOS*, *Linux* seluler (misalnya, *Raspberry Pi*), dan berbagai mikrokontroler. Model tidak dapat dilatih dengan *TensorFlow Lite.* Alur kerjanya adalah melatih model terlebih dahulu menggunakan *TensorFlow* lalu mengonversinya ke format *TensorFlow Lite*, sebelum memuat dan menjalankannya menggunakan penerjemah *TensorFlow Lite* (Moroney, 2021).

TensorFlow Lite dimulai sebagai versi seluler *TensorFlow* yang ditujukan untuk *developer Android* dan *iOS*, dengan tujuan menjadi *toolkit* ML yang efektif untuk kebutuhan mereka. Saat membangun dan menjalankan model di komputer atau layanan *cloud*, masalah seperti konsumsi baterai, ukuran layar, dan aspek lain dari

pengembangan aplikasi seluler tidak menjadi perhatian, namun ketika perangkat seluler ditargetkan, beberapa kendala baru perlu ditangani.

1. Yang pertama adalah *framework mobile* aplikasi harus ringan.
2. *Framework* juga harus memiliki latensi yang rendah.
3. Dengan latensi yang rendah, *framework mobile* membutuhkan format model yang efisien.
4. Peningkatan privasi pengguna serta konsumsi daya.



Gambar 2. 24 Ilustrasi *TensorFlow Lite*

Secara umum *TensorFlow Lite* memiliki dua proses: konverter akan mengambil model *TensorFlow* dan mengonversinya ke format .tflite (mengecilkan dan mengoptimalkannya) dan serangkaian *interpreter* untuk berbagai *runtime* (Gambar 2.23).