KLASIFIKASI CITRA MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DENGAN ARSITEKTUR *INCEPTION V4* BERBASIS *ANDROID* PADA DATASET *FLOWER RECOGNITION*

AINUN MARDIYAH ISTIQAMAH

H13116507

PROGRAM STUDI ILMU KOMPUTER

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

OKTOBER 2020

KLASIFIKASI CITRA MENGGUNAKAN METODE *CONVOLUTIONAL NEURAL NETWORK* ARSITEKTUR *INCEPTION V4* BERBASIS ANDROID PADA DATASET *FLOWER RECOGNITION*

SKRIPSI

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

AINUN MARDIYAH ISTIQAMAH

H13116011

PROGRAM STUDI ILMU KOMPUTER DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

OKTOBER 2020

LEMBAR PERNYATAAN KEOTENTIKAN

Saya yang bertanda tangan di bawah ini menyatakan dengan sungguh-sungguh bahwa skripsi yang saya buat dengan judul:

KLASIFIKASI CITRA MENGGUNAKAN METODE *CONVOLUTIONAL NEURAL NETWORK* ARSITEKTUR *INCEPTION V4* BERBASIS ANDROID PADA DATASET *FLOWER RECOGNITION*

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

Makassar, 02 Oktober 2020

Ainun Mardiyah Istiqamah

NIM. H13116507

KLASIFIKASI CITRA MENGGUNAKAN METODE *CONVOLUTIONAL NEURAL NETWORK* ARSITEKTUR *INCEPTION V4* BERBASIS ANDROID PADA DATASET *FLOWER RECOGNITION*

Disetujui oleh:

Pembimbing Utama	Pembimbing Pertama
Dr. Hendra, S.Si, M.Kom.	**Supri Bin Hj Amir, S.Si., M.Eng.**
NIP. 19760102 200212 1 001	**NIP. 19880504 201903 1 012**

Pada 02 Oktober 2020

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh:

Nama : Ainun Mardiyah Istiqamah

NIM : H13116507

Program Studi : Ilmu Komputer

Judul Skripsi : Klasifikasi Citra Menggunakan Metode *Convolutional Neural Network* Arsitektur *Inception V4* Berbasis Android Pada Dataset *Flower Recognition*

Telah berhasil mempertahankan di hadapan dewan penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Sains pada Program Studi Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin.

DEWAN PENGUJI

		Tanda Tangan
1. Ketua	: Dr. Hendra, S.Si., M.Kom.	(.....)
2. Sekretaris	: Supri Bin Hj. Amir, S.Si., M.Eng.	(.....)
3. Anggota	: Dr.Eng. Armin Lawi, S.Si., M.Eng.	(.....)
4. Anggota	: Andi Muh. Amil Siddik, S.Si, M.Si.	(.....)

Ditetapkan di : Makassar

Tanggal : 02 Oktober 2020

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 "Klasifikasi Citra Menggunakan Metode *Convolutional Neural Network* Arsitektur *Inception V4* Berbasis Android Pada Dataset *Flower Recognition*" yang disusun sebagai salah satu syarat akademik untuk meraih gelar Sarjana Sains pada Program Studi Ilmu Komputer Fakutas 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 materiil, serta dorongan dari berbagai pihak. Oleh karena itu, penulis ingin menyampaikan ucapan terima kasih yang tak terhingga kepada orang tua penulis, Ibunda **Asmi Anwar** dan Ayahanda **Darwis Durahim**, sebagai tempat kembali setelah pergi, dan tempat terlelap dikala lelah, terima kasih atas kasih sayang, doa, dan nasihat yang tulus sebagai bekal kehidupan. Rasa terima kasih juga penulis tujukan kepada Adinda **Asifah Baiq Ramadani,** dan **Abid Ahmad Rajendra** sebagai adik, 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. 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.
2. Bapak **Dr. Hendra S.Si, M.Kom.,** sebagai dosen pembimbing utama sekaligus sebagai tim penguji atas ilmu yang beliau berikan selama proses perkuliahan, dan

kesediaan beliau dalam membimbing, serta memotivasi penulis dalam penyusunan skripsi ini.

3. Bapak **Supri Bin Hj Amir, S.Si., M.Eng.,** sebagai dosen pembimbing pertama sekaligus sekretaris tim penguji atas ilmu yang beliau berikan selama proses perkuliahan, dan bimbingan serta segala bentuk bantuan yang telah beliau berikan dalam penyusunan skripsi ini.
4. Bapak Alm **Dr. Diaraya, M.Ak.,** sebagai anggota tim penguji atas segala ilmu yang telah beliau berikan selama proses perkuliahan bentuk kritik dan masukan yang membangun selama proses penyusunan skripsi ini.
5. Bapak **Dr. Eng. Armin Lawi, S.Si., M.Eng.,** sebagai anggota tim penguji atas segala ilmu yang telah beliau berikan selama proses perkuliahan bentuk kritik dan masukan yang membangun selama proses penyusunan skripsi ini.
6. Bapak **Andi Muh. Amil Siddik, S.Si, M.Si.,** sebagai anggota tim penguji atas segala kritikan dan masukan yang membangun dalam penyusunan skripsi ini.
7. Saudara **Zinedine Kahlil Gibran Zidane,** yang telah meluangkan waktu dalam membagi ilmu. Terimakasih atas segala ilmu yang telah diberikan.
8. Saudara-saudara ku **Berlian Adriani Putri, St. Hestiana Kadir, Nisrina Syadza Dewanty, Rizka Syahfitri, Tasnia Akil, Suci Rahmadana Anwar, Nurmayulina, Marselia Ghanyyu Wahdini, Nirwana Sari Hamka dan ILMU KOMPUTER 2016**, atas kebersamaan, kepedulian, suka-duka, canda tawa yang telah kita lewati selama ini. Semoga persahabatan kita yang telah terjalin tidak pernah usai.
9. Keluarga besar **A16ORITMA 2016** atas segala bentuk dukungan dan bantuan selama proses perkuliahan. Semoga kesuksesan selalu kita dapatkan dalam setiap langkah-langkah kita.
10. Keluarga besar **KKN PPM Parepare** yang secara ikhlas dan tulus mengabdi kepada masyarakat.
11. Seluruh pihak yang yang tidak dapat disebutkan satu per satu atas segala bentuk kontribusi, partisipasi, serta motivasi yang diberikan kepada penulis selama ini.

ɔga apa yang kita berikan, dilipatgandakan oleh Tuhan Yang Maha Kaya.

Penulis menyadari bahwa masih banyak kekurangan dalam tugas akhir ini, untuk itu dengan segala kerendahan hati penulis memohon maaf. Akhir kata, semoga tulisan ini memberika manfaat untuk pembaca.

Makassar, 02 September 2020

Ainun Mardiyah Istiqamah

PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

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

Nama	: Ainun Mardiyah Istiqamah
NIM	: H13116507
Programa Studi	: Ilmu Komputer
Departemen	: Matematika
Fakultas	: Matematika dan Ilmu Pengetahuan Alam
Jenis Karya	: Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Hasanuddin **Hak Prediktor Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*)** atas tugas akhir saya yang berjudul:

"Klasifikasi Citra Menggunakan Convolutional Neural Network Dengan Arsitektur Inception V4 Berbasis Android Pada Dataset Flower Recognition"

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

Demikian surat pernyataan ini saya buat dengan sebenarnya.

Dibuat di Makassar pada 02 Oktober 2020

Yang menyatakan

(Ainun Mardiyah Istiqamah)

ABSTRAK

Klasifikasi bunga adalah pekerjaan penelitian mendasar di bidang Botani. Sampai sekarang, telah ditemukan ratusan ribu spesies bunga. Orang-orang menggunakan kamera, *handphone*, atau perangkat lainnya untuk mengabadikan bunga, tetapi masih banyak yang belum mengetahui jenis bunga dikarenakan kesamaan antara spesies bunga yang berbeda dan perbedaan antara spesies bunga yang sama. Oleh karena itu, membuat pengklasifikasi bunga akan membawa banyak manfaat bagi orang-orang yang tidak mengetahui jenis bunga. Di era modern ini, orang-orang sering membawa handphone ketika berpergian, oleh karena itu pada kasus ini penulis ingin membuat aplikasi android pengklasikfikasi bunga agar orang-orang lebih mudah mencari informasi tentang spesies bunga yang tidak diketahui. Pada penelitian ini penulis menggunakan arsitektur *CNN inception v4* yang digunakan untuk membuat model klasifikasi bunga dan menggunakan *tensorflow lite* agar dapat digunakan di android. Model yang dihasilkan dari pelatihan sebanyak 20 epoch berperforma 98,14% dalam akurasi pada data *training* dan hasil evaluasi pada data *test* menghasilkan akurasi sebesar 88,56%.

Kata Kunci: *Convolutional Neural Network* (CNN), *Machine Learning*, *Inception*, Klasifikasi

ABSTRACT

Flower classification is a fundamental research work in Botany. Thousands of flower species have been discovered. People use their digital camera, handphone or other devices to capture flower images, but there are still many people who do not know the type of flower due to the similarities between different flower species and the differences between the same flower species. Therefore, creating a flower classifier will bring many benefits for people who do not know the types of flowers. In this modern era, people often carry handphone when traveling, therefore in this case the author tries to create an android application of flower classification so that people can easily find information about unknown flower species. In this study, the author uses the Inception v4 CNN architecture for the flower classification model and uses Tensorflow Lite framework so that it can be ported to Android. The resulting model of 20 epoch training performs 98.14% of accuracy in training data and evaluation in test data results in 88.56% of accuracy.

Keywords: *Convolutional Neural Network* (CNN), *Machine Learning*, *Inception*, *Classification*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Klasifikasi bunga adalah pekerjaan penelitian mendasar di bidang Botani. Sampai sekarang, telah ditemukan ratusan ribu spesies bunga. Orang-orang menggunakan kamera, *handphone*, atau perangkat lainnya untuk mengabadikan bunga, tetapi orang juga akan bingung karena tidak mengetahui jenis bunga tersebut. Oleh karena itu, membuat pengklasifikasi bunga akan membawa banyak manfaat bagi orang-orang yang tidak mengetahui jenis bunga. Ada banyak tantangan dalam mengklasifikasikan bunga, seperti *background* gambar bunga yang kompleks dan kesamaan antara jenis bunga. Maka fitur tunggal seperti warna, bentuk atau tekstur untuk membedakan spesies bunga tidak dapat digunakan. Selain itu, spesies bunga yang sama akan berbeda karena bentuk, skala, sudut pandang dan sebagainya.

Metode klasifikasi bunga yang paling primitif adalah untuk mengamati kebiasaan hidup, struktur morfologis, dan fitur bunga lainnya, kemudian dibandingkan dengan bunga yang sudah terdaftar sebelumnya, lalu jenis bunga tersebut ditentukan berdasarkan ciri-cirinya. Metode ini beban kerjanya besar, dan membutuhkan staff profesional yang memiliki pengetahuan dan pengalaman profesional. Metode klasifikasi ini juga tidak otomatis, proses pemilihan fitur memerlukan intervensi manusia, keakuratan pemilihan fitur secara langsung mempengaruhi klasifikasi keseluruhan, dan keakuratannya tidak terlalu tinggi.

Dengan perkembangan teknologi komputer dan citra digital, orang-orang mulai mengeksplorasi metode baru untuk mengklasifikasikan bunga secara otomatis, metode ini didasarkan pada warna, bentuk, tekstur dan fitur bunga untuk menghitung kesamaan antara gambar bunga lainnya kemudian menentukan jenisnya. Selama periode ini, orang lebih memperhatikan gambar segmentasi dan pemilihan fitur buatan.

nvolutional neural network adalah metode yang paling efisien untuk
an yang telah dikembangkan dalam beberapa tahun terkahir. Jaringan ini

menghindari *preprocessing* gambar yang kompleks, dan orang-orang dapat memasukkan gambar asli secara langsung. Menggunakan bidang reseptif lokal, berbagi bobot dan teknologi pengumpulan dan membuat parameter pelatihan sangat berkurang dibandingkan dengan jaringan saraf.

Convolutional neural networks merupakan metode yang populer pada saat ini untuk mengklasifikasikan gambar, dimana telah dilakukan oleh beberapa peneliti seperti, Xialing Xia dan Cui Xu dari *Donghua university* mengklasifikasikan dataset bunga oxford-17 (akurasi pengakuan 95%) dan dataset bunga oxford-102 (akurasi pengakuan 94%). Takeshi Saitoh *et al* dari *Toyohashi university of technology* mengusulkan metode otomatis untuk mengenali bunga mekar, dilakukan untuk 600 gambar dan memperoleh pengakuan bunga 90% pada tahun 2004. Pada 2014, Xiadong Xie menggunakan klasifikasi bunga berbutiran halus pada dataset bunga oxford-17 (akurasi pengakuan 93,14%) dan dataset bunga oxford-1O2 (akurasi pengakuan adalah 79,1%) (Xia & Xu, 2017).

Maka dari itu, penulis tertarik untuk melakukan penelitian dengan menggunakan arsitektur *inception v4* sebab belum ada penelitian terkait arsitektur tersebut pada dataset *flower recognitio*n. Sehingga peneliti memutuskan untuk membuat penelitian yang berjudul **"Klasifikasi Citra Menggunakan Convolutional Neural Network Dengan Arsitektur Inception V4 Berbasis Android Pada Dataset Flower Recognition"**.

1.2 Rumusan Masalah

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

1. Bagaimana membuat model CNN yang mampu mengenali jenis bunga pada *dataset flower recognition* berbasis *android*?
2. Bagaimana tingkat akurasi yang didapatkan pada arsitektur *inception-v4* pada *dataset flower recognition*?

1.3 Batasan Masalah

Batasan masalah pada penilitian ini adalah:

1. *Dataset* yang digunakan adalah *dataset flower recognition*
2. *Dataset* diambil dari Alexander Mamaev di halaman *keagle*.
3. Jenis bunga yang akan diteliti ada 5, yaitu daisy, dandelion, mawar, tulip, dan bunga matahari.

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah, maka tujuan dari penelitian ini adalah:

1. Membuat model CNN yang mampu mengenali jenis bunga pada dataset *flower recognition* berbasis *android.*
2. Mengetahui performa *inception v4* dalam dataset *flower recognition*

1.5 Manfaat penelitian

Hasil penelitian ini diharapkan dapat bermanfaat:

1. Menjadi sumber informasi mengenai performa arsitektur *inception-v4*.
2. Menjadi sumber informasi mengenai penggunaan *convolutional neural network* dengan arsitektur *inception-v4*.
3. Menyediakan aplikasi berbasis *android* yang mampu mengenali objek menggunakan gambar.

BAB II
TINJAUAN PUSTAKA

2.1 Jenis Bunga

Bunga adalah bagian dari tanaman yang berfungsi untuk menghasilkan biji. Penyerbukan dan pembuahan suatu tanaman terjadi didalam bunga, setelah pembuahan terjadi bunga akan berkembang lebih lanjut dengan membentuk buah. Pada tumbuhan yang memiliki bunga, buah merupakan struktur pembawa dan pelindung biji (Pertanian, 2019). Telah ditemukan ratusan ribu spesies bunga didunia ini, dan orang-orang masih banyak yang belum mengetahui jenis bunga. Pada penelitian ini, klasifikasi terhadap jenis bunga akan dilakukan terhadap 5 jenis bunga sebagai berikut.

2.1.1 Daisy

Daisy tergolong dalam satu jenis keluarga tanaman terbesar di dunia. Bunga ini tergolong spesies tumbuhan vascular, yang kaya akan nutrisi dan air. Tumbuhan vaskular ini tergolong dalam 100% jenis tumbuhan berbunga yang tumbuh di bumi. Daisy biasa dijumpai di seluruh daerah yang terdapat di dunia, kecuali Antartika. Bunga daisy pertama kali diperkenalkan oleh seorang ahli botani asal jerman, Paul Dietrich Giseke dan teman dekatnya yang dikenal sebagai “Bapak Taksomi Modern” asal Swedia, yaitu Carl Linnaeus. Bunga daisy juga memiliki berbagai macam warna, putih, kuning, ungu, merah, dan masih banyak lagi (Florist, 2017). Pada gambar 2.1 terdapat 3 warna berbeda dari bunga daisy.

Gambar 2.1. Daisy

2.1.2 Mawar

[awar adalah suatu jenis tanaman semak dari genus rosa sekaligus nama ıng dihasilkan tanaman ini. Mawar liar terdiri dari 100 spesies lebih,

kebanyakan tumbuh dibelahan bumi utara yang berudara sejuk. Spesies mawar umumnya merupakan tanaman semak yang berduri atau tanaman yang memanjat yang tingginya bisa mencapai 2 sampai 5 meter. Walaupun jarang ditemui, tinggi tanaman mawar yang merambat di tanaman lain bisa mencapai 20 meter. Sebagian besar spesies mempunyai daun yang panjangnya antara 5-15cm dengan dua-dua berlawanan pinnate (Christian, 2015). Adapun jenis-jenis tanaman mawar antara lain, old garden roses, climbing roses, wild roses, shrub roses, dan masih banyak lagi. Pada gambar 2.2 terdapat 3 warna berbeda dari bunga mawar.

Gambar 2.2. Mawar

2.1.3 Dandelion

Dandelion *(taraxum officinale weber)* adalah anggota keluarga *asteraceae (compositae)* yang berasal dari eropa dan didistribusikan secara luas di zona beriklim hangat di belahan bumi utara. Dandelion dianggap sebagai ramuan tidak beracun yang dapat berpotensi dimanfaatkan untuk anti rematik dan diuretic. Baru-baru ini, dandelion telah mengumpulkan perhatian untuk aktivitas antioksidan dan kemungkinan efek menguntungkan terhadap perkembangan obesitas, kanker, dan banyak factor risiko kardiovaskular (González-Castejón, dkk., 2012). Dandelion juga memiliki berbagai warna salah satunya putih dan kuning, bisa dilihat pada gambar 2.3.

Gambar 2.3. Dandelion

2.1.4 Bunga Matahari

alam Bahasa latin bunga matahari disebut dengan *Helianthus Annuus L.* i adalah salah satu jenis bunga yang tumbuh mekar dalam satu tahun sekali

dan berasal dari negara di benua amerika latin, seperti meksiko, negara peru, amerika serikat, dan akhirnya bunga ini menyebar luas keseluruh dunia termasuk indonesia. Sehingga membuat bunga ini mempunya nama atau sebutan yang berbeda-beda di setiap negara (Ardinhtc, 2019). Bunga matahari memiliki daun yang lebar dan tumbuh mengembang berwaran hijau tua, dengan batang kehijauan dan tidak terlalu keras. Tangkai bunga berbentuk bulat dan oval dengan daun-daun kecil ditepinya. Sedangkan kelopak bunga berwarna kuning terang tumbuhan mengitari bulatan bunga dengan calon biji ditengah-tengahnya, bisa kita lihat pada gambar 2.4.

Gambar 2.4. Bunga Matahari

2.1.5 Tulip

Bunga tulip merupakan salah satu bunga yang paling banyak mendapatkan perhatian dunia. Padahal jika diperhatikan, bentuk bunga tulip sangatlah sederhana, hanya berbentuk kuncup bunga yang belum mekar sempurna dan tumbuh tegak ke atas. Namun yang menarik adalah, bunga ini hadir dalam berbagai warna-warni yang kontras dengan batang dan daunnya. Mungkin sudah menjadi pengetahuan umum jika bunga tulip atau *tulipa sp* adalah salah satu ikon negara belanda yang sangat mendunia (Bibitbunga, 2015). Pada gambar 2.5 terdapat bunga tulip dengan 3 warna berbeda.

Gambar 2.5. Tulip

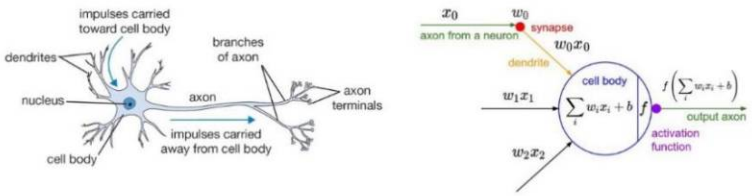
2.2 *Deep Learning*

Deep Learning merupakan salah satu bidang dari *machine learning* yang terinspirasi dari kortex manusia dengan menerapkan jaringan syaraf buatan yang memiliki banyak hidden layer. *Convolutional neural network* merupakan salah satu metode dalam *deep learning* yang dibuat untuk menutupi kelemahan dari metode sebelumnya. Terdapat beberapa kelemahan dari metode sebelumnya, tetapi dengan model ini sejumlah parameter bebas dapat dikurangi dan deformasi gambar input seperti translasi, rotasi, dan skala dapat ditangani (LeCun, dkk., 1998).

Seiring dengan banyaknya pengembangan dan riset tentang *deep learning*, banyak *library* yang bermunculan dengan fokus mempelajari tentang jaringan syaraf tiruan salah satu contohnya yaitu keras. Keras merupakan *library* jaringan syaraf tiruan dengan bahasa *python* dan mampu berjalan di atas *tensorflow*, CNTK, atau *Theano. Library* ini menyediakan fitur yang digunakan dengan fokus mempermudah pengembangan lebih dalam tentang *deep learning* (Data, 2018).

2.3 *Neural Network*

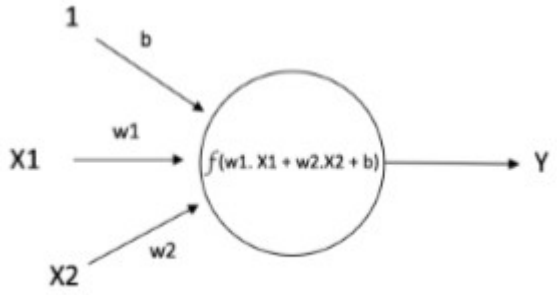
Neural network atau jaringan saraf merupakan sebuah model dari jaringan saraf otak manusia yang ditiru oleh banyak peniliti untuk diadopsi cara kerjanya di berbagai bidang kajian seperti biologi, fisika, ilmu komputer, dll (Setiawan, 2018). Di bidang ilmu komputer terdapat istilah *artifical neural networks* (ANN). Konsep ANN diadopsi dari jaringan saraf otak pada manusia sehingga ANN biasa disebut jaringan saraf tiruan. Model yang dibuat oleh ANN dapat untuk beradaptasi, belajar dan mengelompokkan berbasis pemrosesan parallel. Gambar 2.6 menunjukkan perbedaan neural network pada otak dan tiruan.



Gambar 2.6. Ilustrasi *Neural Network*

Adapun beberapa hal yang mendasar pada komputasi *neural network* adalah tau biasa disebut juga dengan *node*. *Node* dapat menerima input dari *node* tau dari sumber eksternal kemudian dihitung untuk mendapatkan sebuah

output. Setiap *input* memiliki bobot (w) tersendiri, yang dimana bobot ini diberikan dengan dasar hubungannya dengan *input* lainnya. *Node* tersebut menggunakan fungsi ke dalam input yang telah diberi bobot. Contoh ilustrasi *neuron* dapat dilihat pada gambar 2.7.



Gambar 2.7. Ilustrasi *Neural Network*

Sebuah *neuron input* berupa x1 dan x2 dengan bobotnya yaitu w1 dan w2. Selain dari input dan bobot juga terdapat inputan lain berupa 1 dengan bobot b (bias). Fungsi utama dari bias adalah untuk menyediakan setiap *node* dengan sebuah nilai *constant* yang dapat dilatih (sebagai tambahan selain dari *input* normal yang diterima). *Output* pada *neuron* tersebut yaitu *Y* yang merupakan hasil perhitungan dari sebuah fungsi *non-linear f* yang disebut dengan fungsi aktivasi. Tujuan dari sebuah fungsi aktivasi adalah untuk menunjukkan *non-linear* terhadap *output* dari *neuron* tersebut (Setiawan, 2018).

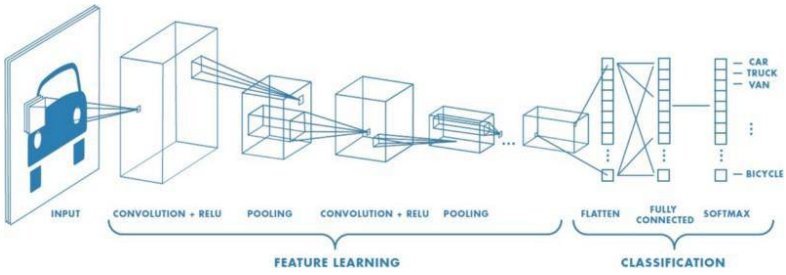
2.3.1 *Batch Size & Epoch*

Batch size adalah jumlah sampel data yang disebarkan ke *neural network*, sedangkan *epoch* adalah ketika seluruh dataset sudah melalui proses *training* pada *neural network* sampai dikembalikan ke awal untuk sekali putaran, karena satu *epoch* terlalu besar untuk dimasukkan (*feeding*) kedalam komputer maka dari itu perlu membaginya kedalam satuan kecil (*batches*) (Imam, 2018).

2.4 *Convolutional Neural Network*

Convolutional neural network (CNN) merupakan sebuah variasi *dari multi layer perceptron* (MLP) yang terinspirasi dari penelitian Hubel pada tahun 1968 [illegible] unakan dalam pengolahan data citra. Konvolusi adalah matriks yang [illegible] fungsi untuk melakukan *filter* pada citra (Ludwig, 2013). *Convolutional* [illegible]*twork* memiliki beberapa *layer* yang difungsikan untuk melakukan *filter*

pada setiap prosesnya. Gambar 2.8 disebut dengan proses *training*. Pada proses *training* terdapat 3 tahapan yaitu *convolutional layer*, *pooling layer*, dan *fully connected layer*.



Gambar 2.8. Proses *Convolutional Neural Network*

Berdasarkan citra diatas, tahap pertama pada arsiterktur CNN adalah tahap konvolusi. Tahap ini dilakukan dengan menggunakan sebuah kernel dengan ukuran tertentu. Perhitungan jumlah kernel yang dipakai tergantung dari jumlah fitur yang dihasilkan, kemudian dilanjutkan menuju fungsi, selanjutnya setelah keluar dari proses fungsi aktivasi kemudian melalui proses *pooling*. Proses ini diulang beberapa kali sampai didapatkan peta fitur yang cukup untuk dilanjutkan ke *fully connected neural network*, dan dari *fully connected network* adalah *output class*.

2.4.1 *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* lalu *vertical*. 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 diperhatikan bahwa dengan menggunakan *stride* yang kecil tidak selalu akan mendapatkan performa yang bagus (Paradistia, 2019).

Padding atau *zero padding* adalah parameter yang menentukan jumlah *pixels* (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 sixe dari input lebih besar
ı dengan size dari filter. Untuk suatu input citra berukuran $n\ x\ n$, padding
erubah size input menjadi $(n + 2p)\ x\ (n + 2p)$ (Paradistia, 2019).

2.4.2 Konvolusi

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 hal pengolahan citra, konvolusi berarti mengaplikasikan sebuah kernel pada citra. Rumus konvolusi dapat dilihat pada (1).

Dalam *machine learning*, *input* citra merupakan data berbentuk *array* multidimensi dan kernel merupakan parameter benrbentuk *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:

$$S(i,j) = (I * K)(i,j) = \sum_a \sum_b I(a,b)K(i-a, j-b) \quad (1)$$

konvolusi bersifat komunitatif, yang berarti persamaan (1) dapat ditulis:

$$S(i,j) = (K * I)(i,j) = \sum_a \sum_b I(i-a, j-b)K(a,b) \quad (2)$$

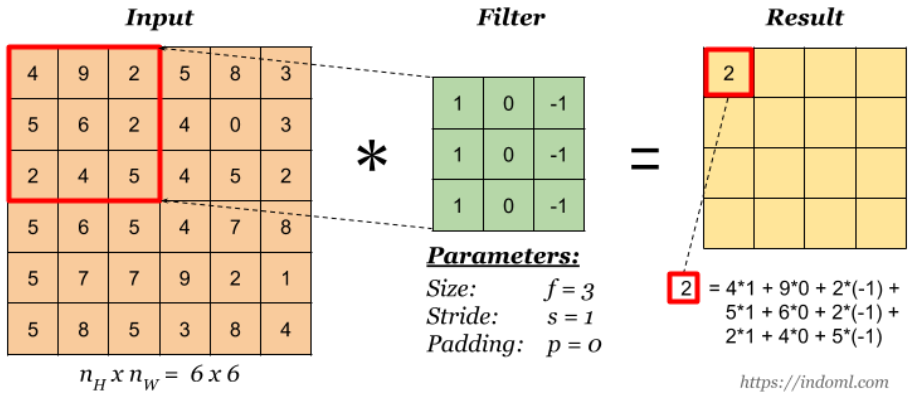
sifat komutatif konvolusi muncul karena kernel telah dibalik relatif terhadap input. Sifat komutatif ini berguna untuk menulis bukti (*proof*) tapi sifat ini bukan merupakan sifat yang essensial pada implementasi jaringan saraf tiruan. Terdapat fungsi serupa yang disebut *cross-correlation* yang sama seperti konvolusi namun tanpa membalik kernel:

$$S(i,j) = (I * K)(i,j) = \sum_a \sum_b I(i+a, j+b)K(a,b) \quad (3)$$

Banyak *library machine learning* yang menerapkan *cross-correlation* namun menyebutnya konvolusi. Pada penelitian ini fungsi itu juga disebut konvolusi (Goodfellow, dkk., 2016)

Tujuan dilakukannya konvolusi pada citra adalah untuk mengekstraksi fitur dari citra *input*. Konvolusi akan menghasilkan transformasi linear dari data *input.* Pada gambar 2.9 diilustrasikan bagaimana operasi konvolusi pada matriks hasil representasi citra. Matriks *input* merupakan representasi citra dalam bentuk

- Operasi konvolusi akan dilakukan antara matriks input ukuran 6x6 dengan *lter* dengan ukuran 3x3. Stride atau perpindahan sebanyak 1 dengan *zero* Hingga hasilnya seperti pada matriks *result*.



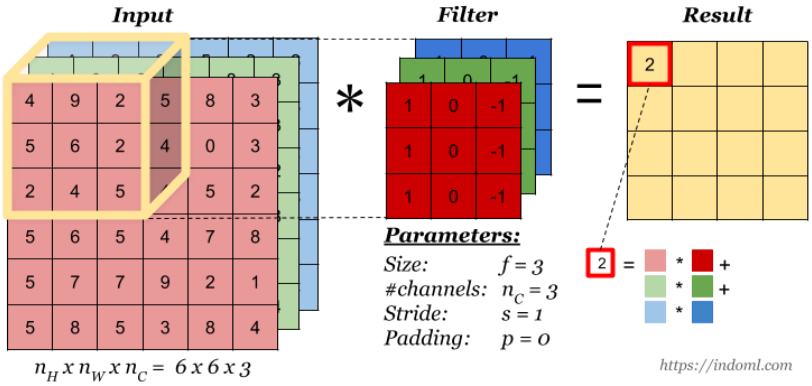
Gambar 2.9. Ilustrasi Operasi Konvolusi

Operasi konvolusi pada gambar 2.9 hanya berlaku pada citra *grayscale* yang hasil representasi gambarnya hanya 1 lapisan. Untuk citra RGB, representasi gambarnya akan menghasilkan 1 lapis matriks dari operasi konvolusi. Masing-masing lapisan matriks akan dioperasikan dengan *filter*. Jika *input* citra berukuran $nxnxn_c$ dengan n_c adalah jumlah *channel* atau *layer* pada *input* citra, maka persamaan (3) menjadi:

$$S(i,j) = (I * K)(i,j) = \sum_{c}^{nc} \sum_{a} \sum_{b} I(i+a, j+b, c) K(a,b,c) \quad (4)$$

dengan $c = 1 \ldots n_c$

Hasil operasi konvolusi dari masing-masing lapisan akan saling dijumlahkan untuk menghasilkan matriks hasil konvolusinya seperti pada gambar 2.10.



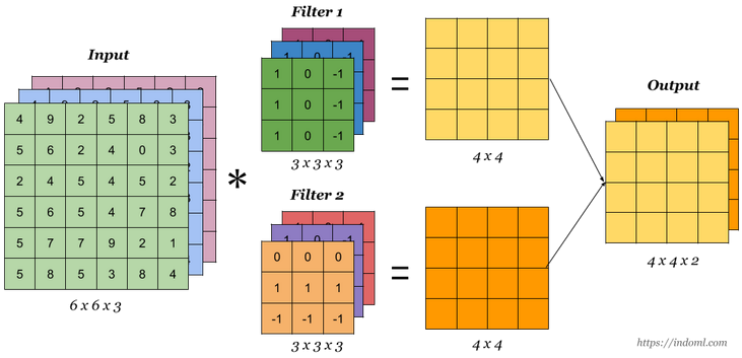
Gambar 2.10. Ilustrasi operasi konvolusi pada citra RGB

Adapun operasi konvolusi pada CNN, biasanya tidak hanya menggunakan *filter*. Pada gambar 2.11 diilustrasikan bagaimana operasi konvolusi pada

citra RGB dengan 2 *filter*. Jika n_f adalah jumlah *filter* yang digunakan dan $k = 1 \dots n_f$ maka persamaan (4) menjadi:

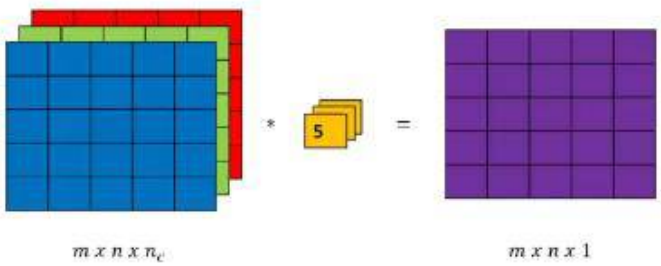
$$S(i,j,k) = (I * K_k)(i,j) = \sum_{c}^{nc} \sum_{a} \sum_{b} I(i+a, j+b, c) K_k(a,b,c) \tag{5}$$

Operasi untuk masing-masing *filter* sama dengan ilustrasi pada gambar 2.10. Jumlah lapisan *output* dari hasil operasi konvolusi dengan banyak lapisan sama dengan banyaknya *filter* yang digunakan (Prijono, 2018).



Gambar 2.11. Ilustrasi operasi konvolusi pada citra RGB dengan layer filter lebih dari satu

Pada gambar 2.12 diilustrasikan, konvolusi 1x1 yang mereduksi *input* dengan n_c *channel* menjadi satu *channel*, hal ini dikarenakan setiap *channel* dikalikan dengan suatu bilangan skalar pada kernel kemudian dijumlahkan.



Gambar 2.12. Konvolusi 1x1

Konvolusi 1x1 yang digunakan sebelum operasi yang membutuhkan waktu lama untuk mempercepat perhitungan. Pada *inception network*, konvolusi 1x1 digunakan untuk mereduksi komputasi dari konvolusi 3x3 dan 5x5 yang mahal (Szegedy, dkk.,

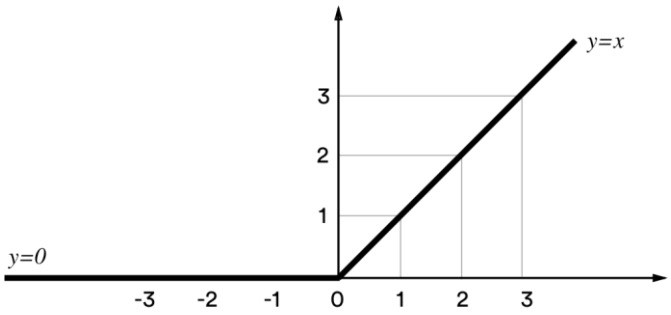
2.4.3 Fungsi Aktivasi

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, et al., 2018). Bentuk fungsi ReLU:

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} \quad (6)$$

Pada fungsi aktivasi ReLU, semua nilai x *negative* akan dipetakan ke 0, seperti pada 2.13.



Gambar 2.13. Grafik Fungsi ReLU

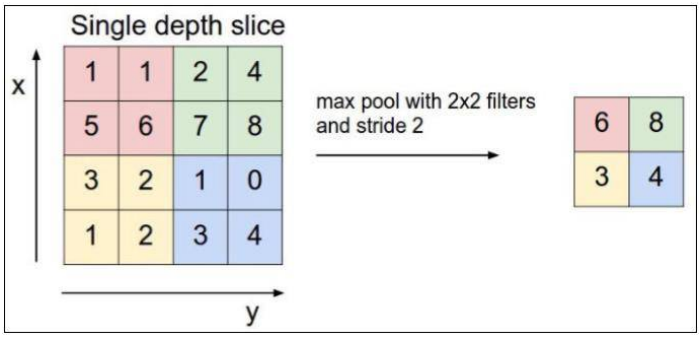
Kelebihan fungsi ReLU yaitu:

1. Fungsi ReLU *konvergen* terhadap *stochastic gradient descent* dibandingkan dengan fungsi *sigmoid/tanh*.
2. Operasi *neuron* pada fungsi ReLU lebih ringan dibanding fungsi *sigmoid/tanh* yang melibatkan operasi eksponensial. ReLU hanya melakukan *treshholding* sebuah matriks aktivasi pada nilai 0. (Chen, dkk., 2018).

2.4.4 *Pooling Layer*

ɔoling layer adalah lapisan yang menggunakan fungsi dengan *feature map* nasukan dengan mengolahnya dengan berbagai macam operasi statistik :an nilai piksel terdekat. Pada model CNN, lapisan *pooling* biasanya

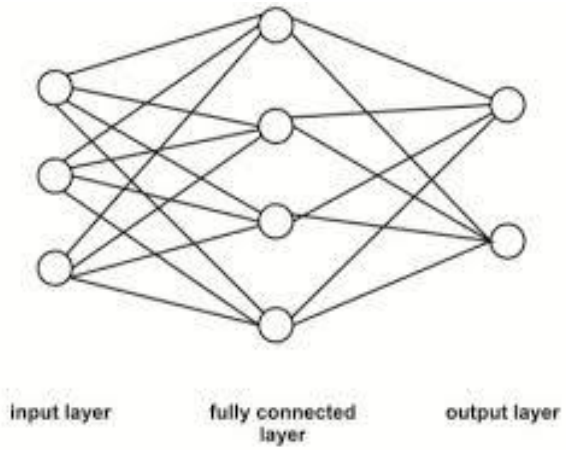
disisipkan secara teratur setelah beberapa lapisan konvolusi. Pada model CNN, lapisan *pooling* biasanya disisipkan secara teratur setelah beberapa lapisan konvolusi. Lapisan *Pooling* yang dimasukkan diantara lapisan konvolusi secara berturut-turut dalam arsitektur model CNN dapat secara progresif mengurangi ukuran *volume output* pada *feature map*, sehingga mengurangi jumlah parameter dan perhitungan dijaringan, dan untuk mengendalikan *overfitting*. Lapisan *pooling* bekerja di setiap tumpukan *feature map* dan mengurangi ukurannya. Bentuk lapisan *pooling* yang paling umum adalah dengan menggunakan *filter* berukuran 2x2 yang diaplikasikan dengan langkah sebanyak 2 dan kemudian beroperasi pada setiap irisan dari *input*. Bentuk seperti ini akan mengurangi *feature map* hingga 75% dari ukuran aslinya (Prijono, 2018). Contoh operasi *max pooling* ditunjukkan dalam gambar 2.14.



Gambar 2.14. Contoh operasi *max pooling*

2.4.5 *Fully Connected Layer*

Fully connected layer mengambil input dari hasil output *pooling layer* yang berupa *feature map*. *Feature map* tersebut masih berbentuk *multidimensional array* maka lapisan ini akan melakukan *reshape feature map* dan menghasilkan vektor sebanyak n-dimensi dimana n adalah jumlah kelas *output* yang harus dipilih *program*. Misalnya lapisan terdiri dari 500 *neuron*, maka akan diterapkan sebagai klasifikasi akhir dari jaringan (Dutt & Dutt, 2017). Gambar 2.15 menampilkan proses yang ada dalam *fully connected layer*.



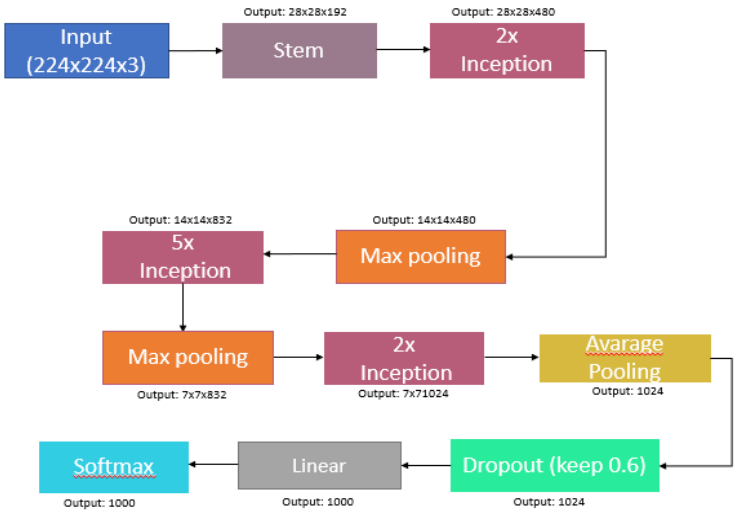
Gambar 2.15. Proses *fully connected layer*

2.5 *Inception*

Inception merupakan pengembangan dari *convolutional neural network* yang pertama kali diperkenalkan oleh Szegedy, dan kawan-kawan pada tahun 2014 dalam *paper* berjudul *"Going Deeper with Convolutions"*. *Very deep convolutional networks* telah menjadi pusat pengembangan dalam performa *image recognition* belakangan ini. Contohnya adalah arsitektur *inception* yang menghasilkan performa yang sangat baik dengan komputasi yang relatif rendah.

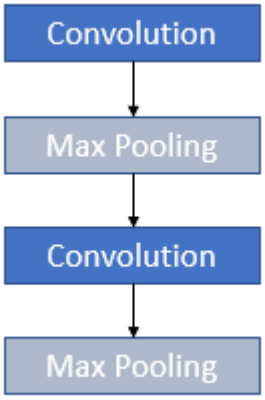
2.5.1 *Inception v1*

Inception v1 merupakan versi pertama dari *inception network*. *Inception* versi ini digunakan untuk menganalisis suatu permasalahan sederhana yang bisa dipecahkan. Pada gambar 2.16 merupakan diagram arsitektur dari *inception v1*



Gambar 2.16. Arsitektur *inception v1*

pada gambar 2.17 merupakan proses *stem* dari *inception v1*



Gambar 2.17. *Stem inception v1*

pada tabel 2.1 merupakan arsitektur rinci dari *inception v1* yang dikelompokkan berdasarkan layer. Jumlah *channel input* pada masing-masing *layer* ditentukan oleh jumlah kernel pada *layer* sebelumnya yang dimana sudah ditentukan oleh arsitektur *inception* sendiri, hal yang serupa berlaku pada *inception v2, v3 dan v4*.

Table 2.1 Arsitektur *inception v1*

Type	*Kernel size/stride Or remarks*	*Input Size*
Conv	7x7/2	*224x224x3*
Max pool	3x3/2	*112x112x64*
Conv	3x3/1	*56x56x64*
Max pool	3x3/2	*56x56x192*
Inception		*28x28x192*
Inception		*28x28x256*
Max pool	3x3/2	*28x28x480*
Inception		*14x14x480*
Inception		*14x14x512*
Inception		*14x14x512*
Inception		*14x14x528*
Inception		*14x14x832*
Max pool	3x3/2	*14x14x832*
Inception		*7x7x832*
Inception		*7x7x1024*

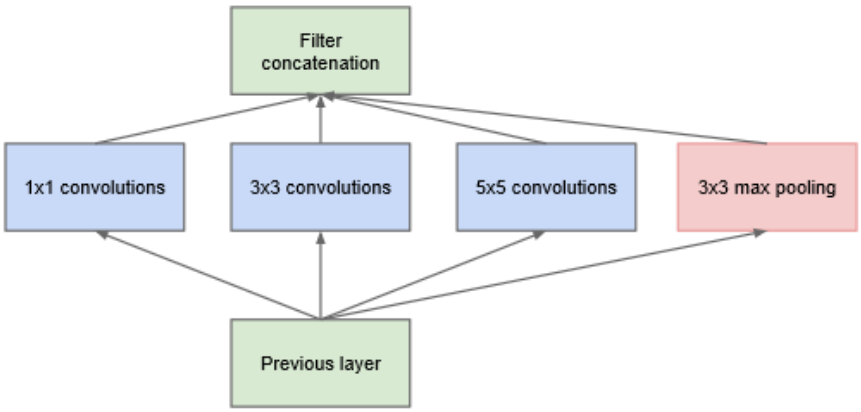
Type	*Kernel size/stride Or remarks*	*Input Size*
Avg pool	7x7/1	*7x7x1024*
Dropout (40%)		*1x1x1024*
Linear		*1x1x1000*
Softmax		*1x1x1000*

Biasanya sebuah gambar memiliki bagian yang terlihat paling menonjol, bagian yang menonjol tersebut dapat memiliki variasi ukuran yang sangat besar. Misalnya, gambar seekor anjing dalam beberapa ruang gambar berikut ini. Area yang ditempati oleh anjing berbeda dalam setiap gambar. Gambar 2.18 dari kiri, seekor anjing yang menempati sebagian besar ruang gambar, seekor anjing yang menempati sebagian ruang gambar, dan seekor anjing yang menempati sedikit ruang gambar.

Gambar 2.18. Dari kiri: Bunga daisy yang menempati sebagian besar ruang gambar, bunga daisy yang menempati sebagian ruang gambar, dan bunga daisy yang menempati sedikit ruang gambar

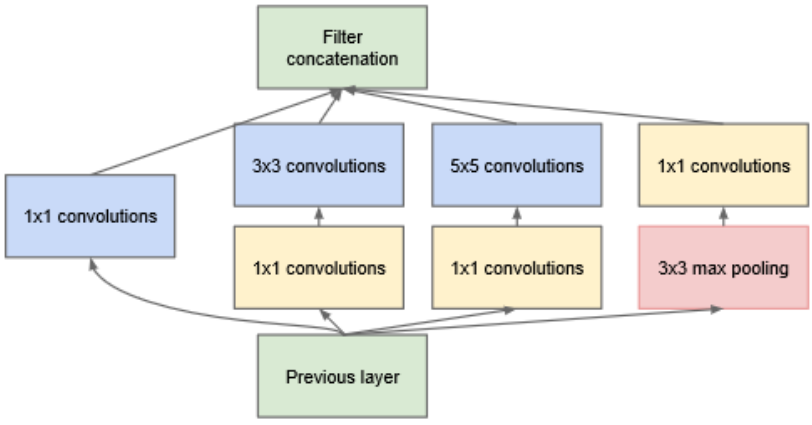
Karena banyaknya variasi informasi dari lokasi di dalam gambar, memilih ukuran kernel yang tepat untuk operasi *convolution* menjadi sulit. Kernel yang lebih besar lebih disukai untuk menyalurkan informasi secara *global*, sedangkan kernel yang lebih kecil lebih disukai untuk menyalurkan informasi secara lokal. Jaringan yang sangat *"deep"* atau mendalam *(very deep learning)* rentan terjadi *overfitting*, sehingga sulit untuk memperbarui *gradien* pada seluruh *network*. Operasi *convolution* yang menumpuk dalam jumlah besar juga dapat menyebabkan waktu komputasi yang lama atau mahal. Dengan memberikan *filter* yang dioperasikan pada *level* yang sama. Dengan demikian ada kemungkinan ukuran *network* akan sedikit lebih lebar (*wider*) dari pada lebih dalam (*deeper*). Hal tersebut gambaran awal dalam merancang *inception module*. Gambar 2.17

merupakan *naive inception module*. Pada gambar 2.19 dilakukan *convolution* pada input dengan 3 ukuran *filter* yang berbeda yaitu 1x1, 3x3, dan 5x5. Selain itu juga dilakukan *max pooling*, sedangkan *output* dirangkai dan dikirim ke *inception module* berikutnya.



Gambar 2.19. *Inception module native*

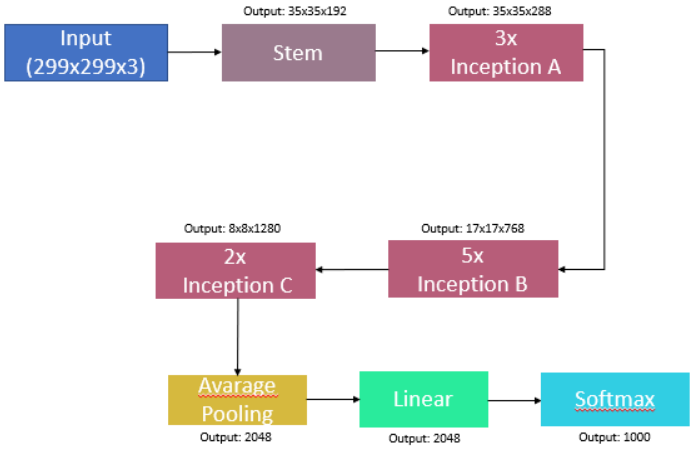
Deep neural network mahal secara komputasi (tidak efisien). Untuk mengatasinya, dengan cara membatasi jumlah *channel input* dengan menambahkan *convolution* 1x1 sebelum 3x3 dan 5x5. Meskipun dengan menambahkan operasi tersebut tampak berlawanan secara intuisi, tetapi *convolution* 1x1 jauh lebih murah dari pada *convolution* 5x5, dan juga dapat diatasi dengan mengurangi jumlah *input channelnya*. Dibandingkan dengan gambar 2.19, pada gambar 2.20 dibawah ini *convolution* 1x1 dimasukkan setelah *layer max pooling* dibawah ini *convolution* 1x1 dimasukkan setelah *layer max pooling* (Szegedy, dkk., 2015).



Gambar 2.20. *Inception module* dengan pengurangan dimensi

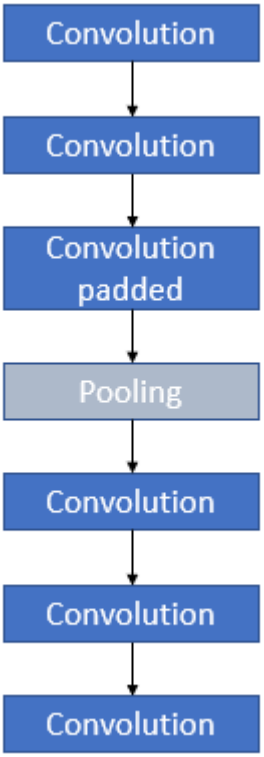
2.5.2 *Inception v2* dan *Inception v3*

Arsitektur dari *Inception v2* dirancang untuk mengurangi kompleksitas CNN, yang dilakukan dengan cara menyusun arsitektur yang lebih melebar dari pada mendalam. Pada gambar 2.21 merupakan arsitektur dari *inception v2* dan *v3*.



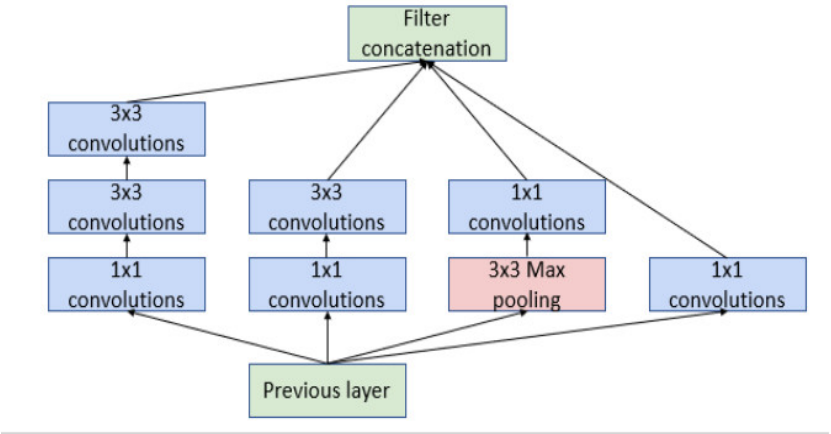
Gambar 2.21. Arsitektur *Inception v2* & *v3*

pada gambar 2.22 merupakan proses *stem* dari *inception v2 & v3*

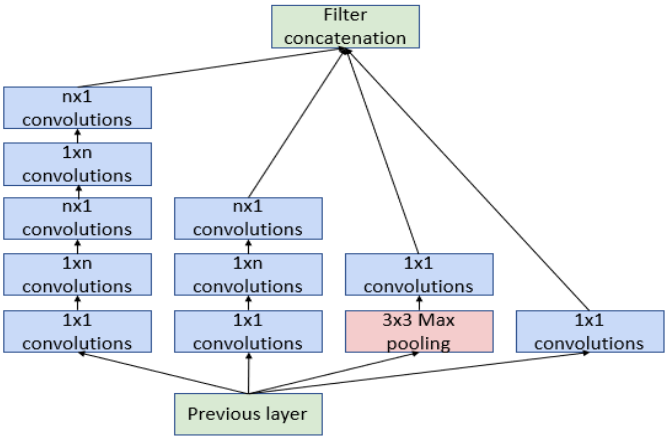


Gambar 2.22 *Stem inception v2* & *v3*

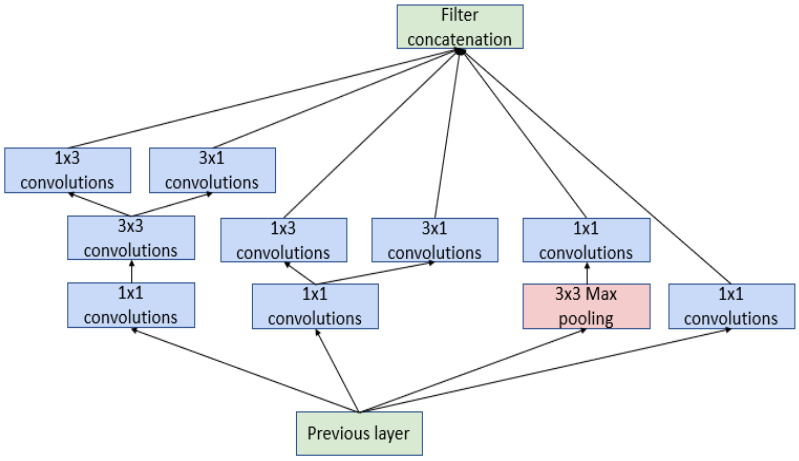
Inception v2 memiliki 3 modul yang ditunjukan oleh gambar 2.23, 2.24, 2.25, 2.26. Modul pertama (a) menggantikan konvolusi 5x5 menjadi 3x3. Selanjutnya pemfaktoran konvolusi dilakukan pada modul (b). Selanjutnya modul diubah lebih melebar untuk mengurangi kompleksitas jaringan konvolusi (c). Terakhir mengecilkan *grid size input* dari 35x35 menjadi 17x17 (d) (Szegedy, dkk., 2016).



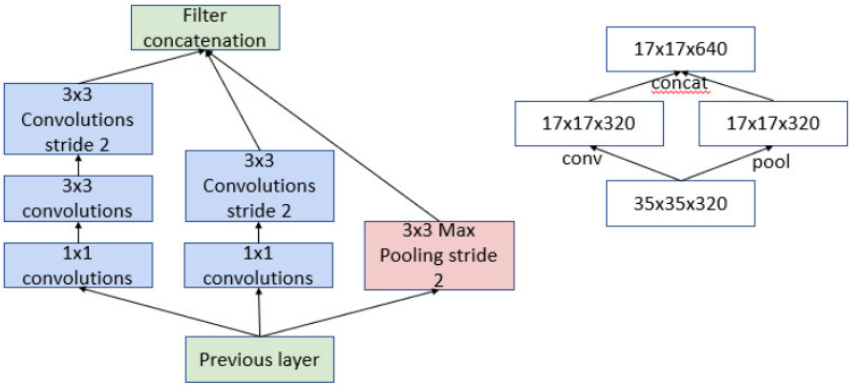
Gambar 2.23. Modul a *inception v2* & *v3*



Gambar 2.24. Modul b *inception v2 & v3*



Gambar 2.25. Modul c *inception v2 & v3*



Gambar 2.26. Modul d *inception v2 & v3*

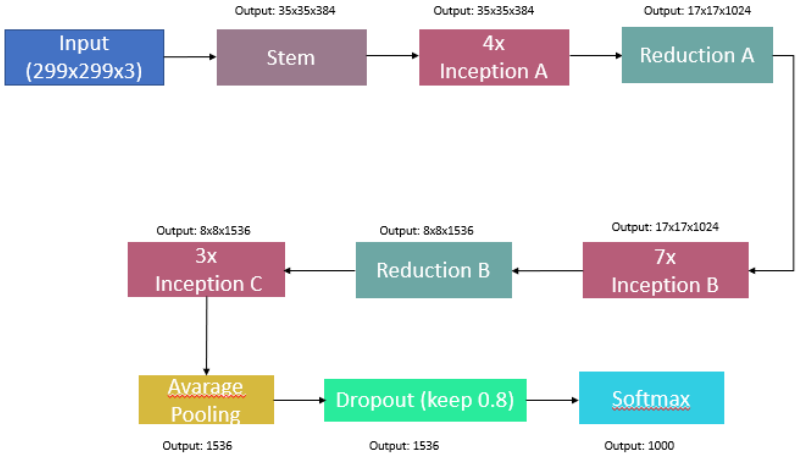
Table 2.2. Arsitektur *Inception v2 dan Inception v3*

Type	*Patch size/stride Or remarks*	*Input Size*
Conv	3x3/2	299x299x3
Conv	3x3/1	149x149x32
Conv padded	3x3/1	147x147x32
Pool	3x3/2	147x147x64
Conv	3x3/1	73x73x64
Conv	3x3/2	71x71x80
Conv	3x3/1	35x35x192
3x *Inception A*		35x35x288
5x *Inception B*		17x17x768
2x *Inception C*		8x8x1280
Pool	8x8	8x8x2048
Linear	*Logits*	1x1x2048
Softmax	*classifier*	1x1x1000

Ukuran *output* masing-masing modul adalah ukuran input dari layer berikutnya. menggunakan variasi teknik pengurangan digambarkan table 2.2 untuk mengurangi ukuran grid antara blok awal setiap kali berlaku.

2.5.3 *Inception v4*

Membuat modul lebih seragam. Diperhatikan bahwa beberapa modul lebih rumit dari yang diperlukan. Sehingga kinerja perlu ditingkatkan yaitu dengan menambahkan lebih banyak lagi modul-modul seragam (*uniform modules*) tersebut. Pada gambar 2.27 merupakan diagram arsitektur dari *inception v4*



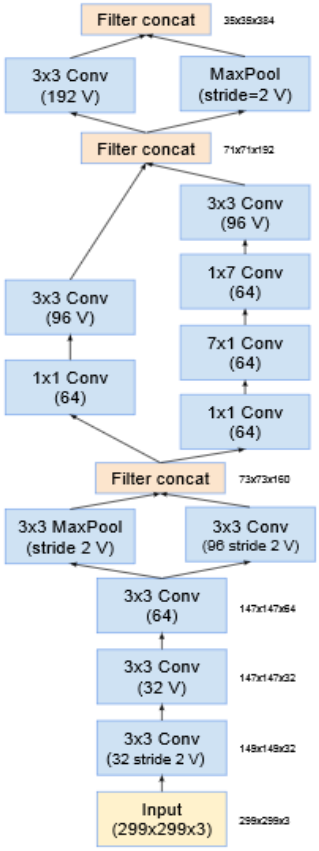
Gambar 2.27. Arsitektur *inception v4*

pada tabel 2.3 merupakan arsitektur rinci dari *inception v4* yang dikelompokkan berdasarkan *layer*.

Table 2.3. Arsitektur *Inception v4*

Type	*Patch size/stride Or remarks*	*Input Size*
Stem		299x299x3
4x *Inception A*		35x35x384
Reduction A		35x35x384
7x *Inception B*		17x17x1024
Reduction B		17x17x1024
3x *Inception C*		8x8x1536
Avarage pool	8x8	8x8x1536
Dropout (20%)		1x1x1536
Softmax		1x1x1536

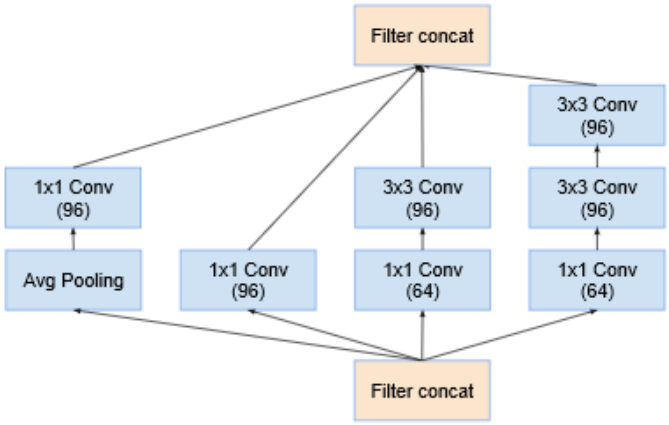
Gambar 2.28 menunjukkan arsitektur lengkap dari *inception v4* yang dimana diawali dari *input* ke *stem*. *Stem*, dalam hal ini mengacu pada operasi *initial set* yang dilakukan sebelum memperkenalkan blok-blok *inception*.



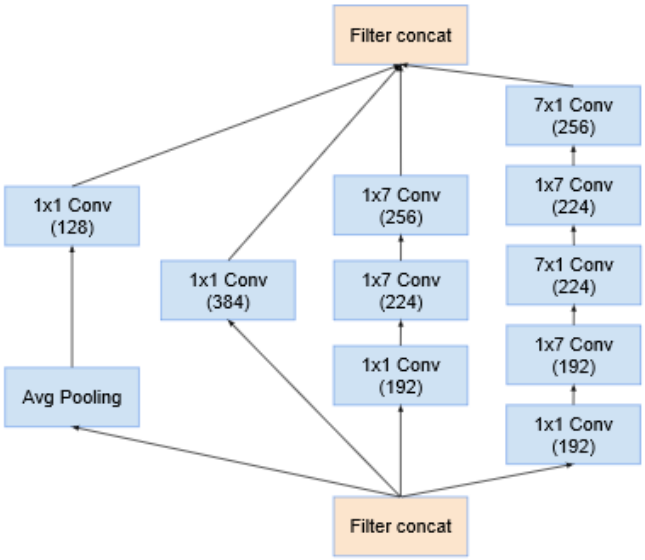
Gambar 2.28. *Stem inception v4*

Pada gambar 2.28, *convolution* yang ditandai dengan "V" menggunakan *same padding* sehingga dimensi *output* akan sama dengan dimensi *input*. Sebaliknya, *convolution* yang tidak ditandai dengan "V" menggunakan *valid padding*.

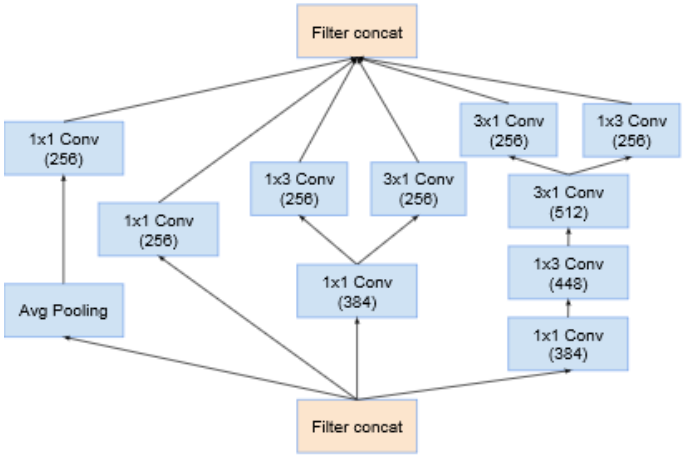
Inception v4 dibagi menjadi 3 modul, yaitu modul A, B, dan C yang terdapat pada gambar 2.29, 2.30, dan 2.31. modul-modul ini terlihat mirip dengan *inception v2*.



Gambar 2.29. Modul A *inception v4*

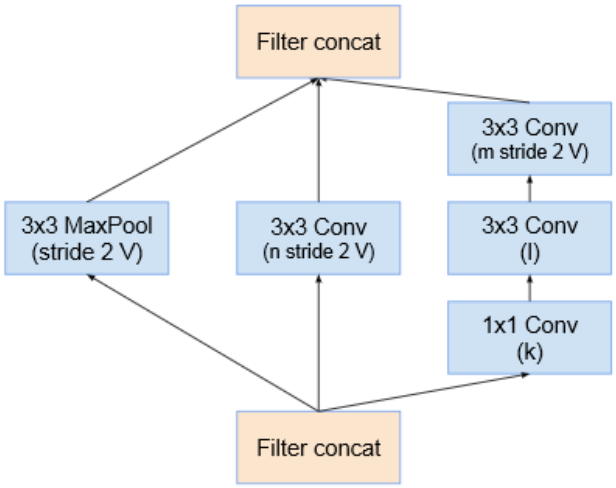


Gambar 2.30. Modul B *inception v4*

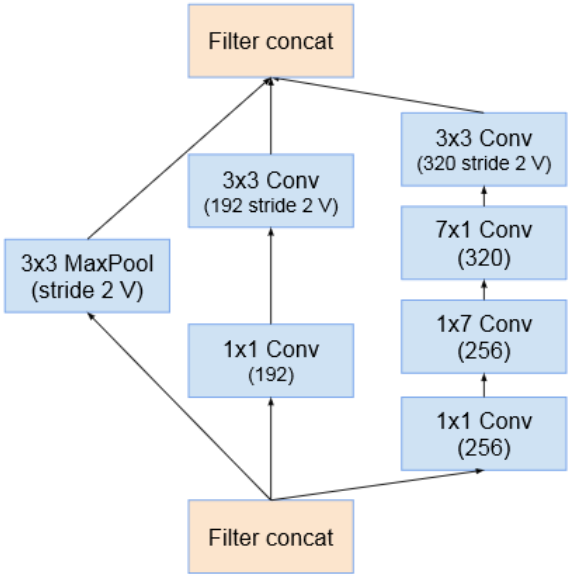


Gambar 2.31. Modul C *inception v4*

Inception v4 memperkenalkan *reduction blocks* khusus yang digunakan untuk mengubah lebar dan tinggi *grid*. Versi sebelumnya tidak secara eksplisit memiliki *reduction blocks*, tetapi fungsionalitas ini juga diterapkan. *Reduction blocks* bisa kita lihat pada gambar 2.32 dan 2.33.



Gambar 2.32. *Reduction block A inception v4*



Gambar 2.33. *Reduciton block b inception v4*

2.6 *Tensorflow*

Tensorflow adalah sebuah perpustakaan bersumber terbuka untuk komputasi numerik dan pembelajaran mesin berskala besar. *Tensorflow bundle* bersama-sama membunuh pembelajaran mesin mendalam (jaringan saraf) model dan algoritma dan membuat mereka berguna dengan cara metafora yang umum. Menggunakan *python* untuk menyediakan API *Front-end* yang nyaman untuk membangun aplikasi dengan *framework*, sementara mengeksekusi aplikasi tersebut dalam C++ berkinerja tinggi.

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 (Yegulalp, 2019).

2.6.1 *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 (Louis, dkk., 2019).

2.7 *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 (Sena, 2018). 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 *finetuning* (Akçay, dkk., 2016)

2.8 *Softmax*

Fungsi *softmax* menghitung probabilitas dari setiap kelas target atas semua kelas target yang memungkinkan dan akan membantu untuk menentukan kelas target untuk *input* yang diberikan (Sofia, 2018).

Jika diberikan suatu *convolutional neural network* dengan jumlah *node output* n_o(kelas) maka peluang masing-masing kelas *o* dapat dinyatakan

$$P\ (y = o) = \frac{a_o}{\sum_{i=1}^{n_o} a_i} \tag{7}$$

dengan $a_i$adalah nilai aktivasi pada *node* output ke i.

2.9 *Categorical Crossentropy*

Categorical crossentrophy adalah *loss function* yang digunakan dalam *multiclass classification*. Dimana sebuah sampel hanya bisa menjadi salah satu dari banyak kategori yang mungkin.

Categorical crossentropy loss function dapat dihitung dengan menggunakan persamaan 4

$$Loss = -\sum_{i=1}^{n_o} yi \cdot log\hat{y}\imath \tag{8}$$

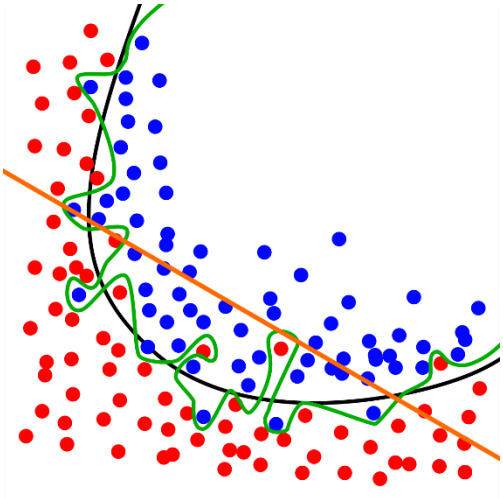
Dimana $\hat{y}\imath$ adalah prediksi nilai skalar pada *output* ke i, sedangkan yi adalah nilai target yang sesuai, dan n_o adalah jumlah *node* pada *output layer* (Peltarion, 2018).

2.10 *Overfitting & Underfitting*

Overfitting adalah salah satu masalah terbesar dalam pelatihan jaringan saraf tiruan. *Overfitting* bertarti bahwa jaringan saraf pada waktu tertentu selama periode pelatihan tidak meningkatkan kemampuannya untuk memecahkan masalah lagi. *Overfitting* terjadi karena model lebih fokus mempelajari *random error* pada data atau *noise* dibanding dengan hubungannya secara umum (Jabbar & Khan, 2015). Pada jaringan saraf tiruan *overfitting* dapat dengan mudah diatasi dengan menambahkan *dropout layer* (Srivastava, dkk., 2014)

Underfitting adalah kebalikan dari *overfitting*. Hal ini terjadi ketika model tidak mampu menangkap variabilitas data. Sebagai contoh suatu model *linear* untuk mempelajari dataset yang distribusi datanya berbentuk parabola rsebut tidak akan memiliki kekuatan prediksi (Jabbar & Khan, 2015).

Underfitting dapat diatasi dengan menggunakan model yang lebih kompleks. Pada gambar 2.34 garis hijau menunjukkan model yang mempelajari training data dengan sempurna tanpa generalisasi dan pada garis jingga menunjukkan model yang tidak mampu menangkap variabilitas data. Hal ini jg biasa disebut overgeneralisasi.



Gambar 2.34. Model yang *underfitting*(jingga) & *overfitting*(hijau)