SKRIPSI

KLASIFIKASI BAKTERI DENGAN METODE DEEP LEARNING

Disusun dan diajukan oleh

ALIF TRI HANDOYO

D42116504

DEPARTEMEN TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS HASANUDDIN

MAKASSAR

2021

LEMBAR PENGESAHAN SKRIPSI

KLASIFIKASI BAKTERI DENGAN METODE DEEP LEARNING

Disusun dan diajukan oleh:

ALIF TRI HANDOYO

D421 16 504

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Teknik Informatika Fakultas Teknik Universitas Hasanuddin
pada tanggal 1 Februari 2021
dan dinyatakan telah memenuhi syarat kelulusan

menyetujui,

Pembimbing Utama,

Dr. Ir. Ingrid Nurtanio, M.T.
NIP. 19610813 198811 2 001

Pembimbing Pendamping,

Anugrayani Bustamin, S.T., M.T.
NIP. 19901201 201807 4 001

Ketua Program Studi,

Dr. Amil Ahmad Ilham, S.T., M.IT
NIP. 19731010 199802 1 001

PERNYATAAN KEASLIAN

Yang bertanda tangan dibawah ini :

Nama	: Alif Tri Handoyo
NIM	: D421 16 504
Program Studi	: Teknik Informatika
Jenjang	: S1

Menyatakan dengan ini bahwa karya tulisan saya berjudul :

KLASIFIKASI BAKTERI DENGAN METODE DEEP LEARNING

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

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, 1 Februari 2021

Yang menyatakan,

(Alif Tri Handoyo)

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa karena berkat rahmat dan karunia-Nya sehingga skripsi yang berjudul "**Klasifikasi Bakteri dengan Metode *Deep Learning***" ini dapat terselesaikan dengan baik sebagai salah satu syarat dalam menyelesaikan jenjang Strata-1 pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin.

Penulis menyadari bahwa banyak kendala yang dihadapi dalam menyelesaikan skripsi ini, namun berkat dorongan, dukungan, bimbingan, serta motivasi yang diberikan sehingga penyusunan skripsi ini dapat terselesaikan dengan baik. Ucapan terima kasih serta penghargaan yang setinggi-tingginya penulis sampaikan kepada:

1. Tuhan Yang Maha Esa yang melalui berkat dan rahmat-Nya sehingga penulis dapat menyelesaikan tugas akhir ini.
2. Kedua Orang tua penulis, Bapak Ir. Hadi Purwanto, M.T., dan Ibu St. Syarfiah yang selalu memberikan dukungan, doa, semangat dan kasih sayang serta selalu sabar dalam mendidik penulis sejak kecil.
3. Ibu Dr. Ir. Ingrid Nurtanio, M.T., selaku pembimbing I dan Ibu Anugrayani Bustamin, S.T., M.T., selaku pembimbing II yang selalu menyediakan waktu, tenaga, pikiran dan memberikan bimbingan dalam penyusunan skripsi ini.
4. Bapak Dr. Amil Ahmad Ilham, S.T., M.IT., selaku ketua Prodi Teknik Informatika Fakultas Teknik Universitas Hasanuddin atas ilmu, bimbingan dan arahannya selama masa perkuliahan penulis.

5. Segenap Dosen dan Staff Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin yang telah banyak membantu penulis selama masa perkuliahan.
6. Teman-teman Laboratorium AIMP *Research Group* FT-UH yang telah memberikan bantuan selama penelitian, dan diskusi mengenai penyusunan skripsi.
7. Teman-teman Prodi Teknik Informatika angkatan 2016 atas dukungan dan semangat yang telah diberikan.
8. Serta berbagai pihak atas segala dukungan dan bantuannya yang tidak dapat penulis tuliskan satu persatu.

Akhir kata, penulis berharap semoga Allah SWT. membalas segala kebaikan dari semua pihak yang telah membantu penulis dalam penyusunan skripsi ini, dan semoga skripsi ini dapat memberikan hal yang bermanfaat serta menambah wawasan ilmu untuk pembaca dan juga bagi penulis sendiri.

Makassar, 1 Februari 2021

Penulis,

(Alif Tri Handoyo)

ABSTRAK

Mikroorganisme seperti bakteri menjadi penyebab utama berbagai penyakit infeksi seperti kolera, botulisme, gonore, penyakit lyme, radang tenggorokan, TBC dan sebagainya. Oleh karena itu, identifikasi dan klasifikasi bakteri sangat penting dalam dunia kedokteran untuk membantu para ahli mendiagnosa penyakit yang diderita oleh pasien. Namun, untuk identifikasi dan klasifikasi bakteri secara manual dibutuhkan individu profesional dan waktu yang lama. Dengan bantuan kecerdasan buatan kita dapat secara efektif dan efisien mengklasifikasikan bakteri dan menghemat banyak waktu serta tenaga manusia. Dalam penelitian ini, dibuat sistem untuk mengklasifikasikan bakteri dari sampel gambar mikroskopis. Sistem ini menggunakan *deep learning* dengan metode *transfer learning*. Arsitektur Inception V3 dimodifikasi dan di*training* ulang menggunakan 108 sampel citra dengan *label* lima jenis bakteri yaitu *Acinetobacter baumanii, Escherichia coli, Neisseria gonorrhoeae, Propionibacterium acnes dan Veionella.* Data kemudian dibagi menjadi *training* dan *validation* dengan metode *k-fold cross validation.* Selanjutnya, fitur yang telah diekstraksi oleh model di *training* dengan konfigurasi *minibatchsize 5, maxepoch 5, initiallearnrate 0,0001, dan validationfrequency 3.* Model kemudian diuji dengan data *validation* dengan melakukan sepuluh kali percobaan dan mendapatkan nilai rata-rata akurasi sebesar 94.43% dan rata-rata waktu eksekusi selama 44 detik.

Kata kunci: Klasifikasi bakteri, *Deep Learning, Inception V3*, *Transfer Learning*.

ABSTRACT

Microorganisms such as bacteria are the main cause of various infectious diseases such as cholera, botulism, gonorrhea, Lyme disease, sore throat, tuberculosis and so on. Therefore, identification and classification of bacteria is very important in the world of medicine to help experts diagnose diseases suffered by patients. However, manual identification and classification of bacteria takes a long time and a professional individual. With the help of artificial intelligence we can effectively and efficiently classify bacteria and save a lot of time and human labor. In this study, a system was created to classify bacteria from microscopic image samples. This system uses deep learning with the transfer learning method. Inception V3 architecture was modified and retained using 108 image samples labeled with five types of bacteria, namely Acinetobacter baumanii, Escherichia coli, Neisseria gonorrhoeae, Propionibacterium acnes and Veionella. The data is then divided into training and validation using the k-fold cross validation method. Furthermore, the features that have been extracted by the model are trained with the configuration of minibatchsize 5, maxepoch 5, initiallearnrate 0.0001, and validation frequency 3. The model is then tested with data validation by conducting ten experiments and getting an average accuracy value of 94.43% and average execution time of 44 seconds.

Keywords: Bacterial Classification, Deep Learning, Inception V3, Transfer Learning.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1. Latar Belakang

Menurut *Food and Agriculture Organization* (FAO), setiap tahunnya jumlah korban meninggal akibat infeksi bakteri mencapai hingga 700.000 jiwa. Pengenalan genera dan spesies bakteri dibutuhkan karena pengetahuan biologis mikroorganisme sangat penting dalam ilmu kedokteran, kedokteran hewan, biokimia, industri makanan atau pertanian. Meskipun sebagian besar mikroorganisme berdampak positif pada berbagai bidang kehidupan, mereka dapat menjadi penyebab banyak penyakit, termasuk penyakit menular. (Zieliński *et al.*, 2017).

Ahli biologi mengidentifikasi dan mengklasifikasikan berbagai jenis bakteri yang memiliki biokimia dan bentuk yang berbeda. Mereka menggunakan atribut bakteri yang berbeda untuk klasifikasi. Misalnya, bentuk sel bakteri (spiral, silindris, dan bola), ukuran dan struktur koloni yang dibentuk oleh bakteri diperiksa untuk membedakan spesies bakteri. Sel dari beberapa jenis bakteri memiliki ukuran dan struktur yang berbeda tergantung pada kondisi lingkungan. Beberapa spesies bakteri memiliki bentuk yang sangat mirip. Meskipun setiap spesies bakteri memiliki karakteristiknya sendiri, reaksi biokimia yang dilakukan oleh bakteri dan aktivitas metabolisme yang mereka lakukan bersama-sama membantu untuk mengklasifikasikan spesies. Namun, klasifikasi spesies bakteri bukanlah tugas yang mudah bahkan bagi seorang spesialis yang berpengalaman (Talo, 2019).

Secara umum, analisis citra mikrobiologi dengan metode laboratorium tradisional terdapat kesalahan pengenalan bakteri, dan membutuhkan pengalaman ekstra serta waktu pengerjaan yang lama. Oleh karena itu, teknik klasifikasi otomatis citra bakteri lebih bermanfaat dibanding pengamatan visual tradisional bagi ahli biologi karena pengklasifikasian yang akurat, biaya rendah, dan diagnosis cepat (Mohamed dan Afify, 2019).

Penelitian sebelumnya terkait klasifikasi bakteri dilakukan pada tahun 2018 oleh basma, dkk melakukan penelitian dengan mengklasifikasi sepuluh macam bakteri menggunakan *feature extraction Bag of words* serta metode *Support Vector Machine* (SVM) dengan hasil akurasi 97%.

Pada tahun 2019 Treesukon, dkk melakukan penilitian dengan mengklasifikasi dua macam bakteri dengan metode *deep learning* berbasis *python* dan menggunakan arsitektur LeNET dengan hasil akurasi lebih dari 75 persen. Pada tahun 2018 lei huang menggunakan metode CNN dengan arsitektur AlexNET untuk mengklasifikasi 18 kategori bakteri dengan hasil akurasi 73%.

Pada penelitian ini penulis akan membuat sistem yang mengklasifikasi bakteri dengan metode *deep learning* CNN menggunakan arsitektur *InceptionV3.* Arsitektur *InceptionV3* digunakan karena dalam penelitian sebelumnya banyak digunakan untuk mengklasifikasi dengan hasil akurasi yang cukup tinggi seperti klasifikasi penderita penyakit covid-19 (Asif *et al.*, 2020), dan juga klasifikasi penderita penyakit paru-paru (Wang *et al.*, 2019).

1.2. Rumusan Masalah

Rumusan masalah yang akan diuraikan dalam skripsi ini antara lain:

1. Bagaimana proses klasifikasi bakteri dari sebuah *dataset image* menggunakan algoritma CNN dengan arsitektur *InceptionV3*?
2. Bagaimana unjuk kerja algoritma CNN dengan arsitektur *InceptionV3* dari sisi akurasi dan waktu proses *training* pada klasifikasi bakteri?

1.3. Tujuan Penelitian

Tujuan akhir dari penelitian ini antara lain:

1. Untuk melihat hasil proses klasifikasi bakteri dari sebuah *dataset image* menggunakan algoritma CNN dengan arsitektur *InceptionV3*.
2. Untuk mengetahui unjuk kerja algoritma CNN dengan arsitektur *InceptionV3* dari sisi akurasi dan waktu proses *training* dalam mengklasifikasikan bakteri.

1.4. Manfaat Penelitian

Dengan dilakukannya penelitian ini, diharapkan manfaat yang didapatkan antara lain:

1. Bagi masyarakat, penelitian ini diharapkan secara tidak langsung dapat membantu masyarakat dengan membantu perkembangan dari sistem kesehatan.
2. Bagi peneliti, penelitian ini dapat dijadikan referensi terkait kemampuan arsitektur *InceptionV3* dalam mengklasifikasi bakteri.
3. Bagi institusi pendidikan, penelitian ini dapat digunakan sebagai referensi ilmiah untuk penelitian-penelitian selanjutnya.

1.5. Batasan Masalah Penelitian

Yang menjadi Batasan masalah dalam skripsi ini adalah:

1. Data yang digunakan diambil dari *misztal.edu.pl/software/databases/dibas/*.
2. *Dataset* penelitian terdiri dari lima macam bakteri berbahaya menurut *CDC's Antibiotic Resistance Threats in the United States, 2019 (2019 AR Threats Report),* yaitu bakteri *Acinetobacter baumanii, Escherichia coli, Neisseria gonorrhoeae, Propionibacterium acnes dan Veionella*.
3. Sistem Klasifikasi dibuat menggunakan *Matlab R2020b* dengan menggunakan arsitektur *InceptionV3*.

1.6. Sistematika Penulisan

Untuk memberikan gambaran singkat mengenai isi tulisan secara keseluruhan, maka akan diuraikan beberapa tahapan dari penulisan secara sistematis, yaitu :

BAB I PENDAHULUAN

Bab ini menguraikan secara umum mengenai hal yang menyangkut latar belakang, perumusan masalah dan bahasan masalah, tujuan, dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini berisi teori-teori tentang hal-hal yang berhubungan dengan klasifikasi bakteri, visi komputer, pemrosesan citra, *deep learning*, dan metode yang digunakan.

BAB III METODOLOGI PENELITIAN

Bab ini berisi tentang tahap penelitian, instrumen penelitian, dan penerapan algoritma serta teknik pengolahan data.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi tentang hasil dan pengolahan data serta pembahasan yang disertai tabel hasil penelitian.

BAB V PENUTUP

Bab ini berisi tentang kesimpulan yang didapatkan berdasarkan hasil penelitian yang telah dilakukan serta saran-saran untuk pengembangan lebih lanjut.

BAB II

TINJAUAN PUSTAKA

2.1. Pengertian Bakteri

Bakteri merupakan penyebab utama berbagai penyakit. Hingga pertengahan abad ke-20, penyakit infeksi bakteri menjadi penyebab utama kematian di kalangan lansia. Sanitasi yang lebih baik, vaksin, dan antibiotik telah berhasil menurunkan tingkat kematian akibat infeksi bakteri, meskipun seiring berjalan nya waktu, ada beberepa jenis bakteri yang mulai kebal terhadap antibiotik sehingga menyebabkan penyakit yang ditimbulkan nya semakin sulit untuk diobati. Bakteri menyebabkan penyakit dengan mengeluarkan racun (seperti dalam botulisme), dengan memproduksi racun secara internal, yang dilepaskan saat bakteri membelah diri (seperti pada bakteri penyebab penyakit tipus), atau dengan menginduksi kepekaan terhadap sifat antigeniknya (seperti pada bakteri penyebab penyakit tuberkulosis). Penyakit bakteri serius lainnya termasuk kolera, difteri, meningitis bakterial, tetanus, penyakit lyme, gonore, dan sifilis (Britannica, 2018).

2.2. Sejarah Klasifikasi Bakteri

Tabel 2.1 Sejarah Klasifikasi Bakteri

Rentang waktu (Tahun)	Klasifikasi utama berdasarkan
Akhir abad ke -19	*Morphology, Growth Requirements, Pathogenic potential.*
1900 – 1960	*Morphology, Physiology, Biochemistry.*

Rentang waktu (Tahun)	Klasifikasi utama berdasarkan
1960 – 1980	*Chemotaxonomy, Numerical Taxonomy, DNA–DNA Hybridization.*
1980 – sekarang	*Genotypic Analyses, Multilocus Sequence Analyses, Average Nucleotide Identity, Whole Genome Analysis.*

Sejarah dari klasifikasi bakteri dengan jelas menunjukkan bahwa perubahan disebabkan oleh muncul nya teknik - teknik baru dalam proses klasifikasi, hal ini dapat dilihat diatas pada Tabel 2.1. Akhir abad ke-19 adalah awal taksonomi bakteri dan Ferdinand Cohn pada tahun 1872 adalah orang pertama yang mengklasifikasikan enam genera bakteri (sebagai anggota tumbuhan) yang sebagian besar berdasarkan morfologi mereka. Namun, saat itu mayoritas ilmuwan tertarik dengan gambaran bakteri patogen. Sebenarnya, banyak bakteri patogen yang dikenal saat ini ditemukan antara tahun 1880 dan 1900. Pada saat itu, selain morfologi, persyaratan pertumbuhan dan potensi patogen merupakan penanda taksonomi yang paling penting.

Pada awal abad ke-20 semakin banyak data fisiologis dan biokimia yang digunakan, selain morfologi sebagai penanda penting untuk klasifikasi dan identifikasi mikroorganisme, berbagai sifat biokimia dan fisiologis kultur bakteri ditentukan untuk karakterisasi dan identifikasi mereka. Kemudian, enzim dipelajari

dan jalur metabolisme dijelaskan. Edisi pertama dari *Bergey's Manual of Determinative Bacteriology* mengklasifikasikan bakteri pada tahun 1923 atas dasar sifat fenotipik sebagai "tipikal tumbuhan *unicellular*", yang disebut *schizomycetes*. Bahkan dalam *bergey's* manual edisi ke-7 yang diterbitkan pada tahun 1957, bakteri masih diklasifikasikan sebagai anggota tumbuhan (*protophyta*, tumbuhan primitif). Berdasarkan sekuens parsial dari gen *16S ribosomal RNA* (rRNA), *Archaea* (awalnya bernama *Archaebacteria*) pertama kali diklasifikasikan sebagai kategori terpisah pada tahun 1977.

Protistolog Prancis Edouard Chatton, mentor dan teman lama A. Lwoff, menyebutkan untuk pertama kalinya pada tahun 1925 dua kategori *prokariota* dan *eukariota* tetapi hanya untuk membedakan *prokariotik* dari *protista eukariotik* saja. Namun, proposalnya tidak diketahui secara umum. Kemudian, A. Lwoff menyebarkan perbedaan ini dan akhirnya meyakinkan R. Stanier, bersama dengan C.B. van Niel pada tahun 1962, untuk menjelaskan pembagian organisme *prokariotik* (bakteri) dan *eukariotik* (hewan, tumbuhan) yang rinci dan diterima dengan baik. Dalam *bergey's manual* edisi ke-8, yang diterbitkan pada tahun 1974, bakteri tidak lagi dianggap sebagai tumbuhan dan diakui sebagai anggota kategori *Procaryotae*. Namun, semua gagasan sebelumnya tentang *filogeni* dan hubungannya dibuang dan bakteri diatur dalam kelompok berdasarkan pada pewarnaan gram, morfologi dan kebutuhan oksigen (Schleifer, 2009).

2.3. Metode Klasifikasi Bakteri

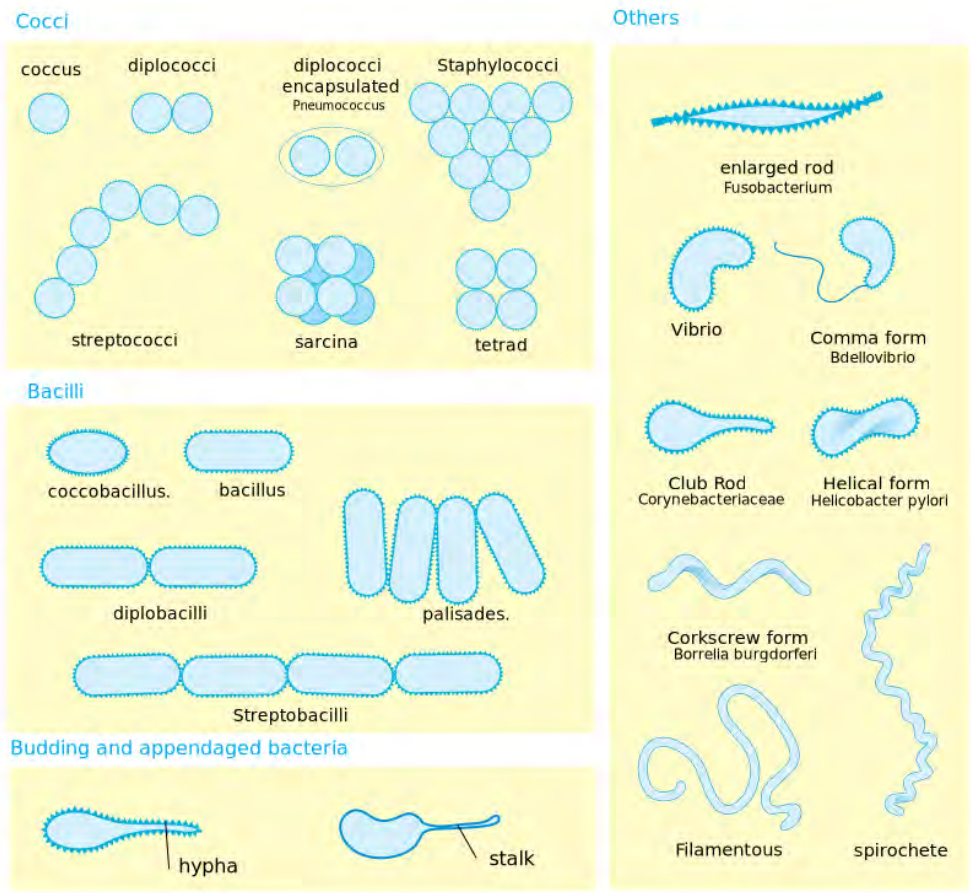
Bakteri dapat diklasifikasikan berdasarkan bentuk, pewarnaan Gram, suhu, dan kebutuhan oksigennya. Berikut penjelasannya:

2.3.1. Klasifikasi Berdasarkan Bentuk

Jika dilihat dari bentuknya, bakteri dapat dibagi menjadi bakteri *coccus*, *bacilli*, dan *spiral*. *Coccus* memiliki bentuk bulat atau bujur telur. Jenis yang satu ini bisa hidup sendiri, tapi bisa juga hidup dalam formasi dengan bakteri sejenis lainnya. Dua *coccus* yang bergabung disebut *diplococci*, sementara empat *coccus* yang membentuk kotak disebut *tetrad*. Susunan yang umum didapatkan dari jenis ini adalah rantai bakteri yang disebut *streptococci*.

Bacilli atau *bacillus* adalah kelompok yang membentuk batang, walaupun nama *bacillus* juga digunakan sebagai *genus*. Kebanyakan bakteri di kategori ini berbentuk batang tunggal, walaupun ada juga *diplobacilli* yang muncul berpasangan dan *streptobacilli* yang muncul berantai.

Terakhir, spiral atau *spirochetes* adalah bakteri yang membentuk melengkung. Banyak jenis ini yang sifat tubuhnya kaku dan punya kemampuan untuk bergerak. Jenis spiral dibagi lagi menjadi *vibrio, spirilla,* dan *spirochetes*. *Vibrio* berbentuk seperti karakter koma, sementara *spirilla* punya struktur spiral yang kaku. Terakhir, Spirochetes membentuk spiral yang fleksibel (Putri, 2017). Ilustrasi bentuk bakteri ini dapat dilihat pada Gambar 2.1.



Gambar 2.1 Klasifikasi Bakteri Berdasarkan Bentuk (Putri, 2017).

2.3.2. Klasifikasi Berdasarkan Pewarnaan Gram

Pewarnaan Gram merupakan pewarnaan yang sangat umum dalam bidang bakteriologi. Dengan pewarnaan ini, kelompok bakteri dapat dibedakan menjadi dua, yaitu kelompok bakteri Gram positif dan bakteri Gram negatif . Ada banyak modifikasi dari teknik pewarnaan ini, namun semua teknik tersebut berdasarkan pada prinsip yang sama, yaitu :

1. Mewarnai mikroorganisme dengan pewarna dasar, yaitu dengan kristal violet atau gentian violet.

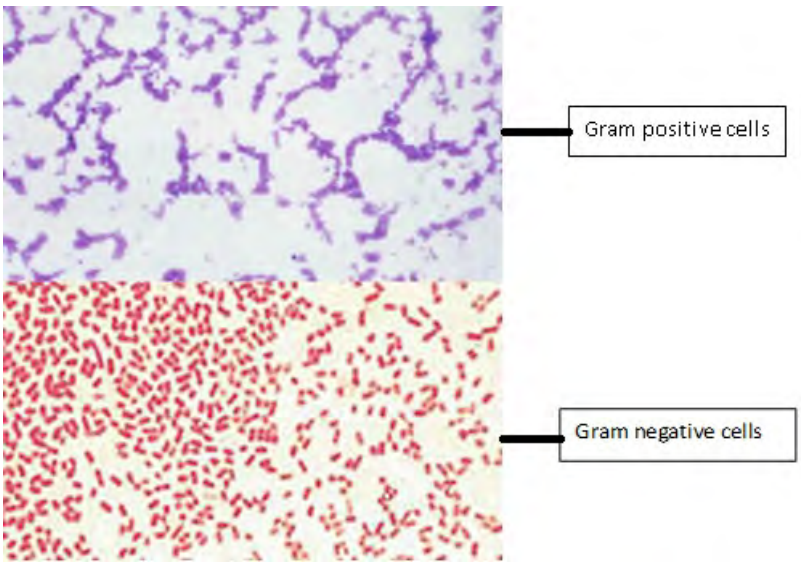
2. Fiksasi warna, yaitu untuk menguatkan perlekatan warna dasar, misalnya dilakukan dengan garam iodin (modifikasi larutan lugol).

3. Pencucian atau penghapusan warna dasar dengan alkohol, aseton, atau campuran alkohol dengan aseton.

4. Pewarnaan kembali dengan pewarna pembanding atau kontras yang berbeda dengan pewarna dasar, yaitu untuk mewarnai sel-sel yang telah hilang warnanya oleh penghapusan warna. Misalnya dalam hal ini dipakai safranin atau karbol fuhsin.

Bakeri yang setelah diwarnai dengan pewarna dasar warnanya tidak terhapus oleh alkohol akan berwarna violet karena terwarnai oleh kristal violet, dan tidak lagi menyerap pewarna kontras. Kelompok bakteri yang mempunyai sifat ini dikelompokkan menjadi bakteri gram positif. Sedangkan, kelompok bakteri yang telah diwarnai dengan pewarna dasar dan warnanya terhapus setelah diperlakukan dengan alkohol, akan menyerap pewarna safranin atau karbol fuhsin yang dipakai sebagai pewarna kontras, sehingga dalam preparat akan terlihat warna merah(warna safranin atau karbol fuhsin). Kelompok bakteri yang demikian disebut dengan bakteri gram negatif (Sujaya, 2016).

Ilustrasi bakteri melalui pewarnaan gram dapat dilihat pada Gambar 2.2.



Gambar 2.2 Klasifikasi Bakteri Berdasarkan Pewarnaan Gram (Putri, 2017).

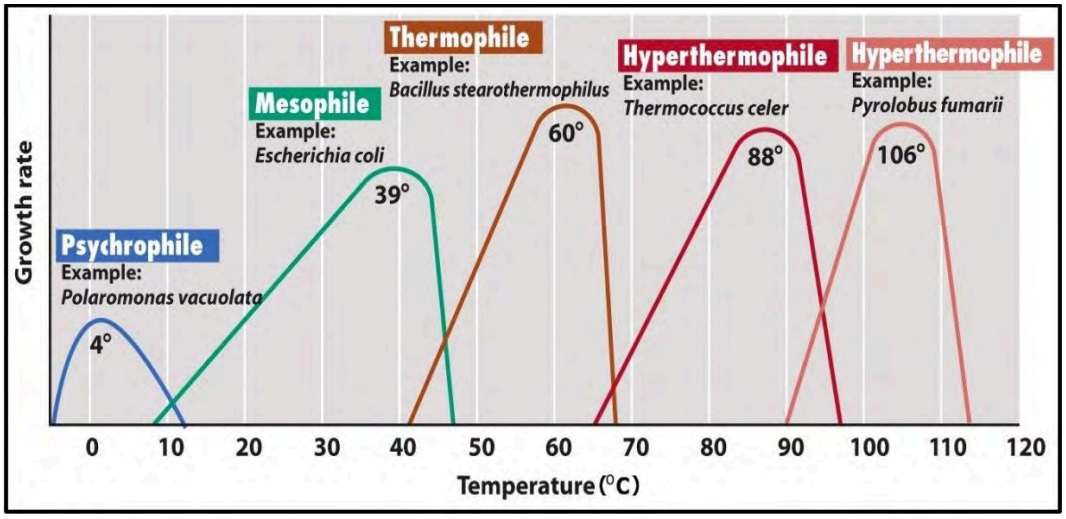
2.3.3. Klasifikasi Berdasarkan Suhu

Berdasarkan kemampuan adaptasinya terhadap suhu, bakteri terbagi menjadi tiga jenis, yaitu *thermophile, mesophile,* dan *psychrophile. Thermophile* mampu bertahan di lingkungan bersuhu tinggi, yaitu 41-122 derajat celsius. Jenis ini biasanya ditemukan di wilayah hangat di bumi, seperti mata air panas, lautan dalam *hidrotermal*, dan *kompos*.

Mesophile adalah jenis yang pertumbuhan optimalnya berada di suhu sedang, yaitu 20-45 derajat celsius. Biasanya, jenis ini terdapat pada keju dan yogurt. Sebagian besar patogen yang menyerang manusia juga termasuk pada *mesophile*.

Psychrophile adalah kelompok bakteri yang dapat tumbuh dan bereproduksi di suhu dingin, dari 20-10 derajat celsius. Ciri-cirinya adalah membran sel lipid yang secara kimia tahan terhadap suhu dingin dan sering membuat protein antibeku

untuk menjaga cairan *internal* dan DNA mereka (Putri, 2017). Ilustrasi klasifikasi bakteri berdasarkan suhu dapat dilihat pada Gambar 2.3.



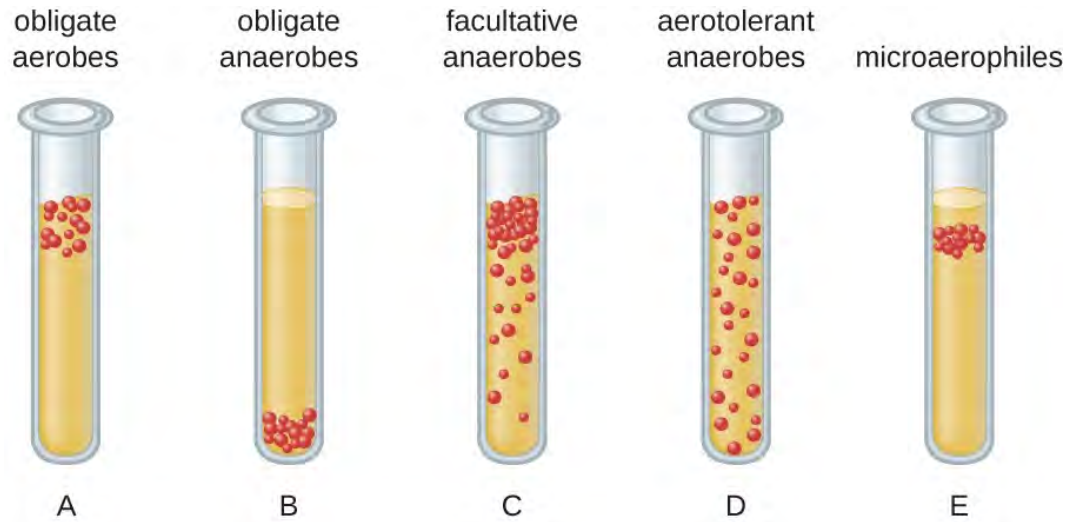
Gambar 2.3 Klasifikasi Bakteri Berdasarkan Suhu (Putri, 2017).

2.3.4. Klasifikasi Berdasarkan Kebutuhan Oksigen

Berdasarkan kebutuhannya akan oksigen, bakteri dibagi menjadi *aerobik, anaerobik, anaerobik fakultatif,* dan *micro-aerophilic*. *Aerobik* adalah bakteri yang memerlukan oksigen untuk proses metabolisme dan respirasi seluler. Jenis ini menggunakan oksigen untuk melakukan metabolisme senyawa seperti karbohidrat dan lemak untuk menghasilkan energi. Jenis ini juga dapat menghasilkan lebih banyak energi ATP daripada respirasi anaerobik atau fermentasi, tapi ia rawan terkena stres oksidatif.

Anaerobik dibagi menjadi tiga, yaitu *anaerobik obligat, anaerobik aerotoleran*, dan *anaerobik fakultatif*. *Anaerobik obligat* adalah jenis yang tidak membutuhkan oksigen untuk pertumbuhannya. Malah, jenis ini bisa mati jika terkena oksigen, walaupun dengan kadar yang berbeda-beda.

Anaerobik fakultatif membuat energi ATP dengan respirasi aerobik jika terdapat oksigen di lingkungannya, tapi bisa berganti menjadi respirasi anaerobik atau fermentasi jika tidak ada oksigen. Terakhir, *micro-aerophilic* adalah jenis yang membutuhkan oksigen untuk bertahan hidup, tapi hanya dalam konsentrasi rendah. Bakteri jenis ini memerlukan oksigen karena tidak mampu melakukan fermentasi ataupun respirasi anaerobik. Tapi, jenis ini akan teracuni jika terpapar konsentrasi oksigen yang tinggi (Putri, 2017). Untuk mengklasifikasi bakteri berdasarkan kebutuhan oksigennya, bakteri di kultur menggunakan tabung tioglikolat, ilustrasi nya dapat dilihat pada Gambar 2.4.



Gambar 2.4 Klasifikasi Bakteri Berdasarkan Kebutuhan Oksigen (Putri, 2017)

2.4. *Artificial Neural Network*

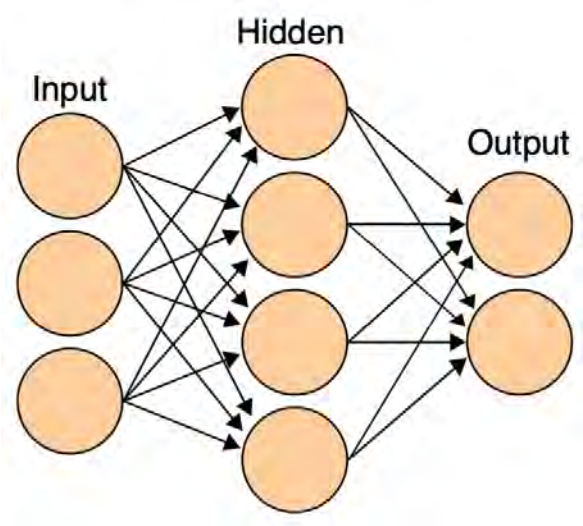
Artificial Neural Network atau ANN adalah model pembelajaran yang terinspirasi dari sistem biologis pembelajaran dan klasifikasi. Secara sederhana, *ANN* bekerja pada banyak lapisan yang terhubung melalui *neuron*:

• ***Input Layer*:** Dimana data diberikan sebagai *input*.

• ***Hidden Layer*****:** Data *input* kemudian melewati neuron berdasarkan fungsi aktivasi.

• ***Output Layer*****:** *Output* dihasilkan di sini.

Secara visual kita dapat menggambarkan ANN seperti pada Gambar 2.5



Gambar 2.5 Visualisasi *Artificial Neural Network* (Rayed Bin Wahed, 2016).

Jika kita membahas mengenai ANN, terdapat dua fungsi penting yaitu *basis function* dan *activation function. Basis function* adalah fungsi yang memproses *input* yang masuk ke *hidden layer.* Maka untuk *shallow* ANN*, output* dihasilkan dengan fungsi berikut:

$$y(\mathrm{x}, \mathrm{w}) = f\left(\sum_{j=1}^{M} w_j \, \Phi_j \, (\mathrm{x})\right) \tag{2.1}$$

Di sini, f diganti dengan fungsi aktivasi nonlinier, dan $\Phi_j \, (x)$ adalah fungsi basis yang disesuaikan dengan keofisien wj yang diberikan melalui vector (w).

Fungsi aktivasi adalah fungsi yang mengaktifkan neuron untuk melanjutkan *input* ke layer selanjutnya. Secara umum, fungsi aktivasi ini dapat berupa fungsi *nonlinear sigmoidal*.

Adapun fungsi aktivasi secara umum yaitu *logistic sigmoid* dengan persamaan :

$$\sigma(a) = \frac{1}{1+e^{-a}} \tag{2.2}$$

Adapun untuk masalah *multi-class*, yang digunakan adalah fungsi *softmax activation* dengan persamaan :

$$\text{softmax}\,(x)i = \frac{\exp(x_i)}{\sum_j \exp\,(x_j)} \tag{2.3}$$

Adapun fungsi aktivasi lain seperti *hyperbolic tan function :*

$$f(x) = tanh(x) = \frac{2}{1+e^{-a}} - 1 \tag{2.4}$$

Rectified Linear Units atau RelU*:*

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases} \tag{2.5}$$

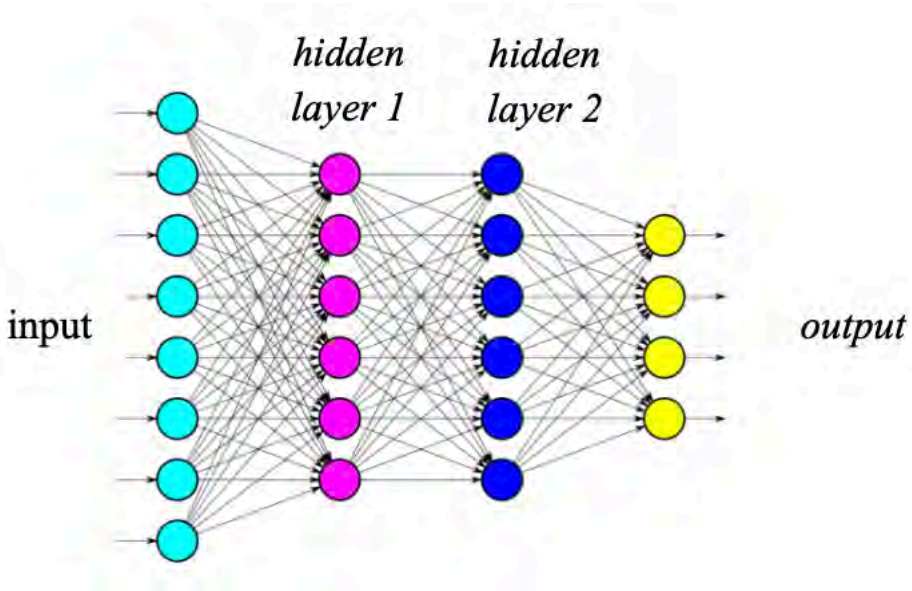
Exponential Linear Units atau ELU*:*

$$f(x) = \begin{cases} a(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases} \tag{2.6}$$

(Rayed Bin Wahed, 2016)

2.5. *Deep Neural Network*

Deep Neural Network (DNN) merupakan artificial neural network yang memiliki banyak layer. Umumnya DNN memiliki lebih dari tiga layer yaitu *input* layer, N *hidden layers* dan *output* layer. Proses pembelajaran pada DNN disebut *deep learning*. Secara visual kita dapat menggambarkan *deep neural network* seperti pada Gambar 2.6.



Gambar 2.6 Visualisasi *Deep Neural Network* (Putra, 2019)

Gambar 2.6 merupakan DNN dengan lima layer, sehingga *output* terakhir nya dapat dihitung dengan persamaan berikut :

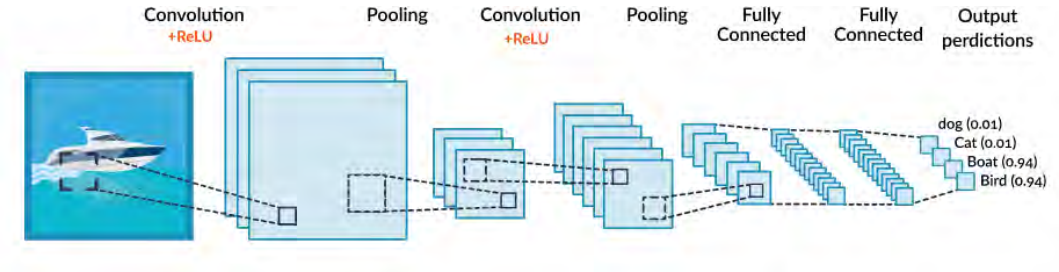
$$\left(\sum_{j=1}^{H_2} u_j \, , i^{\sigma}\left(\sum_{k=1}^{H_1} v_k \, , i^{\sigma}\left(\sum_{m=1}^{M} x_m \, w_{m,k} + \beta_k\right) + \gamma_i\right) + \lambda_i\right) \qquad (2.7)$$

dimana β, γ, λ adalah *noise* atau bias (Putra, 2019).

2.6. *Convolutional Neural Network*

Convolutional neural network (CNN) merupakan arsitektur yang mampu mengenali informasi prediktif suatu objek seperti gambar, teks, potongan suara, dan lain sebagainya. CNN merupakan pengembangan dari *multilayer perceptron* (MLP) yang didesain untuk mengolah data dalam bentuk citra. CNN termasuk dalam jenis *Deep neural network* karena kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra. Penelitian tentang CNN pertama kali dilakukan oleh Hubel dan Wiesel (1968) tentang *visual cortex* pada indera penglihatan kucing. Arsitektur CNN terdiri atas beberapa layer yaitu *convolution*

layer, fungsi *activation layer*, *pooling layer*, dan *fully connected layer*. Ilustrasi jaringan arsitektur Convolutional Neural Network dapat dilihat pada Gambar 2.7.



Gambar 2.7 Ilustrasi Arsitektur *Convolutional Neural Network* (CNN) (Choudari, 2019)

Tahap pertama dalam arsitektur CNN adalah tahap konvolusi. Kemudian dilanjutkan fungsi aktivasi menggunakan fungsi aktivasi ReLu (*Rectifier Linear Unit*), kemudian dilanjutkan dengan proses *pooling*. Proses ini diulang terus menerus sampai didapatkan peta fitur yang cukup untuk dilanjutkan ke *fully connected neural network*, sehingga dapat dihasilkan *output class*. Penjelasan secara detail dari tahap arsitektur CNN tersebut dijelaskan sebagai berikut.

2.6.1. *Convolution Layer*

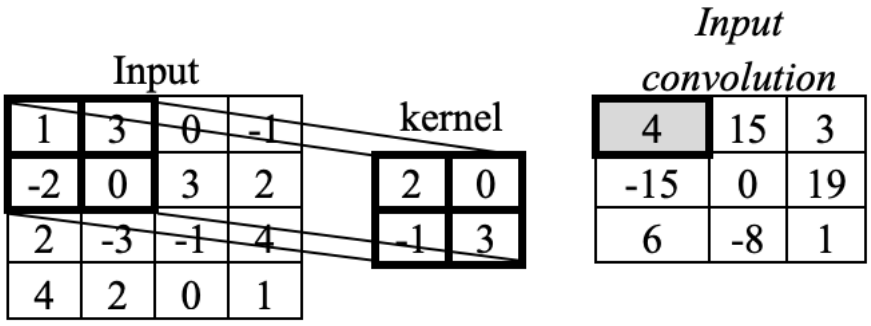
Convolution layer merupakan proses utama yang mendasari jaringan arsitektur CNN dan terdiri atas kernel. Kernel-kernel pada lapisan ini disebut filter konvolusi. Kernel berfungsi mempelajari fitur-fitur lokal pada feature map. Tahap *convolutional layer* melakukan operasi konvolusi pada *output* dari *layer* sebelumnya. Konvolusi adalah istilah matematis dimana pengaplikasian sebuah fungsi pada output fungsi lain secara berulang. Persamaan konvolusi merupakan persamaan pada dua fungsi argument bernilai riil. Operasi konvolusi $s(t)$dapat ditunjukkan pada persamaan berikut:

$$s(t) = \sum_a I(a).K(t-a) \quad (2.8)$$

Dimana $I(a)$ adalah input dan $K(a)$ adalah kernel. *Input convolution layer* merupakan gambar yang direpresentasikan menjadi sebuah matriks. Operasi konvolusi menghasilkan nilai tinggi dan rendah pada posisi tertentu pada *feature map*. Posisi tertentu dari konvolusi kernel, merupakan perkalian untuk setiap nilai pada sel kernel dan nilai piksel gambar yang tumpang tindih dengan sel kernel(Khan *et al.*, 2018). Adapun operasi perkaliannya menggunakan persamaan sebagai berikut:

$$h_{i,j} = \sum_{k=1}^{m} \sum_{l=1}^{m} W_{k,l} X_{i+k-1,j+l-1} \quad (2.9)$$

Dimana m adalah lebar dan tinggi kernel, h adalah *input convolution*, x adalah input, dan w adalah *convolutional kernel*. Sebagai contoh, ilustrasi pada proses *convolution layer* dengan nilai stride 2 dapat dilihat pada Gambar 2.8.



Gambar 2.8 Ilustrasi Proses *Convolutional Layer* (Khan *et al.*, 2018)

Adapun perhitungan nilai pada *input convolution* yang diperoleh dengan menggunakan persamaan 2.9 adalah sebagai berikut:

$h1,1 = (1.2) + (3.0) + (-2.-1) + (0.3) = 4$
$h1,2 = (3.2) + (0.0) + (0.-1) + (3.3) = 15$
$h1,3 = (0.2) + (-1.0) + (3.-1) + (2.3) = 3$

$$h2,1 = (-2.2) + (0.0) + (2.-1) + (-3.3) = -15$$

$$h2,2 = (0.2) + (3.0) + (-3.-1) + (-1.3) = 0$$

$$h2,3 = (3.2) + (2.0) + (-1.-1) + (4.3) = 19$$

$$h3,1 = (2.2) + (-3.0) + (4.-1) + (2.3) = 6$$

$$h3,2 = (-3.2) + (-1.0) + (2.-1) + (0.3) = -8$$

$$h3,3 = (-1.2) + (4.0) + (0.-1) + (1.3) = 1$$

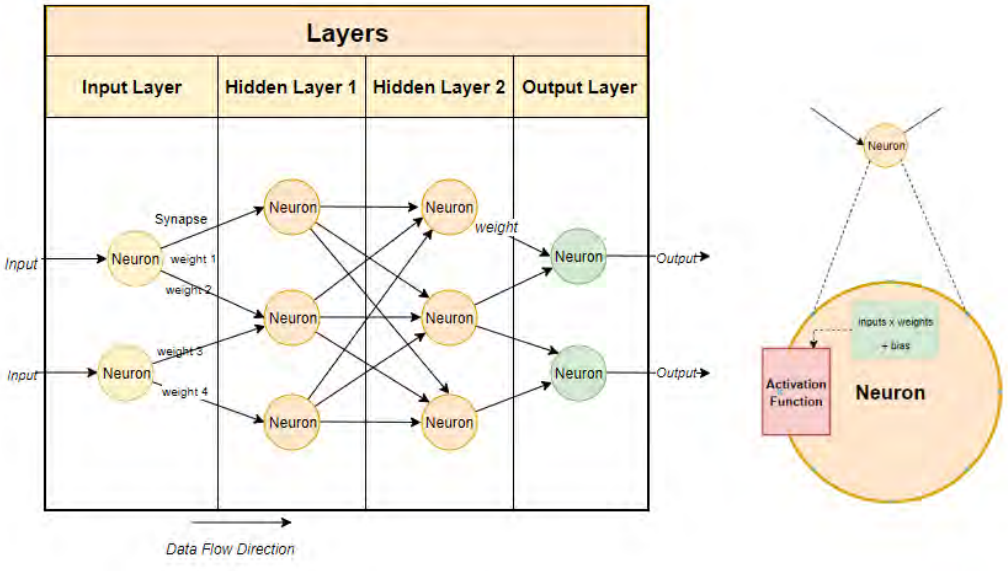
Dari perhitungan diatas, di peroleh matriks *input convolution* sebagai berikut :

$$\begin{matrix} 4 & 15 & 3 \\ -15 & 0 & 19 \\ 6 & -8 & 1 \end{matrix}$$

(Khan *et al.*, 2018).

2.6.2. *Weights dan Bias*

Ketika *input* ditransmisikan antar *neuron*, *weights* diterapkan dan diteruskan ke fungsi aktivasi bersama dengan bias. Untuk lebih jelasnya perhatikan ilustrasi weights dan bias pada Gambar 2.9.



Gambar 2.9 Ilustrasi *Weights* dan *Bias* pada *Layer* (Malik, 2019)

Weights mengontrol sinyal (atau kekuatan koneksi) antara dua neuron. Dengan kata lain, bobot menentukan seberapa besar pengaruh input terhadap output. Saat jaringan neural dilatih pada set pelatihan, jaringan akan diinisialisasi dengan satu set *weights. Weights* kemudian dioptimasi selama periode pelatihan untuk menghasilkan *weights* yang optimal.

Bias adalah nilai konstanta yang ditambahkan ke layer *input* dengan *weights. Bias* digunakan untuk mengimbangi hasil dan untuk menggeser hasil fungsi aktivasi ke nilai positif atau negatif.

Dalam proses perhitungan nya, neuron akan menghitung *weighted sum* dari *input* dengan persamaan:

$$Y = \sum(weight * input) + bias \tag{2.10}$$

Dimana *input* merupakan :

$$x_{1,}x_{2.}, \dots, x_n \tag{2.11}$$

Dan *weight* adalah :

$$w_{1,}w_{2.}, \dots, w_n \tag{2.12}$$

Maka *weighted sum* dapat dihitung dengan persamaan :

$$x_1w_1 + x_2w_2 + \cdots + x_nw_n + bias \tag{2.13}$$

2.6.3. Fungsi Aktivasi

Fungsi aktivasi merupakan operasi matematik yang dikenakan pada sinyal output y. Fungsi aktivasi berfungsi menentukan apakah suatu *neuron* aktif atau tidak berdasarkan *weighter sum* dari suatu *input*. Beberapa jenis fungsi aktivasi yang sering digunakan pada *deep learning* adalah *sigmoid, Tanh, algebraic*

sigmoid, ReLU, noisy ReLU, Leakly ReLU/PReLU, Randomized Leakly ReLU, dan *Eksponential Linear Unit*. Penelitian ini menggunakan dua fungsi aktivasi yaitu fungsi aktivasi ReLU dan fungsi aktivasi *softmax.* Penjelasan dari kedua fungsi tersebut adalah sebagai berikut.

a. ReLU

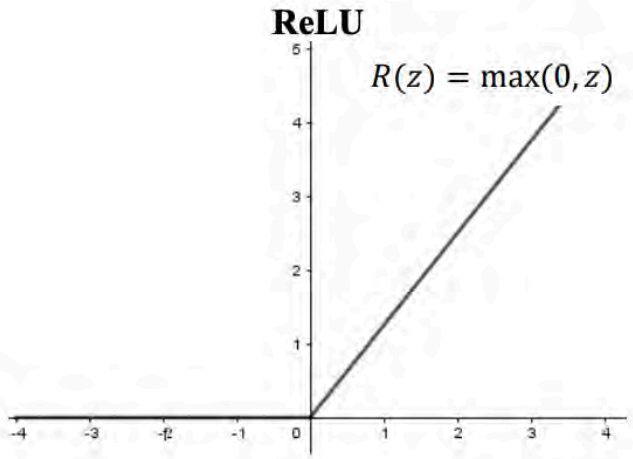
Fungsi aktivasi *Rectifier Linear Unit*(ReLU) merupakan fungsi aktivasi sederhana yang memiliki kepentingan praktis khusus karena perhitungannya yang cepat. Kelebihan fungsi aktivasi ReLU dibandingkan dengan fungsi aktivasi lain adalah sebagai berikut.

1. Fungsi aktivasi ReLU merupakan fungsi aktivasi default ketika mengembangkan *multilayer perceptron* dan *convolutional neural network*.
2. Fungsi aktivasi ReLU mengatasi masalah *gradient descent* yang hilang, yang memungkinkan model belajar lebih cepat dan berkinerja lebih baik.
3. Menemukan cara melatih jaringan dengan lebih cepat, sehingga mengurangi kemungkinan terjadinya *overfitting*.

Fungsi aktivasi ReLU memetakan input ke 0 jika negatif dan mempertahankan nilainya jika positif. Representasi fungsi ReLU adalah sebagai berikut

$$f_{ReLU\ (x)}=\max\ (0,x) \tag{2.14}$$

Adapun grafik nya dapat dilihat pada Gambar 2.10.



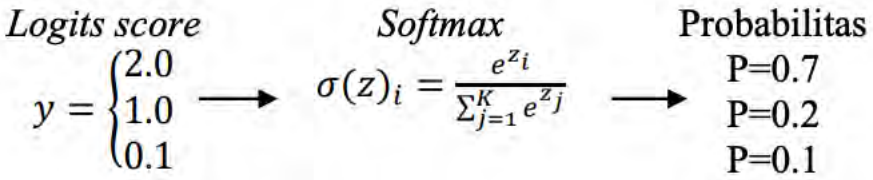
Gambar 2.10 Grafik Fungsi Aktivasi ReLU (Khan *et al.*, 2018)

b. Fungsi Aktivasi *Softmax*

Fungsi aktivasi *softmax* merupakan fungsi input vektor dari bilangan real *K*, yang kemudian dinormalkan menjadi distribusi probabilitas yang terdiri atas probabilitas *K* yang proposional ke eksponensial input. Komponen vektor pada *softmax* memiliki interval (0,1). Fungsi softmax merupakan lapisan yang menghubungkan antara *fully connected layer* dengan *dense connection*. *Softmax* berfungsi untuk menghitung probabilitas pada setiap kelas target yang memungkinkan dan akan membantu menentukan kelas target pada input yang diberikan. Nilai *softmax* berada pada rentang probabilitas output dari 0 hingga 1 dan jumlah semua probabilitas sama dengan satu. Definisi fungsi *softmax* $\sigma = \mathbb{R}K \rightarrow \mathbb{R}K$ dituliskan sebagai persamaan berikut:

$$\sigma(\mathrm{z})_i = \frac{e^{z_i}}{\sum_{\mathrm{j}=1}^{\mathrm{K}} e^{z_j}} \tag{2.15}$$

dimana $i = 1,\ldots,K$ dan $z = (zi , \ldots , zK) \in \mathbb{R}K$. Adapun contoh ilustrasi fungsi aktivasi *softmax* dapat dilihat pada Gambar 2.11 sebagai berikut.

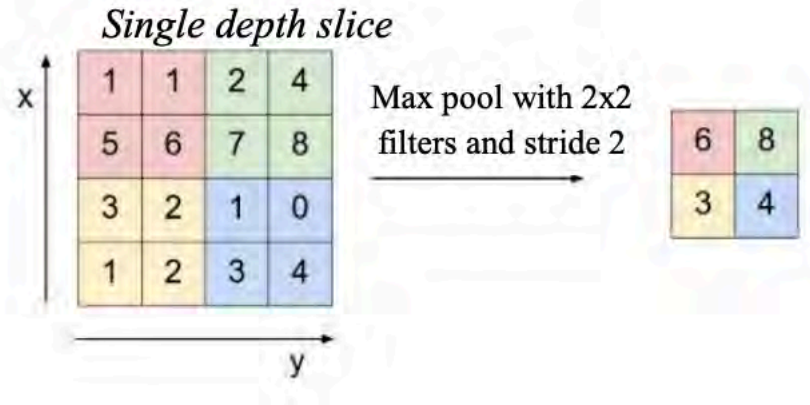


Gambar 2.11 ilustrasi fungsi aktivasi *softmax*

Logits score menunjukkan lapisan *neuron* terakhir sebagai *output* mentah pada lapisan terakhir neural network sebelum proses aktivasi berlangsung. Setelah *output* diproses dengan *softmax* akan menghasilkan nilai probabilitas dengan jumlah 1(Khan *et al.*, 2018).

2.6.4. *Pooling Layer*

Pooling layer terletak setelah *convolution layer. Pooling layer* terdiri atas sebuah *filter* dengan ukuran dan *stride* tertentu yang secara bergantian bergeser pada seluruh *area feature map*. Jenis *pooling layer* yang biasa digunakan yaitu *average pooling* dan *max pooling*. Nilai yang diambil pada *average pooling* adalah nilai rata-rata dan pada *max pooling* yang diambil adalah nilai maksimal. Lapisan *pooling* yang dimasukkan diantara lapisan konvolusi pada model CNN dapat mengurangi ukuran *volume output* pada *feature map*, sehingga mengurangi jumlah parameter dan perhitungan di jaringan. Adapun proses dari *max pooling* dapat dilihat pada Gambar 2.12.



Gambar 2.12 Operasi *max pooling* (Khan *et al.*, 2018)

Output dari proses *pooling* adalah sebuah matriks dengan dimensi yang lebih kecil dibandingkan dengan citra awal. *Pooling layer* beroperasi pada blok *input feature map* dan menggabungkan aktivasi fitur. Operasi kombinasi antara *pooling layer* dan fungsi aktivasi didefinisikan sebagai fungsi penyatuan seperti fungsi rata-rata atau maksimal.

2.6.5. *Fully Connected Layer*

Fully connected layer merupakan sebuah lapisan dimana semua *neuron* aktivasi dari lapisan sebelumnya terhubung dengan neuron lapisan selanjutnya. Perbedaan *fully connected layer* dengan konvolusi biasa adalah *neuron* pada lapisan konvolusi terhubung hanya ke daerah tertentu sedangkan *fully connected* memiliki *neuron* yang semuanya terhubung (Khan *et al.*, 2018).

2.7. *Inception V3*

Inception V3 berfokus pada penggunaan daya komputasi yang lebih sedikit dengan memodifikasi arsitektur Inception sebelumnya. Ide ini diusulkan dalam makalah *Rethinking the Inception Architecture for Computer Vision*, yang diterbitkan pada tahun 2015. Ide ini disusun bersama oleh *Christian Szegedy,*

Vincent Vanhoucke, Sergey Ioffe, dan *Jonathon Shlens.* Dibandingkan dengan *VGGNet*, *Inception Networks* (*GoogLeNet / Inception v1*) terbukti lebih efisien secara komputasi, baik dari jumlah parameter yang dihasilkan oleh jaringan maupun *cost* memori dan sumber daya lainnya. Jika ada perubahan yang ingin dilakukan pada *Inception Network*, harus dilakukan secara hati-hati untuk memastikan bahwa keunggulan komputasi tidak hilang. Dengan demikian, adaptasi jaringan *Inception* untuk berbagai kasus berbeda ternyata menjadi masalah karena ketidakpastian efisiensi jaringan baru. Dalam model *Inception v3*, beberapa teknik untuk mengoptimalkan jaringan telah disarankan untuk mengurangi kendala agar adaptasi model menjadi lebih mudah. Teknik tersebut meliputi *factorized convolutions, regularization, dimension reduction,* dan *parallelized computations* (Kurama, 2020).

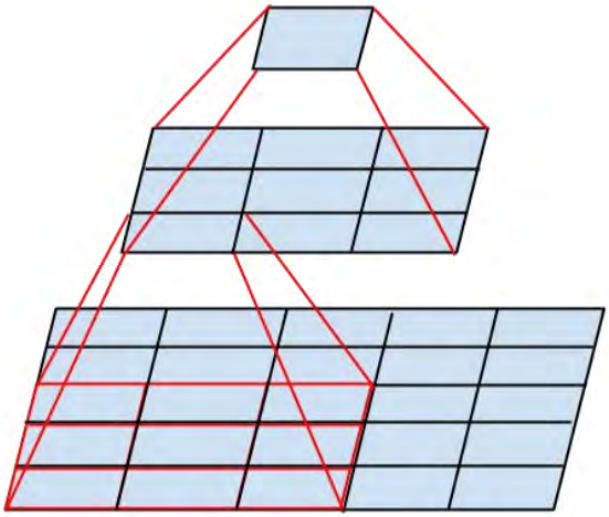
Arsitektur jaringan *Inception v3* dibangun secara progresif, langkah demi langkah, seperti yang dijelaskan di bawah ini:

1. Factorizing Convolutions

Tujuan dari faktorisasi konvolusi adalah untuk mengurangi jumlah koneksi atau parameter tanpa mengurangi efisiensi jaringan.

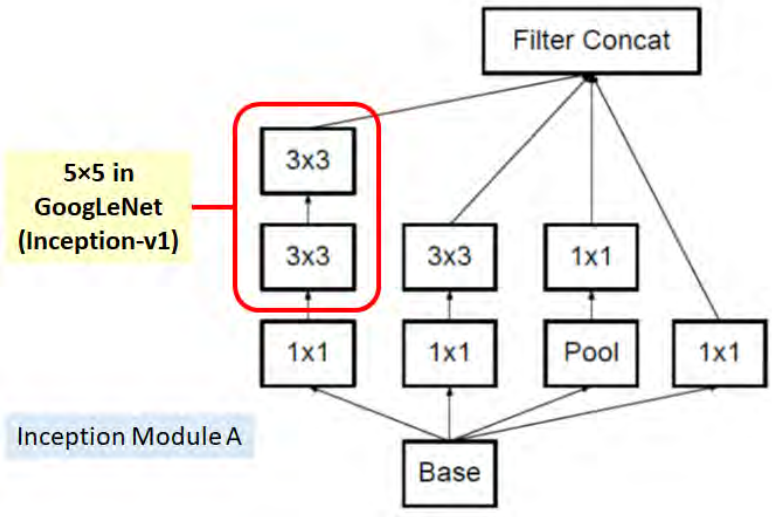
1.1. Factorization Into Smaller Convolutions

Dua konvolusi 3x3 digantikan dengan satu konvolusi 5x5 seperti pada Gambar 2.13



Gambar 2.13 dua konvolusi 3x3 digantikan oleh konvolusi 5x5 (Szegedy *et al.*, 2016)

Dengan menggunakan 1 layer filter 5x5, maka jumlah parameter nya adalah = 5x5 = 25, jika menggunakan 2 layer filter 3x3 maka jumlah parameter nya adalah = 3x3+3x3=18, sehingga jumlah parameter nya berkurang sebanyak 28%. Teknik yang sama juga telah diterapkan oleh arsitektur *VGGNet*. Pada InceptionV3, teknik diterapkan pada *Inception Module A*, Ilustrasi dapat dilihat pada Gambar 2.14.

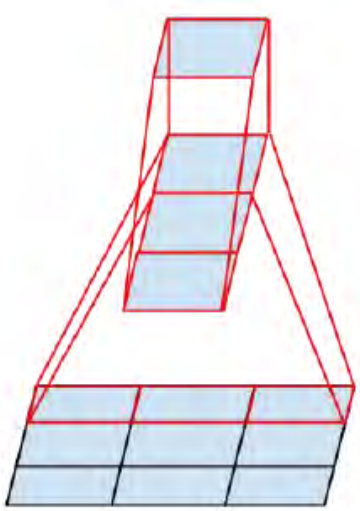


Gambar 2.14 *Inception Module A* dengan *factorization*

(Szegedy *et al.*, 2016)

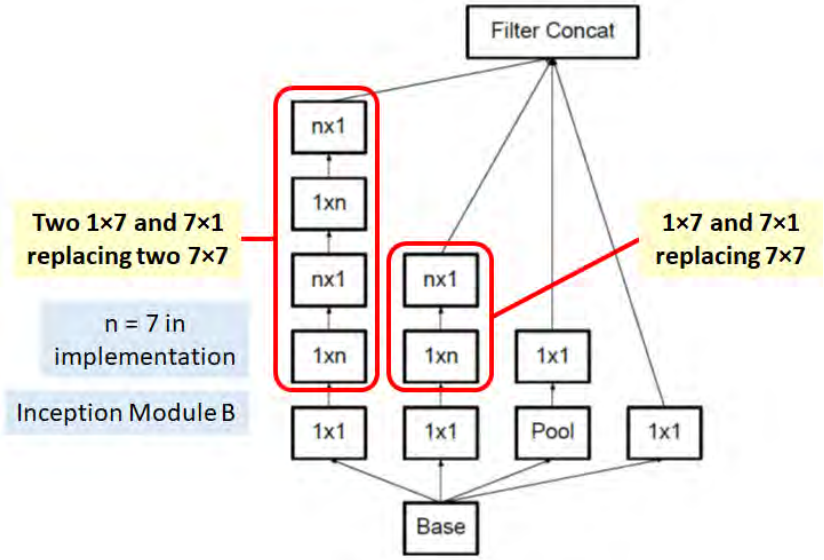
1.2. Factorization Into Asymmetric Convolutions

Satu konvolusi 3x1 yang diikuti oleh konvolusi 1x3 digantikan dengan konvolusi 3x3 seperti pada Gambar 2.15.



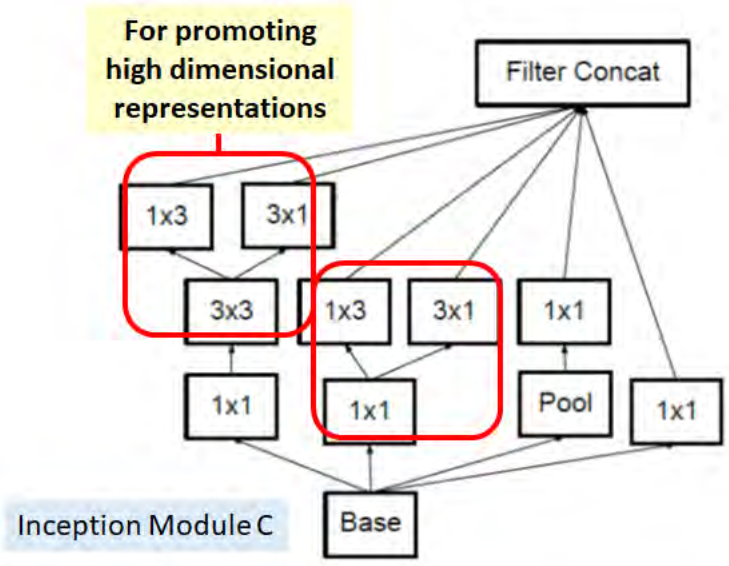
Gambar 2.15 satu konvolusi 3x1 yang diikuti konvolusi 1x3 diganti oleh konvolusi 3x3 (Szegedy *et al.*, 2016)

Dengan menggunakan filter 3x3, maka jumlah parameter nya adalah = 3x3 = 9, jika menggunakan filter 3x1 dan 1x3, maka jumlah parameternya adalah = 3x1+1x3=6, sehingga jumlah parameter nya berkurang sebanyak 33%. Teknik ini diterapkan pada *Inception Module B,* ilustrasi dapat dilihat pada Gambar 2.16



Gambar 2.16 Inception Module B dengan *asymmetric factorization* (Szegedy *et al.*, 2016)

Selanjutnya, *Inception module C* juga digunakan, untuk mempromosikan representasi dimensi tinggi seperti yang ditunjukan pada Gambar 2.17

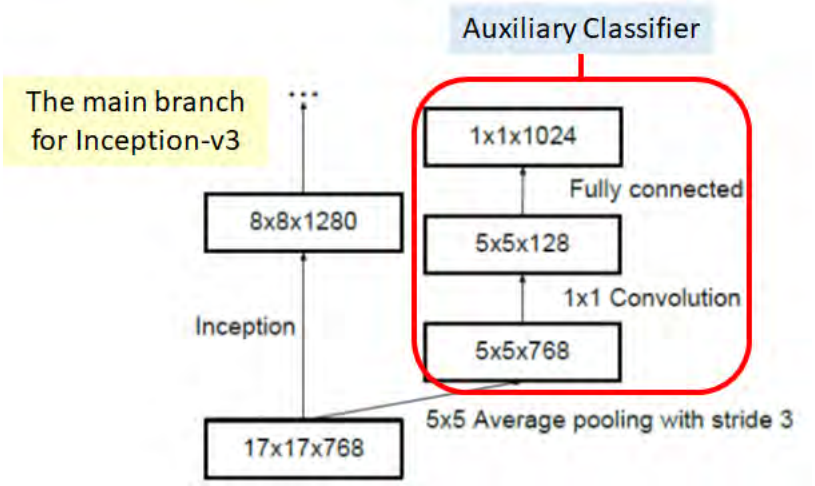


Gambar 2.17 Inception Module C dengan *asymmetric factorization* (Szegedy *et al.*, 2016)

Dengan diterapkannya tiga module inception tadi, jumlah parameter didalam jaringan akan semakin sedikit, sehingga akan memperkecil kemungkinan terjadinya *overfitting* (Tsang, 2018).

2. ***Auxiliary Classifier***

Auxiliary classifier sudah diterapkan sejak arsitektur *GoogLeNet/ InceptionV1*. Ada sedikit modifikasi yang dilakukan pada *InceptionV3,* hanya satu *auxiliary classifier* yang digunakan sebelum *layer* 17x17 terakhir, dibandingkan dengan arsitektur sebelumnya yang menggunakan dua *auxiliary classifier.* Dalam InceptionV3 *auxiliary classifier* digunakan sebagai *regularizer,* ilustrasinya dapat dilihat pada Gambar 2.18.

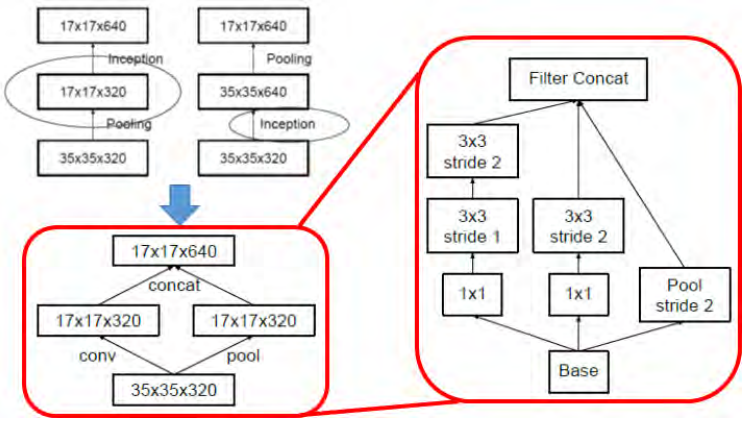


Gambar 2.18 *Auxiliary Classifier* sebagai *regularization* (Szegedy *et al.*, 2016)

3. ***Efficient Grid Size Reduction***

Umumnya seperti yang diterapkan pada arsitektur *AlexNet* dan *VGGNet*, pengurangan *feature map* dilakukan oleh *max pooling.* Tetapi *max pooling* ini memiliki kekurangan yaitu terlalu berlebihan dalam mengurangi *feature map* yang diikuti oleh *layer* konvolusi, atau terlalu boros ketika konvolusi

layer yang diikuti oleh *max pooling.* Pada InceptionV3 *efficient grid size reduction* diterapkan seperti ilustrasi pada Gambar 2.19.

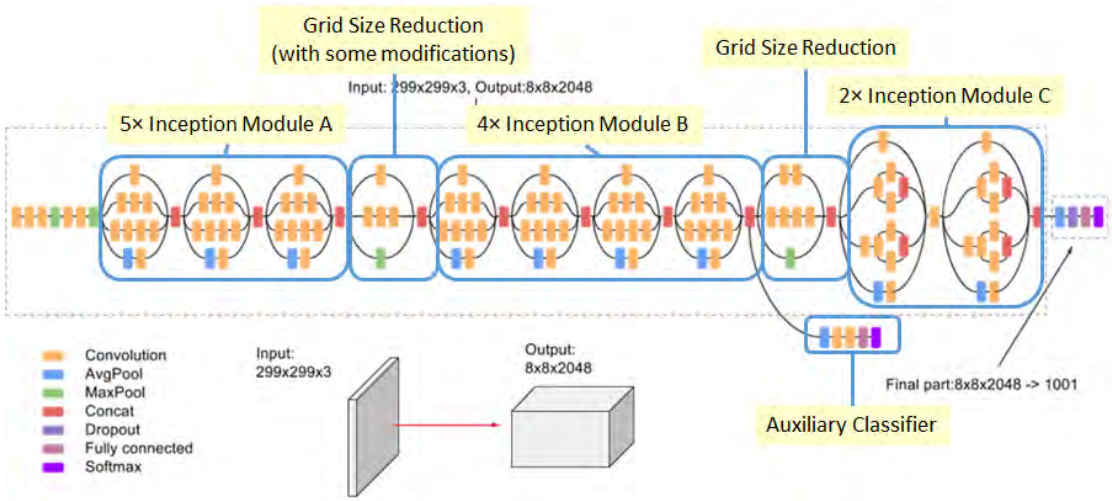


Gambar 2.19 *Conventional downsizing* (atas kiri), *Efficient Grid Size Reduction* (bawah kiri), Detail dari *Efficient Grid Size Reduction* dalam Arsitektur (kanan) (Szegedy *et al.*, 2016)

Dengan *efficient grid size reduction,* 320 *feature maps* dihasilkan oleh konvolusi dengan *stride 2. Feature map* tadi diperoleh dari *max pooling,* kemudian dua set *feature maps* digabungkan menjadi 640 *feature maps* yang kemudian akan diteruskan ke *module inception* selanjutnya (Tsang, 2018).

4. Arsitektur *InceptionV3*

Semua proses diatas diterapkan dalam arsitektur *InceptionV3*, Ilustrasi arsitektur nya dapat dilihat pada Gambar 2.20.



Gambar 2.20 Arsitektur *InceptionV3* (Tsang, 2018)

Parameter dari arsitektur *InceptionV3* dapat dilihat pada Tabel 2.2.

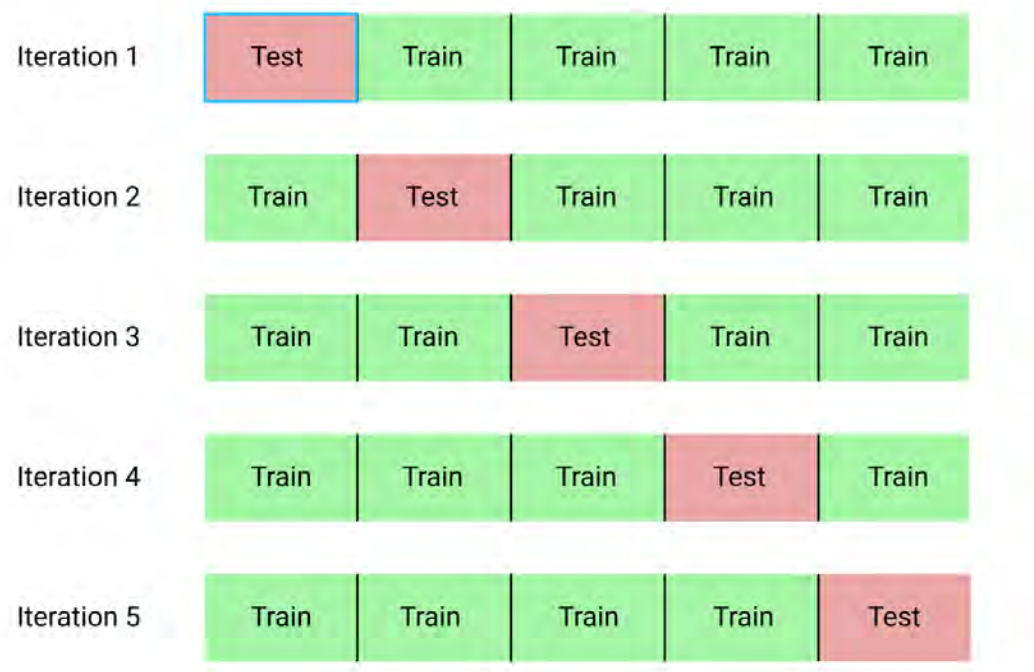
Tabel 2.2. Parameter Arsitektur InceptionV3

Type	**Filter Size / Stride**	**Input Size**
Conv	3 x 3 / 2	299 x 299x 3
Conv	3 x 3 / 1	149 x 149 x 32
Conv padded	3 x 3 / 1	147 x 147 x 32
Pool	3 x 3 / 2	147 x 147 x 64
Conv	3 x 3 / 1	73 x 73 x 64
Conv	3 x 3 / 2	71 x 71 x 80
Conv	3 x 3 / 1	35 x 35 x 192
3 x Inception A	Seperti Gambar 2.13	35 x 35 x 288
5 x Inception B	Seperti Gambar 2.15	17 x 17 x 768
2 x Inception C	Seperti Gambar 2.16	8 x 8 x 1280
Pool	8 x 8	8 x 8 x 2048

Type	**Filter Size / Stride**	**Input Size**
Linear	Logits	1 x 1 x 2048
Softmax	Classifier	1 x 1 x 1000

2.8. *K-Fold Cross Validation*

K-Fold Cross Validation adalah metode yang melakukan pembagian data secara acak menjadi kelompok k atau lipatan dengan ukuran yang kira-kira sama. Lipatan pertama disimpan untuk *validation* dan *model ditraining* pada lipatan k-1. Proses ini diulang sebanyak k kali dan setiap kali lipatan yang berbeda atau kelompok titik data yang berbeda digunakan untuk validasi. Ilustrasi dapat dilihat pada Gambar 2.21.

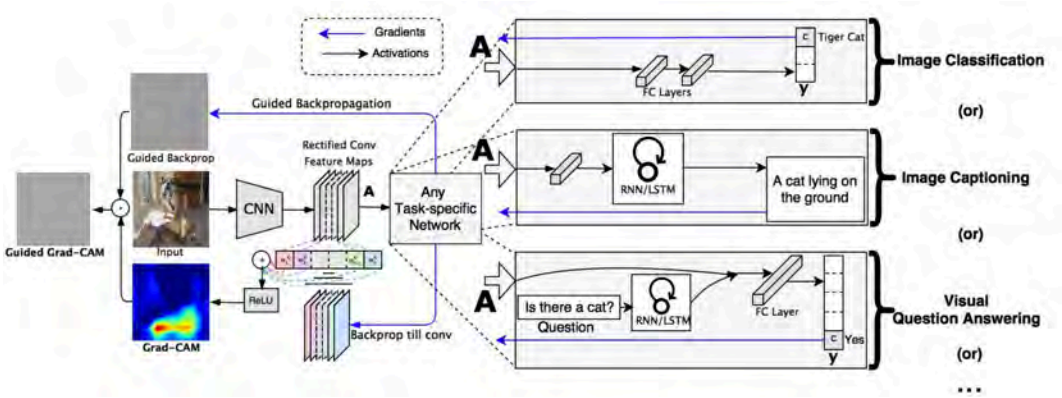


Gambar 2.21 Contoh 5 *fold cross validation* (Shaikh, 2018).

Seiring berjalannya proses hingga banyak k, kita mendapatkan k kali *mean square error* (MSE). MSE_1, MSE_2, *hingga* MSE_K, sehingga MSE nya didapatkan melalui rata-rata dari MSE terhadap lipatan k. *mean square error* dari *k fold cross validation* ini dapat dihitung dengan persamaan berikut (Khandelwal, 2018).

$$cv_k = \frac{1}{k}\sum_{i=1}^{k} MSE_i \tag{2.16}$$

2.9. *Grad-CAM*



Gambar 2.22 Visualisasi *Grad-CAM* (Selvaraju *et al.*, 2019).

Grad-CAM menggunakan informasi gradien yang berada pada lapisan konvolusional terakhir CNN untuk menetapkan nilai penting ke seluruh *neuron* untuk setiap keputusan tertentu. Meskipun teknik ini cukup umum karena dapat digunakan untuk menjelaskan *layer* aktivasi manapun dari dalam *network*, *Grad-CAM*, berfokus pada *layer output* saja. Seperti yang ditunjukkan pada Gambar 2.21, untuk mendapatkan kelas diskriminatif peta lokalisasi *Grad-CAM* $L^c_{\text{Grand}-\text{CAM}} \in \mathbb{R}^{u\,x\,v}$ dari lebar u dan tinggi v untuk setiap kelas c, pertama kita menghitung gradien skor untuk kelas c, y^c (sebelum *softmax*), sehubungan dengan aktivasi

feature map A^k dari *layer* konvolusi seperti $\frac{\partial y^c}{\partial A^k}$. Gradien yang mengalir kembali ini adalah *global average pooled* pada dimensi lebar dan tinggi (masing-masing di indeks oleh i dan j) untuk mendapatkan bobot *neuron* α_k^c yang dapat dituliskan pada persamaan berikut :

$$\alpha_k^c = \overbrace{\frac{1}{Z}\sum_i\sum_j}^{global\ average\ pooling} \underbrace{\frac{\partial y^c}{\partial A_{ij}^k}}_{gradients\ via\ backprop} \tag{2.17}$$

Selama proses komputasi α_k^c, ketika gradien *backpropagating* sehubungan dengan aktivasi, jumlah komputasi yang tepat untuk produk maktriks berturut-turut dari matriks bobot dan gradien sehubungan dengan fungsi aktivasi hingga akhir konvolusi dimana gradien disebarkan. Oleh karena itu, bobot α_k^c ini merepresentasikan linierisasi parsial *deep network* di hilir dari A, dan menangkap 'pentingnya' *feature map* k untuk kelas target c. Grad-CAM melakukan kombinasi *forward activation map*, dan dilanjutkan dengan ReLU untuk mendapatkan :

$$L^c_{\text{Grad-CAM}} = ReLU \underbrace{\left(\sum_k \alpha_k^c A^k\right)}_{\text{linear combination}} \tag{2.18}$$

ReLU diterapkan ke kombinasi linier *maps* karena Grad-CAM hanya tertarik pada fitur yang memiliki pengaruh positif pada *class of interest*, misalnya piksel yang intensitasnya harus ditingkatkan untuk meningkatkan y^c. Piksel negatif cenderung termasuk kategori lain dalam gambar. Seperti yang diduga, tanpa ReLU, *localization maps* terkadang menyorot lebih dari sekadar kelas yang diinginkan dan berperforma lebih buruk dalam *localization*. Secara umum, y^c tidak perlu berupa skor kelas yang dihasilkan oleh klasifikasi gambar CNN. (Selvaraju *et al.*, 2019).

2.10. *Data Augmentation*

Data Augmentation merupakan teknik yang digunakan untuk meningkatkan jumlah data dengan menambahkan salinan yang sedikit dimodifikasi dari data yang sudah ada atau data sintetis yang baru dibuat dari data yang ada. *Image Data Augmenter* mengkonfigurasi serangkaian opsi *preprocessing* untuk augmentasi gambar seperti pengubahan ukuran, rotasi dan refleksi. Berikut penjelasan dari setiap opsi nya:

1. ***FillValue***

   Digunakan untuk menentukan titik di luar batas saat pengambilan sampel ulang, ditentukan sebagai skalar numerik atau vektor numerik.

2. ***RandXReflection***

   Refleksi acak di arah kiri-kanan, ditentukan sebagai skalar logis.

3. ***RandYReflection***

   Refleksi acak dalam arah atas-bawah, ditetapkan sebagai skalar logis.

4. ***RandRotation***

   Rentang rotasi dalam derajat yang diterapkan ke gambar masukan.

5. ***RandScale***

   Rentang skala seragam (isotropik) yang diterapkan pada gambar masukan.

6. ***RandXScale***

   Rentang skala horizontal diterapkan ke gambar *input*.

7. ***RandYScale***

   Rentang skala vertikal diterapkan ke gambar masukan.

8. RandXShear

Rentang pemotongan horizontal yang diterapkan pada citra masukan.

9. ***RandYShear***

Rentang pemotongan vertikal yang diterapkan pada citra masukan.

10. RandXTranslation

Rentang pergeseran horizontal yang diterapkan ke gambar masukan.

11. RandYTranslation

Rentang pergeseran vertikal yang diterapkan ke gambar masukan

(Beale, Hagan and Demuth, 2020).