BAB II
DASAR TEORI

2.1 KAJIAN PUSTAKA

Terkait dengan penelitian yang dilakukan untuk pendeteksian objek berupa wajah manusia, perlu melakukan peninjauan literatur dari penelitian terdahulu untuk mengetahui model atau arsitektur yang terbaik untuk digunakan. Pada penelitian kali ini menggunakan algoritma YOLO dalam pendeteksian objeknya, namun sebenarnya terdapat banyak opsi yang dapat digunakan untuk tujuan mendeteksi objek pada sebuah citra atau gambar, yaitu dengan CNN berbasis TensorFlow, Viola Jones dan *You Only Look Once* (YOLO). Berikut beberapa tinjauan pustaka terkait dengan algoritma untuk deteksi objek berupa wajah beserta alasan penggunaan algoritma YOLO pada penelitian ini.

Penelitian mengenai algoritma CNN dilakukan oleh Wulan Anggraini dengan judul penelitian yaitu "*Deep Learning* Untuk Deteksi Wajah Yang Berhijab Menggunakan Algoritma *Convolutional Neural Network* (CNN) Dengan TensorFlow". Pada penelitian tersebut, digunakan tiga skenario *training model* dengan jumlah data *training* yang berbeda-beda, yaitu 100, 250, dan 300 dengan gambar berupa wajah yang menggunakan hijab. Pada skenario pertama dengan data *training* 100 mendapatkan akurasi sebesar 87% dengan waktu yang dibutuhkan yaitu 950 detik atau sekitar 15 menit. Kemudian, data *training* berjumlah 250 mendapatkan akurasi sebesar 88% dengan waktu yang dibutuhkan hampir sama seperti sebelumnya, yaitu 956 detik. Terakhir adalah data *training* berjumlah 300 yang mendapatkan akurasi sebesar 90% dengan waktu yang dibutuhkan yaitu 1002 detik atau sekitar 16 menit. Dari hasil tersebut CNN telah berhasil mendeteksi wajah yang menggunakan hijan dengan baik, namun membutuhkan waktu *training* yang cukup lama yaitu sekitar 15-16 menit [18].

Kemudian, terdapat penelitian mengenai metode Viola Jones yang dilakukan oleh Adinda Rizkita Syafira & Gunawan Ariyanto dengan judul "Sistem Deteksi Wajah Dengan Modifikasi Metode Viola Jones", pada penelitian ini memakai metode Viola Jones dengan menggunakan deskriptor fitur Haar untuk mendeteksi wajah pada gambar. Metode ini juga memanfaatkan *Integral Image* dan Adaboost untuk memilih dan memilah nilai fitur yang signifikan dalam

membentuk Cascade *Classifier.* Cascade *Classifier* tersebut digunakan untuk mendeteksi wajah pada gambar. Untuk mengevaluasi akurasi sistem, penelitian ini memodifikasi nilai parameter pada metode Viola Jones dan menguji performa sistem menggunakan K-fold *cross validation.* Hasil uji coba menunjukkan bahwa akurasi yang dicapai adalah dengan rentang 84-90,9% untuk gambar wajah dan 65-75,5% untuk gambar bukan wajah dengan menggunakan parameter berupa lima *classifier*, citra yang diberikan memerlukan nilai minimum tinggi/lebar fitur sebesar 8 piksel, dan nilai maksimum tinggi/lebar fitur sebesar 10 piksel [19].

Kemudian, seperti model yang digunakan pada penelitian kali ini yaitu YOLO, terdapat juga referensi dari penelitian sebelumnya mengenai hal ini, seperti yang dilakukan oleh Lusi Susanti, dkk. yang berjudul "Sistem Absensi Mahasiswa Berbasis Pengenalan Wajah Menggunakan Algoritma YOLOv5", dalam penelitian ini, digunakan 5 *dataset* yang terdiri dari 5 mahasiswa Fakultas Ilmu Teknik di Universitas Bina Insan sebagai sampel. Hasil yang didapatkan menunjukkan bahwa sistem memiliki akurasi sekitar 80% dalam mendeteksi wajah. Dataset tersebut terdiri dari sekitar 1500 gambar dengan ukuran 640x640 piksel dan dievaluasi menggunakan 16 batch dengan epoch sebanyak 100, dengan memperhatikan nilai mAP, Precision, dan Recall. Dalam penelitian ini, sistem berhasil mencapai nilai mAP, Precision, dan Recall sebesar 99%, yang menunjukkan bahwa kinerja sistem sudah cukup baik [20].

Terdapat juga penelitian mengenai deteksi wajah menggunakan YOLO seperti yang dilakukan oleh Yessi Hartiwi, dkk. yang membuat sistem manajemen presensi dengan fitur pendeteksi objek berupa wajah menggunakan algoritma YOLO pada *platform* Android dengan ditambahkan fitur GPS, hasil dari pengujian tersebut untuk akurasi tertinggi sebesar 0.93435 dan terendah masih dalam *range* 93%, sedangkan nilai rerata akurasi adalah 93.26% dari 20 data penilaian yang dilakukan sistem menggunakan YOLO, data yang digunakan berupa gambar yang memiliki ukuran 416×416 piksel [21].

Berdasarkan beberapa penelitian di atas, terdapat beberapa opsi yang dapat dipilih untuk model deteksi wajah. Salah satunya adalah YOLO, YOLO digunakan pada penelitian kali ini, karena diyakini memiliki tingkat akurasi yang cukup tinggi, yaitu sekitar 90% keatas. Kemudian, gambar yang digunakan

sebagai data *train* dan data *test* relatif memiliki ukuran yang tinggi pikselnya dibandingkan dengan metode Viola Jones. Oleh karena itu, pada penelitian ini, akan melakukan pendeteksian wajah manusia dengan model YOLO.

2.2 DASAR TEORI

Pada sub-bab ini akan membahas mengenai teori yang berkaitan dengan penelitian, yaitu antara lain:

2.2.1 YOLO (*You Only Look Once*)

Pada bidang AI atau kecerdasan buatan, terdapat ilmu yang mempelajari cara bagaimana sebuah sistem komputer dapat melihat dan mengenali objek seperti layaknya manusia, ilmu tersebut dinamakan *Computer Vision* atau CV. Salah satu contoh penelitian mengenai CV adalah YOLO (*You Only Look Once*), YOLO adalah sebuah sistem yang digunakan untuk mengenali sebuah objek pada suatu citra gambar atau video dengan menggunakan *repurpose classifier* atau *localizer*. YOLO menerapkan sistem lokasi skala pada sebuah citra, sehingga, jika terdapat sebuah daerah yang memiliki skala atau skor yang lebih tinggi, berarti YOLO memiliki keyakinan bahwa daerah tersebut terdapat sebuah objek yang perlu dideteksi [22].

YOLO dalam mendeteksi sebuah objek pada suatu citra menggunakan pendekatan ANN (*Artificiall Neural Network*) atau Jaringan Syaraf Tiruan (JST). Pendekatan dengan JST artinya adalah membuat sebuah citra menjadi beberapa bagian dengan menerapkan *grid*, sehingga dari kotak-kotak yang dihasilkan dari *grid* tersebut akan diprediksi dan dibandingkan dengan setiap probabilitas yang akan diprediksi disetiap *grid*-nya [23].

Menurut Redmon, dkk. YOLO memiliki beberapa keunggulan dan manfaat, yaitu:

1. Algoritma YOLO dalam memproses sebuah citra, menggunakan *grid* untuk memisahkan objek atau wilayah citra lainnya, sehingga dikenal lebih cepat dalam mendeteksi sebuah objek.
2. Pada algoritma yang lain, biasanya melakukan iterasi untuk memproses suatu gambar, hal ini berbeda dengan YOLO, yang mana untuk melihat atau memproses suatu gambar, perlu melakukan sebuah proses *training* dan pengujian, sehingga lebih akurat.

3. Algoritma YOLO juga mempelajari sebuah representasi objek yang dapat digeneralisasikan [24].

Algoritma YOLO memiliki beberapa poin penting agar berjalan dengan baik, diantaranya adalah:

1. *Dataset*

   Dataset adalah sekumpulan atau himpunan data dalam satu rumpun yang sama, *dataset* juga digunakan untuk menyimpan informasi-informasi penting mengenai objek yang akan dipilih untuk dijadikan bahan *training.*

2. Anotasi citra

   Dalam memproses sebuah citra, diperlukan sebuah anotasi terlebih dahulu, pada YOLO sendiri, anotasi diberikan dengan menggunakan *bounding box* beserta memberikan nama kelas pada objek sesuai dengan kebutuhan pada sebuah citra.

3. *Training dataset*

   YOLO perlu melewati tahap *training dataset*, yaitu sebuah tahap untuk melatih sistem agar mengenali objek melalui sebuah *dataset* citra yang telah diberikan anotasi berupa *bounding box* pada setiap objeknya. *Training dataset* ini terdapat beberapa poin yang perlu diketahui, yaitu:

   - *Batch size*

     Batch size adalah banyaknya data yang digunakan pada setiap sesi pelatihan *dataset*, semakin banyak data yang digunakan, ada peluang untuk semakin baik hasilnya.

   - *Epoch*

     Epoch adalah banyaknya proses pengulangan atau iterasi pada tahap pelatihan *dataset.*

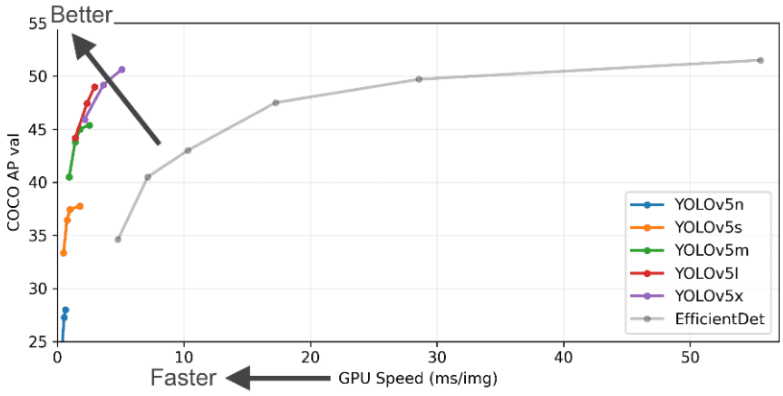
   - *Learning rate*

     Learning rate adalah kecepatan dari proses pelatihan *dataset*, jika lebih cepat, maka semakin mudah untuk mendeteksi sebuah objek [25].

2.2.2 Arsitektur YOLOv5

Awal dari perkembangan YOLO adalah pada tahun 2015, yang dikembangkan oleh Redmon dan Ali Farhadi di University of Washington (AS) [24], pada saat itu, YOLO yang dikembangkan masih sangat rendah akurasinya dibandingkan dengan YOLO masa kini.

Terdapat salah satu YOLO yang dikembangkan oleh Ultralytics [26] (bukan oleh pengembang asli YOLO) dan cukup populer digunakan hingga saat ini, yaitu YOLOv5 yang ditambahkan beberapa fitur di dalamnya. YOLOv5 sendiri memiliki beberapa jenis, salah satunya YOLOv5s yang digunakan pada penelitian kali ini, berikut tampilan grafik perbandingan performa dari berbagai macam jenis YOLOv5.



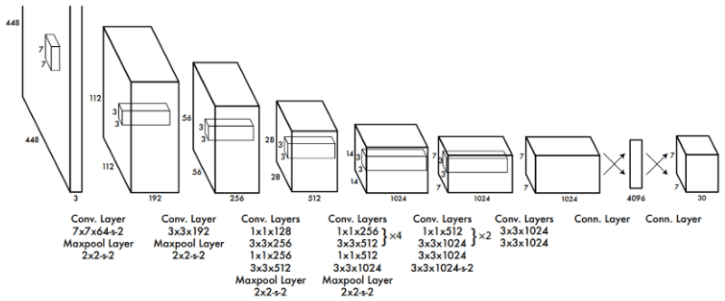
Gambar 2.1 Perbandingan performa berbagai macam jenis YOLOv5 [26].

Terlihat pada grafik di atas, bahwa berbagai macam jenis YOLOv5 memiliki kecepatan GPU yang sangat baik, itu menandakan bahwa proses *training* data, atau proses pendeteksian data bisa dilakukan dengan lebih efisien, sehingga tidak memerlukan *resource* yang sangat banyak. Pada grafik juga terlihat untuk data COCO AP val mendapatkan nilai yang cukup bagus. Oleh karena itu, pada penelitian kali ini, menggunakan YOLOv5.

Dalam pendeteksian objek menggunakan jaringan syaraf YOLO, komponen-komponen terpisah digabungkan menjadi satu, dengan memanfaatkan fitur-fitur dari seluruh gambar, YOLO dapat memprediksi kotak pembatas untuk setiap objek yang muncul dalam gambar secara bersamaan, termasuk untuk semua kelas objek, dengan demikian, YOLO dapat mempertimbangkan seluruh area gambar secara global dan memprediksi semua objek pada gambar.

Dalam YOLO, gambar *input* akan dibagi menjadi petak-petak (*grid*) berukuran S x S. Setiap petak bertanggung jawab untuk mendeteksi objek jika pusat objek berada di dalam petak tersebut.

Dalam YOLO, setiap sel di dalam petak akan memprediksi B kotak pembatas dan menentukan nilai keyakinan untuk setiap kotak. Nilai keyakinan ini menunjukkan seberapa yakin kotak tersebut berisi objek dan seberapa akurat kotak tersebut diprediksi [27]. Ilustrasi arsitektur dasar YOLO, terlihat pada gambar di bawah.



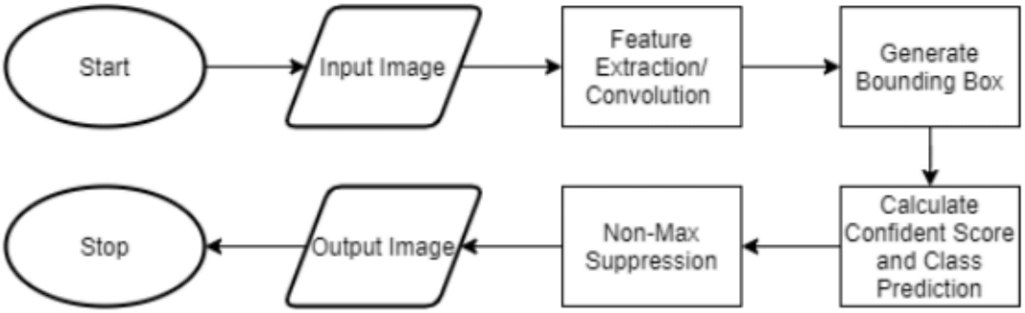
Gambar 2.2 Arsitektur dasar YOLO [27].

Proses pengenalan pada tahap pendeteksian wajah menggunakan citra yang telah diproses pada tahap sebelumnya. Langkah-langkah dalam proses ini adalah sebagai berikut:

1. Normalisasi ukuran citra.
2. Menghitung *boot* pada *convolution layer.*
3. Menghilangkan linearitas yang ada pada hasil *convolution layer*.
4. Mengecilkan *feature map* (*polling*).
5. Mengubah nilai dari sebuah matriks dua dimensi menjadi matriks satu dimensi (*flatten*).
6. Menggabungkan semua proses di atas (*Fully Connected Layer*).
7. Data hasil pelatihan dari metode *Fully Connected Layer* akan disimpan ke dalam *database.* Data tersebut akan digunakan sebagai *database* pada tahap pengujian.
8. Pada tahap pengujian, data dari metode *Fully Connected Layer* akan melalui proses *softmax* untuk menghasilkan data hasil pengenalan ekspresi wajah [28].

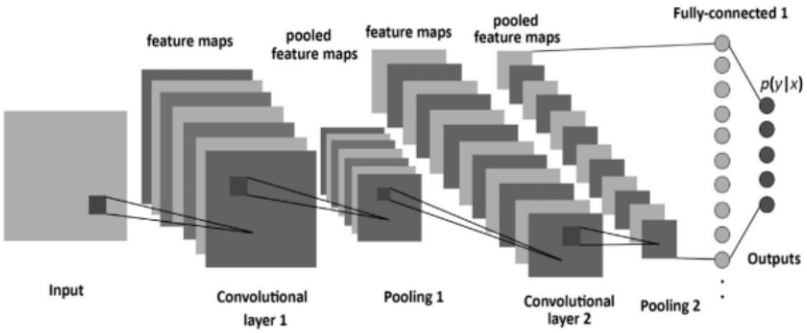
2.2.3 Metode *Feature Map* pada YOLO

Pada YOLO, terdapat sebuah metode untuk mengklasifikasikan objek pada suatu citra, cara tersebut dengan menggunakan *feature map*. Pada dasarnya, sistem deteksi dimulai dengan menerima sebuah gambar masukan. Gambar tersebut akan dilakukan ekstraksi fitur menggunakan jaringan yang digunakan. Kemudian, setiap fitur hasil ekstraksi akan digunakan untuk memprediksi kotak pembatas dan probabilitas kelas yang ada di dalamnya. Hasil *output* yang dihasilkan adalah sebuah gambar dengan kotak pembatas dan label kelas hasil prediksi [29]. Alur deteksi gambar pada YOLO terlihat seperti gambar di bawah.



Gambar 2.3 Alur kerja pendeteksian YOLO [29].

Sebelum algoritma YOLO dilatih dan melakukan pendeteksian objek, data yang diberikan ke algoritma YOLO perlu proses anotasi data terlebih dahulu. Anotasi yang dilakukan adalah memberikan objek pada gambar dengan sebuah *bounding box* yang mengandung titik koordinat X objek, titik koordinat Y objek, Panjang *bounding box,* lebar *bounding box* dan juga nama kelas [28]. Berikut ilustrasi *feature maps* pada YOLO dengan *layer* konvolusi.



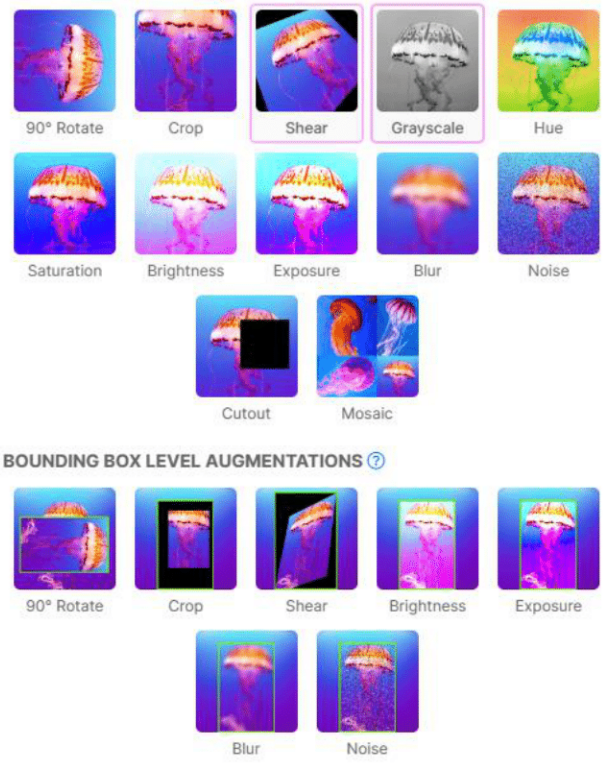
Gambar 2.4 *Feature maps* pada YOLO dengan *layer* konvolusi [28].

Arsitektur YOLO menggunakan *layer* konvolusi di dalamnya. Biasanya disebut dengan *Convolutional Neural Network* (CNN) atau ConvNet. CNN

terdapat beberapa *layer*, yaitu diantaranya adalah *input layer, output layer* dan *hidden layer*. Pada *hidden layer* terdapat sebuah *layer* yang terdiri dari *layer* ekstraksi fitur dan *layer* klasifikasi [28].

2.2.4 Roboflow

Roboflow ialah sebuah platform *web* yang menawarkan berbagai macam fungsi terkait *dataset*. Dengan menggunakan Roboflow, pengguna dapat membagikan dan memproses *dataset* sekaligus. Beberapa fitur yang dimanfaatkan dalam penelitian ini antara lain melakukan anotasi objek dengan *bounding box* untuk deteksi, serta melakukan *pre-processing* pada *dataset* seperti *augmentasi* menggunakan Roboflow. Terdapat beberapa jenis augmentasi konvensional yang disediakan oleh Roboflow, seperti *flip*, *rotate*, *brightness*, *exposure*, *shear* dan masih banyak lagi [30]. Berikut tampilan fitur pada *web* Roboflow.



Gambar 2.5 Fitur pada Roboflow [30].

2.2.5 *Artificial Intelligence* (AI)

Artificial Intelligence (AI) atau biasa disebut dengan kecerdasan buatan adalah sebuah sistem yang dibuat oleh manusia agar berperilaku seperti layaknya manusia, yaitu dapat berpikir, dapat melihat dan dapat membaca serta menulis. Menurut Michael Haenlein dan Andreas Kaplan, AI diciptakan untuk membuat

sistem memiliki kemampuan dalam menafsirkan data dengan benar, data tersebut dipelajari sehingga dapat digunakan untuk mencapai sasaran dan pekerjaan tertentu dengan cara menyesuaikan diri secara fleksibel melalui suatu proses adaptasi [31].

Sedangkan menurut Schell dan Mc Leod, AI adalah sebuah aktivitas mesin seperti komputer yang memiliki kemampuan untuk menirukan perilaku yang cerdas dari seorang manusia [32].

Secara garis besar, AI perlu menirukan cerdas yang dilakukan oleh manusia, seperti contohnya adalah:

1. Dapat bertindak seperti manusia atau *acting humanly*.
2. Dapat berpikir seperti manusia atau *thinking humanly*.
3. Dapat berpikir secara rasional atau *thinking rationally*.
4. Dapat bertindak secara rasional atau *act rationally*.

2.2.6 *Machine Learning & Deep Learning*

Machine Learning dan *Deep Learning* erat kaitannya dengan kecerdasan buatan. *Machine Learning* atau ML adalah sebuah sub-bidang pada AI yang berkaitan dengan pengembangan dan desain algoritma. *Machine learning* juga adalah sebuah teknik pembelajaran untuk meningkatkan performa sistem dari *experience* hasil pelatihan atau *training* data. Pada beberapa kasus, ML akan menjadi sangat kompleks dalam penggunaannya, sehingga pada kasus tersebut, kemampuan mesin perlu ditingkatkan agar bisa meniru cara kerja otak manusia, pada keadaan ini akan disebut dengan *deep learning* [33].

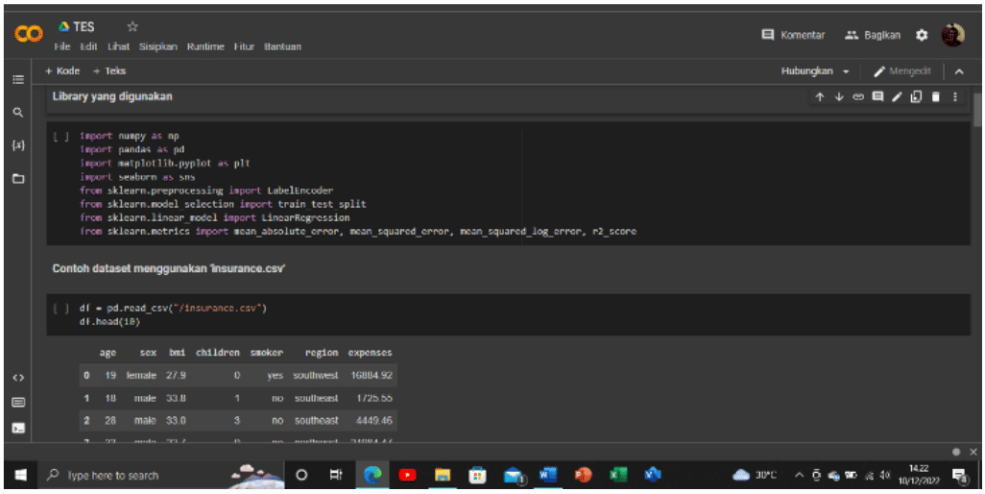
Konsep sederhana dari *deep learning* adalah meniru cara kerja dari otak manusia dengan Jaringan Syaraf Tiruan (JST) atau ANN (*Artificiall Neural Network*) yang arsitekturnya sangat beragam. *Deep learning* akan digunakan ketika membutuhkan komputasi yang lebih tinggi dan mencakup semua perilaku yang dilakukan oleh manusia, seperti contohnya dapat melihat dengan *computer vision* (kemampuan sistem komputer mengenal objek pada citra), dapat mengenal bahasa melalui suara dengan *speech recognition* (mengenal data suara) dan dapat mengenal bahasa melalui teks dengan *Natural Language Processing* (mengenal data teks) [33].

Terdapat banyak sekali arsitektur *deep learning* yang telah digunakan dan dijelaskan pada beberapa studi, contohnya adalah *Deep Neural Networks* (DNNs), *Convolutional Neural Network*s (CNNs), *Recurrent Neural Networks* (RNNs), *Deep Boltzmann Machines* (DBMs), *Deep Autoencoders* (DAs), *Deep Trust Network* (DBNs) dan lain-lain. Secara konsep, *deep learning* adalah sebuah cara komputasi sistem untuk mempelajari sebuah representasi data yang diberikan dengan meningkatkan tingkat abstraksi [34].

2.2.7 Google Colaboratory (Colab)

Perusahaan teknologi ternama Google, mengembangkan sebuah aplikasi *executable document* berbasis *web* yang dapat digunakan untuk menulis, menyimpan, serta membagikan program melalui Google Drive, yaitu Google Colaboratory atau lebih akrab disebut dengan Google Colab [35]. Berikut tertampil logo dari Google Colab dan tampilan dari Google Colab.

Gambar 2.6 Logo Google Colab [36].



Gambar 2.7 Tampilan Google Colab [36].

Aplikasi Google Colab sangat cocok untuk *programmer* pemula yang baru memulai bahasa pemrograman Python, karena penggunaannya yang mudah, karena berbasis *web* sehingga tidak perlu memasang aplikasi apapun di komputer,

pada dasarnya aplikasi ini serupa dengan Jupyter Notebook, hanya saja diciptakan berbentuk *cloud* yang dioperasikan melalui *browser,* sehingga tidak memberatkan kinerja perangkat yang dimiliki, dan dapat digunakan dimanapun dan perangkat apapun [35].

2.2.8 Bahasa Pemrograman Python

Terdapat berbagai macam bahasa pemrograman yang digunakan, salah satunya adalah bahasa pemrograman Python. Bahasa pemrograman ini cukup populer di kalangan *programmer* pemula hingga profesional, karena relatif mudah untuk dipelajari, bahasa pemrograman ini diciptakan oleh *programmer* Belanda yaitu Guido Van Rossum. Python adalah salah satu bahasa pemrograman yang mengeksekusi sejumlah instruksi atau perintah multifungsi secara langsung (interpretatif) dengan metode OOP atau berorientasi pada objek [37].

Python memiliki dukungan yang banyak dari *programmer,* sehingga memiliki segudang fitur yang cocok untuk digunakan. Python juga memiliki tata penulisan *script* yang mudah untuk dipahami, dan dapat mengelola data dan memori secara otomatis. Python juga dapat diaplikasikan pada berbagai sistem operasi seperti Linux, Windows, MacOS, Android, Amiga, Palm, Symbian OS dan lain-lain [37].

2.2.9 *Library*

Pada umumnya, untuk memudahkan penggunaan bahasa pemrograman Python, maka digunakanlah sebuah *library* untuk memuat sebuah paket yang berisi ribuan modul terkait. *Library* Python adalah sebuah kumpulan modul atau data yang berada pada satu fungsi atau rumpun yang sama dan berisi banyak sekali kode yang dapat digunakan berulang kali pada program yang berbeda sesuai dengan kebutuhan [38].

Pada Python, terdapat beberapa contoh *library* yang populer digunakan, khususnya untuk keperluan *data science,* berikut lima contoh *library* tersebut.

1. NumPy

   NumPy atau singkatan dari *Numerical Python* adalah sebuah paket *library* yang fundamental digunakan untuk fungsi numerik pada pemrograman Python. NumPy sendiri diberi dukungan di Github dengan kurang lebih sebanyak 700 kontributor. NumPy adalah sebuah

paket dengan tujuan yang general untuk *array-processing* dengan menyediakan sebuah objek multidimensi yang berkinerja tinggi. NumPy juga menyediakan operator dan fungsi yang beroperasi dengan efisien pada *array* yang dimiliki oleh NumPy, salah satu fungsi utama dari NumPy adalah untuk proses numerik dan operasi komputasi yang melakukan pendekatan dengan *array.*

2. Matplotlib

   Library Matplotlib adalah sebuah *library plotting* yang biasanya digunakan untuk visualisasi data, *library* ini didukung sekitar 700 kontributor di Github. *Library* ini juga dapat membuat grafik dan plot yang berguna untuk keperluan *data science*. Matplotlib juga menyediakan API yang dapat digunakan ke dalam sebuah aplikasi.

3. Pandas

   Library Pandas atau singkatan dari Python *data analysis* adalah sebuah *library* yang digunakan untuk menganalisis sebuah data pada pemrograman Python, biasanya juga akan selalu ada pada *lifecycle data science* sehingga cukup populer di kalangan *scientist data*. Di Github, *library* ini memiliki dukungan kontributor dengan jumlah 1200 kontributor, Pandas juga menyediakan struktur data analisis yang fleksibel dan dapat digunakan secara cepat seperti *data frame* CD, yang memang dirancang untuk digunakan dengan data yang terstruktur secara intuitif dan cepat.

4. SciPy

   SciPy atau singkatan dari *Scientific* Python, adalah sebuah *library* yang digunakan untuk komputasi tinggi dengan tujuan melakukan analisis data. *Library* ini memiliki kontributor sebanyak 600 orang di Github, dan menyediakan berbagai macam perhitungan yang dapat digunakan secara efisien.

5. TensorFlow

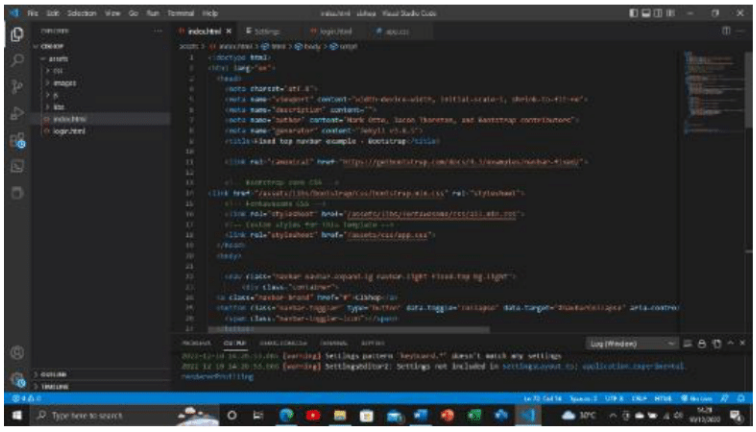
   Library TensorFlow adalah sebuah *library* yang menjalankan sebuah komputasi pada program dengan melibatkan tensor pada sebuah objek yang sudah ditentukan, yang kemudian diharapkan dapat

mengeluarkan sebuah informasi yang bernilai dan berguna. *Library* ini memiliki dukungan lebih dari 1.500 orang di Github [39].

2.2.10 Visual Studio Code

Salah satu aplikasi *code editor* yang populer adalah Visual Studio Code, aplikasi ini juga dapat digunakan pada semua *desktop*, sehingga dapat digunakan pada hampir semua *Operating System* seperti Windows, Linux dan Mac. Visual Studio Code sendiri dikembangkan oleh perusahaan Microsoft [40]. Berikut adalah contoh logo dan tampilan dari Visual Studio Code.

Gambar 2.8 Logo Visual Studio Code [41].



Gambar 2.9 Tampilan Visual Studio Code [41].

Pada Visual Studio Code terdapat banyak sekali ekstensi yang dapat digunakan untuk menunjang produktivitas, dari berbagai macam ekstensi yang ada, tentu saja memiliki fungsi dan kegunaan yang berbeda-beda sesuai dengan kebutuhan. Beberapa contoh ekstensi yang populer digunakan adalah *live server, auto rename tag* dan *error lens* [42].

2.2.11 Bahasa Pemrograman PHP

Pada umumnya, dalam pengembangan *web* pasti menggunakan bahasa pemrograman PHP dalam penggunaannya. Bahasa pemrograman PHP adalah sebuah bahasa pemrograman yang bersifat *open-source* dan berfungsi sebagai

server side scripting. PHP akan menjalankan instruksi pemrograman saat proses *runtime* berjalan. Hasil dari instruksi tersebut tentu akan berbeda sesuai dengan apa yang diproses [43].

Pada saat ini, terdapat kurang lebih 78% situs *web* menggunakan bahasa pemrograman PHP ini, salah satu contohnya adalah perusahaan sosial media ternama yaitu Facebook juga menggunakan bahasa PHP dalam mengembangkan situs *web*-nya. Bahasa PHP sendiri diciptakan oleh Rasmus Lerdorf pada tahun 1995. PHP menjadi sangat populer, karena terdapat beberapa faktor di dalamnya, yaitu:

1. Materi pembelajaran mengenai bahasa PHP sangat banyak tersebar.
2. Relatif mudah untuk dipelajari.
3. Bahasa PHP sendiri bersifat *open-source*.
4. Memiliki proses komputasi dengan kecepatan yang cukup tinggi.
5. *Database* cukup banyak yang mendukung bahasa PHP, mulai dari MySQL hingga *non-relational database* seperti Redis.
6. Memiliki kompatibilitas yang baik dengan HTML.
7. Dapat secara fleksibel digabungkan dengan bahasa pemrograman yang lain.
8. Dapat digunakan pada *multiplatform,* sehingga dapat dioperasikan di berbagai macam OS.
9. Bahasa PHP memiliki perkembangan yang cukup bagus, dikarenakan setiap tahunnya pasti akan ada pembaruan yang dilakukan. Hingga saat ini, PHP sudah memiliki berbagai macam versi, yang terbaru adalah versi 7.4.
10. Mendukung layanan *cloud* [43].

2.2.12 Postman

Terdapat sebuah aplikasi yang berfungsi sebagai *REST CLIENT* untuk uji coba *REST* API yang dinamakan dengan Postman. Aplikasi ini biasa digunakan oleh pengembang aplikasi yang membuat sebuah API. Kemudian, diuji cobakan dengan Postman dari API yang telah *developer* buat [44].

Selain untuk uji coba API yang telah dibuat oleh *developer,* Postman juga dapat membantu proses *development* API dengan beberapa fitur berikut, yaitu:

1. *Collection*

   Postman terdapat fitur *collection* yang berguna untuk mengelompokkan permintaan atau *request* API dalam bentuk folder, sehingga memudahkan *developer* dalam mengelompokkan *request* API sesuai dengan kebutuhan pengembangan yang sedang dikerjakan oleh *developer*. Fitur ini juga dapat merapihkan dan memudahkan API, sehingga mudah untuk digunakan oleh *developer*.

2. *Environment*

   Request API yang terdapat di Postman, dapat dimanipulasikan dengan mengatur *config* untuk menyimpan *attribute*. Fitur untuk memanipulasi *request* API tersebut dinamakan dengan *environment.*

3. *Response*

   Fitur *response* ini berfungsi untuk membuat *mockup* API sebelum *developer* mengaplikasikannya ke dalam sebuah proyek.

4. *Mock Server*

   Fitur ini digunakan ketika *mockup* API pada fitur *response* sudah dapat dianggap oleh *developer* layak untuk di-*deploy*. Pada saat fitur ini digunakan, akan berpengaruh pada *mockup* API yang dapat diakses menggunakan internet.

5. *Script Test*

   Fitur ini digunakan untuk melakukan uji coba dengan melakukan validasi respon dan melakukan tes dengan menuliskan *script test* sesuai dengan kebutuhan *developer* [45].

2.2.13 API (*Application Programming Interface*)

API adalah singkatan dari *Application Programming Interface*. API adalah sebutan untuk sebuah *tools* yang digunakan untuk menyambungkan atau sebagai perantara agar dapat saling berkomunikasi dari aplikasi ke sebuah *server* yang memiliki suatu fungsi dengan berbagai *platform,* baik pada *platform* yang serupa, ataupun berbeda [46]. Terdapat beberapa jenis API, yaitu:

1. *Public* API

   API *public* atau *open* API, adalah istilah API yang digunakan dalam lintas *platform* yang berbeda.

2. *Private* API

   Sesuai dengan namanya, API jenis ini hanya digunakan untuk keperluan internal dalam sebuah perusahaan, sehingga tidak dapat digunakan oleh umum atau tertutup. API ini biasanya juga digunakan oleh pengembang individu yang masih dalam tahap pembelajaran atau pemula.

3. *Partner* API

   API jenis ini, hanya bisa digunakan oleh pihak yang sudah bermitra atau memiliki izin dari pengembang API aslinya, sehingga tidak dapat digunakan jika tidak memiliki izin untuk menggunakannya.

4. *Composite* API

   API jenis ini berfungsi untuk *storage* data dari berbagai *server* dalam satu tempat [47].

2.2.14 Android Studio

Android Studio adalah sebuah aplikasi *multi-platform* yang dikembangkan oleh Google untuk membuat aplikasi berbasis Android, dengan jenis aplikasi berupa *editor* kode IDE atau *Integrated Development Environment* yang didasarkan pada Intellij IDEA. Terdapat banyak sekali fitur yang dapat digunakan pada Android Studio, seperti emulator Android, mengembangkan UI aplikasi berbasis Android, menerapkan perubahan kode ke aplikasi yang sedang berjalan tanpa me-*restart* aplikasi, hingga dapat diintegrasikan dengan GitHub, serta menyediakan banyak *library* yang dapat digunakan untuk mengelola dan mengakses komponen perangkat pada *smartphone* Android seperti sensor [48].