IMPLEMENTASI *DEEP LEARNING* DALAM MENDETEKSI PENYAKIT MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DAN *TENSORFLOW*

(Studi Kasus: Penyakit Saluran Pencernaan (*Gastrointestinal*) *Esophagitis*, *Polyps*, *Normal-Pylorus*, *Dyed-Lifted-Polyps* dan *Dyed-Resection-Margins*)

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana

Program Studi Statistika

Redho Islami

16611071

JURUSAN STATISTIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS ISLAM INDONESIA

YOGYAKARTA

2020

HALAMAN PERSETUJUAN PEMBIMBING

TUGAS AKHIR

Judul	**:**	Implementasi *Deep Learnig* dalam Mendeteksi Penyakit Menggunakan *Convolutional Neural Network* dan *Tensorflow*
Nama Mahasiswa	:	Redho Islami
Nomor Mahasiswa	:	16611071

TUGAS AKHIR INI TELAH DIPERIKSA DAN DISETUJUI UNTUK DIUJIKAN

Yogyakarta, 26 Oktober 2020

Pembimbing

(Muhammad Muhajir, S.Si., M.Sc.)

HALAMAN PENGESAHAN

TUGAS AKHIR

Implementasi *Deep Learnig* dalam Mendeteksi Penyakit Menggunakan *Convolutional Neural Network* dan *Tensorflow*

Nama Mahasiswa : Redho Islami

Nomor Mahasiswa : 16611071

TUGAS AKHIR INI TELAH DIUJIKAN

PADA TANGGAL 9 NOVEMBER 2020

Nama Penguji	**Tanda tangan**
1. Dr. Techn. Rohmatul Fajriyah, M.Si	…….....
2. Rahmadi Yotenka, M.Sc	…….....
3. Muhammad Muhajir, M.Sc	…….....

Mengetahui,
Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

(Prof. Riyanto, S.Pd., M.Si., Ph.D.)

DAFTAR ISI

KATA PENGANTAR

Assalamu'alaikum Warahmatullaahi Wabarakaatuh

Alhamdulillaahirabbil'aalamiin, Puji Syukur kehadirat Allah SWT yang telah melimpahkan rahmat, hidayah, dan inayahnya, sehingga penulis dapat menyelesaikan tugas akhir yang berjudul **"**Implementasi *Deep Learnig* dalam Mendeteksi Penyakit Menggunakan *Convolutional Neural Network* dan *Tensorflow***"** dengan baik dan lancar sebagai salah satu persyaratan yang harus dipenuhi dalam menyelesaikan jenjang strata satu di Program Studi Statistika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Islam Indonesia. Shalawat serta salam semoga selalu tercurah kepada junjungan nabi besar Muhammad SAW semoga mendapatkan safaatnya diakhir hayat nanti.

Penyelesaian tugas akhir ini tidak terlepas dari dukungan, bantuan, arahan, dan bimbingan dari berbagai pihak. Untuk itu pada kesempatan ini penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Kedua Orang tua beserta keluarga besar penulis yaitu Hj. Amna (Ibu) dan H. Mialis, S.Pd (Ayah) yang telah banyak berkorban serta berjuang baik itu dari materi, tenaga, pikiran dan memberi motivasi serta doanya untuk penulis sehingga penulis bisa menyelesaikan Tugas Akhir ini dengan baik.
2. Prof. Riyanto, S.Pd., M.Si., Ph.D. selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Islam Indonesia.
3. Edy Widodo, Dr., S.Si., M.Si. Kepala Program Studi Statistika Universitas Islam Indonesia.
4. Muhammad Muhajir, S.Si., M.Sc. selaku dosen Pembimbing Tugas Akhir yang selalu membimbing penulis.
5. Seluruh dosen dan staff Program Studi Statistika Universitas Islam Indonesia yang telah memberikan ilmu pengetahuan terkait ilmu secara terapan maupun secara aplikasi.

6. Keluarga besar IKS (Ikatan Keluarga Statistika), sebagai Organisasi yang menjadi tempat bagi keluarga mahasiswa statistika FMIPA UII.
7. Teman-teman satu bimbingan yang sudah sama-sama berjuang, saling mengingatkan dan memberi motivasi serta dorongan untuk menyelesaikan Tugas Akhir ini.
8. Teman-teman Statistika UII Angkatan 2016 (ARTCOS) teman seperjuangan untuk bersama-sama mendapat gelar Sarjana Statistika.
9. Serta seluruh pihak lainnya yang tidak dapat penulis sebutkan satu per satu, yang telah banyak membantu menyelesaikan Tugas Akhir ini.

Demikianlah yang dapat disampaikan, semoga Allah SWT senantiasa melimpahkan rahmat dan ridho-Nya kepada semua pihak yang telah membantu penulis. Semoga Tugas Akhir ini dapat memberikan manfaat bagi penulis khususnya dan umumnya bagi semua pihak yang membutuhkan. Akhir kata, semoga Allah SWT senantiasa melimpahkan rahmat serta hidayah-Nya kepada kita semua, Amin amin ya robbal 'alamiin.

Wassalamu'alaikum Warahmatullaahi Wabarakaatuh

Yogyakarta, 23 Oktober 2020

Redho Islami

DAFTAR ISTILAH

Activation Funcion	:	Fungsi yang beroperasi pada jaringan syaraf tiruan yang memberikan *output* pada setiap neuron
Class/Label	:	Variabel atau atribut yang digunakan dalam penelitian
Learning Rate	:	Parameter dari *Gradient Descent Loss*
Loss Function	:	Nilai kerugian yang diperoleh dari proses pelatihan
Batch Size	:	Jumlah sampel data yang disebarkan ke *Neural Network* atau ukuran dari satuan kecil *Epoch* yang dimasukkan (feeding) kedalam komputer
Klasifikasi	:	Penyusunan bersistem dalam kelompok atau golongan menurut kaidah atau standar yang ditetapkan
Stride	:	Parameter yang menentukan berapa jumlah pergeseran *filter/kernel*
Dropout	:	Teknik regulasi jaringan saraf dimana beberapa neuron akan dipilih secara acak dan tidak dipakai selama proses pelatihan
Step	:	Sejumlah langkah yang didefinisikan pada konfigurasi pipeline untuk proses pelatihan yang menentukan tingkat keberhasilan pelatihan Neural Network
Filter/Kernel	:	Matriks untuk menghitung dan mendeteki suatu pola atau ciri yang digunakan untuk perhitungan convolution
Checkpoint	:	Berkas yang dihasilkan dari proses pelatihan yang disimpan dalam format chkp
Prediction Confidence	:	Angka keberhasilan atau skor suatu pendeteksian berupa persentase 0-100%
CNN	:	*Convolutional Neural Network*
OpenCV	:	*Open Computer Vision* Library pada python yang digunakan untuk memproses suatu gambar secara real-time.

ANN	:	*Artificial Neural Network*
Endoskopi	:	Endoskopi adalah pemeriksaan secara visual dan langsung pada lubang atau rongga pada tubuh tertentu untuk melihat kelainan pada tubuh
Validasi	:	Suatu Tindakan pembuktian
TFRecord	:	Format dataset yang dibaca oleh tensorflow
Tensrflow	:	*Framework* komputasi untuk membuat model *mechine learning*

DAFTAR LAMPIRAN

PERNYATAAN

Dengan ini saya menyatakan bahwa dalam tugas akhir ini tidak terdapat karya yang sebelumnya pernah diajukan untuk tugas akhir. Tugas akhir ini diajukan untuk memperoleh gelar sarjana di suatu perguruan tinggi dan sependek pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Yogyakarta, 26 Oktober 2020

Redho Islami

IMPLEMENTASI *DEEP LEARNIG* DALAM MENDETEKSI PENYAKIT MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK* DAN *TENSORFLOW*

(Studi Kasus: Penyakit Saluran Pencernaan (*Gastrointestinal*) *Esophagitis, Polyps, Normal-Pylorus, Dyed-Lifted-Polyps* dan *Dyed-Resection-Margins*)

Redho Islami
Program Studi Statistika Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Islam Indonesia

INTISARI

Kesehatan fisik merupakan hal penting untuk diperhatikan, namun terkadang banyak orang yang abai untuk memperhatikannya. Kerapkali keluhan yang dirasakan seseorang dan gejala penyakit tertentu sulit dideteksi sejak dini oleh yang bersangkutan. Gangguan pada saluran pencernaan (*gastrointestinal*) ialah suatu kelainan atau penyakit pada jalan makanan/pencernaan. Penyakit *gastrointestinal* yang termasuk yaitu kelainan penyakit kerongkongan (*eshopagus*), lambung (*gaster*), usus halus (*intestinum*), usus besar (*colon*), hati (*liver*), saluran empedu (*traktus biliaris*) dan pankreas. Penyakit saluran pencernaan merupakan penyakit yang berbahaya dan menyebabkan kematian, sehingga perlu adanya media bantu berupa sistem mudah memberikan solusi yang tepat untuk menangani permasalahan tersebut. Pemanfaatan *computer vision* sangat berguna bagi bidang kesehatan dengan menggunakan *convolutional neural network* dan *tensorflow* dalam mendeteksi sebuah objek penyakit. Penelitian ini bermaksud untuk memberikan sebuah inovasi kepada tenaga medis yag ahli dibidangnya dalam menganalisa sebuah penyakit *gastrointestinal* dengan menggunakan metode *deep learning*. Hasil *testing* model yang diperoleh dari uji validasi dengan kategori *polyps* nilai akurasi sebesar 83% - 90%, *normal-pylorus* nilai akurasi sebesar 86% - 92%, *dyed-lifted-polyps* nilai akurasi sebesar 83% - 97%, *esophagitis* nilai akurasi sebesar 82% - 92%, dan *dyed-resection-margins* nilai akurasi sebesar 82% - 93%.

Kata Kunci: *Convolutional Neural Network, Deep Learing, Tensorflow, Gastroimtestinal.*

IMPLEMENTATION OF DEEP LEARNIG IN DETECTING DIESEASE USING CONVOLUTIONAL NEURAL NETWORK DAN TENSORFLOW

(Case Study: Digestive Tract Diseases (Gastrointestinal) Esophagitis, Polyps, Normal-Pylorus, Dyed-Lifted-Polyps and Dyed-Resection-Margins)

Redho Islami
Department of Statistics, Faculty of Mathematics and Natural Science
Universitas Islam Indonesia

ABSTRACT

Symptoms of certain diseases are difficult to detect early on by the person concerned. gastrointestinal disorders are abnormalities or diseases of the food/digestive tract. gastrointestinal diseases include esophageal disease disorders (esophagus), gaster, small intestine (intestine), colon, liver, bile duct (biliary tract), and pancreas. gastrointestinal disease is a dangerous disease and causes death, so the need for an auxiliary media in the form of an easy system provides the right solution to deal with the problem. The utilization of computer vision is very useful for the field of health, especially medical personnel who are experts in the field of using a convolutional neural network and tensorFlow in detecting an object of disease. This research aims to provide innovation to medical personnel who are experts in the field in analyzing a gastrointestinal disease using deep learning methods. Testing model results obtained from validation tests with polyps category accuracy value of 83% - 90%, normal-pylorus accuracy value of 86% - 92%, dyed-lifted-polyps accuracy value of 83% - 97%, esophagitis accuracy value of 82% - 92%, and dyed-resection-margins accuracy value of 82% - 93%.

***Keywords**:* Convolutional Neural Network, Deep Learing, Tensorflow, gastroimtestinal.

BAB I
PENDAHULUAN

1.1. Latar Belakang

Kesehatan fisik merupakan hal penting untuk diperhatikan, namun terkadang banyak orang yang abai untuk memperhatikannya. Kerapkali keluhan yang dirasakan seseorang dan gejala penyakit tertentu sulit dideteksi sejak dini oleh yang bersangkutan. Tak heran setiap orang yang menderita penyakit semestinya berkonsultasi dengan dokter spesielis pencernaan. Saluran pencernaan disebut juga dengan saluran *gastrointestinal* (GI), yakni saluran panjang yang masuk melalui tubuh dari mulut ke anus. Saluran ini mencerna, memecah dan menyerap makanan melalui lapisannya ke dalam darah. Gangguan pada saluran pencernaan (*gastrointestinal*) ialah suatu kelainan atau penyakit pada jalan makanan/pencernaan. Penyakit *gastrointestinal* yang termasuk yaitu kelainan penyakit kerongkongan (*eshopagus*), lambung (*gaster*), usus halus (*intestinum*), usus besar (*colon*), hati (*liver*), saluran empedu (*traktus biliaris*) dan pankreas (Hadi, 2002).

Untuk itu sistem pencernaan manusia penting sekali diperhatikan. Sistem pencernaan manusia adalah sebuah sistem yang membantu manusia dalam mencerna makanan dan minuman yang dikonsumsinya untuk menghasilkan energi bagi seluruh anggota tubuh menjadi zat yang lebih mudah dicerna oleh tubuh dan diambil berbagai kandungan di dalamnya yang berguna untuk organ dalam dan bagian tubuh secara keseluruhan. Makanan yang diserap berupa nutrisi dibantu oleh enzim untuk memecah molekul kompleks menjadi molekul yang lebih sederhana sehingga mudah diserap oleh tubuh (Saintif, 2020).

Penyakit pada saluran pencernaan merupakan penyakit yang berbahaya dan banyak menyebabkan kematian. Berdasarkan data dari WHO (*World Health Organization*), penyakit pada saluran pencernaan, diantaranya kanker usus merupakan penyakit yang paling banyak menyebabkan kematian nomor 6 di dunia, dan penyakit diare merupakan penyakit yang menyebabkan kematian nomor 7 di dunia. Hal tersebut disebabkan minimnya pengetahuan akan gejala

awal suatu penyakit yang kurang, pola pikir masyarakat yang ingin hidup serba praktis, kesadaran masyarakat akan kesehatan masih rendah, kurangnya penyampaian informasi melalui media tentang penyakit *gastrointestinal*, serta minimnya angka tenaga medis merupakan masalah yang sedang dihadapi dalam kasus ini, sehingga perlu adanya media bantu berupa sistem mudah memberikan solusi yang tepat untuk menangani permasalahan tersebut. Dalam indrustri kesehatan dan medis keakuratan prediksi sebuah penyakit sangatlah penting dan memerlukan keputusan yang efektif dalam mengambil suatu analisa dan keakuratan prediksi suatu penyakit yang diderita pasien. Aplikasi yang dikembangkan ini bertujuan untuk membantu memberikan informasi yang jelas bagi pasien atau masyarakat umum dan bagi tenaga medis diharapkan dapat membantu dalam penanganannya memberikan solusi yang tepat, dengan hanya memperhatikan gejala- gejala yang dialami (Istiqomah & Fadlil , 2013).

Pada acara APDW 2014 di Nusa Dua, Bali, Minggu (23/11) Sekretaris Jenderal (Sekjen) Kongres Asian Pacific Digestive Week (APDW) Dr Dadang Makmun mengatakan pada agenda penting pertemuan kali ini adalah membahas masalah penyakit pencernaan di perut manusia, karena persentasenya terus meningkat di dunia. Ia mengatakan saat ini penyakit saluran pencernaan masuk dalam 10 besar penyakit mematikan di dunia. Hal yang sama juga terjadi di Indonesia. Berbagai penyakit saluran pencernaan di Indonesia mulai meningkat dari tahun ke tahun. Data terakhir menunjukkan, 30 persen dari pasien di rumah sakit di Indonesia merupakan pasien yang berhubungan saluran pencernaan. Sementara 40-46 persen pasien yang berkunjungan ke klinik, dokter praktek adalah pasien dengan gangguan pencernaan. (Antara, 2014)

Saat ini untuk melihat gangguan pada saluran pencernaan para dokter menggunkan endoskopi. Endoskopi adalah prosedur pemeriksaan menggunakan alat menyerupai selang dengan ujung kamera dan dihubungkan ke layar monitor sehingga dokter dapat melihat secara langsung keadaan yang sesungguhnya terjadi di saluran pencernaan. Endoskopi sendiri harus dilakukan pembaharuan untuk menghasilkan suatu diagnosa yang lebih baik, untuk

mengurangi kemungkinan terjadinya kesalahan diagnosa oleh dokter sangat mungkin terjadi (Dharmawijaya, 2017).

Perkembangan teknologi yang sangat pesat di era digital saat ini tidak bisa lepas dari peranan *artificial intelligence. Artificial intelligence* mempelajari bagaimana membuat computer bisa melakukan sesuatu yang mana orang lakukan, dimana pemikiran atau kecerdasan seperti manusia dapat digunakan oleh peralatan mekanik atau mesin agar dapat melakukan pekerjaan tertentu (Siswanto, 2005).

Seiring dengan perkembangan zaman, *machine learning* pun memiliki evolusi selanjutnya yang masih bagian dari *machine learning* yaitu *deep learning*, dengan metode lebih kompleks tetapi lebih canggih. *deep learning* dapat mempelajari metode komputasinya sendiri menggunakan 'otak'nya sendiri. Teknologi *deep learning* ini salah satu teknologi yang paling populer untuk mengenali suatu kegiatan atau objek yang memiliki tingkat keakuratan lebih tinggi dibanding dengan metode mesin sebelumnya (Primartha, 2018). Untuk memproses dataset seperti klasifikasi gambar, teknik baru telah dikembangkan, yaitu teknik *deep learning* yang merupakan gabungan *mechine learning* dengan AI (*artificial intelligence*). Meningkatkan akurasi dalam menentukan suatu penyakit adalah hal yang sangat penting, karena dapat memperngaruhi pengambilan keputusan pada pengobatan dan tindak lanjut. Endoskopi adalah sumber daya menuntut dan membutuhkan peralatan teknis mahal dan tenaga terlatih. Deteksi otomatis akan berkontribusi untuk mengurangi ketidaksetaraan, meningkatkan kualitas dan mengoptimalkan pengginaan sumber daya medis yang langka. Selain itu, karena pemeriksaan endoskopi adalah pemeriksaan secara langsung (*real time*). Untuk membuat system perawatan kesehatan lebih skalabel dan efektif biaya, padaa dasarnya ilmu komputer dan kedokteran yang bisa melampaui pencitraan medis tradisional dengan menggabungkan kedua bidang ini dengan analisis dan pengambilan data multimedia serta kecerdasan buatan (*artificial intelligence*).

Oleh karena itu, menggunakan *deep learning* diperlukan untuk menyelesaikan masalah ini. Metode yang cocok untuk diterapkan adalah

convolutional neural network dan *tensorflow*. Pada tugas akhir ini akan dibahas mengenai implementasi dan mengukur performansi dari klasifikasi pada penyakit *gastrointestinal* dengan metode *convolutional neural network* dan *tensorflow*.

1.2. Rumusan Masalah

Pada penelitian ini rumusan masalah yang ingin di angkat oleh peneliti adalah sebagai berikut:

1. Bagaimana implementasi *artificial intelligence* dalam mengklasifikasikan penyakit yang ada di dalam saluran pencernaan (*gastrointestinal*) dengan metode *deep learning* menggunakan *convolutional neural network* dan *tensorflow*?
2. Berapa nilai akurasi yang didapatkan dari hasil klasifikasi menggunakan *convolutional neural network* dan *tensorflow*?

1.3. Batasan Masalah

Permasalahan yang akan di bahas pada penelitian ini memiliki ruang lingkup yang luas, adapun batasan masalah sebagai berikut:

1. Penelitian ini menggunakan dataset Kvasir.
2. Penelitian yang di gunakan adalah metode *deep learning* menggunakan *convolutional neural network* dan *tensorflow*
3. Perangkat lunak yang di gunakan dalam penelitian ini adalah aplikasi *Python 3.7* dengan *Library Tensorflow 1.14.0* dan *Sublime Text 3*
4. Jumlah data yang di gunakan adalah 1500 gambar.

1.4. Tujuan Penelitian

Adapun tujuan dari penelitian yang di lakukan adalah:

1. Mengetahui implementasi *artificial intelligence* untuk mengklasifikasikan dan mendeteksi penyakit yang ada di dalam saluran pencernaan (*gastrointestinal*) dengan metode *deep learning* menggunakan *convolutional neural network* dan *tensorflow*.

2. Mengetahui tingkat akurasi yang didapatkan dari hasil klasifikasi dan deteksi menggunakan *convolutional neural network* dan *tensorflow*.

1.5. Manfaat Penelitian

Manfaat dari penelitian tugas akhir ini adalah untuk memudahkan kerja tenaga medis yang ahli dibidangnya sehingga data gambar endoskopi khususnya pada penyakit pencernaan dapat diklasifikasi dan dideteksi secara otomatis dan dapat mengurangi waktu tunggu pasien penderita penyakit gangguan pencernaan (*gastrointestinal*). Selain itu implementasi *deep learning* untuk menghasilkan *artificial intelligence* sangat bermanfaat apabila dikembangkan untuk meningkatkan pelayanan kesehatan di rumah sakit, sehingga pasien dapat memilih pengobatan yang tepat dan dapat membantu dokter dalam memperkuat hasil diagnosa serta dapat mengurangi kemungkinan kesalahan dalam diagnosa.

1.6. Sistematika Penulisan

Sistematika penulisan yang digunakan dalam penulisan tugas akhir ini dapat diuraikan sebagai berikut:

BAB I PENDAHULUAN

Pada bagian ini membahas tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bagian ini memaparkan penelitian-penelitian sebelumnya yang berhubungan dengan permasalahan yang diteliti dan menjadi acuan konseptual.

BAB III LANDASAN TEORI

Pada bagian ini membahas tentang teori-teori dan konsep yang berhubungan dengan penelitian yang dilakukan dan mendukung dalam pemecahan masalahnya.

BAB IV METODOLOGI PENELITIAN

Bab ini memberikan gambaran tentang jumlah populasi dan sampel, jenis dan sumber data, metode analisis data, instrumen penelitian dan tahapan penelitian.

BAB V ANALISIS DAN PEMBAHASAN

Pada bab ini membahas tentang analisa yang sudah dilakukan terhadap hasil pengumpulan, pengolahan dan analisa.

BAB VI PENUTUP

Pada membahas mengenai kesimpulan yang diperoleh dari hasil penelitian, serta saran-saran yang dapat diterapkan dari hasil penelitian sehingga akan berguna dalam penelitian selanjutnya.

BAB II
TINJAUAN PUSTAKA

Dalam menghindari unsur plagiasi dengan penelitian sebelumnya, penulis menjadikan penelitian sebelumnya sebagai acuan. Berikut ini adalah penelitian-penelitian tentang implementasi *artificial intelligence* dengan metode *deep learning* menggunakan *convolutional neural network* dan *tensorflow*.

2.1 Penelitian Menggunakan *Convolutional Neural Network* dan *Tensorflow*

Tabel 2.1 Peneltian Sebelumnya Menggunakan Tensorflow

Peneliti	Judul Penelitian	Hasil
Sirohi . (2020)	*Convolutional neural networks for 5G-enabled Intelligent Transportation System: A systematic review*	Penelitian ini membantu untuk menyelesaikan masalah seperti pengenalan rambu lalu lintas, deteksi lampu lalu lintas, klasifikasi kendaraan dan deteksi pejalan kaki, deteksi dan pelokalan objek, pemilihan fitur dan optimalisasi di Intelligent Transportation System (ITS) berkemampuan 5G
Bernal (2018)	*Artificial Intelligence In Medicine*	Tujuan dari penelitian ini adalah tiga kali lipat. Tujuan utama kami adalah untuk melaporkan bagaimana arsitektur CNN yang berbeda telah berevolusi, mendiskusikan strategi mutakhir, menyingkat hasil yang diperoleh dengan menggunakan kumpulan data publik, dan memeriksa pro dan kontranya. Kedua, makalah ini dimaksudkan sebagai

		referensi rinci kegiatan penelitian di deep CNN untuk analisis MRI otak. Ketiga, penelitian ini menyajikan perspektif tentang masa depan CNN di mana peneliti mengisyaratkan beberapa arah penelitian di tahun-tahun berikutnya.
Lu (2018)	*Lightweight Convolutional Neural Networks for Player Detection and Classification*	Penelitian ini bertujuan untuk mendeteksi dan klasifikasi pemain berbasis *vision-based* dalam aplikasi olahraga. Melakukan eksperimen pada data sepak bola, bola basket, hoki es, dan pejalan kaki. Hasil eksperimen menunjukkan bahwa metode ini dapat mendeteksi pemain secara akurat dalam kondisi yang menantang. Hasil akurasi mutakhir pada tiga jenis permainan (bola basket, sepak bola dan hoki es) dengan 1000 parameter lebih sedikit. pada kumpulan data deteksi pejalan kaki standar di mana metode ini mencapai kinerja yang kompetitif dibandingkan dengan metode canggih.
Mane (2018)	*Moving object detection and tracking Using Convolutional Neural Networks*	Penelitian ini bertujuan untuk deteksi serta melacak suatu objek yang bergerak. Deteksi objek yang kuat merupakan tantangan karena variasi dalam pemandangan. Tantangan terbesar lainnya adalah melacak objek dalam kondisi oklusi. Algoritma pelacakan objek berbasis CNN. Pendekatan yang yang

		digunakan mencapai sensitivitas 92,14%, spesifisitas 91,24% dan akurasi 90,88%.
Liu (2017)	*An Implementation of Number Plate Recognition without Segmentation using Convolutional Neural Network*	Penelitian ini untuk melatih akurasi dalam pengenalan pelat nomorotomatis. Pengenalan karakter pada set tes 796 gambar plat nomor. Hasilnya, kami mencapai akurasi 88,61% dengan set pelatihan hanya 7396 foto yang diperluas dari 3041 gambar plat nomor berbeda, yang merupakan akurasi yang relatif tinggi, terutama untuk CNN dalam yang biasanya membutuhkan sampel dalam jumlah besar. untuk memperkaya kumpulan data dan melakukan pengujian menggunakan jaringan yang lebih sederhana, yang menghasilkan akurasi 90.07% pada satu set pengujian.
Shustanov (2017)	*CNN Design for Real-Time Traffic Sign Recognition*	Penelitian ini implementasi algoritma pengenalan rambu lalu lintas menggunakan jaringan saraf konvolusi. ini juga menunjukkan beberapa arsitektur CNN, yang dibandingkan satu sama lain. Pelatihan jaringan neural diimplementasikan menggunakan library TensorFlow dan arsitektur paralel masif untuk CUDA pemrograman multithread. untuk deteksi dan pengenalan rambu lalu lintas dijalankan secara real time pada mobile GPU. Hasil eksperimen menegaskan efisiensi tinggi dari sistem

		visi komputer yang dikembangkan. klasifikasi rambu lalu lintas menunjukkan hasil yang sangat baik: 99,94% dari citra yang diklasifikasikan dengan benar.

2.2 Penelitian Menggunakan Kvasir

Tabel 2.2 Peneltian Sebelumnya Tentang *Gastrointestinal*

Peneliti	Judul Penelitian	Hasil
Obukhova (2019)	*Method of Endoscopic Images Analysis for Automatic Bleeding Detection and Segmentation*	Penelitian ini berdasarkan segmentasi berbasis blok menggunakan fitur local, Sebagian besar ditentukan oleh karakteristik warna. Dengan hasil klasifikasi yang didapatkan mampu melakukan blok segmentasi yang efektif.
Cogan (2019)	*Accurate identification of anatomical landmarks and diseased tissue in Gastrointestinal tract using deep learning*	Penelitian ini bertujuan untuk memaksimalkan akurasi klasifikasi pada dataset gambar. Untuk data train 85% dan data test 15%. Dengan tingkat akurasi dan presisi yang tinggi, peneliti dapat melakukannya membedakan delapan kelas landmark g*astrointestinal* dan penyakit.
Kasban (2019)	*A robust medical image retrieval system based on wavelet optimization and*	Penelitian ini bertujuan untuk pengambilan citra medis untuk mencari dalam database untuk citra yang mirip dengan citra kueri. Sistem pengembilan gambar medis (MIRS) terdiri dari dua tahap; fase pendaftaran dan fase kueri.

	adaptive block truncation coding	Hasil penelitian menunjukkan bahwa, MIRS yang diusulkan kuat dan efisien untuk database citra medis yang berbeda karena keuntungan membagi citra menjadi blok dan setiap blok dapat diambil secara terpisah sesuai dengan variansnya.
Kirkerod (2019)	*Unsupervised preprocessing to improve generalisation for medical image classification*	Penelitian ini bertujuan untuk klasifikasikan penyakit GI. Dengan koefisien korelasi Matthew 7%. Deteksi penyakit otomatis dalam video dan gambar dari saluran g*astrointestinal* (GI) telah menerima banyak perhatian dalam beberapa tahun terakhir. Namun, kualitas data gambar seringkali berkurang karena hamparan teks dan data posisi. metode preproses gambar tersebut dan menggambarkan pendekatan untuk klasifikasi penyakit GI untuk kumpulan data Kvasir v2.
Owais (2019)	*Artificial Intelligence-Based Classification of Multiple Gastrointestinal Diseases Using Endoscopy Videos for Clinical Diagnosis*	Penelitian ini menggunakan metode AI yang bertujuan menghasilkan kontribusi yang signifikan untuk bidang medis dan diagnose berbasis video. Dengan menggunakan dataset gabungan yang terdiri dari salah satu video endoskopi terbesar dengan 52.471 frame. Hasil eksperimental dari model yang diusulkan mencapai 97,057%. Dengan akurasi rata-rata 92,57%, skor F-1

		93,41%, mAP 94,58% dan mAR 92,28%.
Pogorelov (2017)	*A Multi-Class Image Dataset for Computer Aided Gastrointestinal Disease Detection*	Penelitian ini bertujuan untuk mengklasifikasin menjadi tiga landmark anatomi penting dan tiga temuan klinis. Selain itu, mengandung dua kategori gambar yang terkait dengan penghapusan polip endoskopi.

2.1 Kelebihan dari penelitian sebelumnya

Pada penelitian sebelumnya membahas tentang perbandingan arsitektur *Convolutional Neural Network,* performa dari klasifikasi dan klasifikasi berdasarkan segmentasi berbasis blok. Kelebihan dari penelitian ini dari penelitian-penelitian sebelumnya adalah pada penelitian ini penulis menggunakan data gambar endoskopi untuk mendeteksi dan mengklasifikasikan penyakit yang ada di dalam saluran *gastrointestinal* (GI) yang mana nanti dapat mengidentifikasi lokasi penyakitnya tersebut, pada penelitian ini penulis menggunakan metode *deep learning* dengan *convolutional neural network* dan *tensorflow* dengan kategori *esophagitis, polyps, normal-pylorus, dyed-lifted-polyps* dan *dyed-resection-margins.*

BAB III
LANDASAN TEORI

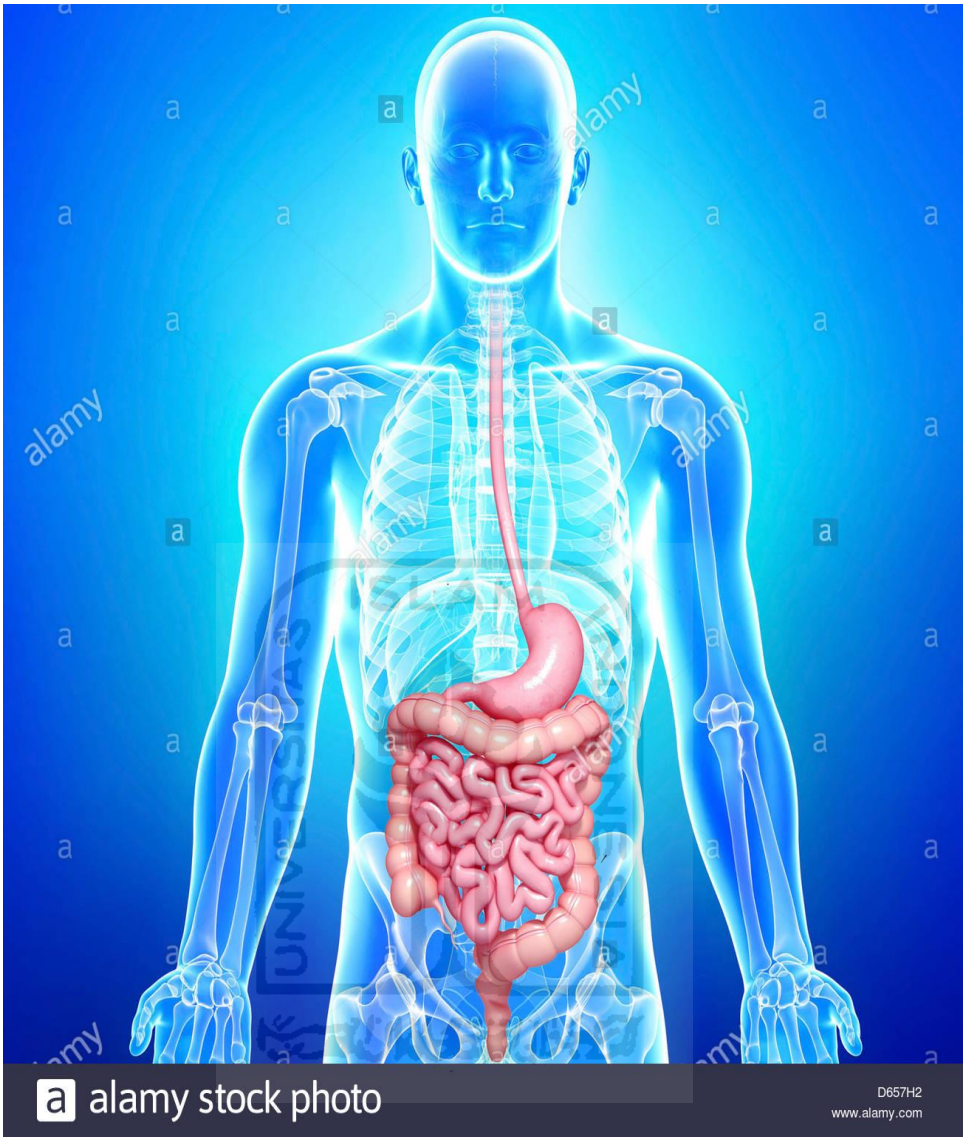
3.1. Sistem Pencernaan Manusia

Sistem pencernaan manusai adalah system yang membantu dalam mencerna makanan untuk menghasilkan energi bagi seluruh anggota tubuh. Makanan yang diserap berupa nutrisi yang dibantu oleh enzim untuk memecah molekul kompleks menjadi molekul yang lebih sederhana sehingga mudah diserap oleh tubuh (Saintif, 2020).

Pencernaan adalah proses penguraian makanan yang semula kasar menjadi halus. Makanan yang sudah dicerna berubah menjadi sari-sari makanan yang lebih mudah diserap oleh pembuluh darah. Lalu sari makanan tersebut diedarkan ke seluruh bagian tubuh oleh darah. Proses pencernaan terjadi di dalam saluran pencernaan dan dibantu oleh enzim yang dihasilkan oleh kelenjar pencernaan (Sudirman, 2017).

Ada dua jenis pencernaan makanan yang terjadi di dalam tubuh, yaitu sebagai berikut:

1. Pencernaan secara mekanis Pencernaan makanan secara mekanis terjadi di dalam mulut. Makanan dilumat oleh gigi sampai hancur agar lebih mudah untuk ditelan.
2. Pencernaan secara kimiawi Pencernaan makanan secara kimiawi dilakukan oleh enzim, yang terjadi di dalam rongga mulut, lambung, dan usus. Tujuan pencernaan dengan bantuan enzim adalah menguraikan makanan menjadi sari makanan yang diserap oleh tubuh.

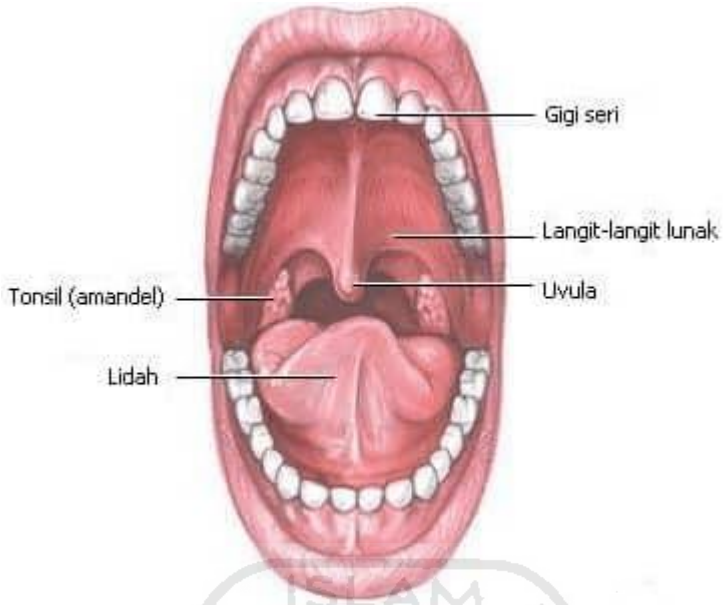


Gambar 3.1. Sistem Pencernaan Manusia
(https://www.alamy.com/stock-photo/digestive-system-blue.html)

3.1.1. Mulut

Bisa dikatakan bahwa mulut adalah pintu gerbang dari sistem pencernaan makanan karena menjadi pintu utama ketika makana masuk. Mulut berfungsi untuk mengunyah makanan menjadi lebih halus agar lebih mudah ditelan.

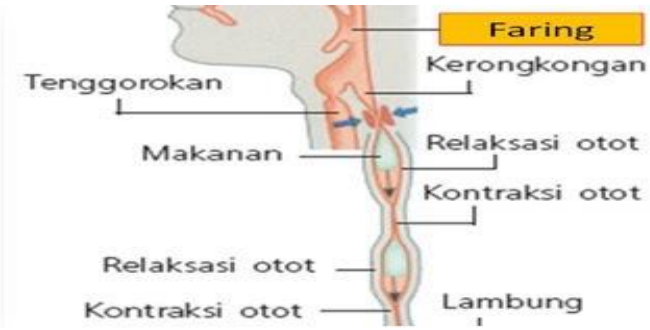
Makanan melalui mulut akan mengalami proses pencernaan secara kimia dan mekanik. Organ yang membantu proses percernaan dalam mulut seperti lidah, gigi dan kelenjar air liur.



Gambar 3.2. Mulut
(https://www.dosenpendidikan.co.id/fungsi-mulut)

3.1.2. Kerongkongan

Setelah makanan melalui mulut dan ditelan, makanan akan melalui tengorokan (*faring*) dan kerongkongan (*esophagus*). Kerongkongan berperan dalam mengantarkan makanan yang sudah ditelan untuk melalui proses selanjutnya dalam lambung. Gerakan kerongkongan yang berkontraksi untuk mendorong makanan ke lambung disebut gerak peristaltik.



Gambar 3.3. Kerongkongan
(https://imateripelajaran.blogspot.com/2018/01/gambar-fungsi-kerongkongan-dalam-sistem.html)

3.1.3. Lambaung

Lambung atau ventrikulus mempunyai bentuk seperti kantong yang menggelembung dan berada pada bagian kiri perut. Lambung mempunyai tiga fungsi utama:

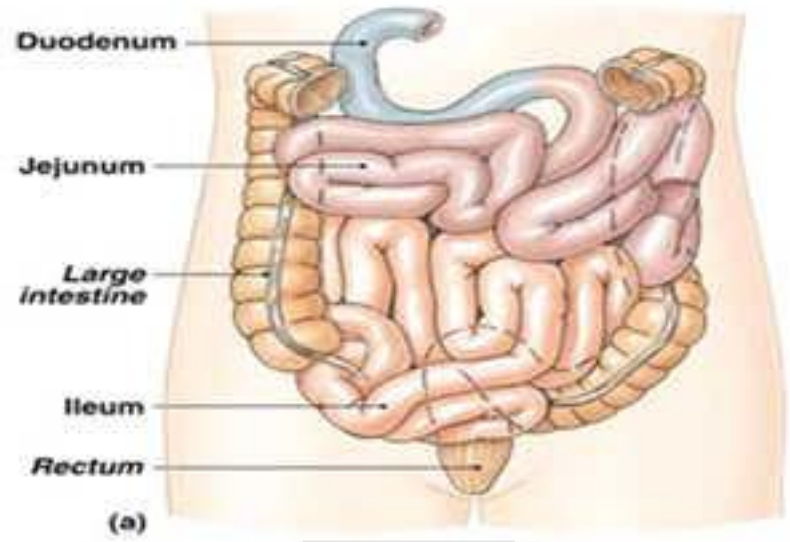
1. Tempat menyimpan makanan sementara sebelum disalurkan ke organ selanjutnya.
2. Memecah dan mengaduk makanan dengan mekanisme gerak peristaltik
3. Mencerna dan menghancurkan makanan dengan bantuan enzim dalam lambung

Gambar 3.4. Lambung
(https://www.zonareferensi.com/fungsi-lambung/)

3.1.4. Usus Halus

Usus halus berbentuk tabung tipis yang panjangnya 10 meter seperti selang yang digulung, dimana permukaan bagian dalamnya penuh dengan tonjolan dan lipatan.

Hasil makanan dari lambung biasanya dalam bentuk semi padat atau chyme. Chyme inilah yang kemudian dilepaskan secara sedikit demi sedikit melalui otot pylori sphincter bagian pertama dari usus halus disebut duodenum (usus 12 jari). Terdapat tiga bagian utama dari usus halus yaitu duodenum (usus 12 jari), jejunum (usus kosong) dan ileum (bagian akhir).



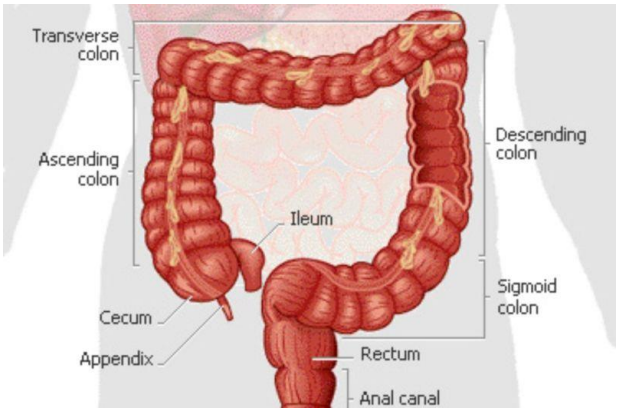
Gambar 3.5. Usus Halus (http://materisekolah45.blogspot.com/2016/08/fungsi-usus-halus-pada-sistem.html)

3.1.5. Usus Besar

Proses penyerapan dari usus halus yang masih belum maksimal kemudian akan dilanjutkan oleh usus besar. Usus besar berbentuk seperti huruf U terbalik yang panjangnya sekitar 5-6 meter. Terdapat tiga bagian utama usus besar yaitu sekum (cecum), kolon dan rektum (rectum).

Sekum berbentuk seperti kantong yang berfungsi menyerap nutrisi yang tidak dapat diserap usus halus. Kolon adalah bagian terpanjang dari usus besar yang berfungsi sebagai tempat cairan dan garam diserap.

Rektum adalah bagian akhir dari usus besar. Rektum terhubung langsung keanus sehingga bagian ini berfungsi untuk tempat penyimpanan tinja sebelum dikeluarkan oleh anus.

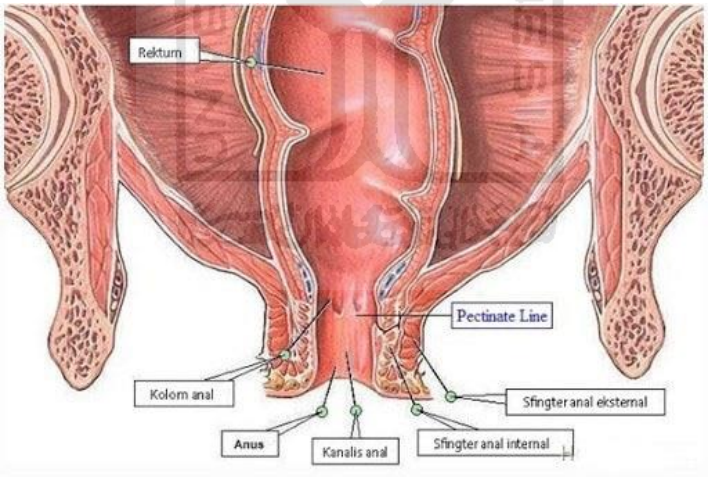


Gambar 3.6. Usus Besar

(https://informazone.com/sistem-pencernaan/)

3.1.6. Anus

Anus berfungsi untuk proses defekasi feses dan mengatur keluarnya fases. Defekasi adalah proses membuang kotoran sisa pencernaan dalam bentuk feses. Hasil akhir dari sistem pencernaan makanan berupa fases atau kotoran.



Gambar 3.7. Anus

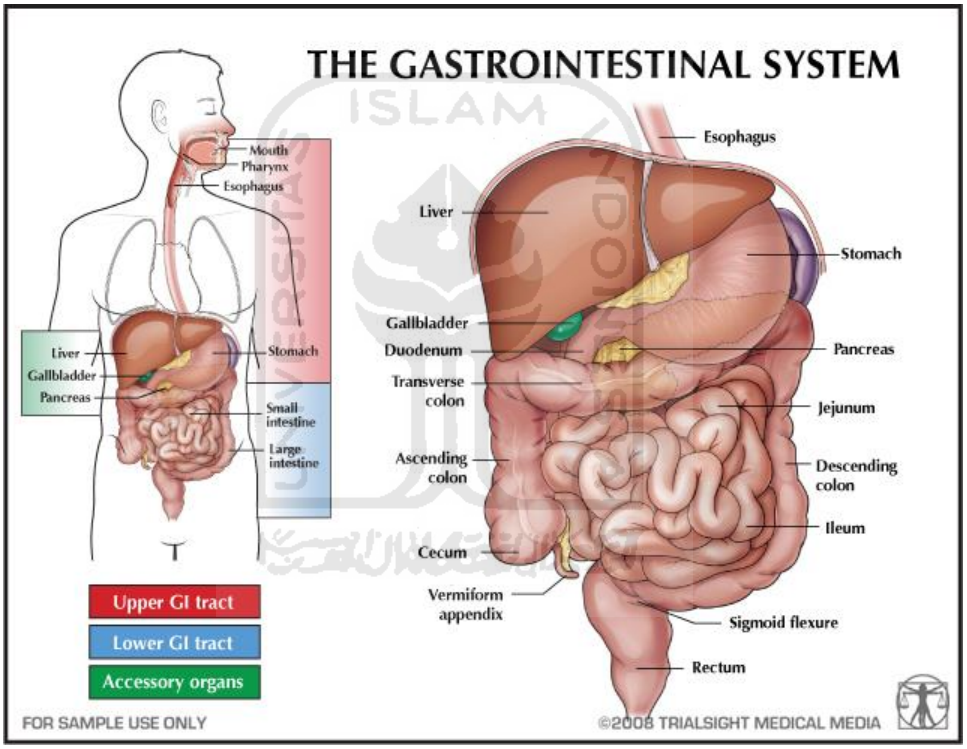
(https://www.ruangbiologi.co.id/fungsi-anus/)

3.2. Gangguan Saluran Pencernaan (*gastrointestinal*)

Gangguan pada *gastrointestinal* ialah suatu kelainan atau penyakit pada jalan makanan/pencernaan. Penyakit *gastrointestinal* yang termasuk yaitu kelainan penyakit kerongkongan (*eshopagus*), lambung (*gaster*), usus

halus (*intestinum*), usus besar (*colon*), hati (*liver*), saluran empedu (*traktus biliaris*) dan pankreas (Hadi, 2002).

Saluran pencernaan (*gastrointestinal*) adalah saluran yang panjang bermula dari mulut sampai anus (Saintif, 2020). Anatomi sistem pencernaan manusia terdiri dari beberapa organ penting yang bertugas untuk mendistribusikan dan mencerna makanan melalui saluran yang dikenal sebagai saluran pencernaan. Organ dalam saluran pencernaan ini sudah sering dikenal seperti mulut, kerongkongan, lambung, usus halus, usus besar dan anus.



Gambar 3.8. *Gastrointestinal*

3.3. Endoskopi

Endoskop adalah alat yang digunakan dalam pemeriksaan endoskopi. Endoskopi adalah pemeriksaan secara visual dan langsung pada lubang atau rongga pada tubuh tertentu untuk melihat kelainan pada tubuh. Pemeriksaan ini langsung di kontrol dari monitor. Alat ini berbentuk pipa kecil panjang yang dapat dimasukkan ke dalam tubuh, misalnya ke lambung, ke dalam sendi, atau ke rongga tubuh lainnya. Di dalam pipa

tersebut terdapat dua buah serat optik. Satu untuk menghasilkan cahaya agar bagian tubuh di depan ujung endoskop terlihat jelas, sedangkan serat lainnya berfungsi sebagai penghantar gambar yang ditangkap oleh kamera. Di samping kedua serat optik tersebut, terdapat satu buah bagian lagi yang bisa digunakan sebagai saluran untuk pemberian obat dan untuk memasukkan atau mengisap cairan. Selain itu, bagian tersebut juga dapat dipasangi alat-alat medis seperti gunting kecil. (Dharmawijaya, 2017).

Gambar 3.9. Endoskopi
(http://www.1800endoscope.com/3meterscopes.htm)

3.4. *Computer Vision*

Sistem penginderaan manusia (dalam hal ini mata) menerima informasi struktur tiga dimensi dari lingkungan yang diamati dan mengolahnya dalam otak untuk kemudian mengambil kesimpulan-kesimpulan tertentu berdasarkan persepsi dan pengalamannya, semisal menentukan jumlah, bentuk, atau warna pada objek yang diamati. Selain itu juga dapat mencari perbedaan ataupun mencari kesamaan suatu objek dengan objek lain. Teknik *computer vision* dikembangkan dengan memanfaatkan fungsi-fungsi matematis untuk menampilkan kembali

informasi tiga dimensi suatu objek disik dalam citra digital. Pemanfaatan *computer vision* telah meluas pada bidang-bidang teknologi seperti rekaan topografi dari kumpulan citra digital, pengamatan objek berjalan, serta deteksi dan pengenalan objek (Szeliski, 2010).

Tujuan utama *computer vision* adalah untuk membuat model dan ekstrak data serta informasi dari gambar, sementara *image processing* adalah tentang mengimplementasikan transformasi komputasi untuk gambar, seperti penajaman, kontras, dan lain-lain (Babatunde, 2015).

3.5. Citra

Citra atau *image* adalah representasi spasial dalam suatu objek yang sebenarnya dalam bidang dua dimensi yang biasanya ditulis dalam koordinat cartesian x-y dan setiap kordinat merepresentasikan satu sinyal terkecil dari ojek (Kulkarni, 2001). Fungsi citra adalah model matematika yang sering digunakan untuk menganalisis dimana semua fungsi analisis digunakan untuk mempertimbangkan citra sebagai fungsi dengan 2 variabel.

Citra atau *image* merupakan fungsi dua dimensi yang dimana terdapat harga ciri atau nilai citra pada setiap titik elemen, nilai yang dimaksud meliputi tingkat keabuan (intensitas) dan vector warna. Sebuah citra atau *image* adalah representasi objek tiga dimensi menjadi dua dimensi. Pada dasarnya citra (*image*) dan gambar (*picture*) merupakan dua hal yang berbeda, gambar tidak dapat diproses dan apabila ingin diproses harus diubah kedalam bentuk citra terlebih dahulu. Sedangkan citra merupakan representasi yang mengandung informasi deskriptif mengenai suatu objek (Siswanto, 2005).

3.5.1. Pixel

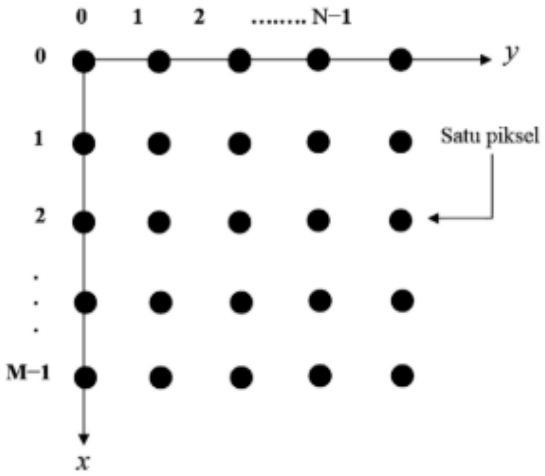
Pixel atau *picture element* merupakan unsur pada gambar atau representasi titik terkecil pada sebuah gambar atau citra digital, bisa dibilang *pixel* merupakan unsur terkecil dalam sebuah gambar. Sebuah gambar atau citra digital dapat tersusun dari ratusan atau bahkan ribuan

pixel. Jumlah *pixel* dapat mempengaruhi kualitas pada suatu citra digital, artinya semakin banyak jumlah *pixel* maka akan semakin bagus pula kualitas gambar tersebut. Pada sebuah citra digital berwarna terdapat 3 buah sub-*pixel* yang berwarna merah, hijau dan biru atau yang biasa disebut dengan *RGB* (*Red, Green, Blue*) (Siswanto, 2005).

3.5.2. Citra Digital

Citra Digital adalah gambar dua dimensi yang dihasilkan dari gambar analog dua dimensi yang kontinus menjadi gambar diskrit melalui proses sampling. Proses perubahan citra menjadi citra digital dinamakan dengan digitasi. Digitasi merupakan proses mengubah sebuah gambar, teks, atau suara dari benda yang dapat diliat ke dalam data elektronik dan dapat disimpan serta diproses untuk keperluan lainnya. Dalam konteks yang lebih luas, pengolahan citra digital lebih mengacu pada pemrosesan setiap dua data dimensi. Pengolahan citra digital adalah sebuah disiplin ilmu yang mempelajari tentang bagaimana teknik pengolahan sebuah citra. Citra yang dimaksud disini adalah sebuah gambar diam (foto) maupun gambar bergerak (Video). Sedangkan digital disini mempuyai maksud penting bahwa pengolahan citra/gambar dilakukan secara digital menggunakan komputer (Sutoyo, T., Mulyanto, E., Suhartono, Dwi Nurhayati Oky, & Wijanarto, 2019).

Citra digital adalah representasi numerik dari citra dua dimensi (Asmara, 2018). Nilai numerik yang di representasikan umumnya adalah nilai biner 8 bit. Nilai biner ini disimpan pada elemen citra yang sering disebut sebagai pixel. Citra digital berisi pixel yang jumlah baris dan kolomnya tetap. Pixel adalah elemen gambar terkecil dari citra digital (Andono, 2017).



Gambar 3.10. Koordinat yang citra digital (Putra, 2010).

Artinya sebuah citra digital dapat ditulis dalam bentuk matriks berikut:

$$f\ (x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,M-1) \\ f(1,0) & f(1,1) & \cdots & f(1,M-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(N-1,0) & f(N-1,1) & \cdots & f(N-1,M-1) \end{bmatrix} \quad (3.1)$$

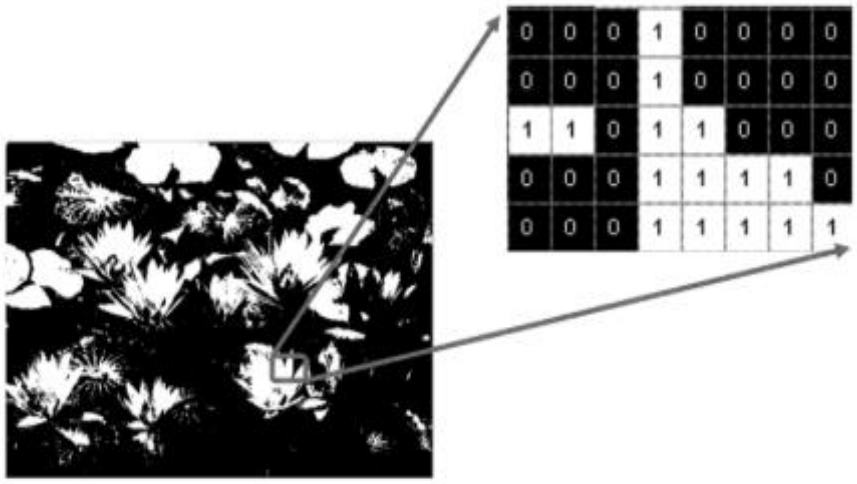
Berdasarkan gambar diatas, maka secara matematis citra digital dapat dituliskan sebagai fungsi intensitas *f(x,y)*, dimana harga x (baris) dan y (kolom) merupakan koordinat posisi dan *f(x,y)* adalah nilai fungsi pada setiap titik *(x,y)* yang menyatakan besar intensitas citra atau tingkat keabuan atau warna dari piksel di titik tersebut.

3.5.3. Jenis Citra Digital

Ada beberapa tipe citra yang sering digunakan dalam penelitian menurut Pulung Nurtantio Andono (2017:3), yaitu:

1. Citra Biner

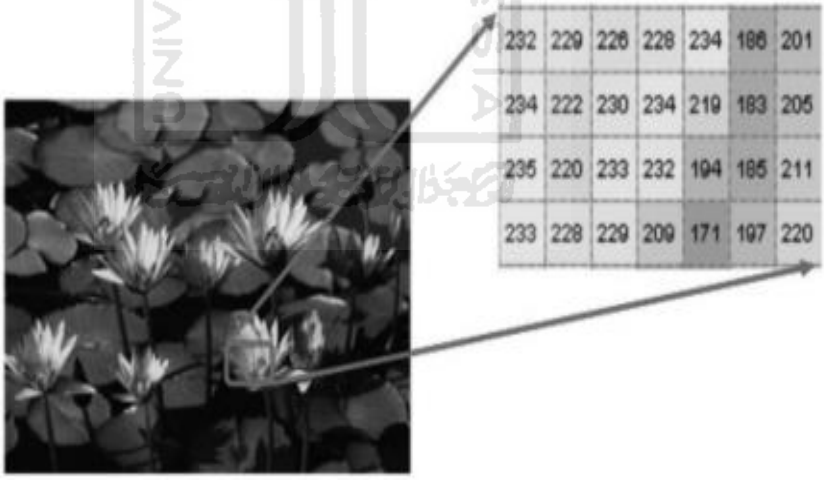
   Pada citra biner, tiap-tiap piksel hanya membutuhkan 1bit memori. Maka dengan demikian, setiap piksel hanya mempunyai dua buah kemungkinan nilai intensitas, yaitu 1 atau 0.



Gambar 3.11. *Citra Biner*

2. Citra *Grayscale*

Citra *grayscale* adalah matriks data yang nilai-nilainya mewakili intensitas setiap piksel berkisar antara 0 sampai dengan 255. Setiap piksel membutuhkan 8bit memori.



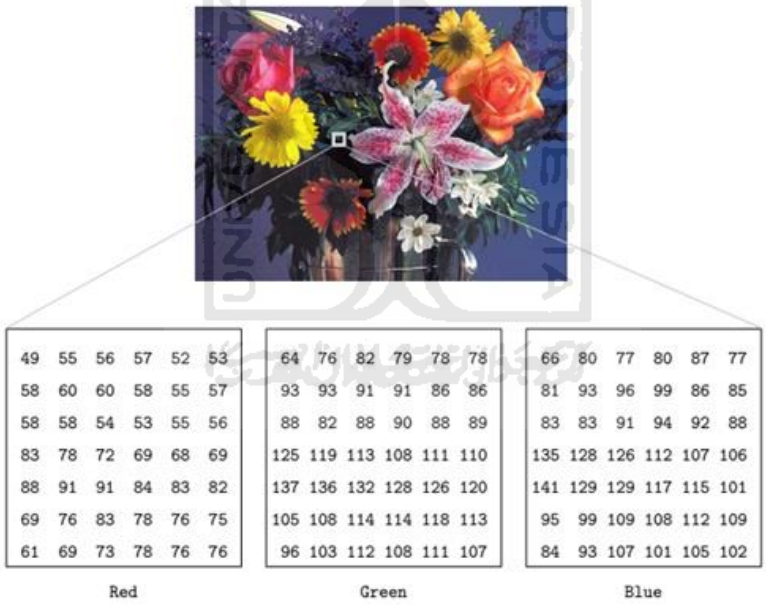
Gambar 3.12. *Citra Greyscale*

3. Citra Warna

Citra warna adalah citra yang masing-masing piksel mempunyai tiga komponen warna yang spesifik, yaitu komponen merah *(red)*, hijau *(green)* dan biru *(blue)*. *RGB* adalah suatu model warna yang terdiri dari merah, hijau, dan

biru, digabungkan dalam membentuk suatu susunan warna yang luas. Setiap warna dasar, misalnya merah, dapat diberi rentang nilai. Untuk monitor komputer, nilai rentangnya paling kecil = 0 dan paling besar = 255 (Basuki, 2016).

Warna setiap piksel ditentukan oleh kombinasi intensitas warna merah, hijau dan biru yang disimpan pada bidang warna di lokasi piksel. Format *file* grafis menyimpan citra warna sebagai citra 24bit, yang berasal dari komponen merah, hijau dan biru masing-masing 8 bit. Hal ini menyebabkan citra warna mempunyai 24 juta kemugkinan warna. Gambar 3.4 menunjukkan citra warna dilihat dari dekat dengan beberapa nilai intensitas piksel.



Gambar 3.13. *Citra Warna*

3.6. Perbaikan Kualitas Citra *(Image Enhancement)*

Perbaikan kualitas citra dilakukan dengan memanipulasi parameter-parameter citra yang bertujuan untuk memperbaiki kualitas citra tersebut (Munir, 2004). Ciri-ciri khusus dari citra akan lebih ditampilkan jika melakukan operasi ini:

a) Perbaikan kontras gelap/ terang

b) Peregangan kontras (*contrast stretching*)
c) Pengubahan histogram citra
d) Perbaikan tepian objek (*edge enhacement*)
e) Pelembutan citra (*image smoothing*)
f) Penajaman citra (*sharpening*)
g) Pemberian warna semu (*pseudocoloring*)
h) Penapiran derau (*noise filtering*)
i) Pengubahan geometri

3.7. *Artificial Intelegence*

Artificial inteligence adalah studi tentang teori dan pengembangan sistem komputer agar mampu melakukan tugas-tugas yang dahulu hanya dapat dilakukan oleh manusia. Seperti membedakan berbagai gambar, menjawab pertanyaan, mengenali dan menerjemahkan Bahasa dan sebagainya. Salah satu implementasi AI adalah pada mobil kendali otomatis. Mobil harus mampu mengenali rambu-rambu lalu lintas, mengamati kondisi lalu lintas, memperhatikan keberadaan manusia dan objek di sekitarnya serta hal-hal lainnya (Primartha, 2018).

Pada (Russel Stuart and Norvig Peter, 1995) mengelompokkan definisi AI ke daalam 4 kategori, yang salah satunya adalah ***acting rationally***: *the rational agent approach.* Artinya membuat inferensi yang logis merupakan bagian dari suatu *rational agent*. Hal ini disebabkan satu-satunya cara untuk melakukan aksi secara rasional adalah dengan menalar secara logis. Dengan melihat secara logis, maka bisa dapat kesimpulan bahwa aksi yang diberikan akan mencapai tujuan atau tidak. Jika mencapai tujuan, maka *agent* dapat melakukan aksi berdasarkan kesimpulan tersebut.

3.8. *Machine learning*

Istila *machine learning* mula-mula diperkenalkan oleh Arthur Samuel pada tahun 1959 melalui jurnalnya yang berjudul "Some Studies in *Machine learning Using the Game of Checkers*". (IBM *Jurnal of Research and Development*). Samuel mencoba mempelajari program komputer untuk bermain catur lebih baik darinya. Tujuannya tercapai, pada tahun 1962

program buatannya dapat mengalahkan juara catur dari negara bagian *connecticut*. Jadi, secara sederhana dapat dijelaskan bahwa *machine learning* adalah pemograman komputer untuk mencapai kriteria/performa tertentu dengan menggunakan sekumpulan data training atau pengalaman di masa lalu (Primartha, 2018).

Gambar 3.14. Samuel arthur sedang mengajari computer bermain catur

Salah satu dampak positif dari *machine learning* adalah menjadi peluang bagi para wirausahawan dan praktisi teknologi untuk terus berkarya dalam mengembangkan teknologi *machine learning*. Terbantunya aktivitas yang harus dilakukan manusia pun menjadi salah satu dampak positif machine learning. Sebagai contohnya adalah adanya fitur pengecekan ejaan untuk tiap bahasa pada *microsoft word*. Pengecekan secara manual akan memakan waktu berhari-hari dan melibatkan banyak tenaga untuk mendapatkan penulisan yang sempurna. Tapi dengan bantuan fitur pengecekan ejaan tersebut, secara *real-time* bisa melihat kesalahan yang terjadi pada saat pengetikan.

Akan tetapi disamping itu ada dampak negatif yang harus diwaspadai. Adanya pemotongan tenaga kerja karena pekerjaan telah digantikan oleh alat teknologi *machine learning* adalah suatu permasalahan yang harus dihadapi. Ditambah dengan ketergantungan terhadap teknologi

akan semakin terasa. Manusia akan lebih terlena oleh kemampuan *gadget*-nya sehingga lupa belajar untuk melakukan suatu aktivitas tanpa bantuan teknologi. (BigsmiLe, 2016).

Inti dari *machine learning* adalah bagaimana membuat komputer dapat menyelesaikan berbagai persoalan dan dapat belajar sendiri seperti manusia belajar sesuatu. Berdasarkan prinsip pembelajaran, algoritma *machine learning* dibagi menjadi *supervised learning, unsupervised learning, semi-supervised learning* dan *reinforcement learning*.

3.8.1. *Supervised Learning*

Supervised learning adalah membangkitkan suatu fungsi yang memetakan *input* (denga sejumlah atribut) ke *output* (label kelas) berdasarkan data berlabel yang diberikan. Kualitas hasil pembelajaran sangat bergantung pada validasi label pada data latih. *Supervised* digunakan untuk meyelesaikan masalah klasifikasi maupun regresi. Beberapa model yang termasuk dalam *supervised* adalah: *deep neural network* (DNN), *convolutional neural network* (CNN), *recurrent neural network* (RNN) dan lainnya sebagainya (Suyanto, Deep Learning Moderisasi Machine Learning untuk Big Data, 2019).

Supervised learning adalah proses belajar yang membutuhkan guru. Yang dimaksud guru di sini adalah sesuatu yang memiliki pengetahuan tentang lingkungan. Guru bisa direpresentasikan sebagai sekumpulan sampel-sampel *input-output*. Pembangunan pengetahuan dilakukan oleh guru dengan memberikan respon yang diinginkan kepada *supervised*. Respon yang diinginkan tersebut mempresentasikan aksi optimum yang dilakukan oleh *supervised*. Parameter-parameter jaringan berubah-ubah berdasarkan vektor latih dan sinyal kesalahan (sinyal kesalahan adalah perbedaan antara keluaran *supervised* dan respon yang diinginkan). Proses perubahan ini dilakukan secara berulang-ulang, selangkah demi selangkah, dengan tujuan agar *supervised* bisa

memiliki kemampuan yang mirip dengan gurunyaa. Dengan kata lain *supevised* dilatih untuk dapat memetaan sekumpulan sampel *input-output* dengan akurasi yang tinggi (Suyanto, Artificial Intelligence Searching, Reasining, Planning dan Learning (Revisi Kedua), 2014).

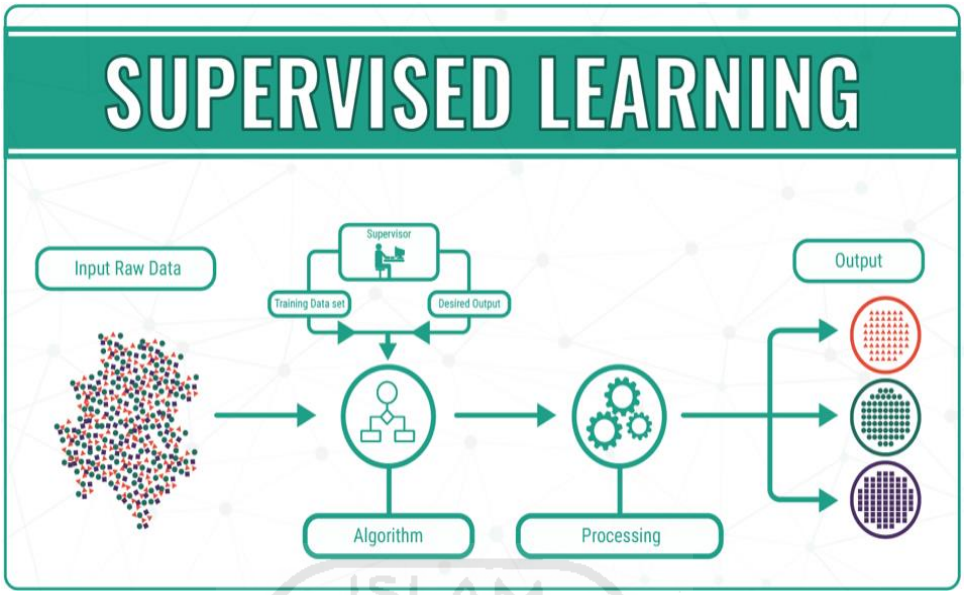
Sebagian besar praktik *machine learning* mengandalkan algoritma *supervised learning*. Algoritmanya dinamakan seperti ini karena training dataset (sekumpulan data untuk training) akan memandu dan mengajari komputer agar menghasilkan *outcome* sesuai harapan. Metode semacam ini mirip dengan seorang guru sedang mengajari anak-anak berhitung (Primartha, 2018).

Supervised learning menggunakan training data yang sudah diberi label untuk mempelari *mapping funcion*, dari *input* variabel (x) ke *output* variabel (y).

$$y = f(x) \tag{3.1}$$

Beberapa algoritma yang termasuk dalam *supervised* antara lain:

- *Decision tree*
- *Naïve Bayes Classifier*
- *Artificial neurol networks*
- *Linear Regression*
- dsb



Gambar 3.15. Ilustrasi *supervised learning*

3.8.2. *Unsupervised Learning*

Unsupervised learning adalah memodelkan sekumpulan *input* secara otomatis tanpa ada panduan (berupa *output* yang diinginkan). Artinya data-data yang dipelajari hanya berupa *input* tanpa label kelas. *Unsupervised* biasanya digunakan untuk masalah klasterisasi: diberikan sebuah himpunan data masukan, kelompokkan data tersebut ke dalam sejumlah klaster berdasarkan kriteria tertentu. Misalnya tingkat kemiripan dalam suatu kelas. (Suyanto, 2019).

Unsupevised learning adalah proses belajar yang tidak membutuhkan guru untuk memantau pross belajar. Dengan kata lain, tidak ada sekumpulan sampel input-output atau fungsi tertentu untuk dipelajari oleh jaringan (Suyanto, 2019).

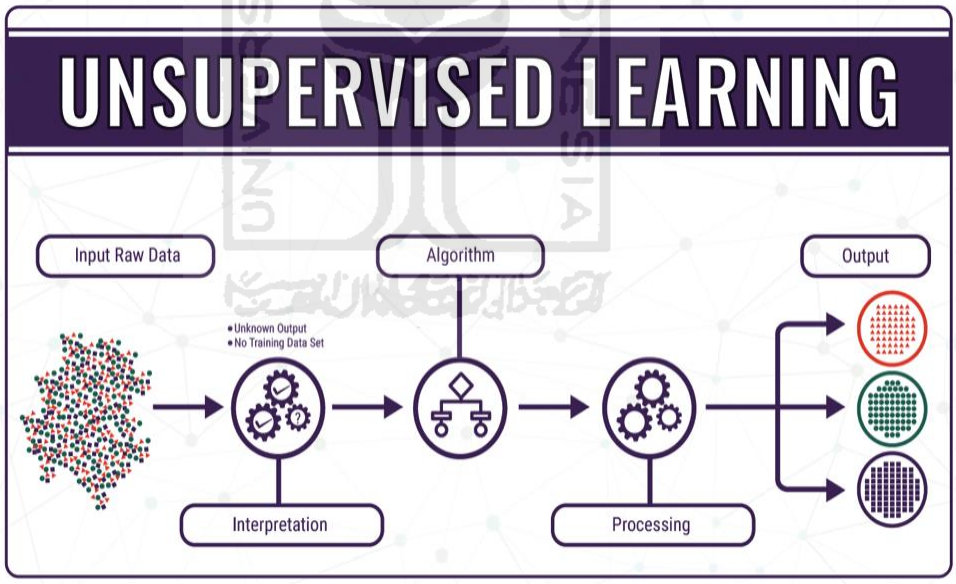
Pada *unsupervised learning* persoalan diproses hanya mengandalkan data yang belum dilatih sebelumnya. *Unsupervised* menggunakan *unlabeled training dataset* untuk memodelkan struktur dari data. Sehingga *unsupervised learning* bersifat lebih subjektif dibandingkan *supervised learning* (Primartha, 2018).

Unsupervised learning bermanfaat untuk kasus-kasus dimana kita ingin menemukan relasi implisit dari *unlabeled dataset* yang disediakan. Jadi pada *unsupervised learning* kita tidak memprediksi masa depan. Sebab *input* variabel (X) tidak memiliki relasi dengan *output* variabel (Y).

$$f(X) \tag{3.2}$$

Beberapa algoritma yang termasuk dalam *unsupervised* antara lain:

- *K-Means, Hierarchical Clustering*
- *Fuzzy C-Means*
- DBSCAN
- *Self-Organizing Maps*
- dsb.



Gambar 3.16. Ilustrasi *unsupervised learning*

3.8.3. *Reinforcement Learning*

Reinforcement learning (RL) adalah suatu metode *learning* yang dipengaruhi oleh *feedbeck* dari lingkungan dengan teknik *learning* yang *iterative* (berulang-ulang) dan *adaptive* (menyesuaikan). RL dipercaya mendekati cara manusia belajar (Primartha, 2018).

Reinforcement learning adalah mempelajari suatu kebijakan bagaimana melakuka aksi berdasarkan hasil pengamatan terhadap lingkungan yang ada. Setiap aksi menghasilkan akibat bagi lingkungan tersebut, dan lingkungan memberikan umpan balik untuk memandu *reinforcement learning* (Suyanto, 2019).

Reinforcement learning (RL) diinspirasi oleh kebiasaan makhluk hidup dalam belajar dan bertindak, khususnya manusia. Pada RL tidak ada *training* dataset. Data-data yang diperoleh berdasarkan pengalaman. Algoritma RL mengijinkan agent untuk memutuskan aksi selanjutnya berdasarkan kondisi saat ini.

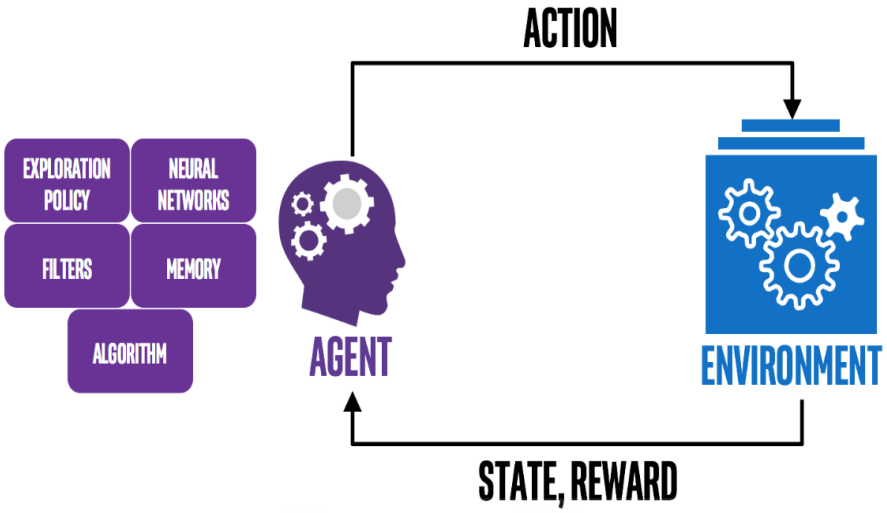
Representasi RL mirip dengan *supervised learning*. Yang membedakan adalah pada RL inputnya tidak hanya *x*, namun *x* dan *y*.

$$y = f(x)\ given\ z \tag{3.3}$$

Algoritma RL tidak memiliki tujuan eksplisit, sebagai gantinya algoritma dipaksa untuk belajar menemukan nilai optimal melalui kegiatan trial dan error. Salah satu contoh penerapan RL adalah pada bidang *robotic.*

Beberapa algoritma yang dikelompokkan dalam RL antara lain:

- *Genetic Algorith* (GA)
- *Dynamic Programming* (DP)
- *Generalized Policy Iteration* (GPI)
- *Monte Carlo*.



Gambar 3.17. Ilustrasi *Reinforcement learning*

3.9. *Deep Learning*

Deep learning adalah merupakan metode *learning* yang memanfaatkan *artificial neural networks* yang berlapis-lapis (*multi layer*). *Artificial neural networks* ini dibuat mirip dengan otak manusia, dimana *neuron-neuron* terkoneksi satu sama lain sehingga membentuk sebuah jaringan *neuron* yang sangat rumit (Primartha, 2018).

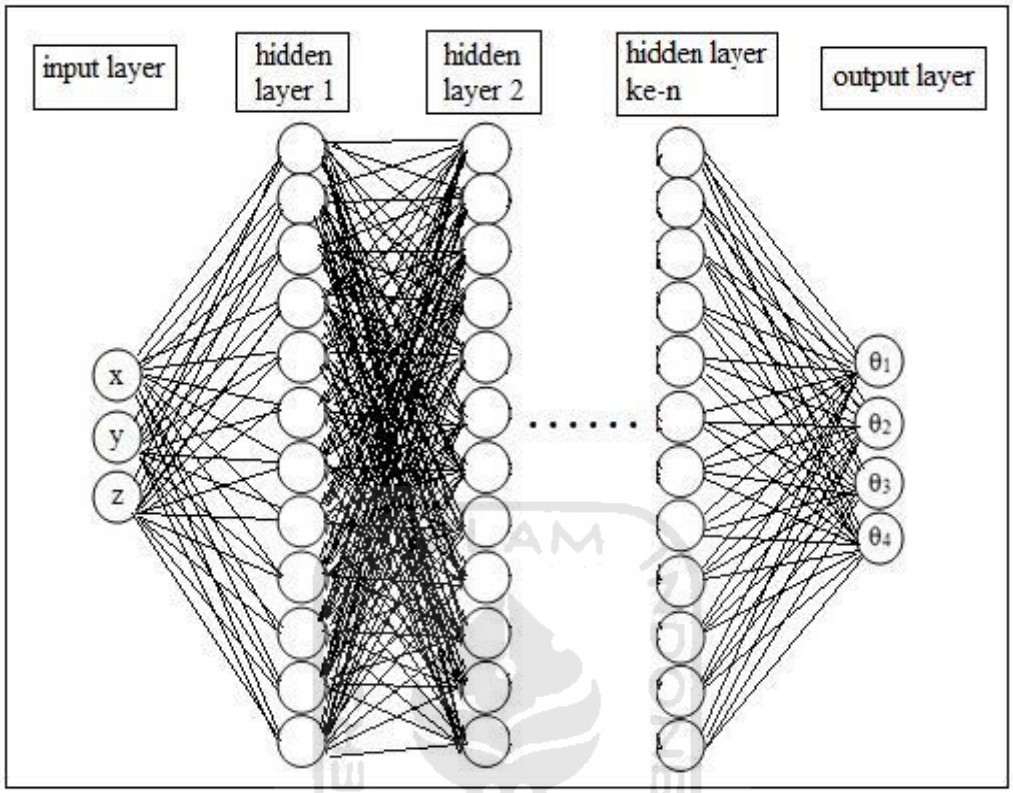
Deep learning merupakan metode learning yang memanfaatkan *multiple non-linear transformation*. Yang diketahui bahwa *deep learning* dapat dipandang sebagai gabungan *mechine learning* dengan AI (*artificial intelligence*).

Deep learning dapat dikelompokkan menjadi empat pendekatan yaitu deep unsupervised learning, deep supervised learning, semi supervised learning dan deep reinforcement learning (Suyanto, 2019).

Beberapa algoritma yang termasuk dalam kategori *deep learning* antara lain:

- *Convolutional Networks*
- *Restricted Boltzmann Machine* (RBM),
- *Deep Belief Network* (DBN),

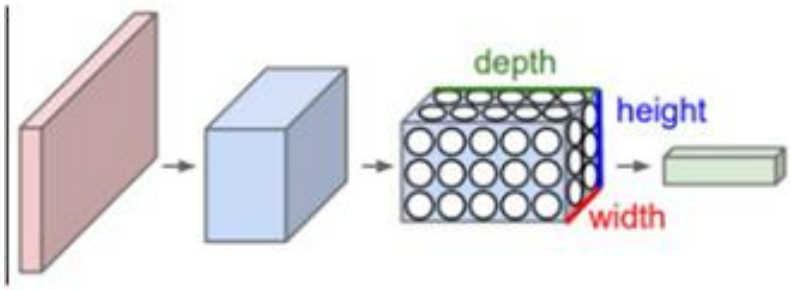
- *Stacked Autoencoders*.



Gambar 3.18. Ilustrasi *multilayer artificial neurol networks*

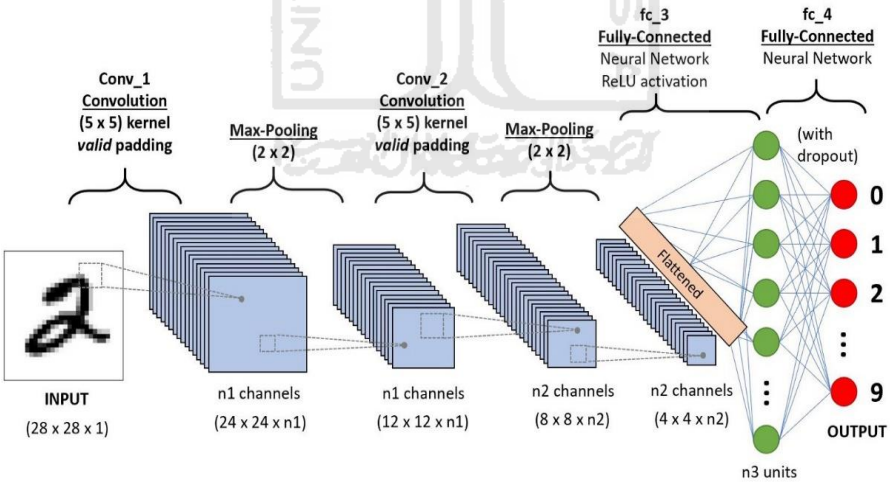
3.10. *Convolutional Neural Network*

Convolutional neural network (CNN) merupakan salah satu model *deep learning* yang banyak digunakan untuk keprluan analisis citra/visual (Primartha, 2018). Secara prinsip, CNN meniru visual cortex pada mamalia. CNN memiliki *neuron-neuron* yang disusun secara tiga dimensi, jadi memiliki panjang, lebar, dan tinggi. Sehingga CNN sangat efektif dan efisien untuk menganalisis *image*/gambar, seperti diilusrasikan pada Gambar 3.10 (Primartha, 2018). Lapisan CNN mentransformasikan volume masukan tiga dimensi (3D *input volume*) ke dalam volume keluaran tiga dimensi aktivasi-aktivasi sel saraf (3D *output volume of neuron activations*). Pada gambar tersebut, masukan berupa citra berwarna, dimana lebar dan tinggi menyatakan dimensi citra tersebut sedangkan dalam (*depth*)-nya adalah 3 yang menyatakan kanal *red, green, blue* (RGB).



Gambar 3.19. Ilustrasi arsitektur CNN secara umum

Arsitektur CNN sangat sederhana: terdiri atas satu lapis masukan (*input layer*), satu lapis keluaran (*output layer*), dan sejumlah lapis tersembunyi (*hidden layers*). Lapis tersembunyi umumnya berisi convolutional layers, pooling layers, normalization layers, ReLu layer, fully connected layers, dan loss layer. Semua lapisan tersebut disusun secara bertumpuk-tumpuk, seperti sepotong sandwich yang berisi roti dibagain bawah, sayuran, daging, keju, saus tomat, mayonnaise, saus sambal, dan roti bagian atas.



Gambar 3.20. Ilustrasi CNN

3.9.1 Convolution Layer

Convolution layer merupakan bagian dari tahap pada arsitektur CNN. Tahap ini melakukan operasi konvolusi pada *output* dari layer sebelumnya. Layer tersebut adalah proses utama yang mendasari

jaringan arsitektur CNN. Konvolusi adalah istilah matematis dimana pengaplikasian sebuah fungsi pada *output* fungsi lain secara berulang (Nurhikmat, 2018).

Operasi ini menerapkan fungsi *output* sebagai *feature map* dari input citra. *Input* dan *output* ini dapat dilihat sebagai dua argumen bernilai riil. Perhitungan untuk *forward pass* dan *backprogagation* di *convolutional layer* mengikuti prosedur standar dalam literatur, dan memiliki filter terlatih dan satu bias latih setiap *map*.

Jika menambahkan istilah bias ke operasi konvolusi, maka nantinya dapat membuat hasil konvolusi positif pada bagian tepi horizontal (vertikal) dengan arah tertentu (misalnya, bagian tepi horizontal dengan piksel di atasnya lebih cerah daripada piksel di bawahnya), dan negatif pada lokasi lain. Pola-pola yang lebih kompleks ini akan lebih lanjut dirangkai oleh lapisan yang lebih dalam untuk mengaktifkan bagian-bagian objek yang bermakna secara semantik atau bahkan jenis objek tertentu, misalnya, anjing, kucing, pohon, pantai, dll. Selain itu juga manfaat dari *convolution layer* adalah semua lokasi spasial membagi kernel konvolusi yang sama, yaitu dengan mengurangi jumlah parameter yang diperlukan untuk lapisan konvolusi. Misalnya, jika beberapa anjing muncul dalam gambar input, fitur "pola seperti kepala anjing" yang sama akan diaktifkan di beberapa lokasi, sesuai dengan kepala anjing yang berbeda (Suyanto, 2014).

Penentuan volume *output* dapat ditentukan dari masing-masing lapisan dengan *hyperparameters*. *Hyperparameter* yang digunakan pada persamaan di bawah ini digunakan untuk menghitung banyaknya neuron aktivasi dalam sekali *output.* Perhatikan persamaan berikut

$$(W - F + 2P)/(S + 1) \quad (3.4)$$

Keterangan:

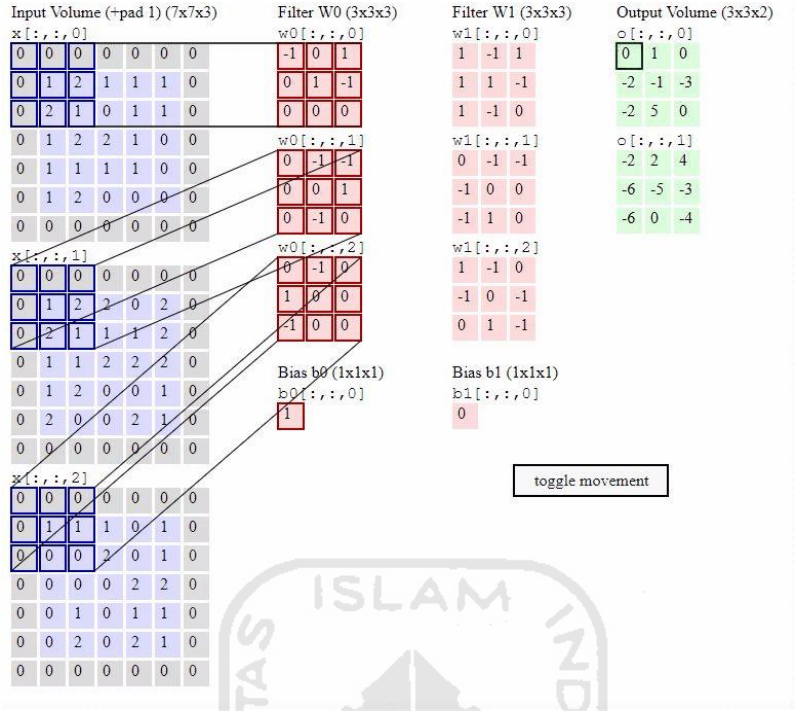
W = Ukuran input gambar

F = Ukuran Filter

P = Nilai *Padding* yang digunakan

S = Ukuran Pergeseran (*Stride*)

Convolution layer terdiri dari *neuron* yang tersusun sedemikian rupa sehingga membentuk sebuah filter dengan panjang dan tinggi (*pixels*). Sebagai contoh, layer pertama pada *feature extraction layer* biasanya adalah *convolution layer*. Layers dengan ukuran 6x6x3, panjang 6 *pixels,* tinggi 6 *pixels* dan tebal/ jumlah 3 buah. Ketiga filter ini akan digeser keseluruh bagian dari gambar. Setiap pergeseran akan dilakukan perkalian *dotproduct* antara *input* dan nilai dari *filter* tersebut sehingga menghasilkan sebuah *output* atau biasa disebut sebagai *activation map* atau *feature map.* Perhatikan ilustrasi berikut:

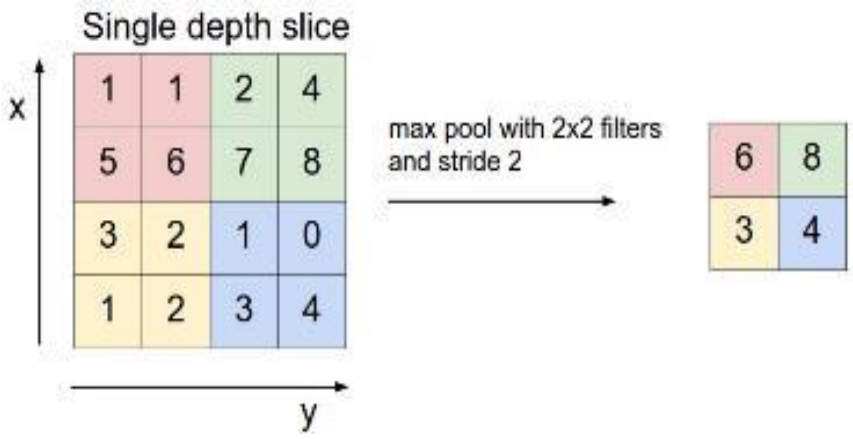


Gambar 3.21. *Convolution Layer*
Sumber: (Sena, 2018)

3.9.2 Pooling Layer

Pooling Layer merupakan lapisan yang menggunakan fungsi dengan *feature map* sebagai masukan dan mengolahnya dengan berbagai macam operasi statistik berdasarkan nilai piksel *terdekat*. Tujuan dari *pooling layers* adalah untuk mencapai invarian spasial dengan mengurangi resolusi *feature map*.

Proses pooling atau *pooling layer* merupakan operasi setelah proses konvolusi, sehingga *activation map* atau *output* yang dihasilkan dari proses konvolusi akan menjadi *input* bagi *poolong layer*. *Pooling layer* berfungsi untuk memperhalus atau mengurangi tingkat sensitifitas terhadap *noise* pada citra digital dengan cara mengurangi dimensi pixelnya. Proses pooling dapat dilakukan dengan 2 cara yaitu dengan mengambil nilai maksimum (*max-pooling*) atau dengan mengambil nilai rata-rata (*average pooling*) dari beberapa pixel (Suyanto, 2014).Berikut gambar contoh operasi *Max Pooling*.



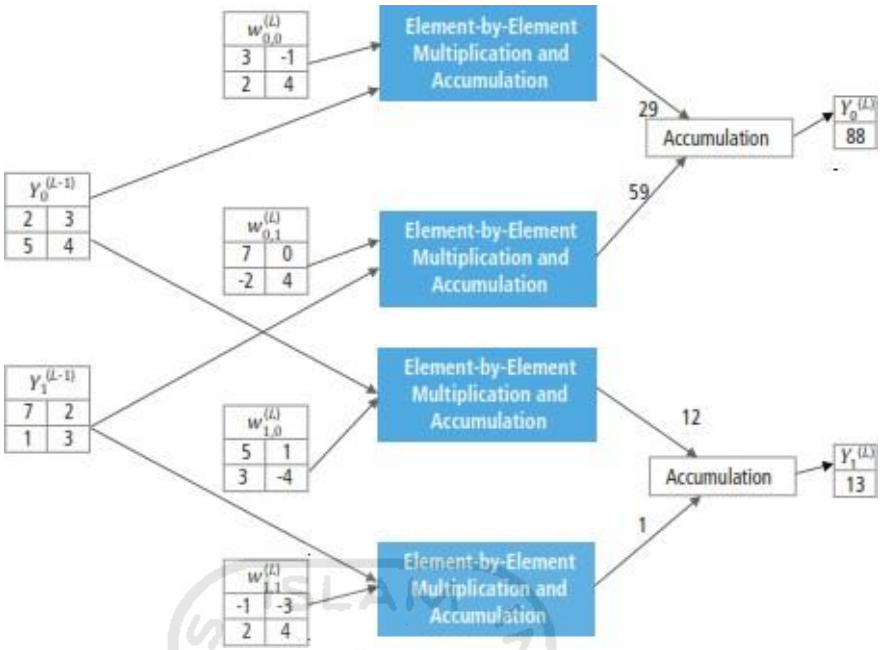
Gambar 3.22. Operasi *Max Pooling*

Lapisan *pooling* akan beroperasi pada setiap irisan kedalaman *volume input* secara bergantian. Pada gambar di atas, lapisan *pooling* menggunakan salah satu operasi maksimal yang merupakan operasi yang paling umum. Dari ukuran input 4x4, pada masing-masing 4 angka pada input operasi mengambil nilai maksimalnya dan membuat ukuran *output* baru menjadi 2x2.

3.9.3 Fully-Conected Layer

Fully-conected layer adalah sebuah lapisan dimana seua *neuron* aktivasi dari lapisan sebelumnya terhubung semua dengan *neuron* di lapisan selanjutnya sama seperti halnya dengan *neural network biasa.* Pada dasarnya lapisan ini biasanya digunakan pada MLP (*multi layer perceptron*) yang mempunyai tujuan untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear (Nurhikmat, 2018).

Setelah melalui proses konvolusi, sebuah citra menjadi nilai input pada *fully-connected layer.* Pada lapisan ini semua nilai yang sebelumnnya berbentuk matriks berubah menjadi sebuah matriks satu dimensi, atau yang lebih di kenal sebagai vektor sebelum nilai dari aktivasi tersebut dapat masuk pada *layer* ini.



Gambar 3.23. *Fully-Connected Layer*

3.9.4 ReLu Layer

ReLu layer ini mengaplikasikan fungsi aktivasi *f*(x) = (0,x). ini meningkatkan sifat nonlinearitas fungsi keputusan dan jaringan secara keseluruhan tanpa memengaruhi bidang-bidang reseptif pada *convolutional layer* (Suyanto, 2019).

3.9.5 Normalization layer

Normalization layer berguna untuk mengatasi perbedaan rentang nilai yang signifikan pada citra masukan. Para ahli telah mengusulkan banyak jenis lapis normalisasi. Namun, saat ini lapis normalisasi tidak banyak digunakan secara praktis karena dampaknya yang relative kecil, atau bahkan tidak ada sama sekali (Suyanto, 2019).

3.9.6 Loss Layer

Loss layer merupakan lapisan terakhir dalam CNN. Menentukan bagaimana pelatihan memberikan penalty atas penyimpangan antara hasil prediksi dan label. Terdapat sejumlah versi *loss fungtion*, diantaranya adalah *softmax loss* yang digunakan untuk memprediksi satu dari sejumlah kelas yang saling eksklusif,

sigmoid cross-entropy loss yang digunakan untuk memprediksi sejumlah nilai probabilitas dalam interval [0, 1] dan *Euclidean loss* yang digunakan untuk regresi nilai kontinu (Suyanto, 2019).

Loss Function yang baik adalah fungsi yang menghasilkan *error* yang diharapkan paling kecil atau paling rendah. Gambaran umum dari algoritma ini adalah meminimalkan kemungkinan log negatif dari dataset, yang merupakan ukuran langsung dari performa prediksi model.

Softmax classifier menerima nilai input sejumlah kelas pada kasus klasifikasi, kemudian mengubahnya menjadi nilai *normalized exponential* menggunakan operasi softmax:

$$q = \frac{e^{f_{yi}}}{\sum_j e^{f_{ji}}} \tag{3.5}$$

Dimana *fyi* adalah nilai label untuk data ke *i* dan *fji* adalah nilai *input* ke *j* pada data ke *i*. Akan dihasilkan sebuah *vector q* dengan nilai untuk masing-masing kelas yang menggambarkan tingkat keyakinan. Label dari data *input* akan ditentukan berdasarkan kelas dengan nilai keyakinan tertinggi. Untuk perhitungan besarnya *error* pada tahap pelatihan, digunakan *cross entropy loss*. Fungsi tersebut akan menghitung *error* antara nilai prediksi *q* dengan nilai sebenarnya *p* menggunakan

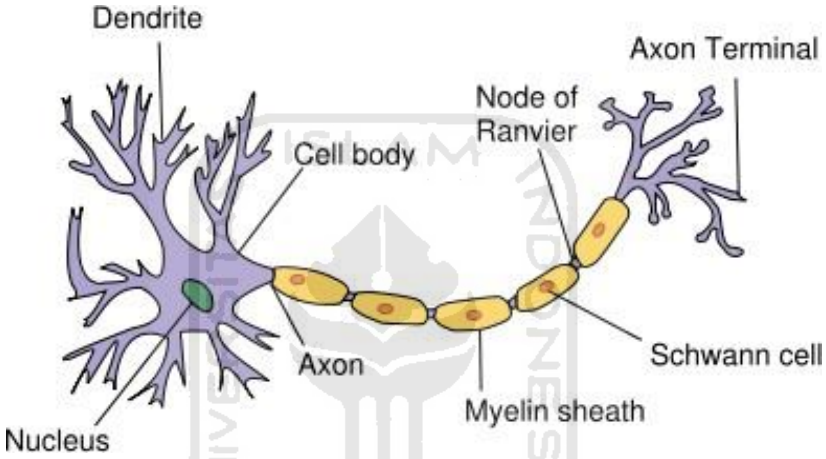
$$H(p,q) = -\sum_{i=1}^{N} p_i \log q_i \tag{3.6}$$

Fungsi softmax menerima input sejumlah kelas, sehingga diperlukan satu layer tambahan sebelum loss layer yang mentransformasi data menjadi sejumlah kelas

3.11. *Artificial Neural Network*

Artificial neural network merupakan suatu sistem yang direpresentasikan dengan jaringan syaraf biologi otak manusia untuk

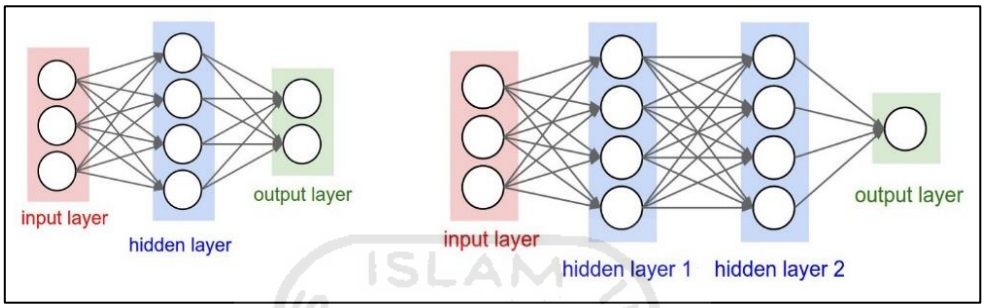
memproses suatu informasi. *Artificial neural network* adalah prosesor yang tersebar paralel yag sangat besar serta memiliki kecendrungan untuk mentimpan pengetahuan yang bersifat pengalaman dan membuatnya siap untuk digunakan. *Artificial neural network* menyerupai otak manusia dalalm dua hal, yaitu pengetahuan diperoleh jaringan melalui proses belajar dan kekuatan hubungan antar sel syaraf yang dikenal sebagai bobot-bobot sinaptik digunakan untuk menyimpan pengetahuan (Suyanto, 2014).



Gambar 3.24. Jaringan syaraf manusia

Cara kerja dari sistem syaraf diatas adalah bermula pada sinyal masuk melalui dendrit menuju *cell body*. Kemudian sinyal akan di proses didalam *cell body* berdasarkan fungsi tertentu (*summation proses*). Jika sinyal hasil proses melebihi nilai ambang batas (*treshold*) tertentu maka sinyal tersebut akan membangkitkan neuron untuk meneruskan sinyal tersebut. Sedang jika dibawah nilai ambang batasnya maka sinyal tersebut akan dihalangi (*inhibited*). Kemudian sinyal yang diteruskan akan menuju ke axon dan akhirnya menuju ke neuron lainnyamelewati *synapse* (Nurhikmat, 2018).

Lapisan *artificial neural network* biasanya terdiri atas 3 *layer*. Pertama adalah Input layer berfungsi untuk menerima input atau informasi dari lingkungan luar. Kedua adalah *hidden layer* berfungsi meneruskan informasi dari hasil input untuk diteruskan ke layer berikutnya. Ketiga adalah *output layer* berfungsi untuk menerima hasil dari *output hidden layer* dan mengirimnya kepada pemakai (Dharma et al., 2011).



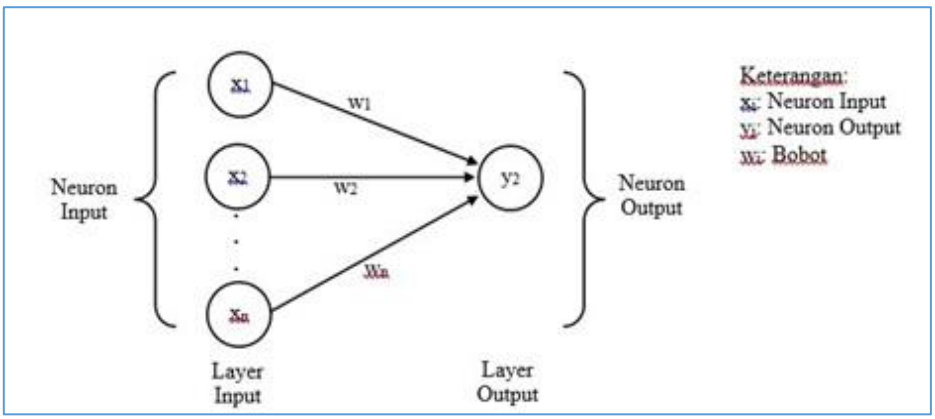
Gambar 3.25. Input layer, *Hidden layer* dan *Output* layer.

3.11.1. Arsitektur Jaringan

Arsitektur jaringan syaraf tiruan merupakan suatu susunan komponen layer dan neuron yang terhubung dengan nilai bobot pada input, *hidden* dan *output* layer. Secara garis besar arsitektur *ANN* terdiri dari tiga dasar yaitu *single layer feedforward network*, *multilayer feedforward network*, *recurrent network* dan (Suyanto, Artificial Intelligence Searching, Reasining, Planning dan Learning (Revisi Kedua), 2014).

1. Single Layer Feedforward Network

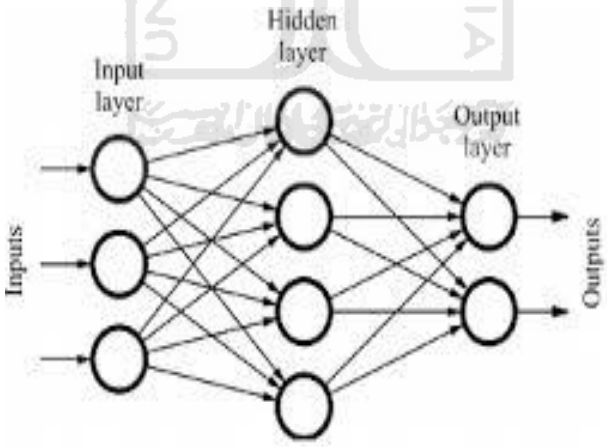
Arsitektur *Single layer feedforward network* adalah jarigan *neuron* yang diorganisasikan dalam bentuk lapisan-lapisan. Pada bentuk jaringan berlapis yang paling sederhana, hanya terdapat *input layer* dengan *node* sumber yang terproyeksi ke dalam *output layer* daari *neuron*, tetapi tidak sebaliknya. Pada Gambar 3.17 dapat dilihat terdapat tiga *node* pada *input layer* dan satu *output layer* (Suyanto, Artificial Intelligence Searching, Reasining, Planning dan Learning (Revisi Kedua), 2014).



Gambar 3.26. *Single layer feedforward network*

2. Multilayer Feedforward Network

Multi-layer feedforward nedwork adalah Jaringan dengan satu atau lebih lapis tersembunyi (*hidden layer*), dengan *computation nodes* yang berhubungan disebut *hidden neurons* atau *hidden units*. Pada Gambar 3.18 dapat dilihat terdapat tiga *node* pada *input layer*, empat *hidden* neurons dan dua *output neurons* (Suyanto, Artificial Intelligence Searching, Reasining, Planning dan Learning (Revisi Kedua), 2014).

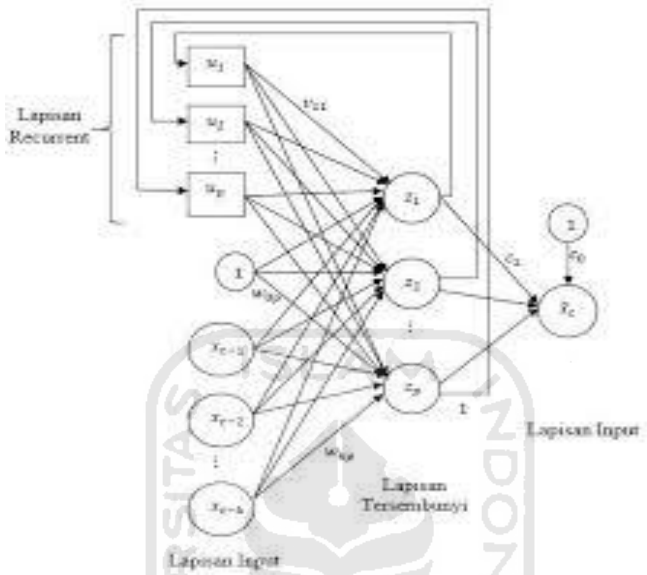


Gambar 3.27. *Multi-layer feedforward network*

3. Recurrent Network

Reccurent network adalah jaringan yang mempunyai minimal satu *feedback loop*. Sebagai contoh, suatu *reccurent network* bisa terdiri dari satu lapisan neuron tunggal dengan

masing-masing neuron memberikan kembali *output*nya sebagai *input* pada semula *neuron* yang lain, seperti diilutrasikan oleh Gambar 3.19 (Suyanto, Artificial Intelligence Searching, Reasining, Planning dan Learning (Revisi Kedua), 2014).



Gambar 3.28. *Recurrent network* (Adriyani, 2019)

3.11.2. Fungsi Aktivasi

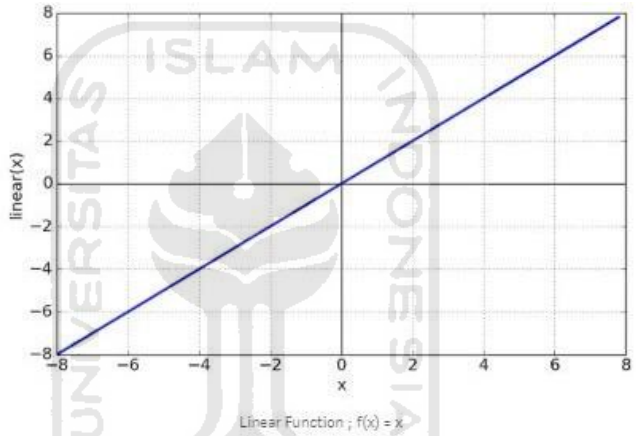
Proses yang terjadi pada fungsi aktivasi pada dasarnya mengikuti cara kerja neuron pada *neural network*, yaitu mengendalikan bagaimana sinyal mengalir dari satu *layer* ke *layer* berikutnya. Sinyal dapat dipropagasi secara efisien karena sinyal pada *output* yang terkait terkait dengan referensi sebelumnya akan mengaktifkan banyak neuron lain. Salah satu jenis fungsi aktivasi yang paling sering digunakan pada algoritma CCN yaitu fungsi aktivasi ReLU (*rectified linear unit*) karena fungsi aktivasi ini dapat mempercepat proses komputasi (Primartha, 2018).

Fungsi aktivasi merupakan fungsi yang menggambarkan hubungan antara tingkat aktivitas internal (*summation function*) yang mungkin berbentuk linear ataupun *non-linear*. Fungsi ini bertujuan untuk menentukan apakah *neuron* diaktifkan atau tidak (Nurhikmat, 2018).

(Sena, Pengenalan *Deep Learning* Part 1: *Neural Network*, 2018) dalam arikelnya menjelaskan beberapa fungsi aktivasi yang sering digunakan dalam *artificial neural network* adalah sebagai berikut:

1. Aktivasi linear

   Aktivasi linear merupakan fungsi yang memiliki nilai output sama dengan nilai *input* nya. Hal ini berkaitan dengan, jika sebuah *neuron* menggunakan *linear activation*, maka keluaran dari *neuron* tersebut adlaah *weighted sum* dari *input* + bias.



Gambar 3.29. Aktivasi linear

2. Aktivasi *sigmoid*

   aktivasi *sigmoid* merupakan fungsi nonlonear. Masukan untuk fungsi aktivasi ini berupa bilangan real dan output dari fungsi tersebut memiliki range antara 0 sampai 1.

$$A = \frac{1}{1 + e^{-x}} \tag{3.7}$$

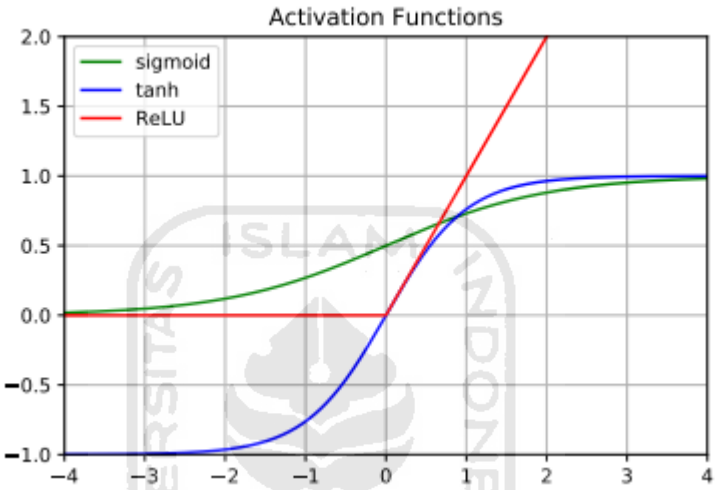
3. Aktivasi *tanh*

   Fungsi aktivasi Tanh merupakan fungsi nonlonear. Masukan untuk fungsi aktivasi ini berupa bilangan real dan output dari fungsi tersebut memiliki range antara -1 sampai 1

   $tanh(x) = 2\sigma(2x)-1$ (3.8)

4. Aktivasi *ReLu*

   Aktivasi *ReLu* (*rectified linear unit*) melakukan "*treshold*" dari 0 hingga *infinity*. Fungsi ini menjadi salah satu fungsi yang populer saat ini



Gambar 3.30. *Sigmoid*, *tanh* dan *Rectified linear unit* (dalam *Neural Network)*

3.12. *Tensorflow*

Tensorflow adalah *framework machine learning* yang bekerja dalam skala besar dan dalam environment yang heterogeneous. *tensorflow* digunakan untuk melakukan eksperimen model deep learning, melatih model pada dataset yang berukuran besar, dan membuatnya layak diproduksi.

Terdapat beberapa *framework* yang sudah didesain untuk membantu dalam menjalankan algoritma *deep learning*, salah satu nya yaitu *tensorflow*. *tensorflow* merupakan *software* atau *framework open source* yang dapat melakukan komputasi numerik yang rumit dengan menggunakan *data flow graphs*. *tensorflow* dikembangkan oleh *Google Brain Team*. *tensorflow* dapat melakukan komputasi secara paralel sehingga proses

komputasi dapat dilakukan dengan lebih cepat. *tensorflow* telah melengkapi beberapa layanan yang dimiliki Google seperti *Gmail*, Google *Photos*, *Speech recognition* dan *Google Search* (Primartha, 2018).

Tensorflow sangat populer karena dapat menggunakan sekeluruhan sistem pada komputer, Tepatnya *Tensorflow* dapat menggunakan komputasi CPU dan GPU. Dan bahkan termasuk pada perangkat *mobile* seperti *smartphone,* dan perangkat pendukung *IoT (Internet of Things)* seperti *Raspbery Pi* dan *Arduino.*

3.12.1. *Object Detection* API

Object detection API, Merupakan salah satu *application programming interface*, yang merupakan salah satu *interface* di sediakan oleh *tensorflow* yang di kembangkan oleh beberapa peneliti yang berkontribusi pada perkembangan *tensorflow,* sehingga *library tensorflow* ini dapat di gunakan oleh kalangan luas dan dapat terus berkembang.

Membangun sebuah model *machine learning* yang akurat sehingga mampu melokalkan dan mengidentifikasi berbagai macam objek dalam gambar menjadi tantangan utama dalam *computer vision* (Hang, J., Rathod, V., Chow, D., Sun, C., Zhu, M., Tang, M., et al, 2017). Pilihan model *object detection* yang dapat dilatih antara lain yaitu:

1. *Single shot multibox detector (SSD) with MobileNet*
2. *SSD with Inception V2*
3. *Region-Based Fully Convolutional Network (R-FCN) with Resnet*
4. *Faster RCNN with Resnet 101*
5. *Faster RCNN with Inception Resnet V2*

Object detection API, Salah satu *API* yang di sediakan oleh beberapa peneliti yang dapat di gunakan untuk melakukan *training* sebuah model pendeteksi objek yang di definisikan oleh peneliti

sendiri. Sehingga hasil yang di dapat adalah dapat melakukan pendeteksian objek yang sudah di tentukan pada sebuah citra digital.

Tabel 3.1. Definisi Operasional Variabel

Model	**Kecepatan**	***COCO mAP***	***Output***
ssd_mobilenet_v1_coco	Cepat	21	Kotak
ssd_inception_v2_coco	Cepat	24	Kotak
rfcn_resnet101_coco	Sedang	30	Kotak
faster_rcnn_resnet101_coco	Sedang	32	Kotak
faster_rcnn_inception_resnet_v2_atrous_coco	lambat	37	Kotak

Pada tabel 3.1 menunjukan informasi mengenai model-model yang sesuai dengan file konfigurasi yang digunakan untuk melatih model pada direktori sampel / konfigurasi, kecepatan masing-masing model, kinerja detektor pada data COCO yang diukur dengan ukuran *COCO mAP* (semakin tinggi nilainya menunjukan bahwa semakin baik kinerja detector), dan juga jenis *output* yang dimana pada saat ini hanya berupa kotak. (Guadarma, S., & Chow, D. J., 2020).

BAB IV
METODOLOGI PENELITIAN

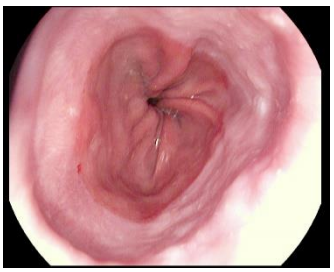
4.1. Populasi dan Sample

Populasi dari penelitian ini adalah seluruh gambar endoskopi organ tubuh. Sedangkan sampel yang digunakan dalam penelitian ini adalah gambar endoskopi organ tubuh pada saluran pencernaan (*Gastrointestinal*) yaitu *esophagitis, polyps, normal-pylorus, dyed-lifted-polyps* dan *dyed-resection-margins.* Gambar-gambar tersebut di jadikan menjadi satu dalam dataset yang bernama Kvasir. Masing-masing kategori berjumlah 240 gambar untuk data training dan 60 gambar untuk data testing, sehingga jumlah keseluruhan gambar berjumlah 1.500 gambar.

4.2. Definisi Operasional Variabel

Variabel yang di gunakan pada penelitian ini di tampilkan dalam tabel 4.1 berikut tentang penjelasan dan definisi operasional penelitian:

Tabel 4.1 Definisi operasional variabel

Gambar GI	Nama Variabel	Definisi Operasional Variabel
	*Esophagitis*	*Esophagitis* adalah peradangan pada lapisan kerongkongan. *Esophagitis* merupakan organ berbentuk pipa yang menyalurkan makanan dari mulut ke lambung. *Esophagitis* dapat menimbulkan rasa sakit dan kesulitan saat menelan serta rasa perih di dada. Bila tidak ditangani, *esophagitis* dapat merusak jaringan esopagus sehingga menimbulkan luka atau penyempitan pada kerongkongan

		dan juga dapat menyebabkan penyakit *Barrett's esophagus*, yang akan meningkatkan risiko terjadinya kanker kerongkongan.
	*Polyps*	*Polyps* adalah sebuah jaringan abnormal dan memiliki tangkai yang tumbuh di dalam tubuh. *Polyps* sering disalah artikan sebagai tumor brbahaya. Padahal *polyps* merupakan tumor jinak, bukanlah sebuah kanker. Namun, *polyps* harus tetap diwaspadai karena berupa jaringan yang tumbuh secara abnormal. Tidak menutup kemungkinan keadaan dapat berkembang menjadi suatu yang ganas berupa kanker.
	*Normal-pylorus*	*Pylorus* adalah saluran yang membawa makanan dan minuman dari lambung ke usus 12 jari. Bila terjadi penyempitan secara terus-menerus maka makanan dan minuman dari lambung tidak bisa memasuki usus 12 jari. Keadaan ini menyebabkan bayi mengalami muntah yang menyembur, dehidrasi, turun berat badan dan merasa lapar setiap hari. *Pylorus* hanya terjadi pada 2 hingga 3

		bayi dari 1000 kelahiran. Keluhan biasanya muncul saat bayi berusia 2-8 minggu, tetapi bisa juga menimbulkan keluhan setelah bayi berusia 6 bulan.
	*Dyed-lifted-polyps*	*Dyed-lifted-polyps* merupakan pengangkatan *polyps.* Pewarna-pewarna ditambahkan untuk memudahkan akurasi dalam identifikasi *margin polyps*.
	*Dyed-resection-margins*	*Dyed-resection-margins* keadaan setelah pengangkatan *polyps,* apakah pengangkatan *polyps* benar-benar hilang atau tidak.

4.3. Metode Pengumpulan Data

Penelitian ini menggunakan data sekunder yang diakses melalui *website* https://www.kaggle.com/dineshm2/the-kvasir-dataset-medical-images/data pada tanggal 4 Juli 2020. Data dikumpulkan pada tahun 2019 berupa data gambar dari hasil endoskopi organ tubuh pada saluran pencernaan (*Gastrointestinal*) yaitu *esophagitis*, *polyps*, *normal-pylorus*, *dyed-lifted-polyps* dan *dyed-resection-margins.* Gambar-gambar tersebut di jadikan menjadi satu dalam dataset yang bernama Kvasir dengan total keseluruhan 5.000 gambar. Dari keseluruhan gambar data yang di gunakan untuk *training* dan *testing* adalah sebanyak 1.500 gambar serta 25 gambar untuk validasi.

4.4. Metode Penelitian

Metode yang digunakan pada penelitian ini adalah metode *deep learning* yang menggunakan algoritma *convolutional neural network* (CNN). *Deep learning* merupakan pengembangan dari MLP (*Multi Layer Preceptron)* yang berfungsi untuk melakukan klasifikasi citra gambar. Metode *deep learning* dengan algoritma *convolutional neural network* digunakan untuk mengklasifikasikan gamba maupun video, serta *tensorflow* API untuk mengidentifikasi lokasi citra.

4.5. Perangkat Penelitian

Perangkat hardware seperti laptop/computer untuk melakukan proses *training* (*Machine learning*) dengan spesifikasi tinggi untuk mendapatkan suatu model yang dapat memprediksi data baru. Berikut ini adalah spesifikasi perangkat yang digunakan.

- Laptop : Asus VivoBook A442U
- Windows : Windows 10
- Processor : Intel Corei7-7500U, up to 3.5GHz CPU
- GPU : Nvidia Geforce 940mx
- RAM : 8GB DDR3.
- HDD : 1 Tb

Selanjutnya adalah beberapa *software* dan *library* yang di gunakan pada penelitian ini adalah:

- Python

  merupakan aplikasi (*Software*) bahasa pemrograman yang digunakan untuk menjalankan *script* perintah. Banyak digunakan oleh *data analyst* dan *progammer*
- Numpy

  Merupakan *library* pada python untuk melakukan operasi matriks dan *vector*.
- *Tensorflow*

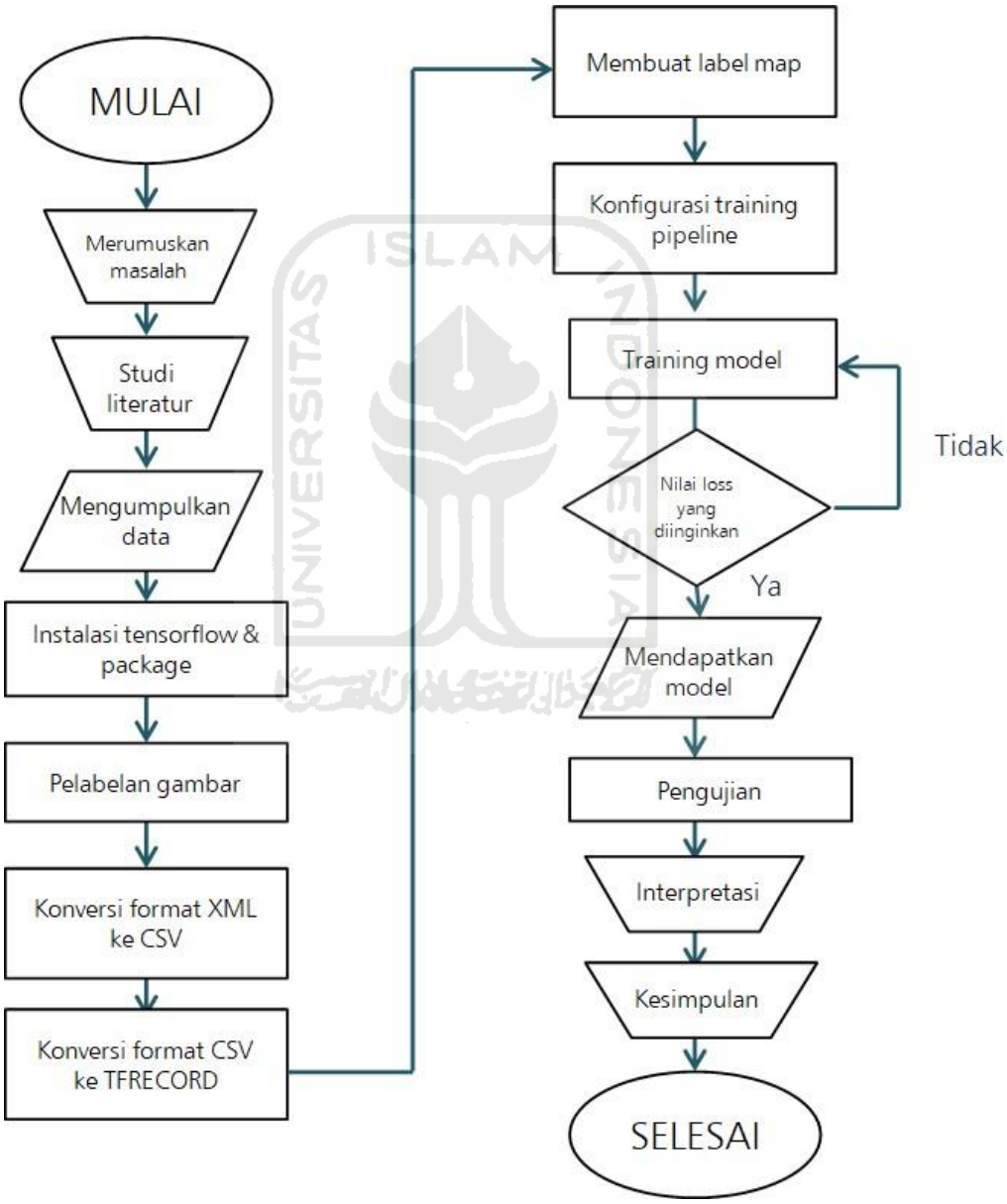
  Merupakan *framework* pada python untuk melakukan *deep learning.*

- OpenCV

  Merupakan *library* pada python untuk melakukan *deep learning*.
- LabelImg

  Modul pada python yang digunakan untuk melakukan *labeling* pada gambar.

4.6. Tahapan Penelitian



Gambar 4.1. Alur Penelitian

Pada penelitian ini adapun tahap-tahap yang di lakukan adalah sebagai berikut:

1. Mulai, penelitian ini dimulai dengan menetukan topik. Topik yang penulis gunakan adalah deteksi penyakit pada saluran pencernaan (*gastrointestinal*). Selanjutnya mengumpulkan informasi tentang *gastrointestinal*,
2. Merumuskan masalah, rumusan masalah yang penulis gunakan berdasarkan melihat dari pekembangan teknologi saat ini khususnya pada bidang kesehatan.
3. *Studi literature*, penulis mencari junal atau penelitian yang terkait dengan topik yang digunakan untuk menjadikannya sebagai referensi.
4. Dalam pengumpulan data penulis melakukan secara sekunder yang diakses melalui *website* https://www.kaggle.com/dineshm2/the-kvasir-dataset-medical-images/data, kemudian penulis menggunakan data untuk *training* dan *testing* adalah sebanyak 2.400 gambar serta 24 gambar untuk validasi.
5. Selanjutnya instalasi tensorflow dan package, penulis melakukan installasi untuk tensorflow yang digunakan adalah *tensorflow* versi 1.14.0. serta installasi *package* yang dibutuhkan seperti *pillow*, *numpy* dan *openCV*
6. Kemudian dalam pelabelan gambar penulis menggunkan *labelImg*. Pelabelan ini berdasarkan kategorinya yaitu *esophagitis*, *polyps*, *normal-pylorus*, *dyed-lifted-polyps* dan *dyed-resection-margins.*
7. Setelah melakukan pelabelan maka muncul folder baru dengan format XML. Selanjutnya melakukan konversi file XML menjadi file CSV
8. Kemudian format CSV dikonversi kedalam bentuk TFRecord. Selanjutnya penulis membuat file baru yang berisi *Number map* yang sesuai dengan kategorinya.
9. Setelah itu penulis melakukan konfigurasi untuk merancang arsitektur CNN.

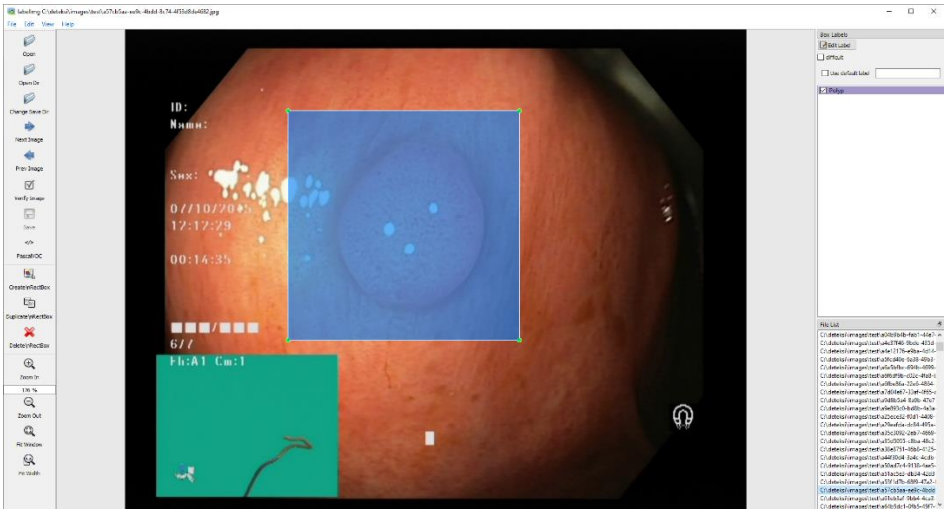
10. Selanjutnya melakukan proses *training*, proses *training* akan membutuhkan waktu yang cukup lama tergantung dengan spesifikasi perangkat yang digunakan. Proses *training* dapat dihentikan apabila nilai *loss* sudah tidak menunjukkan perubahan yang signifikan.
11. Hasil dari *training* berupa model yang akan di *export* ke dalam file *frozen inference graph*.
12. Kemudian model yang didapat akan di uji dengan cara menggunakan data gambar validasi yang mana data tersebut tidak diikutsertakan dalam proses *trining.*
13. Terakhir penulis membuat interpretasi serta kesimpulan dari hasil klasifikasi yang telah diuji sebelumnya.

BAB V
ANALISIS DAN PEMBAHASAN

5.1. Rancangan Sistem

Rancangan sistem dalam penelitian ini dimulai dari pengumpulan data. Data yang penulis gunakan yaitu data sekunder berupa citra *endoskopi* pada *gastrointestinal* yang semua kategori di dataset penulis gunakan yaitu *esophagitis*, *polyps*, *normal-pylorus*, *dyed-lifted-polyps* dan *dyed-resection-margins*. Dataset dapat *Dataset* dapat diakses melalui https://www.kaggle.com/dineshm2/the-kvasir-dataset-medical-images/data.Setelah *dataset* diunggah, penulis melakukan seleksi data untuk digunakan dalam penelitian yang meliputi data *train* dan data *test* di dalam dataset baru, dengan perbandingan 80%-20% yaitu sebanyak 1.200 untuk total data *train* dan 300. Penulis juga menyiapkan data untuk validasi dengan jumlah 25 gambar untuk masing-masing kategori. Data validasi merupakan data yang tidak dilibatkan pada proses *machine learning*.

Dataset baru itu disimpan dalam folder *Images* → *train* dan *test* kemudian dilakukan pelabelan gambar. Pelabelan gambar dilakukan secara manual menggunakan aplikasi labellimg. Pemberian label diberikan sesuai dengan kategori gambar dengan memberikan tanda berbentuk kotak. Hasil output dari proses pelabelan merupakan file dengan format XML.



Gambar 5.1. Proses Label pada gambar.

Hasil *output* pelabelan berupa file dengan format XML *(Xtensible Markup Language*) dan dikonversi dalam bentuk *file csv (comma separated value)*. Konversi format file memiliki tujuan untuk menggenerate berkas *TFRecord*.

Hasil konversi ke *csv* yang berisi 5 variabel untuk merangkum dari keselurunal label yang sudah di buat yaitu *'filename', 'width', 'height', 'class', 'xmin', 'ymin', 'xmax', 'ymax'*.

	A	B	C	D	E	F	G	H
1	filename	width	height	class	xmin	ymin	xmax	ymax
2	a0471134-2faf-40bf-b8d8-ba805fb72b74.jpg	720	576	Dyed-resection-margins	361	230	667	526
3	a0658788-e7f9-4352-9cf7-116ac9ae472b.jpg	720	576	Polyp	289	48	644	399
4	a1179a37-8cee-468c-8ac7-8ada22612e77.jpg	720	576	Polyp	184	140	490	458
5	a209bfe2-8271-46c8-9f78-79d4918734ba.jpg	720	576	Polyp	170	15	592	306
6	a267d728-3aa5-4339-a47d-84145a2b63af.jpg	720	576	Polyp	217	197	439	487
7	a29eafda-dc84-495a-84f8-9980a506b9b9.jpg	720	576	Polyp	129	350	417	540
8	a35579a6-9ab6-4ab7-9a26-a2b2eee95049.jpg	720	576	Polyp	188	305	418	535
9	a359507a-7a7a-4176-93b2-96358509f4fd.jpg	720	576	Dyed-resection-margins	347	87	676	532
10	a38e8751-46b8-4125-8303-4b22c3d02742.jpg	720	576	Polyp	172	237	525	468
11	a4659885-bd5e-4cd1-8506-6ba7d22a6f23.jpg	720	576	Dyed-resection-margins	78	294	391	543
12	a5095f5e-753b-4e31-b737-e424db22e9ad.jpg	720	576	Polyp	96	202	439	441
13	a510691e-08f6-4327-9091-d4c2ebce5b38.jpg	720	576	Dyed-resection-margins	419	175	600	451
14	a5165759-a282-4107-99a7-ac5b84f656c5.jpg	720	576	Polyp	118	262	381	518
15	a53f1d7b-68f9-47a2-b696-bf39661440c8.jpg	720	576	Polyp	261	325	550	555

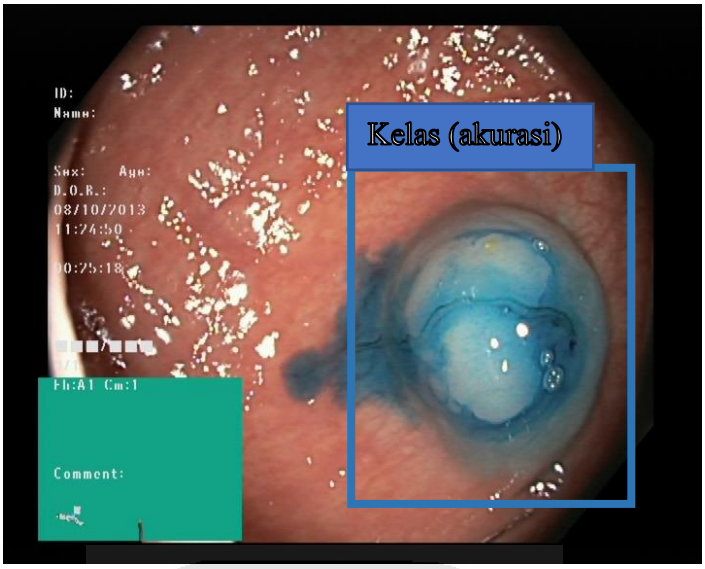
Gambar 5.2. File csv

Selanjutnya adalah mendefinisikan *labelmap*, dimana *labelmap* ini merupakan definisi kategori objek yang akan digunakan yang berbentuk numerik, objek yang didefinisikan pada *numbermap* harus sesuai dengan kelas atau kategori yang digunakan untuk klasifikasi. Untuk *numbermap* dapat dilihat pada gambar 5.3.

```
item {
        id: 1
        name: 'Polyp'
}
item {
        id: 2
        name: 'Esophagitis'
}
item {
        id: 3
        name: 'Normal-pylorus'
}
item {
        id: 4
        name: 'Dyed-lifted-polyps'
}
item {
        id: 5
        name: 'Dyed-resection-margins'
}
```

Gambar 5.3. *Labelmap*

Pada penelitian ini penulis menggunakan 8 kategori yaitu item label dengan nama item *polyp* dengan id= 1, item *eshopagitis* dengan id= 2, item *ulcerative-colitis* dengan id= 3, item *dyed-lifted-polyps* dengan id= 4, item *dyed-resection-margins* dengan id= 5, item *normal-cecum* dengan id= 6, item *normal-pylorus* dengan id= 7, dan item *normal-z-line* dengan id= 8. Jika objek yang akan dideteksi lebih dari 9 objek maka *name* dan *id* juga harus menyesuaikan sebanyak jumlah kelasnya.

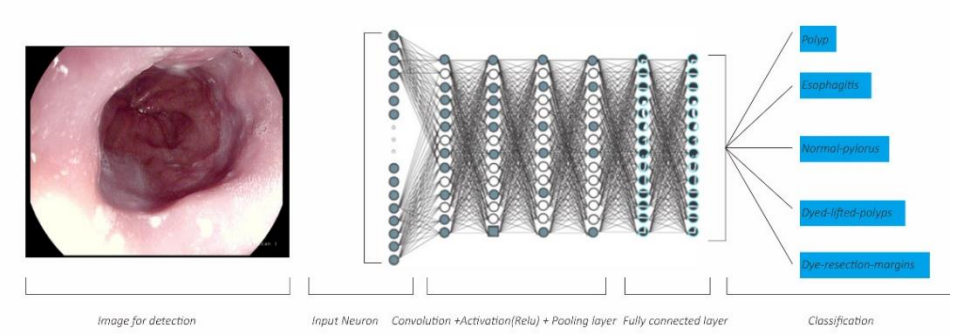


Gambar 5.4. Rancangan *output*

Pada Gambar 5.4 diatas merupakan rancangan dari *output* sistem yang akan dibuat oleh peneliti. Pada gambar diatas dapat ketahui bahwa untuk gambar validasi yang diuji terdapat kotak berwarna yang menunjukkan hasil klasifikasi dan lokasi objek pada gambar tersebut. Informasi yang terdapat pada *output* tersebut adalah kelas dan nilai akurasi yang didapat.

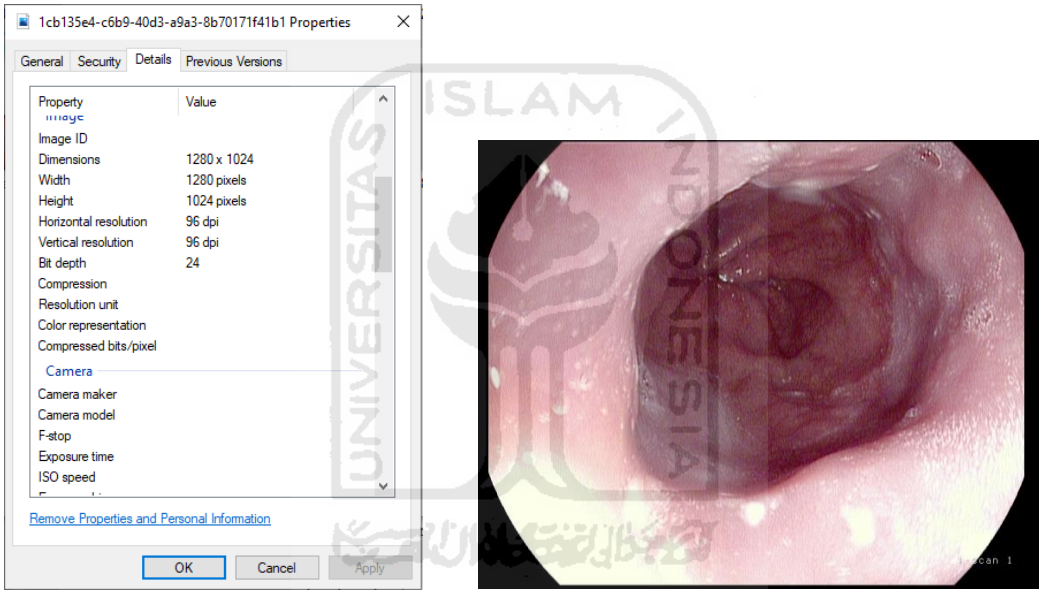
5.2. Sistem Jaringan

Desain arsitektur *convolutional neural network* terdiri dari beberapa bagian jaringan dalam proses pelatihan yaitu *image for detection*, *input neuron*, *convolution* + *activation* (ReLU) + *pooling layer*, *fully connected layer*, *classification* dan *detection output*.



Gambar 5.5. Arsitektur Jaringan

Pada Gambar 5.5 dapat diketahui pada *image for detection* adalah gambar yang akan di deteksi. Pada gambar tersebut ukuran untuk pelatihan sebesar 1280x1024 piksel, yang berarti gambar tersebut memiliki lebar piksel sebanyak 1280 dan Panjang 1024 piksel serta dengan warna RGB (*red*, *green*, *blue*) yaitu sebanyak 3 *channel* sehingga yang masuk kedalam layer pertama atau bagian *input neuron* sebanyak 3.932.160 neuron yang diperoleh dari perhitungan 1280x1024x3. Setiap neuron memiliki nilai parameter dimana parameter yang digunakan dalam jaringan tersebut berkisar antara 0 sampai 225.



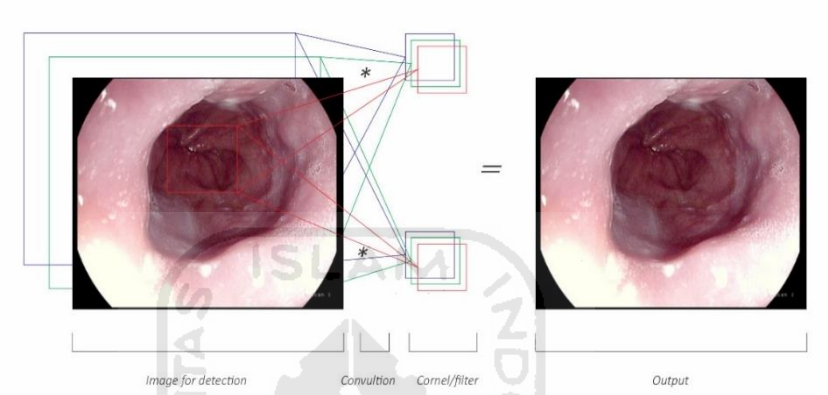
Gambar 5.6. Informasi umum pada citra

5.2.1. *Convolution layer*

Satuan terkecil dari sebuah gambar adalah pixel, sehingga sebuah gambar dapat direpresentasikan sebagai matriks. *Convolutional layer* atau proses konvolusi merupakan *layer* pertama pada proses CNN. Secara umum proses konvolusi melibatkan 3 komponen yaitu matriks gambar input, kernel (filter) dan output (*activation map*).

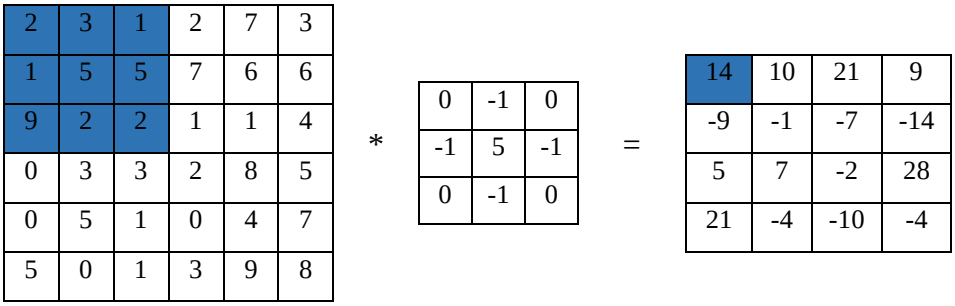
Konvolusi merupakan cara untuk mengkombinasikan dua buah deret angka yang menghasilkan deret angka yang ketiga. Dalam penelitian ini, dua buah deret angka tersebut terdapat dalam

input dan *kernel* atau *filter* sedangkan deret angka yang ketiganya adalah *output*. Input dan *kernel* atau *filter* keduanya memiliki deret angka berupa matriks. Pada *input* deret angka diperoleh berdasarkan tingkat warna yang ada pada masing-masing piksel sedangkan pada kernel atau filter deret angka disesuaikan oleh kebutuhan peneliti. Berikut ini adalah visualisasi proses pada *convolution layer*:



Gambar 5.7. Convolution Layer

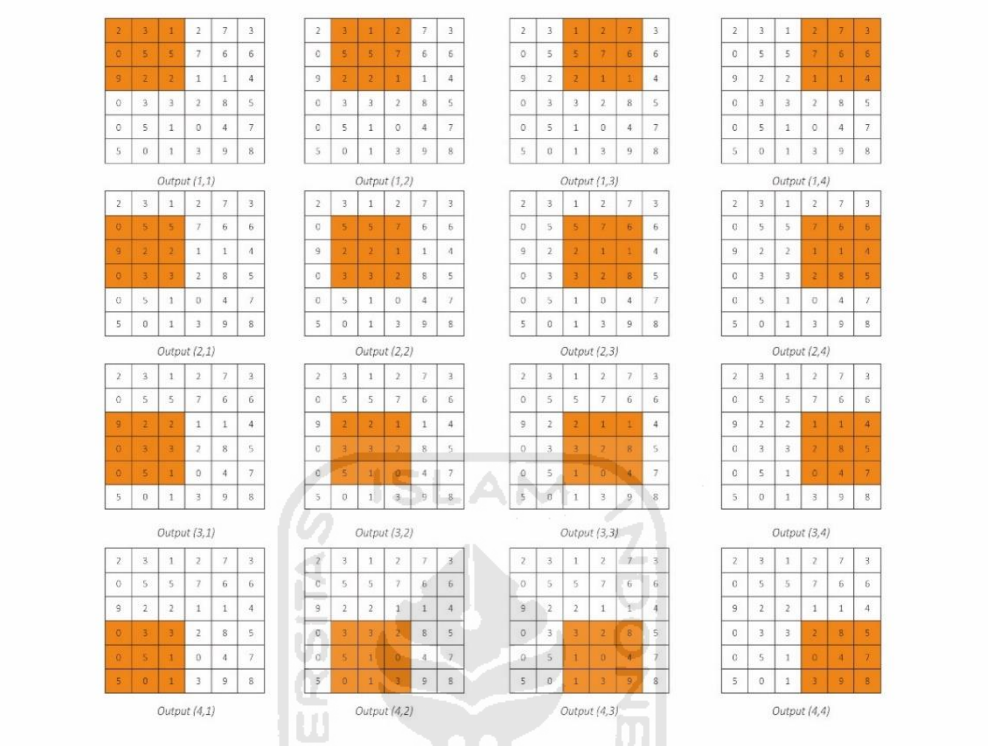
pada bagian *input,* gambar tersebut memiliki lebar 1280 dan panjang 1024. Pada Gambar 5.4 juga memiliki kedalaman, kedalaman tersebut sebesar 3 sesuai *channel* warna dasar yaitu warna merah, hijau dan biru. Untuk mengetahui cara kerja dari proses konvolusi, peneliti akan menggunakan sampel deret angka pada *input* dikarenakan keterbatasan penulisan dengan ukuran 1280x1024 maka peneliti menggunakan sampel deret angka pada *input* dengan ukuran 6x6 dan menggunakan kernel atau *filter* untuk operasi *vertical edge detection* dengan ukuran 3x3. Untuk karnel nya penulis menggunakan penajaman citra *(image sharpening)* yang bertujuan untuk melewatkan komponen yang berfrekuensi tinggi dan menekan komponen yang berfrekuensi rendah. Operasi ini menghasilkan penajaman atau memperjelas pada tepi objek, penulis ilustrasikan seperti gambar dibawah ini.



Gambar 5.8. Konvolusi

Dengan menggunakan filter 3x3 dengan *strided* atau langkah yang digunakan dalam perhitungan konvolusi tersebut adalah 1. Nilai *stride* merupakan nilai yang menentukan berapa jumlah pergeseran matriks kernel pada saat proses konvolusi berlangsung. Jika nilai *stride* yang digunakan sebanyak 1, maka matriks kernel akan bergeser sebanyak 1 piksel pada matriks *input*. Semakin kecil stride maka akan semakin detail informasi yang kita dapatkan dari sebuah input, namun membutuhkan komputasi yang lebih jika dibandingkan dengan stride yang besar. Pada ilustrasi gambar 5.8, matriks input dengan dimensi 6×6 dan dengan kernel berdimensi 3×3 akan menghasilkan *output (activation map)* dengan dimensi 4×4. Matriks kernel akan bergeser secara horizontal, dimulai dari sudut kiri atas pada matriks input hingga pada bagian sudut kanan bawah.

Proses perhitungan konvolusi tersebut dapat divisualisasikan seperti gambar dibawah ini:



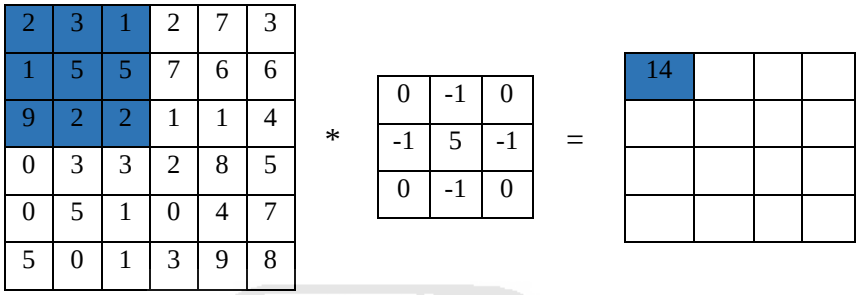
Gambar 5.9. Proses perhitungan konvolusi

5.2.2. *Activation Function*

Sebelum didapat *output* dari hasil proses konvolusi, *activation map* akan dimasukan kedalam fungsi aktivasi terlebih dahulu. Fungsi aktivasi (*activation function*) bertujuan untuk menentukan apakah suatu nilai pada neuron masih aktif atau tidak tergantung pada rentang nilai aktivasi yang digunakan. Fungsi aktivasi akan mentransformasikan nilai pada *activation map* menjadi nilai pada rentang tertentu.

Tahap ini adalah perhitungan nilai dengan *activation function* yang digunakan untuk mencari nilai non-linier pada nilai hasil konvolusi. Pada tahapan ini penulis menggunakan fungsi aktivasi ReLU (*rectified linear unit).* Rumus ReLu yang digunakan adalah $f(x) = \max(x, 0)$. Dengan x merupakan input neuron atau node.

Angka 0 pada rumus ReLU merupakan unit linier yang dikoreksi jika input kurang dari 0. Artinya, jika *input* lebih besar dari 0, *output*nya sama dengan *input*. Sehingga fungsi aktivasi ReLU memiliki rentang nilai dari 0 hingga ∞. Untuk lebih memahami ReLU dapat diilustrasikan dengan gambar dibawah ini:



Gambar 5.10. *Activation function (ReLU)*

Output(1,1) = [input(1,1) × kernel(1,1)] + [input(1,2) × kernel(1,2)] + [input(1,3) × kernel(1,3)] +[input(2,1) × kernel(2,1)] + [input(2,2) × kernel(2,2)] + [input(2,3) × kernel(2,3)] +[input(3,1) × kernel(3,1)] + [input(3,2) × kernel(3,2)] + [input(3,3) × kernel(3,3)]

= (2 x 0) + (3 x (-1)) + (1 x 0) + (1 x 0) + (5 x 5) + (5 x (-1)) + (9 x 0) + (2 x (-1)) + (2 x 0)

= 14

5.2.3. *Pooling Layer*

Pooling layer merupakan lapisan yang berguna untuk mengurangi ukuran gambar. *Pooling layer* biasanya berada setelah *convolutional layer*. sehingga *output* atau *activation map* pada hasil konvolusi akan menjadi *input* bagi *pooling layer*. Pada prinsipnya *pooling layer* terdiri dari sebuah filter dengan ukuran dan *stride* tertentu yang akan bergeser pada seluruh area *feature map*. Salah satu contoh metode dalam *pooling layer* adalah *maxpooling*. Dalam prosesnya *maxpooling* akan mengambil nilai pixel maksimum pada setiap pergeseran kernel terhadap gambar *input*. Berikut merupakan ilustrasi proses *pooling* menggunakan kernel berdimensi 2×2

dengan nilai *stride* 1.

14	10	21	9
-9	-1	-7	-14
5	7	-2	28
21	-4	-10	-4

4 x 4

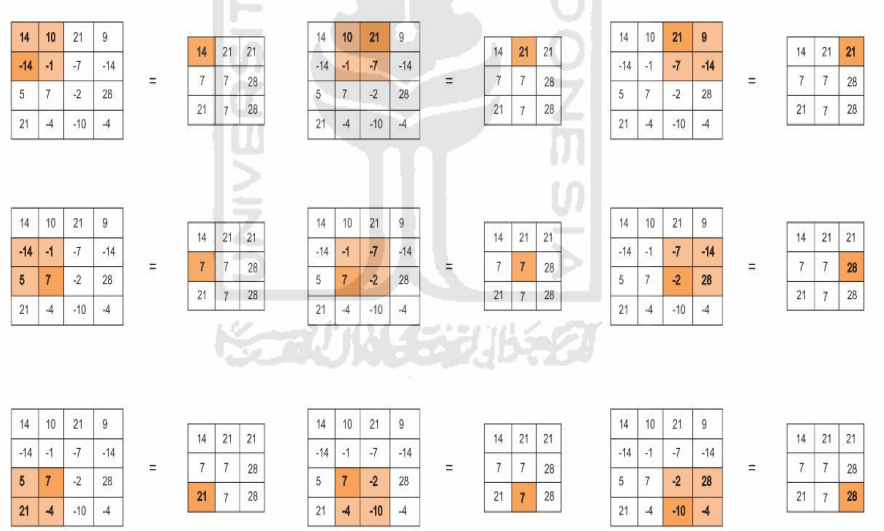
=

14	21	21
7	7	28
21	7	28

3 x 3

Gambar 5.11. *Pooling layer*

Pada Gambar 5.11 terdapat lapisan dengan ukuran 4×4 dengan *stride* 1 maka diperoleh hasil *Max pooling* dengan ukuran 3x3, yang diambil dari nilai maksimum pada setiap posisi pergerakan karnel. Proses perhitungan konvolusi tersebut dapat divisualisasikan seperti gambar dibawah ini:

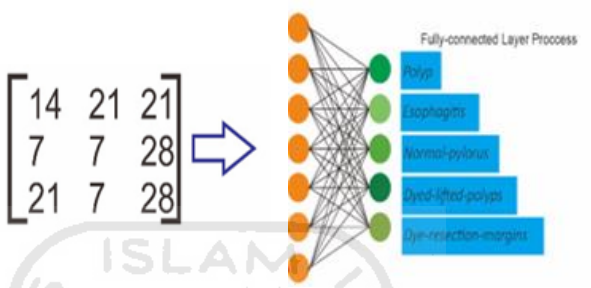


Gambar 5.12. Proses perhitungan *pooling layer*

5.2.4. *Fully Connected Layer*

Pada akhir *convolution layer* dan *pooling layer*, kemudian akan menjadi *input* untuk *flatten layer*. *Flatten layer* berfungsi untuk mengubah data yang semula dalam bentuk matriks menjadi bentuk vector. Data berupa vector tersebut kemudian akan dijadikan sebagai *input layer* pada proses *fully connected layer*. Proses pada *fully connected layer* ini komputasi jaringan syaraf tiruan mulai

digunakan jaringan umumnya menggunakan lapisan yang terhubung sepenuhnya di mana setiap piksel dianggap sebagai neuron terpisah seperti jaringan saraf biasa yang disebut sebagai *Fully Connected Layer*. *Fully Connected Layer* merupakan lapisan terakhir yang terhubung sepenuhnya yang mengandung banyak neuron sebagai jumlah kelas yang harus diprediksi.



Gambar 5.13. Proses Fully Connected Layer

Gambar 5.13 diatas merupakan prose dari proses *fully connected layer* dimana pada proses akhir komputasi ini akan menghasilkan *output* yang dimana *output* tersebut merupakan target kelas/kategori yang digunakan.

Output yang dihasilkan juga akan mempunyai nilai error, sehingga dari nilai *error* yang dipunya akan dapat mengupdate nilai bobot sampai didapat nilai error yang terkecil, disini proses *backpropagation* berjalan dimana komputasi dimulai lagi dari input layer. Proses komputasi dari input layer hingga didapat output dinamakan proses *feedforward.*

5.2.5. *Classification*

Tahapan ini merupakan tahapan klasifikasi pada bagian mana saja pada setiap piksel yang memiliki pola penyakitnya (*esophagitis*, *normal-cecum*, *polyps*, *ulcerative-colotis*, *normal-pylorus*, *normal-z-line*, *dyed-lifted-polyps* dan *dyed-resection-margins*).

5.2.6. *Detection Output*

Detection Output merupakan hasil akhir dari mendeteksi penyakit. Sesuai dengan rancangan *output*, penulis mendapatkan *detection output* dengan keterangan nama label kelas dan keterangan label gambar dengan tingkat akurasi antara 1-99%.

5.3. Konfigurasi System

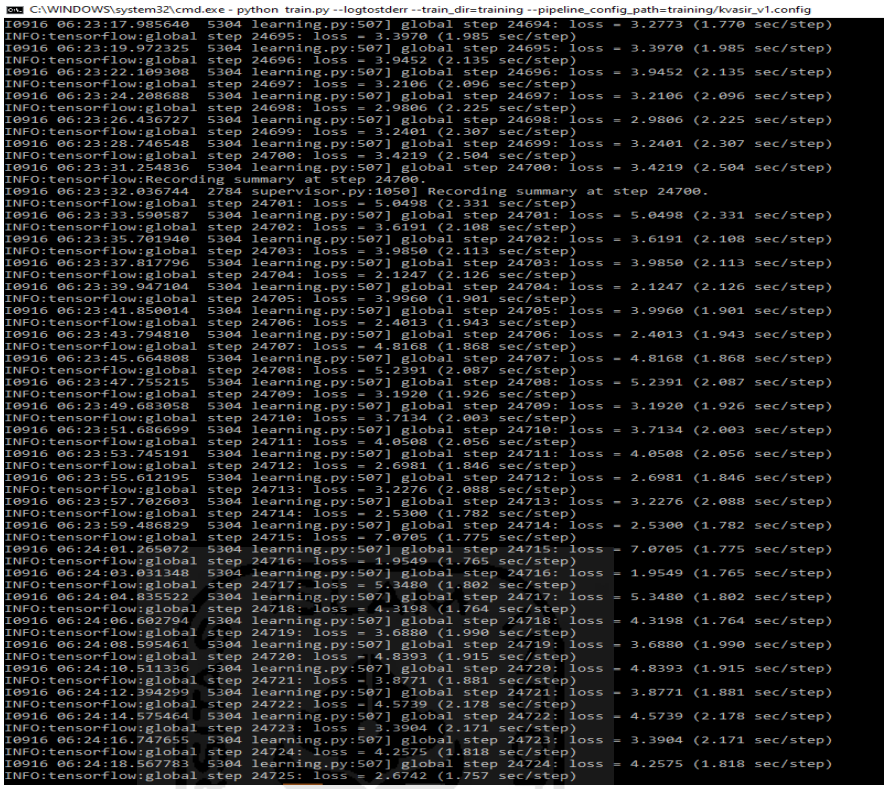
Konfigurasi system dilakukan untuk merancang arsitektur *Convolutional Neural Network* beserta parameternya seperti jumlah *step* (*epoch*), *batch size*, *kernel size*, jenis *optimizer* dan *learning rate*. Konfigurasi *pipeline* yang gunakan yaitu dari *SSD with Mobilenet v1*.

5.4. Model Hasil Training

Pembahasan model merupakan hasil implementasi dan proses pelatihan. Berikut ini akan dijabarkan mengenai model yang telah diperoleh peneliti:

5.4.1. *Training Steps*

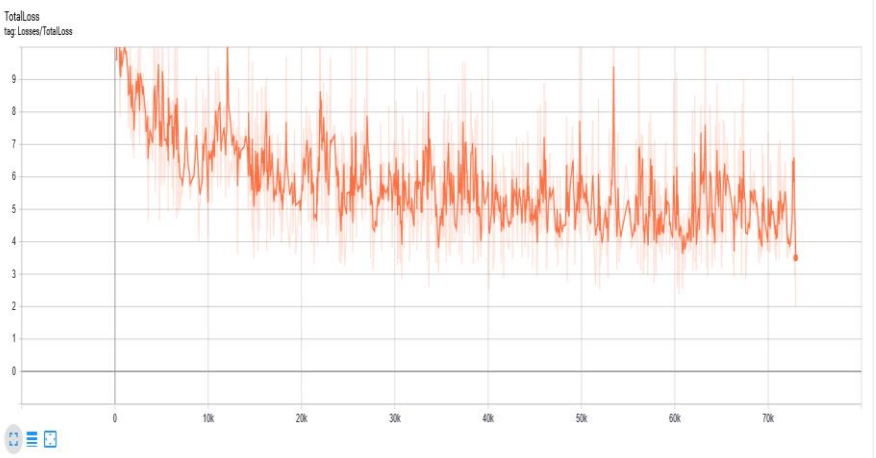
Training Steps adalah langkah dimana proses pelatihan dilakukan yang memvisualisasikan hasil pelatihan. Dengan memasukan data *train* dan data *teset* dalam bentuk *TFRecord* kedalam arsitektur CNN yang sudah dirancang sebelumnya. Pada konfigurasi system penulis telah menentukan untuk maksimal *steps* pada *training* yaitu sebanyak 100.000 *steps*, namun dapat dihentikan apabila nilai *loss* dirasa sudah cukup. Berikut ini adalah hasil pelatihan dataset:



Gambar 5.14. Proses *training*

Gambar 5.14 merupakan hasil pencatatan langkah pada saat proses pelatihan, dengan memberikan informasi nilai *loss* yang dihasilkan disetiap langkah dan setiap langkahnya diselesaikan dengan waktu detik perlangkah.

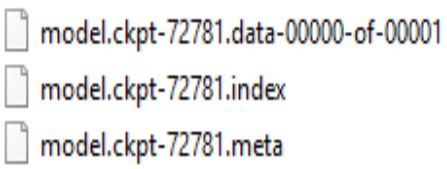
Pada saat proses *training* berlangsung, semua proses terekam dalam sebuah file. File tersebut dapat dilihat pada TensorBoard untuk melakukan visualisasi pada saat *training*. *Tensorboard* merupakan *dashboard* yang disediakan oleh *tensorflow API* untuk yang dapat memvisualisasikan berbagai macam grafik dari hasil proses *training*. Berikut ini adalah grafik *total loss* yang ditampilkan dalam *tensorBoard*:



Gambar 5.15. Grafik *total loss*

Gambar 5.14 merupakan grafik *total loss* yang dihasilkan pada saat proses pelatihan sampai dengan selesai, Pada penelitian ini, penulis melakukan proses pelatihan sebanyak 72.960 langkah dan nilai *loss* 1.98. Proses pelatihan ini membutuhkan waktu 7 hari 3 jam 26 menit 27 detik. Penulis memutuskan untuk menghentikan sementara proses *training* karena nilai *loss* tidak menunjukkan penurunan yang signifikan dan mencoba melihat hasil model yang didapat melalui data validasi.

Sebelum melakukan pengujian, penulis melakukan ekstrasi model dengan menggunakan file *checkpoint* hasil *training* pada step terakhir yang tercatat. Pada penelitian ini penulis menggunakan file *checkpoint* "model.ckpt-72781". Pada proses pelatihan, penulis melakukan proses *training* hingga pada *step* ke-72.960 dan model yang tersimpan yaitu pada *step* ke-72.781.

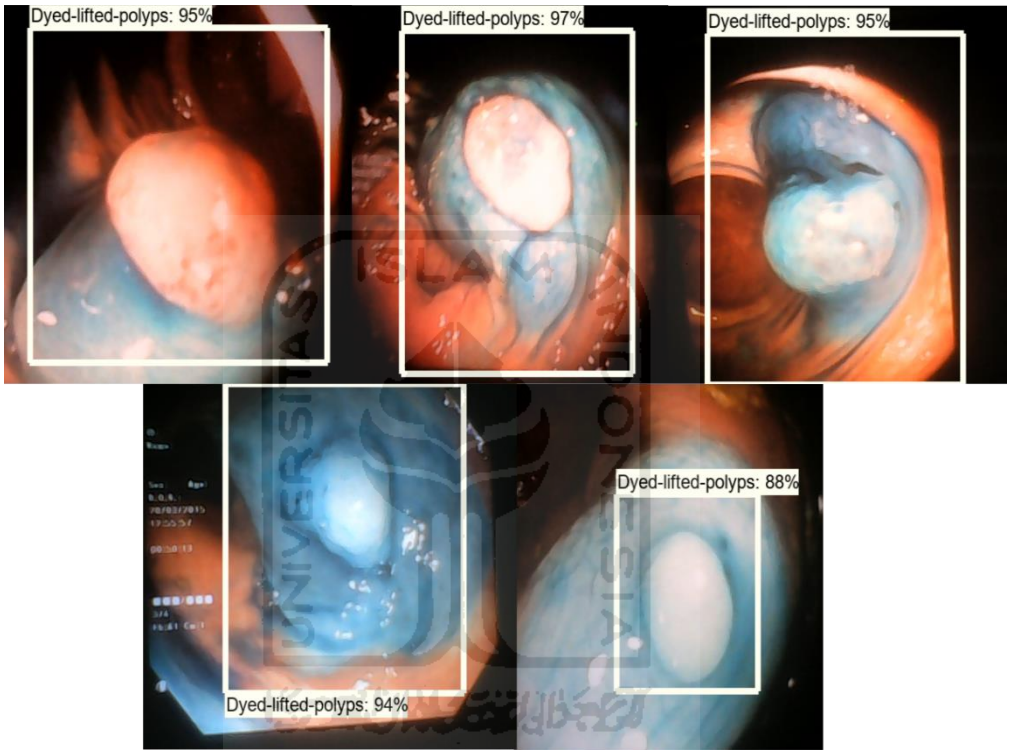


Gambar 5.16. File *checkpoint*

5.5. Hasil Deteksi

Hasil pengujian dari sebuah model yang telah di *training*. Penulis menggunakan 25 data gambar dengan kategori *esophagitis*, *polyps*, *normal-pylorus*, *dyed-lifted-polyps* dan *dyed-resection-margins* dengan masing-masing 5 gambar pada setiap kategori. Berikut merupakan hasil pengujian:

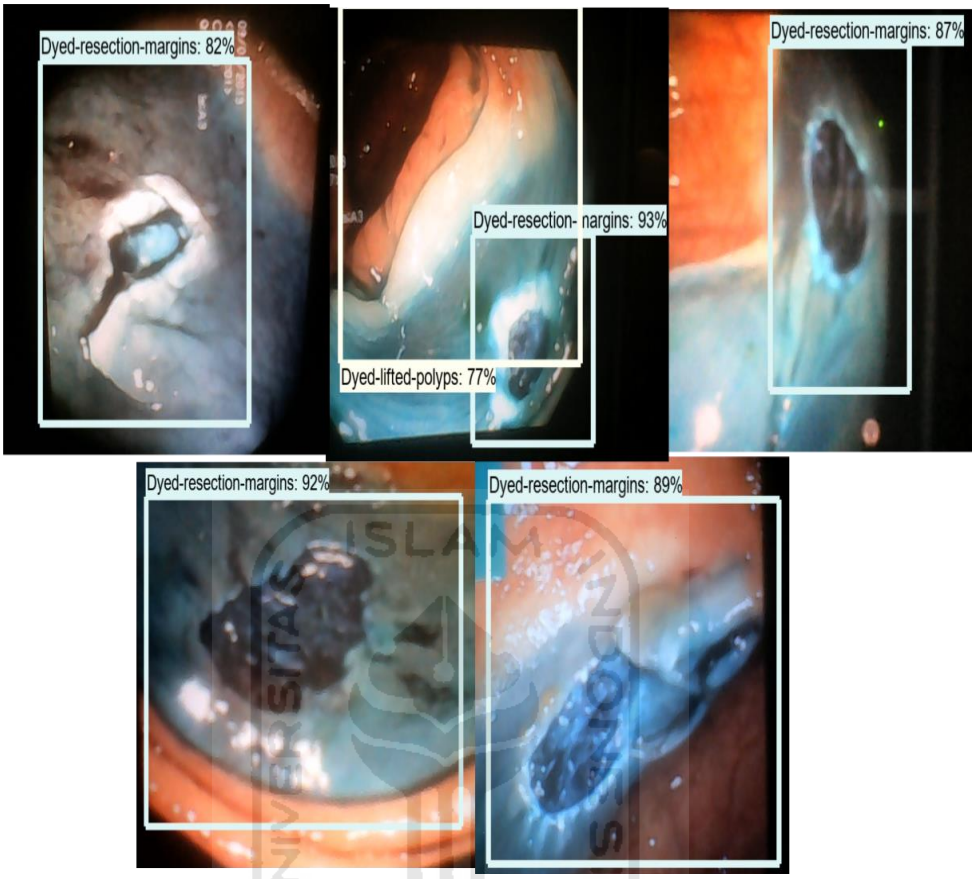
5.5.1. *Dyed-lifted-polyps*



Gambar 5.17. Hasil pengujian *dyed-lifted-polyps*

Berdasarkan hasil pengujian di atas dapat diketahui bahwa seluruh gambar dapat dideteksi dengan baik oleh model. Tingkat akurasi yang didapat yaitu sebesar 83% - 97%. Model yang didapat mampu memprediksi dan mendeteksi seluruh gambar dengan tepat.

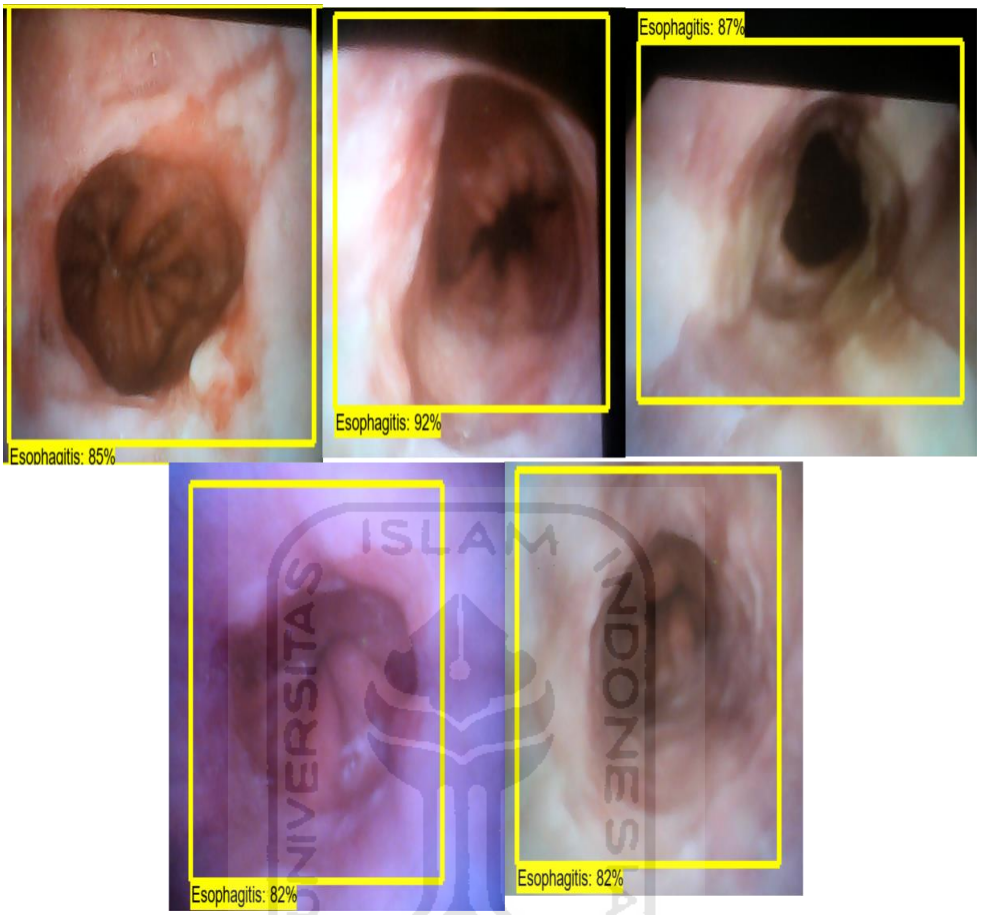
5.5.2. *Dyed-resection-margins*



Gambar 5.18. Hasil pengujian *dyed-resection-margins*

Berdasarkan hasil pengujian di atas dapat diketahui bahwa terdapat 1 gambar yang tidak dapat dideteksi dengan baik karena muncul 2 label pada gambar tersebut, tetapi 4 gambar lainnya dapat dideteksi dengan baik oleh model. Tingkat akurasi yang didapat yaitu sebesar 82% - 93%. Model yang didapat mampu memprediksi dan mendeteksi gambar dengan tepat.

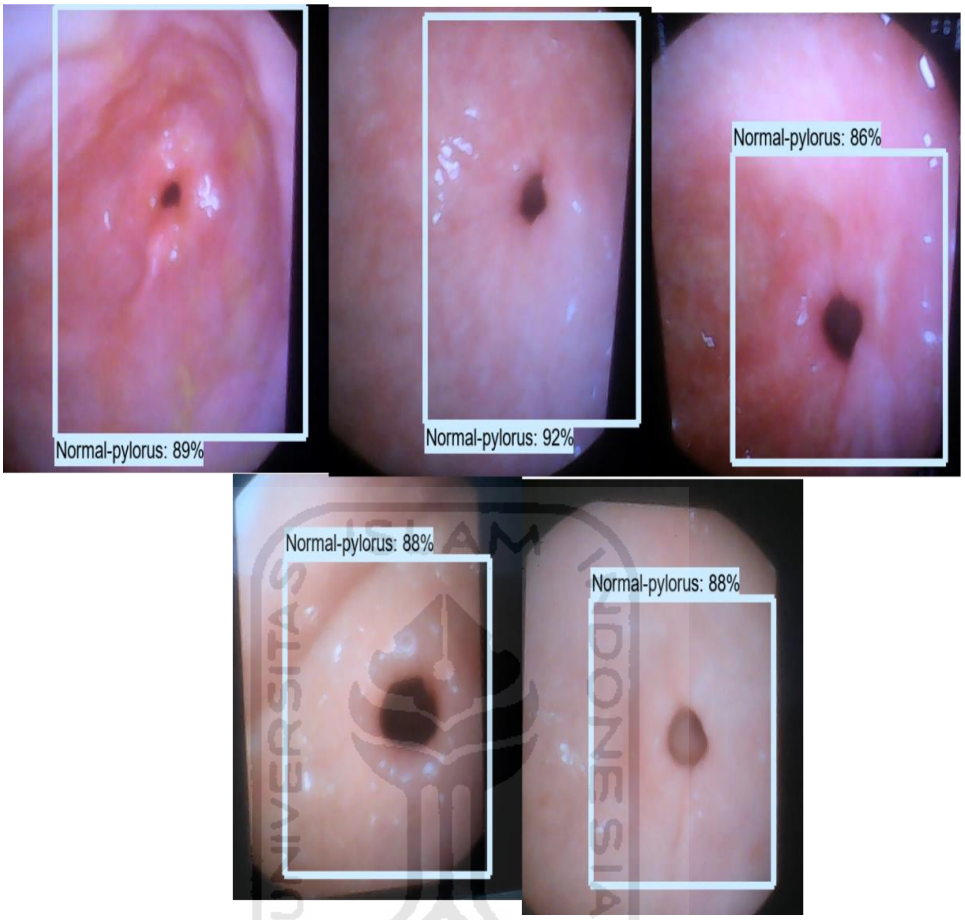
5.5.3. *Esophagitis*



Gambar 5.19. Hasil pengujian *esophagitis*

Berdasarkan hasil pengujian di atas dapat diketahui bahwa seluruh gambar dapat dideteksi dengan baik oleh model. Tingkat akurasi yang didapat yaitu sebesar 82% - 92%. Model yang didapat mampu memprediksi dan mendeteksi seluruh gambar dengan tepat.

5.5.4. *Normal-pylorus*



Gambar 5.20. Hasil pengujian *normal-pylorus*

Berdasarkan hasil pengujian di atas dapat diketahui bahwa seluruh gambar dapat dideteksi dengan baik oleh model. Tingkat akurasi yang didapat yaitu sebesar 86% - 92%. Model yang didapat mampu memprediksi dan mendeteksi seluruh gambar dengan tepat.

5.5.5. *Polyps*

Gambar 5.21. Hasil pengujian *polyps*

Berdasarkan hasil pengujian di atas dapat diketahui bahwa seluruh gambar dapat dideteksi dengan baik oleh model. Tingkat akurasi yang didapat yaitu sebesar 83% - 90%. Model yang didapat mampu memprediksi dan mendeteksi seluruh gambar dengan tepat.

Untuk hasil keseluruhan model yang didapat mampu memprediksi dan mendeteksi dengan baik pada gambar validasi yang digunakan untuk pengujian, tetapi terdapat satu gambar yang tidak dapat dideteksi dengan baik oleh model. Hasil pengujian dapat menampilkan kategori yang dideteksi dan tingkat akurasi dari masing-masing gambar sesuai dengan rancangan *output*.

BAB VI
PENUTUP

6.1. Kesimpulan

Berdasarkan hasil analisis yang telah dilakukan tentang Implementasi *deep learnig* dalam mendeteksi penyakit menggunakan *convolutional neural network* dan *tensorflow*, maka dapat diperoleh beberapa kesimpulan sebagai berikut:

1. Dengan menggunakan 72.960 *step*, nilai *loss* 1.98 dan 1 jumlah perulangan (*batch*) pada proses *training*, dapat menghasilkan model untuk deteksi penyakit pada saluran pencernaan (*gastrointestinal*) dengan tingkat akurasi yang tinggi.
2. Untuk hasil keseluruhan model yang didapat mampu memprediksi dan mendeteksi dengan baik pada gambar validasi yang digunakan untuk pengujian, tetapi terdapat satu gambar yang tidak dapat dideteksi dengan baik oleh model yaitu pada gambar *dyed-resection-margins*. Hasil pengujian dapat menampilkan kategori yang dideteksi dan tingkat akurasi dari masing-masing gambar sesuai dengan rancangan *output.*
3. Hasil klasifikasi yang diperoleh dari uji validasi dengan kategori *polyps* nilai akurasi sebesar 83% - 90%, *normal-pylorus* nilai akurasi sebesar 86% - 92%, *dyed-lifted-polyps* nilai akurasi sebesar 83% - 97%, *esophagitis* nilai akurasi sebesar 82% - 92%, dan *dyed-resection-margins* nilai akurasi sebesar 82% - 93%. Pada gambar *dyed-resection-margins* terdapat satu gambar yang tidak dapat diklasifikasikan dengan baik karena muncul 2 label pada gambar tersebut. namun untuk ke empat gambar yang lain dapat mampu memprediksi dan mendeteksi gambar dengan tepat.

4. Implementasi *deep learning* pada penyakit saluran pencernaan, sangat membantu. khususnya tenaga medis yang ahli dibidangnya dalam menetukan jenis penyakit untuk hasil diagnosa.

6.2. Saran

Dari yang telah dilakukan oleh penulis, maka penulis memberikan saran untuk perkembangan dan penelitian selanjutnya diantaranya adalah sebagai berikut:

1. Untuk penelitian selanjutnya dapat melakukan proses *training* dengan data gambar yang lebih banyak lagi dan waktu yang lebih lama agar mendapatkan hasil yang lebih baik serta nilai *loss* yang lebih kecil.
2. Penelitian selanjutnya dapat menambahkan klasifikasi yang lebih banyak lagi mengenai penyakit pada saluran pencenaan
3. Kedepannya dapat meng *upgrade* alat endoskopi atau didesain suatu alat khusus untuk penyakit pada salauran pencernaan dalam mendukung diagnose pasien.
4. Penelitian selanjutnya dapat menggunakan metode analisis gambar lainnya untuk membandingkan akurasi, mendeteksi dan kemampuan pengklasifikasian gambar penyakit pada saluran pencernaan (*gastrointestinal*).
5. Karena proses klasifikasi menggunakan dataset dengan jumlah dan ukuran yang besar, maka membutuhkan waktu eksekusi yang cukup lama saat menggunakan *central processing unit* (CPU). Oleh karena itu untuk mempersingkat waktu eksekusi, perlu menggunakan bantuan *grafik card* yaitu *graphical processing unit* (GPU) dalam implementasi proses klasifikasi untuk meningkatkan efisiensi waktu.

DAFTAR PUSTAKA

Al-Waisy, A., Qahwaji, R., Ipson, S., Al-Fahdawi, S., & Nagem, T. (2017). A multi-biometric iris recognition system based on a deep learning approach. *Pattern Analysis and Applications*, 1-20.

Andono, P. N. (2017). *Pengolahan Citra Digital.* Yogyakarta: Andi.

Antara. (2014, November 23). *Kesehatan*. Retrieved from Berita Satu: https://www.beritasatu.com/beritasatu/kesehatan/227302/sekjen-apdw-kasus-penyakit-pencernaan-meningkat-di-dunia

Asmara, R. A. (2018). *Pengolahan Citra Digital.* Malang: Polinema Press.

Babatunde, e. a. (2015). A survey of computer-based vision systems for automatic identification of plant species. *Journal of Agricultural Informatics*, 61-71.

Basuki, L. (2016). *Impelementasi Metode Histograms of Oriented Gradients dengan Optimasi Algoritma Frei-Chen untuk Deteksi Citra Manusia. Diakses di.* Bandung: UNIKOM.

Bernal, J., Kushibar, K., Asfaw, D., Valverde, S., Oliver, A., Marti, R., & Llado, X. (2018). Artificial Intelligence In Medicine. *Article in Press*.

BigsmiLe, M. (2016). *Mengenali Teknologi Machine Learning.* Ciawi-Bogor: Direktorat Sistem Informasi dan Teknilogi.

Cogan, T., Cogan , M., & Tamil, L. (2019). Accurate identification of anatomical landmarks and diseased tissue in Gastrointestinal tract using deep learning. *Computers in Biology and Medicine*.

Dharma, I., Putera, I., & Ardana, P. (2011). Artificial Neural Networks Untuk Pemodelan Curah Hujan-Limpasan Pada Daerah Aliran Sungai (Das) Di Pulau Bali. *Jurnal Bumi Lestari*, 9-22.

Dharmawijaya, D. (2017). Endoskopi. 1-2.

Guadarma, S., & Chow, D. J. (2020, 03 31). *Tensorflow detection model zoo.* Retrieved from www.github.com: https://github.com/tensorflow/models/blob/477ed41e7e4e8a8443bc633846eb01e2182dc68a/object_detection/g3doc/detection_model_zoo.md

Hadi, S. (2002). *Gastroentologi. Cetakan ke : 2.* Bandung: PT. Alumni.

Hang, J., Rathod, V., Chow, D., Sun, C., Zhu, M., Tang, M., et al. (2017, 06 15). Retrieved from www.github.com: https://github.com/tensorflow/models

Istiqomah, Y. N., & Fadlil , A. (2013). Sistem Pakar untuk Mendiagnosa Penyakit Saluran Pencernaan Menggunakan Metode Dempster Shafer. 1-10.

Kasban, H., & Salama, H. D. (2019). A robust medical image retrieval system based on wavelet optimization and adaptive block truncation coding. *Springer Science+Business Media*.

Khomsan, A. (2003). *Pangan Dan Gizi Untuk Kesehatan.* Jakarta: PT.Rajagrafindo. Persada.

Kirkerod, M., & Borgli, R. J. (2019). Unsupervised preprocessing to improve generalisation for medical image classification. *International Symposium on Medical Information and Communication Technology*.

Kulkarni, A. (2001). *Computer Vision and Fuzzy Neural System.* New Jersey: Prentice Hall PTR.

Liu, J., Zhang, H., Liu, C., Dou, L., & Ju, L. (2017). An Implementation of Number Plate Recognition without Segmentation using Convolutional Neural Network. *International Conference on High Performance Computing and Communications*.

Lu, K., Chen, J., Little, J., & He, H. (2018). Lightweight Convolutional Neural Networks for Player Detection and Classification. *Computer Vision and Image Understanding*.

Luo, X., Shen, R., Jian, H., Deng, J., Hu, L., & Guan, Q. (2017). A Deep Convolution Neural Network Model for Vehicle Recognition and Face Recognition. *Advances in Information and Communication Technology: Proceedings of 7th International Congress of Information and Communication Technology (ICICT2017)*, 715-720.

Mane, S., & Mangale, P. (2018). Moving object detection and tracking Using Convolutional Neural Networks. *Proceedings of the Second International Conference on Intelligent Computing and Control Systems*.

Munir, R. (2004). *Pengolahan citra digital dengan pendekatan algoritmik.* Bandung: Institut Teknologi Bandung Press.

Nurhikmat, T. (2018, 05 23). *Implementasi Deep Learning Untuk Image Classification Menggunakan Algoritma CNN Pada Citra Wayang Golek.* Retrieved from dspace.uii.ac.id: https://dspace.uii.ac.id/handle/123456789/7843

Obukhova, N., Motyko, A., Pozdeev, A., & Timofeev, B. (2019). Method of Endoscopic Images Analysis for Automatic Bleeding Detection and Segmentation.

Owais, m., Arsalan , m., Choi, J., Mahmood, T., & Park, K. R. (2019). Artificial Intelligence-Based Classification of Multiple Gastrointestinal Diseases Using Endoscopy Videos for Clinical Diagnosis. *Clinical Medicine*.

Pogorelov, K., Randel, K. R., Griwodz, C., Eskeland, S. L., Lange, T. D., Johansen, D., . . . Halvorsen, P. (2017). A Multi-Class Image Dataset for Computer Aided Gastrointestinal Disease Detection. 1-6.

Primartha, R. (2018). *Belajar Machine Learning Teori dan Praktik.* Bandung: Informatika .

Rokhim, A. (2012). *Mengenal Tes Buta Warna.* Yogyakarta: Rona Publising.

Russel Stuart and Norvig Peter. (1995). *Artificial Intelligence: A Modern Approach.* Prentice Hall International : inc.

Saintif. (2020). *Penjelasan Sistem Pencernaan Manusia (Fungsi dan Anatominya)*. Retrieved from Saintif: https://saintif.com

Sena, S. (2018, May 27). *Pengenalan Deep Learning Part 1 : Neural Network*. Retrieved from Medium: https://medium.com/@samuelsena/pengenalan-deep-learning8fbb7d8028ac: https://medium.com

SERRA, X. (2017). *Face recognition using Deep Learning.* Barcelona: Polytechnic University of Catalonia.

Shustanov, A., & Yakimov, P. (2017). CNN Design for Real-Time Traffic Sign Recognition. *Information Technology and Nanotechnology*, 25-27.

Sidharta, H. (2017, October 28). *Introduction to OpenCV*. Retrieved from binus.ac.id: http://binus.ac.id/malang/2017/10/introduction-to-open-cv/

Sirohi, D., Kumar, N., & Rana, P. S. (2020). Convolutional neural networks for 5G-enabled Intelligent Transportation System: A systematic review. *Computer Communications*, 459-498.

Siswanto. (2005). *Kecerdasan Tiruan.* Yogyakarta: Graha Ilmu.

Sudirman, D. (2017). *Bab II Landasan Teori.* Retrieved from docplayer.info: https://docplayer.info/47258484-Bab-ii-landasan-teori.html

Sutoyo, T., Mulyanto, E., Suhartono, Dwi Nurhayati Oky, & Wijanarto. (2019). *Teori Pengolahan Citra Digital.* Yogyakarta: Andi Yogyakarta dan UDINUS semarang.

Suyanto. (2014). *Artificial Intelligence Searching, Reasining, Planning dan Learning (Revisi Kedua).* Bandung: Informatika.

Suyanto. (2019). *Deep Learning Moderisasi Machine Learning untuk Big Data.* Bandung: Informatika.

Szeliski, R. (2010). *Computer Vision: Algorithms and Applications.* London: Springer-Verlag London.

Zhu, X. (2017). Semi-supervised learning. *Encyclopedia of Machine Learning and Data Mining*, 1142-1147.

LAMPIRAN

Lampiran 1. *Syntax Xml to csv*

```
import os
import glob
import pandas as pd
import xml.etree.ElementTree as ET


def xml_to_csv(path):
    xml_list = []
    for xml_file in glob.glob(path + '/*.xml'):
        tree = ET.parse(xml_file)
        root = tree.getroot()
        for member in root.findall('object'):
            value = (root.find('filename').text,
                     int(root.find('size')[0].text),
                     int(root.find('size')[1].text),
                     member[0].text,
                     int(member[4][0].text),
                     int(member[4][1].text),
                     int(member[4][2].text),
                     int(member[4][3].text)
                     )
            xml_list.append(value)
    column_name = ['filename', 'width', 'height', 'class', 'xmin', 'ymin', 'xmax', 'ymax']
    xml_df = pd.DataFrame(xml_list, columns=column_name)
    return xml_df


def main():
    for folder in ['train','test']:
        image_path = os.path.join(os.getcwd(), ('images/' + folder))
        xml_df = xml_to_csv(image_path)
        xml_df.to_csv(('images/' + folder + '_labels.csv'), index=None)
        print('Successfully converted xml to csv.')


main()
```

Lampiran 2. *Syntax generate tfrecord*

```
from __future__ import division
from __future__ import print_function
from __future__ import absolute_import
import os
import io
import pandas as pd
import tensorflow.compat.v1 as tf
import sys
sys.path.append("C:\\deteksi\\models\\research\\")
sys.path.append("C:\\deteksi\\models\\research\\object_detection\\utils")
sys.path.append("C:\\deteksi\\models\\research\\slim")
sys.path.append("C:\\deteksi\\models\\research\\slim\\nets")
from PIL import Image
from object_detection.utils import dataset_util
from collections import namedtuple, OrderedDict
flags = tf.app.flags
flags.DEFINE_string('type', '', 'Type of CSV input (train/test)')
flags.DEFINE_string('csv_input', '', 'Path to the CSV input')
flags.DEFINE_string('output_path', '', 'Path to output TFRecord')
FLAGS = flags.FLAGS
def class_text_to_int(row_label):
    if row_label == 'Polyp':
        return 1
    if row_label == 'Esophagitis':
        return 2
    if row_label == 'Dyed-lifted-polyps':
        return 3
    if row_label == 'Dyed-resection-margins':
        return 4
    if row_label == 'Normal-pylorus':
        return 5
    else:
        return 0
def split(df, group):
    data = namedtuple('data', ['filename', 'object'])
    gb = df.groupby(group)
    return [data(filename, gb.get_group(x)) for filename, x in zip(gb.groups.keys(), gb.groups)]
def create_tf_example(group, path):
    with tf.gfile.GFile(os.path.join(path, '{}'.format(group.filename)), 'rb') as fid:
        encoded_jpg = fid.read()
    encoded_jpg_io = io.BytesIO(encoded_jpg)
    image = Image.open(encoded_jpg_io)
    width, height = image.size
    filename = group.filename.encode('utf8')
    image_format = b'jpg'
    xmins = []
    xmaxs = []
    ymins = []
    ymaxs = []
    classes_text = []
    classes = []
    for index, row in group.object.iterrows():
        xmins.append(row['xmin'] / width)
        xmaxs.append(row['xmax'] / width)
        ymins.append(row['ymin'] / height)
        ymaxs.append(row['ymax'] / height)
        classes_text.append(row['class'].encode('utf8'))
        classes.append(class_text_to_int(row['class']))
    tf_example = tf.train.Example(features=tf.train.Features(feature={
        'image/height': dataset_util.int64_feature(height),
        'image/width': dataset_util.int64_feature(width),
        'image/filename': dataset_util.bytes_feature(filename),
        'image/source_id': dataset_util.bytes_feature(filename),
        'image/encoded': dataset_util.bytes_feature(encoded_jpg),
        'image/format': dataset_util.bytes_feature(image_format),
        'image/object/bbox/xmin': dataset_util.float_list_feature(xmins),
        'image/object/bbox/xmax': dataset_util.float_list_feature(xmaxs),
        'image/object/bbox/ymin': dataset_util.float_list_feature(ymins),
        'image/object/bbox/ymax': dataset_util.float_list_feature(ymaxs),
        'image/object/class/text': dataset_util.bytes_list_feature(classes_text),
        'image/object/class/label': dataset_util.int64_list_feature(classes),
    }))
    return tf_example
def main(_):
    writer = tf.python_io.TFRecordWriter(FLAGS.output_path)
    path = os.path.join(os.getcwd(), 'images/{}'.format(FLAGS.type))
    examples = pd.read_csv(FLAGS.csv_input)
    grouped = split(examples, 'filename')
    for group in grouped:
        tf_example = create_tf_example(group, path)
        writer.write(tf_example.SerializeToString())
    writer.close()
    output_path = os.path.join(os.getcwd(), FLAGS.output_path)
    print('Successfully created the TFRecords: {}'.format(output_path))
if __name__ == '__main__':
    tf.app.run()
```

Lampiran 3. *Script* Label Map

```
item {
    id: 1
    name: 'Polyp'
}
item {
    id: 2
    name: 'Esophagitis'
}
item {
    id: 3
    name: 'Normal-pylorus'
}
item {
    id: 4
    name: 'Dyed-lifted-polyps'
}
item {
    id: 5
    name: 'Dyed-resection-margins'
}
```

Lampiran 4. *Script* Konfigurasi *Object Detection Pelatihan Pipeline*

```
model {
  ssd {
    num_classes: 5
    box_coder {
      faster_rcnn_box_coder {
        y_scale: 10.0
        x_scale: 10.0
        height_scale: 5.0
        width_scale: 5.0
      }
    }
    matcher {
      argmax_matcher {
        matched_threshold: 0.5
        unmatched_threshold: 0.5
        ignore_thresholds: false
        negatives_lower_than_unmatched: true
        force_match_for_each_row: true
      }
    }
    similarity_calculator {
      iou_similarity {
      }
    }
    anchor_generator {
      ssd_anchor_generator {
        num_layers: 6
        min_scale: 0.2
        max_scale: 0.95
        aspect_ratios: 1.0
        aspect_ratios: 2.0
        aspect_ratios: 0.5
        aspect_ratios: 3.0
        aspect_ratios: 0.3333
      }
    }
    image_resizer {
      fixed_shape_resizer {
        height: 300
        width: 300
      }
    }
    box_predictor {
      convolutional_box_predictor {
        min_depth: 0
        max_depth: 0
        num_layers_before_predictor: 0
        use_dropout: false
        dropout_keep_probability: 0.8
        kernel_size: 1
        box_code_size: 4
        apply_sigmoid_to_scores: false
        conv_hyperparams {
```

```
        activation: RELU_6,
        regularizer {
          l2_regularizer {
            weight: 0.00004
          }
        }
        initializer {
          truncated_normal_initializer {
            stddev: 0.03
            mean: 0.0
          }
        }
        batch_norm {
          train: true,
          scale: true,
          center: true,
          decay: 0.9997,
          epsilon: 0.001,
        }
      }
    }
    feature_extractor {
      type: 'ssd_mobilenet_v1'
      min_depth: 16
      depth_multiplier: 1.0
      conv_hyperparams {
        activation: RELU_6,
        regularizer {
          l2_regularizer {
            weight: 0.00004
          }
        }
        initializer {
          truncated_normal_initializer {
            stddev: 0.03
            mean: 0.0
          }
        }

  batch_norm {
        train: true,
        scale: true,
        center: true,
        decay: 0.9997,
        epsilon: 0.001,
      }
    }
    }
    loss {
      classification_loss {
        weighted_sigmoid {
          anchorwise_output: true
```

```
        }
      }
      localization_loss {
        weighted_smooth_l1 {
          anchorwise_output: true
        }
      }
      hard_example_miner {
        num_hard_examples: 3000
        iou_threshold: 0.99
        loss_type: CLASSIFICATION
        max_negatives_per_positive: 3
        min_negatives_per_image: 0
      }
      classification_weight: 1.0
      localization_weight: 1.0
    }
    normalize_loss_by_num_matches: true
    post_processing {
      batch_non_max_suppression {
        score_threshold: 1e-8
        iou_threshold: 0.6
        max_detections_per_class: 100
        max_total_detections: 100
      }
      score_converter: SIGMOID
    }
  }
}train_config: {
  batch_size: 2
  optimizer {
    rms_prop_optimizer: {
      learning_rate: {
        exponential_decay_learning_rate {
          initial_learning_rate: 0.004
          decay_steps: 800720
          decay_factor: 0.95
        }
      }
      momentum_optimizer_value: 0.9
      decay: 0.9
      epsilon: 1.0
    }
  }
  fine_tune_checkpoint: "ssd_mobilenet_v1_coco_2017_11_17/model.ckpt"
  from_detection_checkpoint: true
  num_steps: 100000
  data_augmentation_options {
    random_horizontal_flip {
    }
  }
  data_augmentation_options {
    ssd_random_crop {
```

```
    }
  }
}train_input_reader: {
  tf_record_input_reader {
    input_path: "data/train.record"
  }
  label_map_path: "data/kvasir_number_map.pbtxt"
}eval_config: {
  num_examples: 30
  max_evals: 10
}eval_input_reader: {
  tf_record_input_reader {
    input_path: "data/test.record"
  }
  label_map_path: "data/kvasir_number_map.pbtxt"
  shuffle: false
  num_readers: 1
  num_epochs: 1
}
```

Lampiran 5. *Syntax pengujian*

```
# coding: utf-8
# In[1]:
import numpy as np
import os
import six.moves.urllib as urllib
import sys
import tarfile
import tensorflow as tf
import zipfile

# In[2]:
from collections import defaultdict
from io import StringIO
from matplotlib import pyplot as plt
from PIL import Image

# In[16]:
import cv2
cap = cv2.VideoCapture(0)

# In[5]:
# This is needed since the notebook is stored in the object_detection folder.
sys.path.append("..")

# In[7]:
from utils import label_map_util
from utils import visualization_utils as vis_util

# In[8]:
# What model to download.
MODEL_NAME = 'kvasir_baru'
#MODEL_NAME = 'ssd_mobilenet_v1_coco_11_06_2017'
MODEL_FILE = MODEL_NAME + '.tar.gz'
DOWNLOAD_BASE = 'http://download.tensorflow.org/models/object_detection/'

# Path to frozen detection graph. This is the actual model that is used for the object detection.
PATH_TO_CKPT = MODEL_NAME + '/frozen_inference_graph.pb'

# List of the strings that is used to add correct label for each box.
PATH_TO_LABELS = os.path.join('data', 'kvasir_number_map.pbtxt')
NUM_CLASSES = 8

# In[7]:

# In[9]:
detection_graph = tf.Graph()
with detection_graph.as_default():
  od_graph_def = tf.GraphDef()
  with tf.gfile.GFile(PATH_TO_CKPT, 'rb') as fid:
    serialized_graph = fid.read()
    od_graph_def.ParseFromString(serialized_graph)
    tf.import_graph_def(od_graph_def, name='')
```

```
# In[10]:
label_map = label_map_util.load_labelmap(PATH_TO_LABELS)
categories = label_map_util.convert_label_map_to_categories(label_map, max_num_classes=NUM_CLASSES, use_display_name=True)
category_index = label_map_util.create_category_index(categories)

# In[11]:

def load_image_into_numpy_array(image):
  (im_width, im_height) = image.size
  return np.array(image.getdata()).reshape(
      (im_height, im_width, 3)).astype(np.uint8)

# In[12]:

# If you want to test the code with your images, just add path to the images to the TEST_IMAGE_PATHS.
PATH_TO_TEST_IMAGES_DIR = 'test_images'
TEST_IMAGE_PATHS = [ os.path.join(PATH_TO_TEST_IMAGES_DIR, 'image{}.jpg'.format(i)) for i in range(1, 3) ]
# Size, in inches, of the output images.
IMAGE_SIZE = (12, 8)

# In[ ]:
with detection_graph.as_default():
  with tf.Session(graph=detection_graph) as sess:
    while True:
      ret, image_np = cap.read()
      # Expand dimensions since the model expects images to have
      shape: [1, None, None, 3]
      image_np_expanded = np.expand_dims(image_np, axis=0)
      image_tensor = detection_graph.get_tensor_by_name('image_tensor:0')
      # Each box represents a part of the image where a particular object was detected.
      boxes = detection_graph.get_tensor_by_name('detection_boxes:0')
      # Each score represent how level of confidence for each of the objects.
      # Score is shown on the result image, together with the class label.
      scores = detection_graph.get_tensor_by_name('detection_scores:0')
      classes = detection_graph.get_tensor_by_name('detection_classes:0')
      num_detections = detection_graph.get_tensor_by_name('num_detections:0')
      # Actual detection.
      (boxes, scores, classes, num_detections) = sess.run(
          [boxes, scores, classes, num_detections],
          feed_dict={image_tensor: image_np_expanded})
      # Visualization of the results of a detection.
```

```
94          # Visualization of the results of a detection.
95          vis_util.visualize_boxes_and_labels_on_image_array(
96              image_np,
97              np.squeeze(boxes),
98              np.squeeze(classes).astype(np.int32),
99              np.squeeze(scores),
100             category_index,
101             use_normalized_coordinates=True,
102             line_thickness=8)
103         cv2.imshow('object detection', cv2.resize(image_np, (1000,800)))
104         if cv2.waitKey(25) & 0xFF == ord('q'):
105           cv2.destroyAllWindows()
106           break
```