BAB II

Tinjauan Pustaka

2.1 Malaria

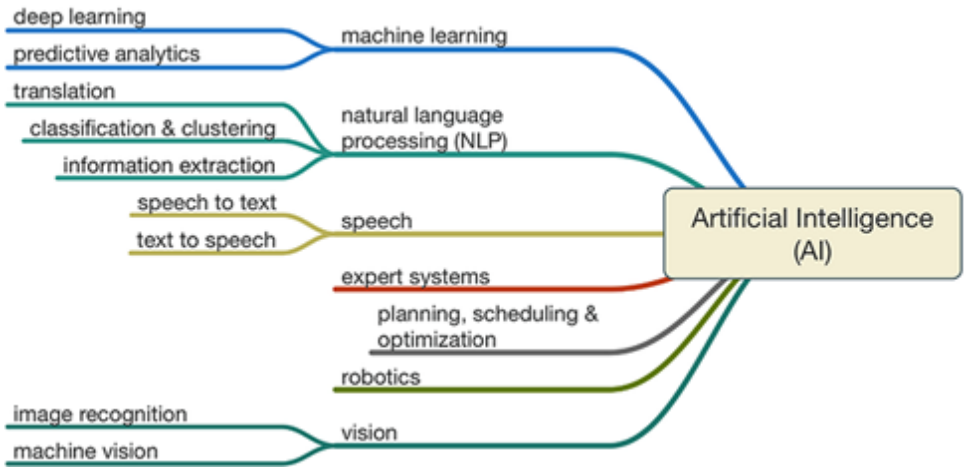
Malaria adalah salah satu penyakit paling mematikan di dunia. Menurut Zein (2019) penyakit ini adalah penyakit infeksi serius yang di sebabkan oleh parasit darah perifer dari genus Plasmodium. Penyakit ini dibawa oleh gigitan nyamuk yang dari seseorang/binatang yang sudah terinfeksi malaria. Penyakit ini juga bisa ditularkan melalui transplantasi organ, transfusi darah, dan penggunaan jarum suntik bersama dengan penderita. Malaria menjadi salah satu penyakit yang bisa mengakibatkan komplikasi pada pengidapnya jika tidak diatasi dengan tepat (Makarim, 2020). Saat parasit Plasmodium ini masuk ke tubuh, parasit ini akan menyebar ke organ hati dan berkembang biak. Secara perlahan, parasit akan menyerang sel darah merah yang sangat berperan sebagai pembawa oksigen di tubuh (Fadli, 2020).

Orang yang terkena penyakit malaria biasanya akan merasakan sangat kesakitan, yang disertai dengan demam tinggi serta menggigil kedinginan. Gejala penyakit malaria baru dapat diketahui 1-2 minggu atau lebih setelah terkena gigitan nyamuk yang terinfeksi. Gejala awalnya-pun sangat terlihat umum seperti gejala flu yaitu demam, menggigil, sakit kepala, mudah lelah, dan mudah pegal-pegal, sehingga sering dideteksi sebagai penyakit lain. Gejala lainnya seperti sakit perut, diare, otot terasa sakit, dan lain-lain. Jika penyakit ini tidak diatasi secara cepat dan tepat akan mengakibatkan komplikasi seperti anemia, gagal hati, gagal ginjal, kejang-kejang, koma, penyakit kuning, bahkan kematian (Fadlah, 2020).

Diagnosis infeksi malaria dilakukan dengan mengamati citra apusan darah tipis yang didapatkan dari darah yang dicampur dengan Giemsa yang dipantau di bawah mikroskop. Apusan darah tipis ini bisa mengidentifikasi spesies Plasmodium penyebab malaria. Pemeriksaan ini dikenal sebagai tes kualitatif. Pada pemeriksaan manual, ada cara yang bernama mikroskopi manual untuk pemeriksaan apusan darah (ditemukan pada akhir abad ke-19). Diagnosis menggunakan mikroskop membutuhkan pelatihan khusus dan keahlian yang cukup. Telah ditunjukkan dalam

beberapa studi lapangan bahwa mikroskop manual bukan metode skrining yang dapat diandalkan bila dilakukan oleh non-ahli karena kurangnya pelatihan terutama di daerah pedesaan (Zein, 2019).

2.2 Artificial Intelligence



Gambar 2.1 Artificial Intelligence

(sumber: https://medium.com/@shivakumar.goniwada/microservices-architecture-in-artificial-intelligence-60bac5b4485d)

Kata “Artificial” memiliki makna yaitu sesuatu yang tidak nyata dan merupakan hasil simulasi. Sedangkan kata “Intelligence” berasal dari bahasa Latin yaitu “Intelligo” yang memiliki makna “saya paham”. Artificial Intelligence (AI) atau Kecerdasan Buatan adalah salah satu bagian dari ilmu komputer yang mempelajari tentang bagaimana membuat mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan manusia bahkan lebih baik daripada yang dilakukan manusia (Dahria, 2008). AI merupakan teknologi yang memerlukan data untuk dijadikan pengetahuan/pembelajaran, sama seperti manusia. AI membutuhkan pengalaman dan pembelajaran dari data agar kecerdasannya bisa bertambah baik. Poin penting dalam proses AI adalah *learning*, *reasoning* dan *self-correction*. Proses belajar AI-pun tidak selalu disuruh oleh manusia, AI bisa belajar dengan sendirinya berdasarkan pengalamannya saat digunakan oleh manusia. Contoh aplikasinya adalah *Self Driving Cars* dan *Sign and Voice Recognizer (SVR)*. Pada gambar 2.1, cabang ilmu Artificial Intelligence yaitu Machine Learning, NLP,

Speech, Expert Systems, Planning, Scheduling, dan Optimization, Robotics, dan Vision. Pada penelitian ini menggunakan Deep Learning yang merupakan cabang dari Machine Learning.

2.3 Machine Learning

Istilah Machine Learning pertama kali didefinisikan oleh Arthur Samuel ditahun 1959. Menurut Arthur Samuel, Machine Learning adalah salah satu bidang ilmu komputer yang memberikan kemampuan pembelajaran kepada komputer untuk mengetahui sesuatu tanpa pemrogram yang jelas. *"Machine Learning (ML) is a subset of Artificial Intelligence (AI) technique which use statistical methods to enable machines to improve with experience",* Garg, 2019. Machine Learning memberikan kemampuan kepada sistem komputer untuk belajar secara mandiri menggunakan data yang disediakan dan membuat prediksi yang akurat. Data adalah komponen penting dalam Machine Learning. Data akan diproses oleh algoritma dan metode yang berbeda pada Machine Learning melalui berbagai percobaan untuk memperoleh hasil yang optimal. Pada Machine Learning data akan dibagi menjadi 2 yaitu data *training* dan data *testing*. Data *training* digunakan untuk melakukan pelatihan data pada algoritma dalam mencari model yang cocok, sedangkan data testing digunakan untuk melihat performa dari model yang sudah dilatih sebelumnya ketika menemukan data baru yang belum pernah ditemukan sebelumnya.

Machine Learning dibagi menjadi 4 jenis yaitu:

1. Supervised Learning

   Supervised Learning adalah tipe pembelajaran di mana data mempunyai variabel *input* (masukan) dan variabel *output* (keluaran) dan menggunakan satu algoritma atau lebih untuk mempelajari fungsi pemetaan dari *input* ke *output*. Tujuan Supervised Learning ini adalah untuk memperkirakan fungsi pemetaannya, sehingga ketika kita mempunya *input* baru, kita dapat memprediksi *output* yang tepat untuk *input* tersebut (Satria, 2018). Pada Supervised Learning, *input* dan *output* disediakan sehingga komputasi akan lebih mudah.
2. Unsupervised Learning

Unsupervised Learning adalah tipe pembelajaran di mana data hanya mempunyai data *input* tetapi tidak ada variabel *output*-nya. Tujuan dari Unsupervised Learning adalah untuk memodelkan struktur dasar atau distribusi dalam data dengan tujuan untuk mempelajari data dan menyimpulkan fungsi yang mendeskripsikan atau menjelaskan data (Satria, 2018).

3. Semi Supervised Learning

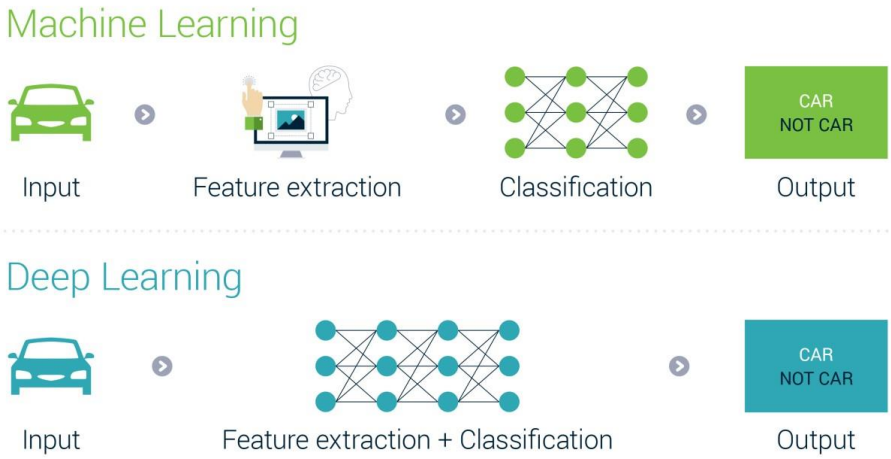
   Semi Supervised Learning berada di antara Supervised Learning dan Unsupervised Learning. Semi Supervised Learning adalah tipe pembelajaran di mana data mempunyai data *input* dalam jumlah besar dan hanya beberapa dari data *input* tersebut yang memiliki variabel *output* (Satria, 2018).

4. Reinforcement Learning

   Reinforcement Learning adalah ketika algoritma disajikan dengan contoh yang kekurangan label (*output*), namun disertakan contoh dengan *feedback* positif atau negatif bergantung pada solusi yang ditawarkan oleh algoritma tersebut. Melalui pembelajaran ini, dapat menentukan tingkah laku ideal terhadap sebuah konteks yang spesifik secara otomatis, dengan tujuan untuk memaksimalkan performanya (Satria, 2018).

Contoh aplikasi dari Machine Learning adalah Filter Snapchat dan rekomendasi film pada Netflix.

2.4 Deep Learning

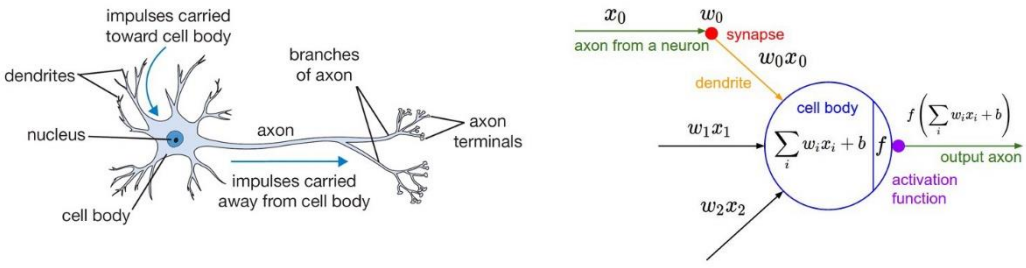


Gambar 2.2 Machine Learning VS Deep Learning

(sumber: https://www.inteliment.com/blog/our-thinking/lets-understand-the-difference-between-machine-learning-vs-deep-learning/)

Deep Learning merupakan salah satu cabang pada bidang Machine Learning yang mengadopsi cara kerja otak manusia agar mesin bisa belajar secara mandiri. Deep learning dirancang untuk terus menganalisa data seperti pada otak manusia dalam mengambil keputusan (Peryanto *et al.*, 2019). Pada gambar 2.2 adalah struktur pemodelan jaringan pada Machine Learning dan Deep Learning. Terdapat perbedaan pada saat *feature extraction*. Pada Machine Learning *feature extraction* dilakukan dengan bantuan manusia, namun pada Deep Learning *feature extraction* dilakukan oleh komputer secara mandiri.

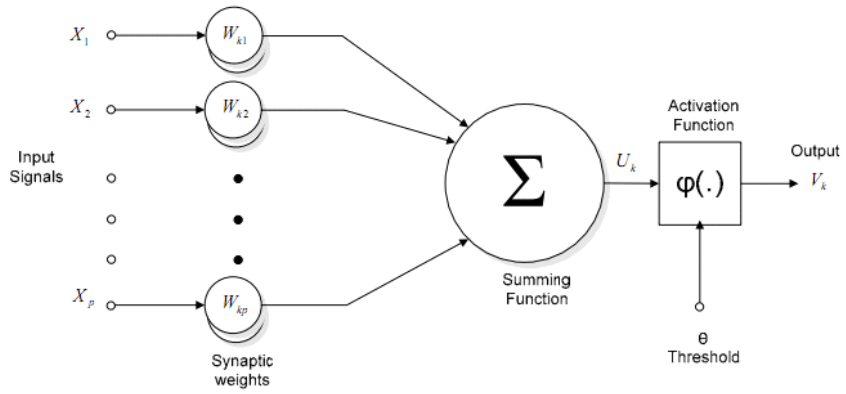
2.5 Artificial Neural Network



Gambar 2.3 Neuron dan Neural Network

(sumber: https://medium.com/@atriadplt/get-to-know-how-artificial-neural-network-formed-in-computer-science-794897d48fd)

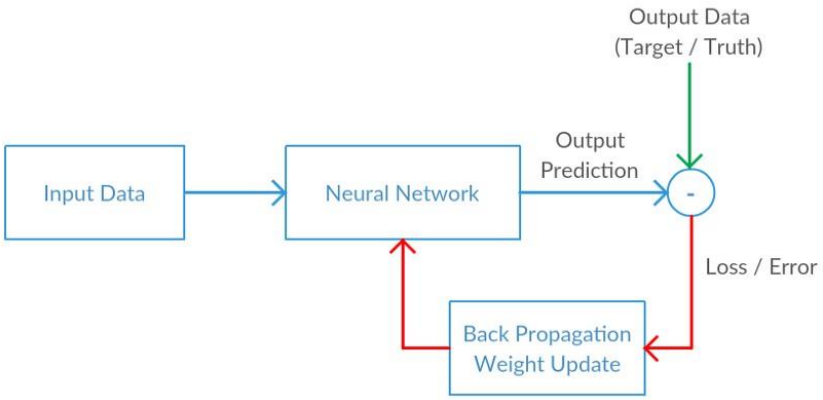
Artificial Neural Network atau Jaringan Syaraf Tiruan (JST) terinspirasi dari cara kerja neuron pada otak manusia. Menurut Widiputra (2016), Artificial Neural Network merupakan sebuah teknik atau pendekatan pengolahan informasi yang terinspirasi oleh cara kerja sistem saraf biologis, khususnya pada sel otak manusia dalam memproses informasi. Elemen kunci dari teknik ini adalah struktur sistem pengolahan informasi yang bersifat unik dan beragam untuk tiap aplikasi.



Gambar 2.4 Arsitektur Artificial Neural Network

(sumber: https://www.researchgate.net/publication/326648350_Model_Pendugaan_Produktivitas_Perikanan_Pukat_Cincin_Di_Laut_Jawa/figures?lo=1)

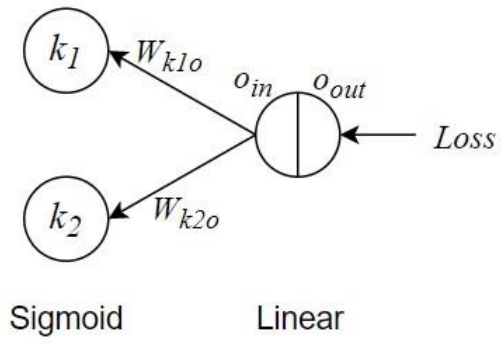
Neural Network terdiri dari beberapa elemen (neuron) yang saling terhubung dan bekerja bersama-sama untuk menyelesaikan sebuah masalah (prediksi/klasifikasi). Hasil prediksi akan dipengaruhi oleh bobot (W) yang akan terus berganti hingga mencapai hasil yang maksimal. Terdapat 2 jenis ANN yaitu Single Layer Perceptron dan Multi Layer Perceptron (MLP) Perbedaannya, Single Layer Perceptron tidak memiliki *hidden layer*, sedangkan Multi Layer Perceptron memiliki *hidden layer*. Neural Network paling sederhana adalah Single Layer Perceptron, yang hanya memiliki *single layer output node*. *Input node* langsung menuju ke *activation function* lalu ke *output node* melalui serangkaian pembobotan atau *weight*. Gambar 2.4 merupakan contoh arsitektur Artificial Neural Network Single Layer Perceptron. Sedangkan Multi Layer Perceptron, *input node* menuju *hidden layer* (bisa lebih dari 1), lalu menuju ke activation function dan *output node*.



Gambar 2.5 Artificial Neural Network

(sumber: https://medium.com/@samuelsena/pengenalan-deep-learning-part-3-backpropagation-algorithm-720be9a5fbb8)

Gambar 2.5 menjelaskan alur kerja ANN, *input* data akan diproses pada model yang akan menghasilkan *output* prediksi, lalu akan dicocokkan dengan output yang sebenarnya/sesungguhnya, maka akan dihasilkan *loss* atau *error*. Setelah didapatkan *error*, akan dilakukan pembaharuan bobot dengan backpropagation. Pada tahap backpropagation, loss akan mengalir menuju semua node pada hidden layer untuk dicari gradient nya. lalu bobot akan diupdate. Alur backpropagation adalah seperti pada gambar 2.6.



Gambar 2.6 Backpropagation

(sumber: https://medium.com/@samuelsena/pengenalan-deep-learning-part-3-backpropagation-algorithm-720be9a5fbb8)

Beberapa *Activation Function* pada Neural Network:

1. Linear Function

$$f(x) = cx \tag{2-1}$$

Linear Function adalah activation sederhana, yaitu dengan cara mengalikan *input* (x) dengan bobot (c) lalu menjumlahkan semuanya.

2. Sigmoid

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

Sigmoid adalah *activation function* yang mengubah nilai input (x) menjadi sebuah nilai yang memiliki range mulai dari 0 sampai 1.

3. Tanh

$$f(x) = \frac{1}{1 + e^{-2x}} - 1 \tag{2-3}$$

Tanh adalah *activation function* yang mengubah nilai input (x) menjadi sebuah nilai yang memiliki range mulai dari -1 sampai 1.

4. ReLu (Rectified Linear Unit)

$$f(x) = \max(0, x) \tag{2-4}$$

ReLu adalah *activation function* yang mengubah nilai input (x). Jika nilai x $\leqslant$ 0, maka akan menjadi 0, sedangkan jika x > 0, maka f(x) menjadi x. Range ReLu adalah 0 hingga *infinity*.

5. Leaky ReLu

$$f(x) = f(x) = \begin{cases} 0.01x, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{2-5}$$

Leaky ReLu adalah *activation function* yang mengubah nilai input (x). Jika nilai x $\leqslant$ 0, maka akan menjadi 0,01x, sedangkan jika x > 0, maka f(x) menjadi x. Range ReLu adalah *infinity* hingga *infinity*.

Beberapa *Loss Function* pada Neural Network:

1. Binary Crossentropy

$$CE = -\frac{1}{output\ size} \sum_{i=1}^{output\ size} -\tilde{y}_i \log \tilde{y}_i + (1-\tilde{y}_i) \log(1-\tilde{y}_i) \qquad (2\text{-}6)$$

Loss function ini digunakan untuk masalah klasifikasi biner. Klasifikasi yang di mana nilai target berada pada set {0, 1} atau hanya klasifikasi untuk 2 kelompok saja, sehingga *output size* = 2. y_i adalah output hasil prediksi dan $\tilde{y}_i$ adalah output sesungguhnya.

2. Categorical Crossentropy

$$CE = -\sum_{i=1}^{output\ size} y_i \log \tilde{y}_i \qquad (2\text{-}7)$$

Categorical Crossentropy dapat digunakan untuk masalah klasifikasi *multi-class*. Klasifikasi yang di mana nilai target berada pada set {0, 1, 2, 3, …, n} atau bisa digunakan untuk klasifikasi lebih dari 2 kelompok. y_i adalah output hasil prediksi dan $\tilde{y}_i$ adalah output sesungguhnya.

3. Mean Squared Error (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \qquad (2\text{-}8)$$

MSE memberi informasi seberapa dekat garis regresi dengan sekumpulan titik. Jarak dari titik ke garis regresi (*error*) dikuadratkan agar menghindari adanya hasil negatif serta memberi bobot lebih pada perbedaan yang lebih besar. MSE bekerja dengan mencari rata – rata kesalahan (*error*) dari sekumpulan titik. y_i adalah output hasil prediksi dan $\tilde{y}_i$ adalah output sesungguhnya.

4. Mean Absolute Error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \tilde{y}_i| \qquad (2\text{-}9)$$

MAE mengukur besarnya rata – rata kesalahan dalam serangkaian prediksi, tanpa mempertimbangkan arahnya. Hampir sama dengan MSE, namun pada MAE jarak dari titik ke garis regresi (*error*) tidak

dikuadratkan, namun diabsolutkan. y_i adalah output hasil prediksi dan $\tilde{y}_i$ adalah output sesungguhnya.

Artificial Neural Network memiliki kelebihan dan kelemahan. Menurut Saraswati (2019), kelebihan dan kelemahan Artificial Neural Network adalah sebagai berikut.

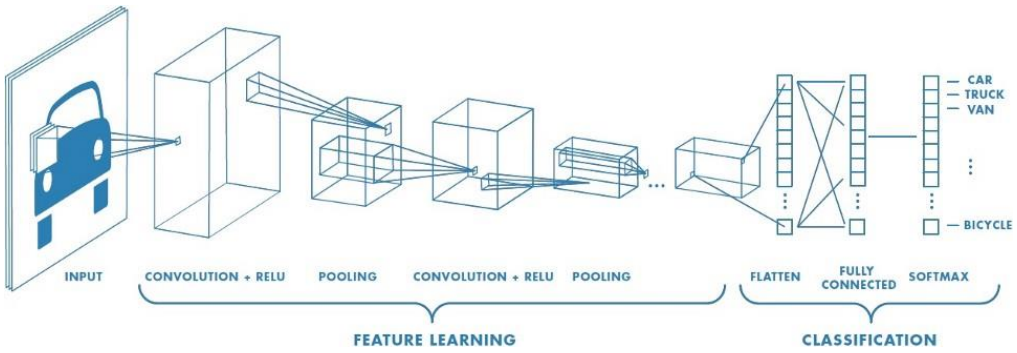
Kelebihan :

1. Dapat melaksanakan tugas yang suatu program linier tidak bisa lakukan
2. Ketika suatu unsur jaringan salah/gagal, ANN masih dapat melanjutkan tanpa masalah selanjutnya
3. ANN bisa diimplementasikan pada berbagai aplikasi
4. ANN bisa diimplementasikan tanpa masalah yang berarti

Kelemahan :

1. ANN membutuhkan "pelatihan" terlebih dahulu sebelum beroperasi
2. Arsitektur dari ANN berbeda dari arsitektur kebanyakan *microprocessor*, sehingga membutuhkan proses emulasi
3. Membutuhkan waktu *processing* yang lama untuk ANN dengan ukuran yang besar

2.6 Convolutional Neural Network



Gambar 2.7 Arsitektur CNN

(sumber: https://www.inteliment.com/blog/our-thinking/lets-understand-the-difference-between-machine-learning-vs-deep-learning/)

Convolutional Neural Network (CNN) adalah salah satu perkembangan cabang Machine Learning yaitu Computer Vision. Convolutional Neural Network

adalah salah satu metode Machine Learning dari pengembangan Multi Layer Perceptron (MLP) yang didesain untuk mengolah data dua dimensi (Sofia, 2018). Gambar 2.7 menunjukkan arsitektur dari CNN. Arsitektur dari CNN dibagi menjadi 2 bagian besar yaitu *Feature Learning* dan *Classification*.

Feature Learning melakukan *encoding* pada citra *input* menjadi berbentuk angka-angka dalam vektor. *Feature Learning* terdiri dari *Convolutional layer* dan *Pooling layer*.

- *Convolutional layer* terdiri dari neuron yang telah disusun sedemikian rupa sehingga dapat membentuk filter dengan *Pixels*. Filter tersebut akan bergeser ke seluruh bagian citra. Pada tiap pergeseran dilakukan operasi dot antara *input* dan filter yang akan menghasilkan *output* (*activation map*). Fungsi Aktivasi Rectified Linear Unit (ReLU) akan menghilangkan *vanishing gradient* dengan cara menerapkan fungsi aktivasi element sebagai f(x) = max(0,x) alias aktivasi elemen akan dilakukan saat berada di ambang batas 0. *Pooling layer* adalah lapisan yang mengurangi dimensi sehingga mempercepat komputasi karena parameter akan menjadi semakin sedikit dan dapat mengatasi *over-fitting*.
- *Pooling* yang biasa digunakan adalah *Max-pooling* dan *Average-pooling*. *Max-pooling* menggunakan nilai maksimum pada tiap pergeseran filter, sementara *Average-pooling* menggunakan nilai rata-ratanya.

Classification berguna untuk mengklasifikasikan tiap neuron yang telah diekstraksi fitur pada sebelumnya. Lapisan ini terdiri dari *Flatten*, *Fully-connected*, dan *Softmax*.

- *Flatten* membentuk ulang fitur (*re-shape feature map*) menjadi sebuah vektor 1 dimensi agar bisa digunakan sebagai *input* dari *Fully-connected layer*.
- *Fully-connected* akan menghitung skor setiap kelas yang ada.
- Fungsi *Softmax* menghitung probabilitas dari setiap kelas yang ada dan akan menentukan kelas dengan probabilitas tertinggi sebagai *output* dari *input* yang diberikan.

2.6.1 ShallowNet

ShallowNet adalah salah satu arsitektur CNN yang memiliki 1 *Convolutional layer*, 2 *Pooling layer*, dan 2 *Fully-Connected layer* untuk *input shape* 64x64 dan 1 *Convolutional layer*, 3 *Pooling layer*, dan 2 *Fully-Connected layer* untuk *input shape* 128x128. Arsitektur dari model ShallowNet dengan *input shape* 64x64 adalah seperti tabel 2.1. Sedangkan arsitektur dari model ShallowNet dengan *input shape* 128x128 adalah seperti pada tabel 2.2.

Tabel 2.1 Arsitektur Model ShallowNet 64x64

ShallowNet 64 x 64

Layer	Output Shape
Conv2D	(None, 64, 64, 32)
Activation	(None, 64, 64, 32)
MaxPooling2D	(None, 32, 32, 32)
MaxPooling2D	(None, 16, 16, 32)
Flatten	(None, 8192)
Dense	(None, 4000)
Dense	(None, 2)
Activation	(None, 2)

Tabel 2.2 Arsitektur Model ShallowNet 128x128

ShallowNet 128 x 128

Layer	Output Shape
Conv2D	(None, 128, 128, 32)
Activation	(None, 128, 128, 32)
MaxPooling2D	(None, 64, 64, 32)
MaxPooling2D	(None, 32, 32, 32)
MaxPooling2D	(None, 16, 16, 32)
Flatten	(None, 8192)
Dense	(None, 4000)
Dense	(None, 2)
Activation	(None, 2)

Activation function arsitektur ShallowNet adalah ReLu dan *Optimizer*-nya adalah RMSProp. Namun, percobaan yang dilakukan juga menggunakan *Optimizer* Adam sebagai pembanding.

2.6.2 MicroVGGNet

Arsitektur MicroVGGNet memiliki 4 *Convolutional layer*, 2 *Pooling layer*, dan 2 *Fully-Connected layer*. *Activation function* MicroVGGNet adalah ReLu, *Optimizer*-nya Adam, dan *input shape*-nya adalah 64x64. Arsitektur MicroVGGNet adalah seperti pada tabel 2.3.

Tabel 2.3 Arsitektur MicroVGGNet

MicroVGGNet 64 x 64

Layer	Output Shape
Conv2D	(None, 64, 64, 32)
Activation	(None, 64, 64, 32)
Conv2D	(None, 64, 64, 32)
Activation	(None, 64, 64, 32)
MaxPooling2D	(None, 32, 32, 32)
Conv2D	(None, 32, 32, 64)
Activation	(None, 32, 32, 64)
Conv2D	(None, 32, 32, 64)
Activation	(None, 32, 32, 64)
MaxPooling2D	(None, 16, 16, 64)
Flatten	(None, 16384)
Dense	(None, 5000)
Dense	(None, 2)
Activation	(None, 2)

2.6.3 BaselineNet

Arsitektur BaselineNet adalah arsitektur CNN yang menggunakan *input shape* 64x64. Ada beberapa jenis BaselineNet yang diuji pada praktik kreja lapangan ini. *Optimizer* BaselineNet adalah Adam, namun juga dilakukan pengujian dengan menggunakan *optimizer* RMSProp. *Activation function*-nya ReLu, namun juga dilakukan pengujian dengan Leaky ReLu. Pada tabel 2.4 adalah arsitektur BaselineNet, tabel 2.5 adalah arsitektur BaselineNet dengan menggunakan *activation function* Leaky ReLu, dan tabel 2.6 adalah gambar arsitektur BaselineNet dengan no padding.

Tabel 2.4 Arsitektur BaselineNet

BaselineNet 64 x 64

Layer	Output Shape
Conv2D	(None, 64, 64, 32)
MaxPooling2D	(None, 32, 32, 32)
Conv2D	(None, 32, 32, 64)
MaxPooling2D	(None, 16, 16, 64)
Conv2D	(None, 16, 16, 128)
MaxPooling2D	(None, 8, 8, 128)
Conv2D	(None, 8, 8, 128)
MaxPooling2D	(None, 4, 4, 128)
Flatten	(None, 2048)
Dense	(None, 1000)
Dense	(None, 2)

Tabel 2.5 Arsitektur BaselineNet dengan Leaky ReLu

BaselineNet_LeakyReLu 64 x 64

Layer	Output Shape
Conv2D	(None, 64, 64, 32)
LeakyReLU	(None, 64, 64, 32)
MaxPooling2D	(None, 32, 32, 32)
Conv2D	(None, 32, 32, 64)
LeakyReLU	(None, 32, 32, 64)
MaxPooling2D	(None, 16, 16, 64)
Conv2D	(None, 16, 16, 128)
LeakyReLU	(None, 16, 16, 128)
MaxPooling2D	(None, 8, 8, 128)
Conv2D	(None, 8, 8, 128)
LeakyReLU	(None, 8, 8, 128)
MaxPooling2D	(None, 4, 4, 128)
Flatten	(None, 2048)
Dense	(None, 1000)
LeakyReLU	(None, 1000)
Dropout	(None, 1000)
Dense	(None, 2)

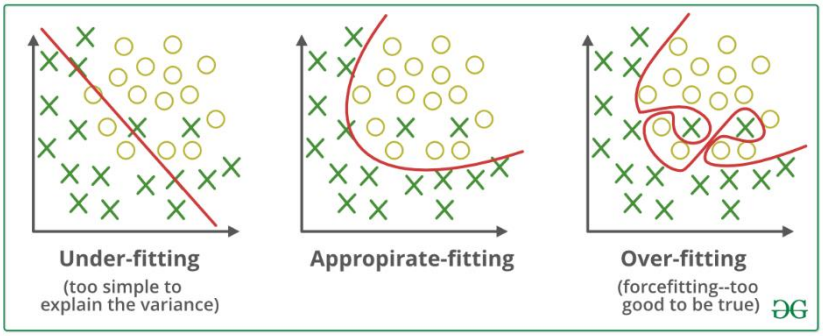
Tabel 2.6 Arsitektur BaselineNet dengan No Padding

BaselineNet_NoPad 64 x 64

Layer	Output Shape
Conv2D	(None, 64, 64, 32)
MaxPooling2D	(None, 31, 31, 32)
Conv2D	(None, 29, 29, 64)
MaxPooling2D	(None, 14, 14, 64)
Conv2D	(None, 12, 12, 128)
MaxPooling2D	(None, 6, 6, 128)
Conv2D	(None, 4, 4, 256)
MaxPooling2D	(None, 2, 2, 256)
Flatten	(None, 1024)
Dense	(None, 500)
Dropout	(None, 500)
Dense	(None, 500)
Dropout	(None, 500)
Dense	(None, 500)
Dropout	(None, 500)
Dense	(None, 2)

2.7 Regularization

Regularization atau regularisasi adalah sebuah teknik yang memberikan sedikit modifikasi pada algoritma pembelajaran sehingga model dapat digeneralisasi lebih baik. Regularisasi dibutuhkan untuk meningkatkan kinerja model menjadi lebih baik serta bisa menghindarkan dari Over-fitting dan Under-fitting.



Gambar 2.8 Under-fitting, Appropriate-fitting, dan Over-fitting

(sumber: https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/)

Over-fitting adalah kondisi di mana hasil akurasi dari data *training* memiliki hasil yang baik, namun hasil akurasi dari data *testing* memiliki hasil yang buruk/tidak baik. Sedangkan *Under-fitting* adalah kondisi di mana hasil akurasi dari data *training* maupun data *testing* memiliki hasil yang buruk/tidak baik. Regularisasi bisa mencegah adanya kondisi *Over-fitting* dan *Under-fitting*, sehingga hasil dari akurasi data *training* maupun data *testing* sama-sama baik.

Regularisasi memiliki berbagai macam teknik yaitu seperti:

1. L1 & L2 yang memperbaiki/mereduksi bobot

   Rumus L1:

   $$Cost\ function = Loss + \frac{\lambda}{2m} * \sum \|w\| \qquad (2\text{-}10)$$

   Rumus L2:

   $$Cost\ function = Loss + \frac{\lambda}{2m} * \sum \|w\|^2 \qquad (2\text{-}11)$$

   Lambda adalah parameter regularisasi yang nilainya dioptimalkan untuk hasil yang lebih baik. Bobot dari L1 memungkinkan akan berkurang hingga ke angka 0, sementara L2 nilainya akan mendekati 0, namun tidak mencapai 0.
2. Dropout yang menghilangkan sementara sebagian neuron yang berupa *Hidden Layer* maupun *Visible Layer* yang berada di dalam jaringan.
3. Image Augmentation yang berperan mengubah bentuk gambar misalnya merotasi gambar dan menskalakan gambar.

Early Stopping yang memberhentikan pelatihan model pada saat validasi model semakin memburuk.

2.8 Python

Gambar 2.9 Logo Python

Python merupakan bahasa pemrograman tingkat tinggi berorientasi objek yang diciptakan oleh Guido Van Rossum pada tahun 1990 di Stichting Mathematisch Centrum (CWI), Amsterdam yang bertujuan untuk menggantikan bahasa pemrograman ABC. Bahasa Python telah menjadi bahasa pemrograman yang populer yang banyak digunakan oleh Data Analysts, Data Scientists dan para Software Engineers untuk pengolahan data pada Machine Learning. Python dilengkapi banyak fitur yaitu *library* yang dapat digunakan untuk Machine Learning maupun Deep Learning.

2.9 Library Numpy

Gambar 2.10 Logo Numpy

NumPy (kependekan dari Numerical Python) adalah salah satu *library* teratas yang dilengkapi dengan sumber daya yang berguna untuk membantu para Data Scientist mengubah Python menjadi alat analisis dan pemodelan ilmiah yang kuat (Purwanto, 2018). Menurut Rohman (2019) NumPy adalah *library* Python yang berfokus pada *scientific computing*. NumPy memiliki kemampuan untuk membentuk objek *N-dimensional array* yang mirip dengan list pada Python. Keunggulan NumPy array dibandingkan dengan list pada Python adalah penggunaan *memory* yang lebih kecil serta *runtime* yang lebih cepat. NumPy juga

memudahkan melakukan komputasi pada Aljabar Linear, terutama operasi pada Vector (1-*dimension* array) dan Matrix (2-*dimension* array). *Libary open source* terpopuler ini tersedia di bawah lisensi BSD.

2.10 Library OpenCV

Gambar 2.11 Logo OpenCV

OpenCV adalah *library open source* yang digunakan untuk mengolah gambar dan video dengan tujuan untuk meng-ekstrak informasi di dalamnya. Library OpenCV sering digunakan dalam komputasi Computer Vision. Contoh penggunaannya adalah untuk mendeteksi wajah manusia. Menurut Aprilia (2018), kelebihan *library* ini adalah:

1. Memiliki dokumentasi yang cukup banyak
2. Dapat bekerja secara cepat pada komputer yang menggunakan *processor* Intel
3. Komputasi yang lebih ringan dibandingkan menggunakan Matlab dalam hal pengolahan citra digital (gambar)
4. Dapat bekerja secara *real-time*

2.11 Library Keras

Gambar 2.12 Logo Keras

Keras adalah *library* berbasis *open source* yang dirancang untuk menyederhanakan model dari kerangka Deep Learning. Keras dapat dijalankan di atas *framework* kecerdasan buatan seperti TensorFlow, Microsoft Cognitive Toolkit, dan Theano (Fachrizal, 2019). *Library* ini dikembangkan oleh François Chollet sebagai bagian dari *research project* dan dirilis sebagai proyek *open source* pada Maret 2015. Keras dengan cepat mendapatkan popularitas karena kemudahan penggunaan, fleksibilitas, dan desainnya yang indah. Sebagian besar fungsi Keras, seperti *losses*, *regularizers*, *constraints*, *initializers*, *metrics*, *activation functions*, *layers*, dan bahkan model lengkap dapat disesuaikan dengan cara yang sama. Sering kali, hanya perlu menulis *simple function*, dengan *input* dan *output* yang sesuai (Aurélien Géron, 2019).

Keras menyediakan banyak fungsi *built-in* terkait Neural Network untuk membuat Model keras dan lapisan Keras. Menurut Mutiara (2020), beberapa fungsi tersebut adalah sebagai berikut:

- Activation modules: fungsi aktivasi adalah konsep penting dalam Neural Network dan modul aktivasi ini menyediakan banyak fungsi aktivasi seperti Softmax dan ReLu
- Loss module: menyediakan fungsi-fungsi *loss* seperti Mean Squared Error (MSE), Mean Absolute Error (MAE), Poisson, dan lain-lain
- Optimizer module: menyediakan fungsi-fungsi optimizer seperti Adam, SGD, dan lainnya
- Regularizers: menyediakan fungsi-fungsi Regularizer seperti L1 Regularizer, L2 Regularizer, dan sebagainya

2.12 Google Colab

Gambar 2.13 Logo Google Colab

Google Colab adalah salah satu produk buatan Google berbasis *cloud* yang bisa digunakan bersama secara *real-time* (seperti Google Docs dan Google Sheets). Google Colab bisa digunakan gratis maupun berbayar, tentunya dengan fitur lebih untuk yang berbayar. Kegunaan Google Colab ini adalah ditujukan bagi pengguna yang perlu mengerjakan *coding* bersama rekan secara *real-time* dan lebih berfokus pada pengguna yang tidak memiliki komputer/laptop dengan kapasitas yang mumpuni untuk membuat program atau dengan kata lain Google Colab meminjami kapasitas komputernya. Menurut Adam (2019), manfaat dari Google Colab adalah sebagai berikut:

1. Free GPU: Google Colab memudahkan untuk menjalankan program pada komputer dengan spesifikasi tinggi (GPU Tesla, RAM hingga 12GB, Disk hingga 300GB yang masih bisa tersambung dengan Google Drive, akses internet cepat untuk *download file* besar) dan *running* dalam waktu yang lama (untuk versi gratisnya, Google Colab mengizinkan *running* program hingga 12 jam)
2. Colaborate: Google Colab juga memudahkan kolaborasi dengan rekan dengan cara membagi *coding* secara *online* (mirip Google Doc). Dengan bergitu akan lebih mudah bereksperimen secara bersamaan, atau sekadar menggunakan fitur ini untuk mempelajari *coding* orang lain yang telah rapi
3. Mudah berintegrasi: Google Colab terbilang sangat fleksibel dalam hal integrasi. Google Colab dapat dengan mudah dihubungkan dengan Jupyter Notebook di komputer pribadi (*local runtime*), menghubungkan dengan Google Drive, atau dengan Github
4. Fleksibel: Google Colab memungkinkan *running* program via *Smartphone*. Hal ini bisa terjadi karena pada Google Colab hanya perlu *running* di browser. Dengan begitu pengawasan *training* atau proses jalannya *coding* bisa dipantau dengan mudah karena bisa diawasi dengan menggunakan *Smartphone* saja