BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

2.1.1 Pengertian Jalan

Jalan diartikan sebagai jalan yang dibuat pada permukaan tanah, di atas permukaan tanah, di bawah permukaan tanah, dan/atau di atas permukaan air, kecuali jalan kereta api, jalan lori, dan jalan kabel berdasarkan PP No 34 Thn 2006 (Dan et al., 2018).

2.1.2 Jenis Kerusakan Jalan

Kerugian jalan umumnya diklasifikasikan menjadi dua kategori: kerusakan struktural, yang mencakup kegagalan perkerasan atau kerusakan satu atau lebih bagian perkerasan, yang menyebabkan perkerasan tidak dapat menahan beban lalu lintas dan kerusakan fungsional, yang menyebabkan keamanan dan kenyamanan pengguna jalan terganggu dan meningkatkan biaya operasi kendaraan (Sulaksono, 2001).

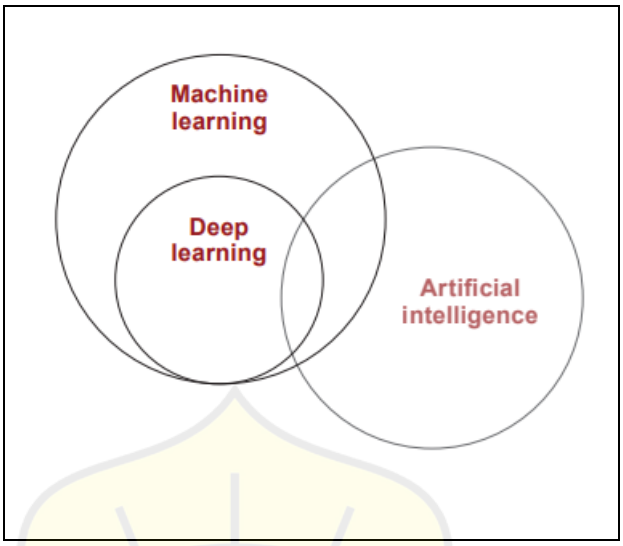
2.1.3 Machine Learning, Deep Learning dan Sebagainya

2.1.3.1 Machine Learning

Machine Learning (ML) adalah bidang studi yang berfokus pada desain dan analisis algoritma yang memungkinkan komputer untuk belajar. ML juga dapat diartikan sebagai sebuah komputer yang memiliki kemampuan belajar tanpa diprogram secara eksplisit, program-program ini menggunakan data untuk membangun model dan mengambil keputusan berdasarkan model yang telah dibangun sebelumnya (Id, 2006).

2.1.3.2 *Deep Learning*

Bidang ini berfokus pada pembentukan dan pembelajaran mesin yang dapat melakukan tugas secara mandiri. Analisis citra gambar, pemrosesan teks, dan pengenalan suara adalah beberapa pekerjaan industri yang membutuhkan pembelajaran mendalam. Singkatnya, jaringan saraf tiruan adalah sebuah kelas algoritma yang terinspirasi oleh otak manusia dan merupakan bagian dari metode pembelajaran mesin yang menggunakan pemahaman mendalam (Courville, 2016).



Gambar 2.1 Ilmu Kecerdasan Buatan

(Courville, 2016)

2.1.3.3 *Artificial Neural Network*

Artificial Neural Network atau ANN adalah jaringan syaraf tiruan yang metode cara kerjanya mengikuti cara belajar manusia dengan meniru arsitektur syaraf otak dan menerapkannya pada perangkat lunak komputer. ANN bersifat *black box*, dalam arti proses kerjanya tidak dapat dilihat dengan jelas dari luar.

Lapisan ANN terdiri dari 3 jenis, yaitu *input layer, hidden layer* dan *output layer*:

1. Input Layer

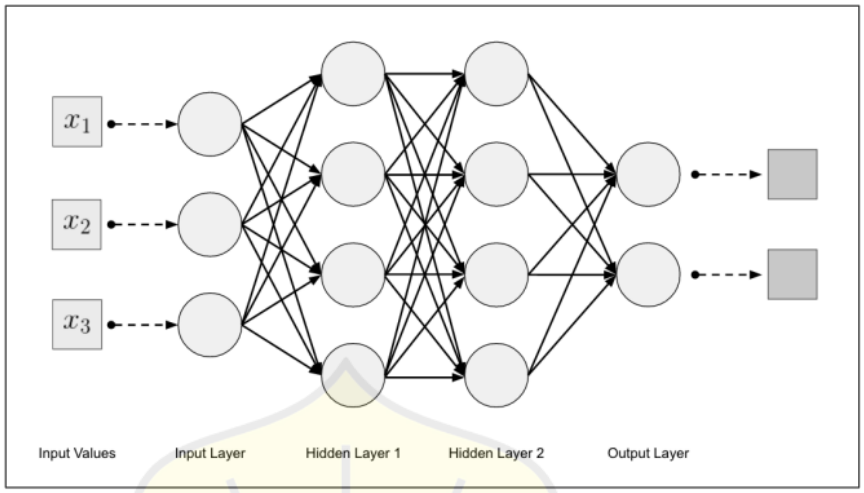
Input layer biasanya digunakan untuk memasukkan data masukan (vektor) ke dalam jaringan kita. Jumlah neuron pada *input layer* biasanya memiliki jumlah yang sama dengan jumlah fitur masukan dalam jaringan.

2. Hidden layer

Hidden layer berjumlah satu atau lebih dalam jaringan saraf, nilai bobot pada koneksi antar lapisan adalah bagaimana jaringan saraf mencari dan menyandikan informasi yang dipelajari yang diambil dari data pelatihan mentah. *Hidden layer* adalah kunci untuk memungkinkan jaringan saraf memodelkan fungsi nonlinier.

3. Output Layer

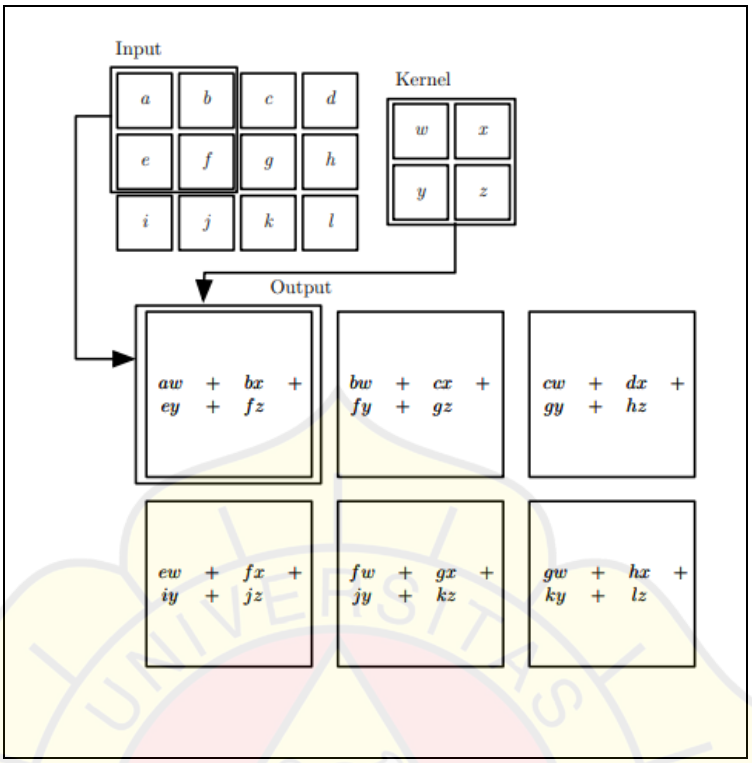
Output layer merupakan layer yang memberikan jawaban atau prediksi dari model yang diolah pada *hidden layer* (Patterson & Gibson, 2017).



Gambar 2.2 Arsitektur Neural Network
(Patterson & Gibson, 2017).

2.1.3.4 Convolutional Neural Network

Convolutional Neural Network (CNN) adalah jenis jaringan saraf yang dirancang untuk memproses data dengan topologi grid. Konvolusi adalah operasi matematika khusus yang digunakan dalam proses kerja *Convolutional Neural Network*. Ini digunakan sebagai pengganti matriks perkalian umum dan sering digunakan dalam perhitungan lapisan (Goodfellow et al., 2016).

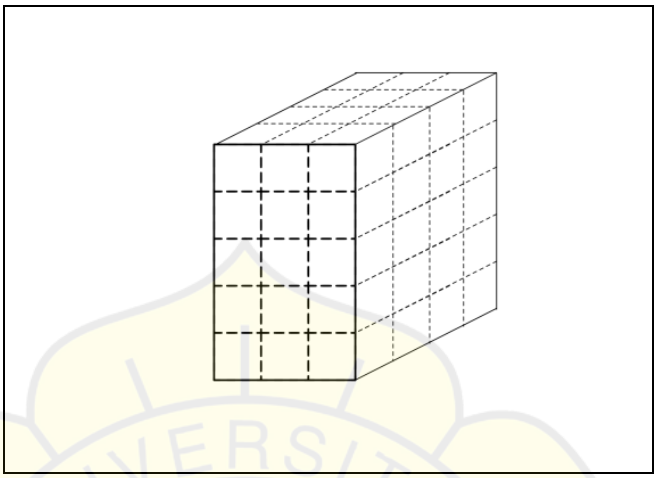


Gambar 2.3 Mekanisme perhitungan konvolusi 2-D
(Goodfellow et al., 2016)

Menurut (Patterson & Gibson, 2017) Fungsi dasar *Convolutional Neural Network* terbagi menjadi 4 bagian yaitu :

1. Input Layer

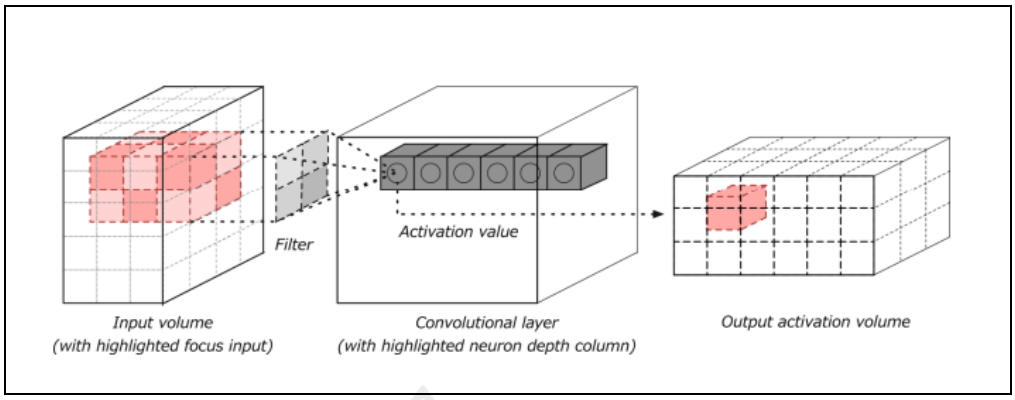
Input layer adalah lapisan yang digunakan untuk memuat dan menyimpan data masukan dari data mentah gambar untuk diproses dalam jaringan. Data masukan ini berupa data lebar, tinggi, dan jumlah saluran. Biasanya, jumlah saluran input layer adalah tiga, yang berisikan nilai RGB untuk setiap piksel.



Gambar 2.4 Input Layer dengan Volume 3D
(Patterson & Gibson, 2017).

2. Convolutional Layer

Lapisan konvolusional dianggap sebagai blok penyusun inti arsitektur CNN. Seperti yang diilustrasikan Gambar 2.5, lapisan konvolusional mentransformasikan data masukan dengan menggunakan satu petak neuron yang menghubungkan secara lokal dari lapisan sebelumnya. Lapisan akan menghitung perkalian titik antara wilayah *neuron* pada lapisan masukan dan bobotnya yang mana mereka terhubung secara lokal di lapisan keluaran.



Gambar 2.5 Lapisan Konvolusi Dengan Input dan Output Volume (Patterson & Gibson, 2017).

3. *Pooling Layer*

Lapisan ini berada di lapisan konvolusional yang terikat. *Pooling layer* mengurangi representasi data secara progresif melalui jaringan dan membantu mengontrol *overfitting*. Lapisan ini melakukan proses *downsampling* pada volume input disepanjang dimensi spasial memasukan data. Artinya jika gambar masukan berukuran lebar 32 piksel dan tinggi 32 piksel, gambar keluaran akan lebih kecil lebar dan tingginya menjadi lebar 16 piksel dan tinggi 16 piksel.

4. *Fully Connected Layer*

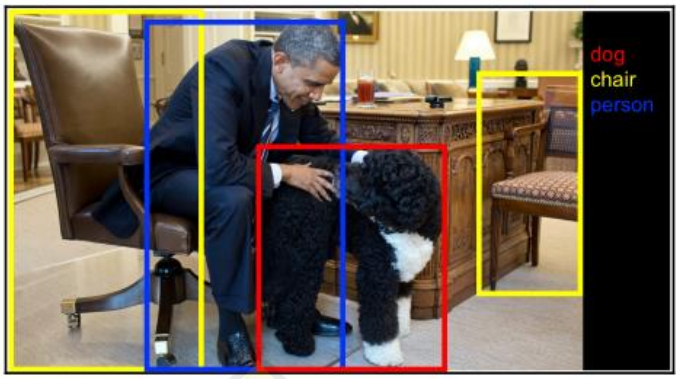
Fully Connected Layer digunakan untuk menghitung skor kelas yang akan digunakan sebagai *output* jaringannya, dimensi volume output adalah [1 x 1 x N]. Nilai N merepresentasikan jumlah kelas pada suatu dataset yang akan di uji.

2.1.3.5 Augmentasi Data

Augmentasi data adalah teknik untuk membuat data pelatihan baru secara artifisial dari yang sudah ada pada data pelatihan. Hal ini dilakukan dengan menerapkan teknik khusus domain pada contoh pelatihan data yang membuat contoh pelatihan baru dan berbeda. Augmentasi data gambar adalah jenis augmentasi data yang paling terkenal dan melibatkan pembuatan versi transformasi gambar dalam kumpulan data pelatihan yang termasuk dalam kelas yang sama dengan gambar asli. Proses ini dilakukan untuk manipulasi gambar, seperti menggeser, membalik, memperbesar, dan banyak lagi.

2.1.3.6 Deteksi Objek

Deteksi objek adalah proses menemukan lokasi suatu objek dalam suatu gambar. Perbedaan mendasar dengan klasifikasi gambar adalah klasifikasi gambar memberi label pada gambar secara keseluruhan. Sedangkan deteksi objek mampu melakukan proses untuk menemukan posisi benda selain label pada objek, proses tersebut biasa disebut lokalisasi objek. Biasanya, posisi objek ditentukan oleh koordinat persegi panjang. Menemukan banyak objek dalam gambar dengan koordinat persegi panjang disebut deteksi (Gollapudi, 2019).

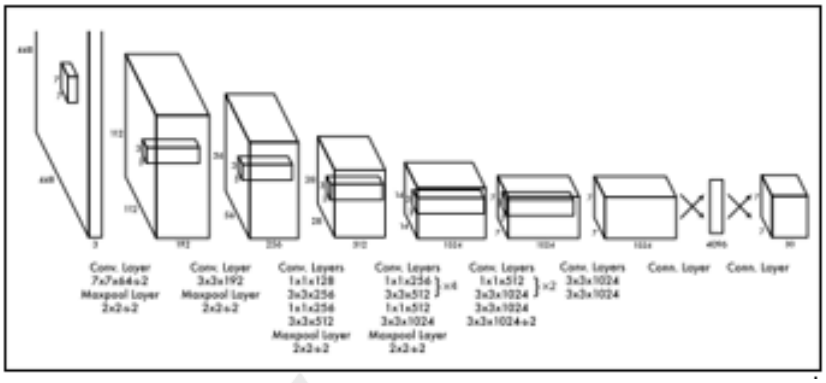


Gambar 2.6 Deteksi Objek Deep Learning
(Gollapudi, 2019)

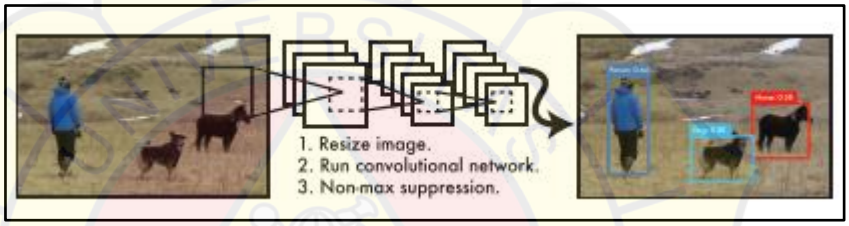
2.1.3.7 You Only Look Once (YOLO)

YOLO merupakan sebuah model deteksi objek secara *real time*, model ini bekerja dengan membagi data gambar yang masuk menjadi kisi-kisi sebuah sel, setiap sel bertanggung jawab untuk memprediksi kotak pembatas jika pusat kotak pembatas berada di dalamnya. Setiap sel grid memprediksi kotak pembatas yang melibatkan koordinat x, y serta lebar dan tinggi dan nilai *confidence*. Prediksi dari suatu kelas juga didasarkan pada setiap sel. Misalnya, sebuah gambar dibagi menjadi grid 7 × 7 dan setiap sel dalam grid dapat memprediksi 2 kotak pembatas, maka selanjutnya YOLO akan menghasilkan 98 prediksi kotak pembatas yang diusulkan.

YOLO merubah tugas deteksi objek menjadi masalah regresi tunggal dengan mengolah piksel gambar langsung menjadi probabilitas kelas dan koordinat kotak pembatas. Sistem ini hanya memerlukan satu kali melihat gambar untuk mengenali jenis dan lokasi objek. Semua komponen deteksi objek disatukan dalam satu jaringan saraf oleh YOLO. Dalam memprediksi setiap kotak pembatas, YOLO menggunakan fitur keseluruhan gambar untuk setiap kelas objek yang ada, sehingga mempertimbangkan gambar secara menyeluruh dan semua kelas objek sekaligus. Ini menunjukkan bahwa YOLO memperhitungkan keseluruhan gambar, termasuk setiap objek di dalamnya. Model ini memproses data gambar yang dimasukan menjadi kotak berukuran S×S. Apabila pusat suatu objek terletak di dalam salah satu petak, petak tersebut akan mendeteksi objek tersebut. Setiap petak menghasilkan prediksi untuk B kotak pembatas beserta nilai kepercayaannya; nilai ini menunjukkan seberapa akurat prediksi kotak tersebut dan seberapa yakin bahwa kotak tersebut berisi objek (Redmon et al., 2016).



Gambar 2.7 Arsitektur YOLO
(Redmon 2016)



Gambar 2.8 Proses Deteksi YOLO
(Redmon 2016)

Arsitektur YOLO terdiri dari 24 lapisan konvolusi dengan dua lapisan yang benar-benar terhubung. Lapisan reduksi 1×1 digunakan sebelum lapisan konvolusi 3×3. Jaringan lengkap arsitektur ini digambarkan pada Gambar 2.7.

YOLO menggunakan fungsi aktivasi linier untuk lapisan akhir dan *bounding box* semua lapisan lainnya menggunakan aktivasi *leaky rectified* berikut ini:

$$Pr(\text{Class i}|\text{Object}) * Pr(\text{Object}) * \text{IOUtruth/pred} = \text{Pr(Class i)} * \text{IOUtruth/pred} \quad (2.1)$$

YOLO memiliki banyak keuntungan dibandingkan dengan metode lain. Pertama dan terpenting, model ini memiliki keunggulan

kecepatan untuk mengubah deteksi objek menjadi sederhana dan tidak membutuhkan proses yang kompleks. YOLO hanya menggunakan gambar untuk memprediksi deteksi dengan CNN. Dibandingkan dengan sistem *real-time* lainnya, YOLO memiliki kemampuan untuk membaca objek video secara langsung dengan latensi sangat cepat sekitar 25 milidetik serta tingkat akurasi rata-rata *(MAP)* lebih dari dua kali lipat.

Kedua, saat YOLO membuat prediksi, mereka mempertimbangkan gambar secara keseluruhan. Ini berbeda dengan pendekatan sliding window dan area proposal. Karena metode deteksi terbaik, seperti *Fast R-CNN*, tidak dapat mengenali objek secara luas dan menghasilkan lebih sedikit kesalahan deteksi *background* daripada Fast R-CNN.

terakhir, model ini memiliki kemampuan untuk beradaptasi dan menerapkan pengenalan objek baru. Model ini berfungsi dengan baik ketika diterapkan ke domain baru atau input yang tidak diduga karena kemampuan generalisasi yang luar biasa (Redmon et al., 2016).

2.1.3.8 Confusion Matrix

Confusion Matrix adalah tabel yang menunjukkan jumlah data uji yang benar diklasifikasikan dan jumlah data uji yang salah diklasifikasikan merupakan metode yang biasanya digunakan untuk melakukan perhitungan akurasi konsep *data mining*.

Tabel 2.1 *Confusion matrix*

Kebenaran Klasifikasi	Klasifikasi Berdasarkan	
	Prediksi "+"	Prediksi "-"
Aktual "+"	True Positive	False Negative
Aktual "-"	False Positive	True Negative

- *True Positives (TP)* adalah jumlah *record* data positif yang diklasifikasikan sebagai nilai positif.
- *False Positives (FP)* adalah jumlah *record* data negatif yang diklasifikasikan sebagai nilai positif.
- *False Negatives (FN)* adalah jumlah *record* data positif yang diklasifikasikan sebagai nilai positif.
- *True Negatives (TN)* adalah jumlah *record* data negatif yang diklasifikasikan sebagai nilai negative.

2.1.3.9 Mean Average Precision

Mean Average Precision adalah Nilai rata-rata *precision* (AP), yang merupakan metrik evaluasi untuk mengukur kinerja deteksi objek, diperoleh dari perhitungan *precision* dan perhitungan *recall.*

1.7.1 CRISP-DM

2.1.4.1 Pemahaman Bisnis

Setelah mempelajari tujuan dan kebutuhan proyek dari sudut pandang bisnis, fase awal ini mengubah pengetahuan tentang data menjadi definisi masalah penambangan data dan rencana awal yang dirancang untuk mencapai tujuan.

2.1.4.2 Pemahaman Data

Fase ini dimulai dengan pengumpulan data awal dan dilanjutkan dengan tindakan seperti memahami jenis data, menemukan masalah kualitas data, dan menemukan informasi menarik dari bagian data untuk membuat hipotesis tentang informasi tersembunyi.

2.1.4.3 Persiapan Data

Tahap ini diperlukan untuk membuat dataset akhir dari data mentah awal yang akan digunakan dalam alat pemodelan termasuk dalam tahap persiapan data, yang biasanya dilakukan berulang kali dan tidak dalam urutan yang tetap. Tahap persiapan data juga mencakup pemilihan tabel, catatan, dan atribut, serta transformasi dan pembersihan data untuk keperluan pemodelan.

2.1.4.4 Permodelan

Pada tahap ini, berbagai metode pemodelan dipilih dan diterapkan, dan parameter disesuaikan untuk mencapai nilai yang ideal. Normalnya, ada sejumlah teknik yang dapat diterapkan untuk setiap jenis masalah *data mining*, tetapi beberapa teknik membutuhkan jenis data tertentu. Akibatnya, untuk mendapatkan model yang baik, seringkali diperlukan untuk kembali ke tahap persiapan data.

2.1.4.5 Evaluasi

Pada titik ini, setelah membuat model yang sesuai dengan kebutuhan, dilanjutkan dengan evaluasi dan peninjauan langkah-langkah pembuatan untuk memastikan bahwa itu sesuai dengan tujuan bisnis dan untuk menemukan masalah bisnis yang belum terselesaikan. Agar dapat melanjutkan ke tahap selanjutnya, keputusan harus dibuat tentang bagaimana hasil data mining akan digunakan.

2.1.4.6 Development

Pada tahap ini model yang telah disetujui masuk pada fase pengembangan menjadi sebuah aplikasi baik berupa web atau aplikasi, model diterapkan langsung sehingga klien dapat menggunakannya.

2.1.5 Aplikasi dan Library Pendukung

2.1.5.1 TensorFlow

TensorFlow merupakan library *machine learning* untuk skala besar dan mampu melakukan proses perhitungan berat. *Tensorflow* dibuka untuk umum pada November 2015 oleh *Google* dan sudah banyak digunakan dalam banyak project *machine learning* maupun *deep learning*. *TensorFlow* mampu melakukan berbagai macam tugas pembelajaran seperti klasifikasi gambar, pemrosesan bahasa alami, pembuatan sistem rekomendasi, prediksi penjualan dan masih banyak lagi. *TensorFlow* tidak hanya dapat di jalankan di *Windows*, *Linux*, dan *MacOS*, tetapi *TensorFlow* juga dapa dijalankan pada perangka seluler menggunakan *TensorFlow Lite* (Géron, 2019).

2.1.5.2 Pyhton

Python merupakan bahasa pemrograman yang muncul pada tahun 1991, sejak kemunculannya *python* telah menjadi salah satu program penafsiran paling populer. *Python* sering disebut bahasa scripting, karena dapat digunakan untuk menulis program kecil dengan cepat, dan menjalankan skrip untuk mengotomatiskan tugas lainnya. Sekarang *python* menjadi salah satu bahasa pemrograman terpenting untuk ilmu data, seperti *machine learning*, *deep learning* dan bahasa

pemrograman untuk pengembangan perangkat lunak umum di dunia akademis dan industri (Mckinney, 2017).

2.1.5.3 Flutter

Flutter adalah SDK *opensource* yang dibuat oleh *Google, Flutter* adalah SDK yang benar-benar lengkap untuk membuat sebuah aplikasi. *Flutter* adalah sebuah platform yang menyediakan banyak fitur dalam membangun sebuah aplikasi. *Flutter* menggunakan bahasa pemrograman *Dart* dalam proses pengoperasiannya, *Dart* sangat populer digunakan oleh *mobile developer* dalam membangun sebuah aplikasi (Rischpater, 2020)

2.1.5.4 Visual Studio Code

Visual Studio Code adalah sebuah platform kode editor yang dikembangkan oleh *Microsoft* dan memiliki penghargaan sebagai kode editor terpopuler dalam survei *Stack Overflow* pada tahun 2019. *Visual Studio Code* memiliki dukungan bawaan hanya untuk *JavaScript, HTML, CSS* namun tetap mendukung bahasa pemrograman lainnya seperti *Python* dan harus menginstall lewat ekstensi (Speight, 2021).

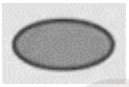
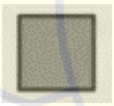
2.1.6 Unified Modeling Language

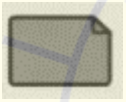
Bahasa pemodelan perangkat lunak *Unified Modeling Language* (UML) adalah sebuah bahasa yang telah distandardisasi sebagai media penulisan cetak biru perangkat lunak (Pressman). Visualisasi, spesifikasi, kontruksi, dan dokumentasi beberapa komponen sistem perangkat lunak biasanya dilakukan dengan UML. UML digunakan untuk membantu *programmer* atau *developer* dalam membangun perangkat lunak(Abdillah, 2021).

2.1.6.1 Use Case Diagram

Salah satu komponen UML adalah use case diagram, yang digunakan untuk memvisualisasikan fungsionalitas yang diharapkan dari sebuah sistem. Sebuah use case menunjukkan bagaimana seseorang berinteraksi dengan sistem. Sebuah tugas tertentu, seperti mengakses sistem, membuat laporan, dan sebagainya, disebut *use case*. Seorang aktor adalah orang atau mesin yang berinteraksi dengan sistem untuk melakukan dan menyelesaikan tugas tertentu. Saat menyusun persyaratan sistem, berkomunikasi dengan klien, dan merancang *test case* untuk setiap fitur sistem, diagram ini sangat membantu

Tabel 2. 2 Simbol Use Case Diagram

No	Simbol	Nama	Keterangan
1		Actor	Representasi entitas manusia dalam berinteraksi atau berkomunikasi bersama sistem
2		User Case	Aktivitas atau tindakan yang dilakukan oleh aktor dalam sistem.
3		Association	Hubungan aktor dalam kasus penggunaan.
4		System	Sistem yang sedang di kembangkan
5		Include	Hubungan antara dua kasus penggunaan di mana satu kasus penggunaan memasukkan atau mengandung fungsionalitas dari yang lain
6		Extend	Merupakan hubungan antara dua kasus penggunaan, Di mana satu kasus

			penggunaan memasukkan atau mengandung fungsionalitas dari yang lain
7		Generalization	Hubungan di mana satu kasus penggunaan mewarisi perilaku dari kasus penggunaan lain
8		Collaboration	Dua atau lebih actor atau use case yang terhubung
9		Note	Penjelasan tambahan tentang aktivitas yang terjadi
10		Anchor	Hubungan teks note dengan simbol diagram lain

2.1.6.2 Activity Diagram

Tahap ini menggambarkan proses dari sebuah sistem yang dirancang, dimulai dari proses berawal, keputusan yang akan terjadi, dan poses sistem selesai. Selain itu, diagram aktivitas dapat menampilkan proses bersamaan yang akan terjadi pada berbagai proses yang berjalan. Diagram aktivitas adalah bagian diagram yang khusus di mana sebagian besar diagram adalah tindakan dan transisi yang berasal dari bagian sebelumnya. Akibatnya, aktivitas diagram

tidak secara akurat memvisualisasikan proses sistem berjalan dalam keadaan internal dan koneksi antar subsistem, tetapi lebih memvisualisasikan proses dan alur aktivitas dari tingkat atas keseluruhan.

Tabel 2. 3 Simbol Activity Diagram

1		Start	Simbol untuk menandai awal aktivitas sistem di mulai
2		End	Simbol untuk menandai akhir aktivitas sitem
3		Activity	Simbol untuk memberikan informasi aktivitas atau tindakan yang sedang di lakukan oleh sistem
4		Decision	Simbol apabila terdapat satu atau lebih aktivitas yang sedang berjalan
5		Join	Simbol untuk penggabungan aktivitas

1.8 Kajian Penelitian Terdahulu

Tabel dibawah ini merangkum referensi penelitian sebelumnya menggunakan metode *deep learning* untuk deteksi objek jalan berlubang.

Tabel 2.4 Kajian atau Jurnal Penelitian Terdahulu

No	Judul	Penulis	Tahun terbit	Klasifikasi
1	Deteksi Kerusakan Jalan Menggunakan Pengolahan Citra Deep Learning di Kota Semarang	Sasmito, et all	2023	Jurnal Ilmiah Bidang Ilmu Kerekayasaan, TEKNIK, vol. 44, no. 1, pp. 7-14, May. 2023. e-ISSN: 240-9919, S2
Metode: Deep Learning YOLO Hasil: Penelitian ini bertujuan untuk membuat sebuah model yang dapat melakukan deteksi objek jalan berlubang menggunakan teknologi deep learning. Kesimpulan dari penelitian ini adalah akurasi deteksi objek jalan berlubang 88% dan akurasi kappa 86%. Ini juga menunjukkan nilai tes uji rata-rata akurasi sebesar 41% dan nilai kehilangan rata-rata sebesar 0,6428.				
2	Pendeteksian Lubang Pada Jalanan Menggunakan Metode SSD-MobileNet	Pakpahan & Dewi	2021	Indonesia Journal of Electronics and Instrumentation Systems (IJEIS), Vol. 11 No. 2, October 2021, pp 213-222 ISSN : 2460-7681, S2
Metode: Deep Learning SSD-MobileNet Hasil :. Penelitian ini bertujuan untuk menentukan jenis dan angka hyperparameter terbaik untuk mendapatkan akurasi yang tinggi. Penelitian ini dilakukan dengan menggunakan tiga jenis dataset yakni dataset normal yang berasal dari kaggle, dataset dashboard dan closeup. Penelitian ini dilakukan dengan 3 variasi hyperparameter learning rate, yakni 0.004, 0.0004, dan 0.00004. Kesimpulan penelitian ini adalah akurasi tertinggi terdapat pada				

dataset closeup sebesar 76%, dataset normal sebesar 56% dan dataset dashboard sebesar 50%.				
3	Deteksi Objek Lubang pada Citra Jalan Raya menggunakan Pengolahan Citra Digital	Yusuf Budiarto & Sutikno	2017	Jurnal Komputer Terapan Vol. 3, November 2017, 109-118, ISSN : 2443-4159, S4
Metode : Backpropagation Hasil : Penelitian ini dilakukan dengan jumlah dataset dan learning rate berbeda, untuk dataset pelatihan yang tersedia adalah sebanyak 75, 150, dan 300 gambar jalan berlubang. Sedangkan untuk proses parameter learning rate yang di uji adalah sebebsar 0,9, 0,5 dan 0,1. Kesimpulan pada penelitian ini adalah model dengan data pelatihan sebanyak 75 masih belum maksimal dalam mendeteksi objek di bandingkan dengan data uji pelatihan sebanyakan 150, dan 300.				
4	SISTEM DETEKSI KONDISI JALAN MENGGUNAKAN METODE Z-DIFF PADA SMARTPHONE ANDROID	Putri et all	2018	Jurnal Telematika Vol. 11 No. 2 Agustus 2018 -ISSN : 2442 - 4528 65, S2
Metode : Z-Diff Hasil : Data yang dikumpulkan dari sensor accelerometer disaring dalam penelitian ini menggunakan metode alghoritma Z-diff. Penelitian ini menemukan 281 titik jalan berlubang di beberapa kecamatan di timur Semarang yang dianggap memiliki tingkat kerusakan jalan yang tinggi. Hasil pengukuran presisi, recall, dan F-measure menunjukkan akurasi 79%.				
5	DETEKSI JALAN BERLUBANG MENGGUNAKAN METODE GREY LEVEL CO-OCCURRENCE MATRIX DAN	Mahrus ali	2022	Jurnal Kecerdasan Buatan, Komputasi dan Teknologi Informasi, Vol. 3 No.1 Tahun 2022 ISSN : 2774-7875,S2

	NEURAL NETWORK			
Metode : Gray Level Co-occurrence Matrix & Neural Network Hasil : Dalam penelitian ini, metode Gray Level Co-occurrence Matrix (GLCM) dan Neural Network digunakan untuk mengklasifikasikan data video untuk mengembangkan sistem deteksi jalan berlubang. Deteksi terdiri dari dua langkah. Ciri-ciri gambar jalan diekstraksi, dan label manual diberikan untuk instruksi. Selanjutnya, nilai dari data pelatihan digunakan untuk menguji data gambar. Pengujian Matriks Kekacauan menunjukkan hasil Recall sebesar 0,80, hasil Ketepatan sebesar 0,06, hasil Ketepatan sebesar 0,79, dan hasil Kesalahan Rate sebesar 0,20.				