Kaji Awal Pendeteksi Api Menggunakan Kamera dengan Program Machine Learning

Khusni Mubarok[1], Teguh Wibowo[2], Singgih Satrio Wibowo [3]

[1]Jurusan Teknik Mesin, Politeknik Negeri Bandung, Bandung 40012
E-mail : khusni.mubarok.aer19@polban.ac.id

ABSTRAK

Pendeteksi api sangat beragam, jenisnya, dari mulai yang mendeteksi api dari asap, karbon monoksida hingga secara visual dengan kamera. Mendeteksi api dengan visual mempunyai keuntungan tersendiri yaitu memiliki waktu respon yang relative cepat dari api dinyalakan. Artikel ini menjelaskan mengenai pembuatan sebuah detektor api yang menggunakan kamera sebagai sensor. Pembuatan detektor api menggunakan *machine learning* dengan Tensorflow, kemudian menghubungkan kamera dengan program menggunakan Open-CV. Semua itu dilakukan dengan Bahasa Python yang ditulis di sebuah teks editor Bernama Visual Studio Code. Model diberi *input* gambar berbagai foto api dengan posisi dan *background* yang berbeda-beda kemudian di-*training* sebanyak 5000 kali. Hasilnya, model bisa mendeteksi api sejauh satu meter lebih hanya dalam waktu kurang lebih tiga detik.

Kata Kunci :
Fire detection, Python, machine learning

1. PENDAHULUAN

Sensor api atau kebakaran adalah sebuah perangkat yang memberikan pemberitahuan jika terdapat api atau asap di suatu tempat. Pemberitahuan itu bisa beragam caranya seperti melalui *buzzer* maupun lampu berwarna. Cara mendeteksi api pun beragam, ada yang menggunakan asap sebagai parameter, karbon monoksida maupun api itu sendiri. Yang paling sering digunakan di berbagai tempat umum adalah detektor asap. Seluruh jenis pendeteksi api atau asap memiliki kelebihan dan kekurangannya masing-masing, salah satu contohnya adalah detektor asap atau karbon monoksida yang biasa digunakan di langit-langit, maka harus menunggu sebagian asap mengenai alat itu terlebih dahulu untuk kemudian diidentifikasi sebagai asap kebakaran. Hal itu cenderung memakan waktu lama untuk menunggu asap sampai pada pendeteksi. Berbeda dengan menggunakan pandangan secara visual komputer. Dengan cara memasang kamera pada salah satu sisi dinding, api akan langsung terdeteksi saat kamera menangkap gambar dari api tersebut, dengan demikian, waktu untuk api terdeteksi akan lebih cepat. Namun, pendeteksi api dengan kamera juga memiliki kekurangan yaitu tidak bisa mendeteksi api yang masih tertutup dengan benda lain, artinya, jangkauan kamera memang cepat namun tidak bisa menembus pandang pada api yang berada di balik suatu benda yang menutupinya. Namun hal itu bisa dikurangi dengan cara memasang lebih dari satu kamera dengan sudut pandang yang berbeda dari kamera lain.

Oleh karena itu, perlunya untuk mengetahui factor-faktor apa saja yang memengaruhi kecepatan deteksi daripada pendeteksi api dengan menggunakan kamera, agar bisa meminimalisir kesalahan saat pemasangan maupun pembuatan detektor api dengan kamera.

2. TINJAUAN PUSTAKA

2.1 Dasar Pemrograman

2.1.1 Algoritma

Algoritma berarti [1] langkah-langkah logis yang disusun sistematis guna menyelesaikan masalah. Artinya setiap langkah dalam lagoritma harus jelas dan memiliki kejelasan antara benar atau salahnya. Berikut adalah lima ciri algoritma:

a. Algoritma harus memiliki ujung atau bisa dibilang garis finish setelah melakukan sesuatu.

b. Setiap langkah harus dimengerti oleh pengguna atau si penulis program.

c. Algoritma memiliki nol atau lebih masukan(input).

d. Algoritma memiliki nol atau lebih keluaran(output).

e. Algoritma harus efektif.

2.1.2 Algoritma pada Komputer

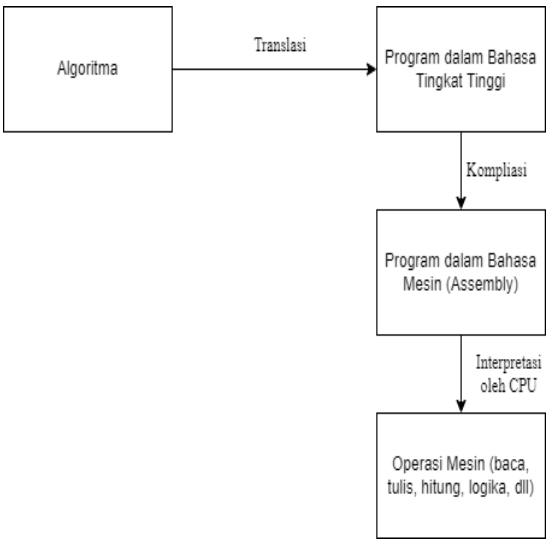
Pemrograman pada komputer [1] memberikan interuksi secara sistematis dan berurutan pada komputer. Dalam hal ini, komputer hanyalah sebagai pemroses yang dapat diperintah sesuai masukan yang kita berikan. Bahasa pemrograman lah yang bisa menjadi jembatan untuk pengguna dalam memerintah komputer.

Pada komputer modern, pemrosesan algoritma yang telah ditulis membutuhkan empat komponen komputer yaitu prosesor, memori, perangkat input, dan perangkat output. Prosesor berguna untuk memproses instruksi sementara memori berguna untuk penyimpanan sementara data yang akan dijalankan oleh prosesor. Perangkat *input* berguna untuk menerima masukkan dari pengguna berupa data yang dikirimkan melalui komponen tertentu seperti *mouse, keyboard* dan lain-lain. Sementara *output* merupakan hasil dari input yang dimasukkan, *output* bisa berupa perintah, LCD dan sebagainya.

Komputer memiliki bahasa *Assembly* sebagai bahasa yang bisa ia mengerti sendiri. Bahasa *Assembly* disebut bahasa tingkat rendah karena merupakan bahasa mesin dengan sintaks-sintaks rumit komputer yang jauh dari bahasa manusia.

Oleh karena sulit untuk memahami bahasa *assembly*, terciptalah bahasa pemrograman yang lebih ramah bagi manusia di era digital seperti sekarang ini. Bahasa yang diciptakan pun beragam dan mempunyai kegunaan masing-masing. Dalam tingkatannya, bahasa tersebut dibagi menjadi tiga yaitu: Bahasa tingkat rendah, bahasa tingkat menengah dan bahasa tingkat tinggi. Di mana semakin tinggi tingkatan bahasa, maka akan semakin mudah dipahami oleh manusia.

Lalu bagaimana komputer dapat mengerti bahasa tingkat tinggi ini, sedangkan komputer hanya mengerti bahasa assembly. Dalam kasus ini harus memerlukan adanya “penerjemah" bahasa dari bahasa tingkat tinggi ke bahasa tingkat rendah.



Gambar 2.1 Alur algoritma [1]

Dalam dunia pemrograman perangkat lunak yang bertugas "menterjemahkan bahasa tingkat tinggi ke bahasa tingkat rendah disebut juga sebagai “Translator" atau "Kompilator". Tugas kompilator adalah membuat sebuah file program yang isinya adalah instruksi-instruksi dalam kode biner atau assembly yang sesuai dengan algoritma yang dibuat, sehingga komputer dapat langsung mengerti dan menjalankan program tersebut. Urutan pemrosesan algoritma pada gambar di atas dapat dijelaskan sebagai berikut:

a. Algoritma dalam ide manusia ditranslasikan ke dalam bahasa tingkat tinggi menggunakan suatu editor teks atau software pemrograman.

b. Instruksi dalam bentuk teks yang berisi bahasa tingkat tinggi dikompilasi oleh Kompilator menjadi file berisi instruksi-instruksi bahasa mesin atau bahasa assembly.

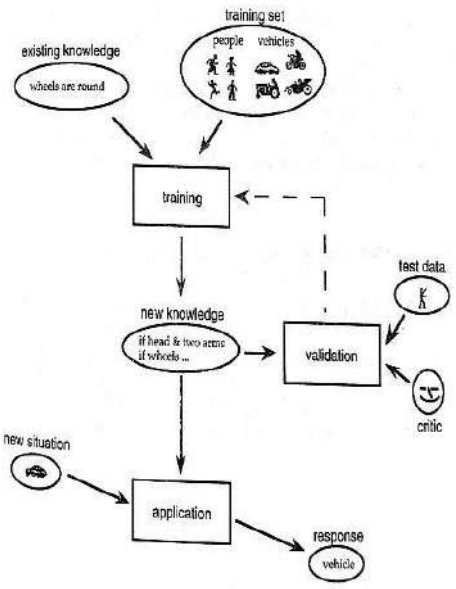
c. Bahasa mesin atau bahasa assembly tersebut diinterpretasi oleh komputer atau CPU menjadi aksi-aksi dalam komputer.

2.2 Python

Python merupakan bahasa pemrograman tingkat tinggi yang diracik oleh Guido van Rossum. Python banyak digunakan untuk membuat berbagai macam program, seperti: program CLI, Program GUI (desktop), Aplikasi Mobile, Web, IoT, Game, Program untuk *Hacking*, dan sebagainya. Python juga dikenal dengan bahasa pemrograman yang mudah dipelajari, karena struktur sintaknya rapi dan mudah dipahami.

2.3 Machine Learning

Machine Learning [2] adalah sebuah cabang dari AI (Artificial Intelegence) atau bisa dibilang sebagai kecerdasan buatan. Intinya adalah memberikan pelatihan pada mesin untuk kemudian dipelajari olehnya, di mana tahap dari pembelajaran tersebut terdapat pada gambar di bawah ini.



Gambar 2.2 Tahap *machine learning* [2]

2.4 Tensorflow

Tensorflow diluncurkan oleh Google [3] dengan maksud memperkenalkan suatu ekosistem yang menyediakan sebuah alur kerja untuk melatih serta mengembangkan model sesuai keinginan pengguna. Google juga bertujuan agar bisa mengimplementasikan *machine learning* ke hampir semua aplikasi. Sebenarnya, secara sadar atau tidak, penggunakan Tensorflow itu sudah banyak di kehidupan dekat atau dalam genggaman ponsel seperti Google Photo atau Google Voice. Model ini bekerja pada kelompok besar perangkat keras Google. Komponen iniiti Tensorflow adalah tensor dan grafik komputasi yang melintasi *node* hingga *edge.*

2.5 Open CV

Open CV [4] , atau singkatan dari *Open Source Computer Vision* adalah sebuah alat yang dikembangkan oleh intel untuk keperluan *computer vision.* Open CV ini merupakan *library* gratis yang bisa siapapun gunakan, oleh karena itu sangat mudah mendapatkannya. Hanya dengan mengetikkan perintah install Open CV pada Python, maka *library* tersebut langsung akan bisa terinstall.

Sebenarnya di dalam Open CV sudah memiliki banyak fitur pendeteksi yang telah dibuat untuk hal-hal umum seperti pendeteksi wajah, pengenalan wajah dan masih banyak jenis metode *Artificial Intelegent* yang lain yang berkaitan dengan *computer vision.*

Pada 1999, Intel meluncurkan Open CV namun masih memerlukan *library* dari Intel *image processing* untuk dapan menjalankan fungsi Open CV secara optimal. Namun akhirnya *dependency* itu akhirnya dihilangkan sehingga terciptalah Open CV yang bisa berdiri sendiri sebagai *library*. *Library* tersebut saat ini sudah bisa mencakup banyak *platform* penting seperti Windows, Linux, Mac, bahkan Android.

Dalam hal *computer vision*, Open CV memiliki peran sangat penting, hal itu didukung oleh fitur canggih yang juga dimiliki *library* ini seperti yang disubutkan di bawah:

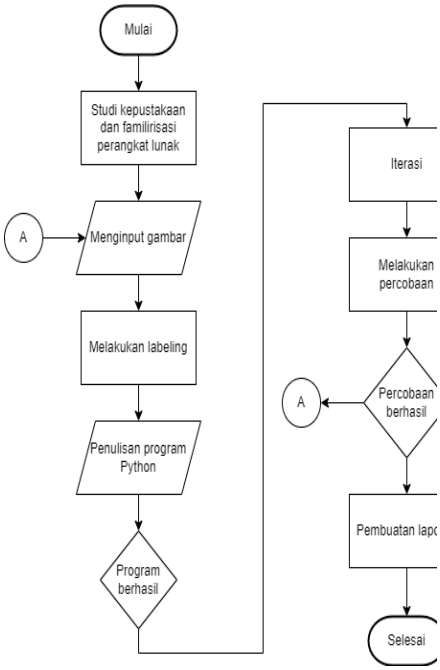
a. *Image and video I/O*, dengan ini dapat dilakukan pembacaan sebuah data gambar dari sebuah *file* di mana biasanya digunakan untuk identifikasi gambar maupun video.
b. *Computer Vision* adalah pengolahan data secara langsung dari kamera yang terhubung dengan program. Hasil daripada *Computer vision* bisa didapatkan dari *machine learning* maupun dari sebuah *file* yang memuat data-data.
c. Metode untuk AI dan *Machine Learning*, seperti yang telah disebutkan pada poin sebelumnya, Open CV bisa sebagai wadah untuk menjalankan hasil daripada *machine*

learning. Lebih tepatnya untuk *real time detection.*

3. METODE PENELITIAN

Metode penyelesaian masalah pada Jurnal ini dilakukan dengan perangkat lunak Visual Studio Code dengan menggunakan bahasa Python versi 3.8.6. Di dalam Python tersebut juga membutuhkan Tensorflow sebagai machine learning dan OpenCV sebagai pembaca video.

Aplikasi di atas digunakan untuk membuat program kamera pendeteksi api yang akan dijabarkan secara ringkas pada diagram *flowchart* di bawah ini.



Gambar 3.1 Diagram alir penyelesaian Jurnal

3.1 Studi Kepustakaan dan Familiarisasi Perangkat Lunak

Studi kepustakaan mencakup pencarian referensi jurnal juga mempelajari materi yang akan digunakan dalam menyelesaikan Jurnal ini. Setelah itu, dilakukanlah penginstalan perangkat lunak yang dibutuhkan. Dalam hal ini, penginstalan aplikasi mencakup Visual Studio Code dan Python.

Visual Studio Code berfungsi sebagai teks editor yang dikhususkan untuk menulis program, sehingga penulisan program pada aplikasi tersebut lebih mudah. Sementara Python sendiri berfungsi sebagai bahasa yang digunakan dalam program yang akan dibuat di Jurnal ini.

3.2 Menginput Gambar dan Melakukan Labeling

Gambar dimasukkan ke dalam sebuah folder kemudian dilakukan *labeling*, *labelling* adalah sebuah proses pemisahan *background* dengan benda yang akan dideteksi, dalam hal ini api.

3.3 Penulisan Program Python

Program ditulis di dalam Visual Studio Code. Program menjalankan fungsi-fungsi utama seperti membuat direktori, mengistal ekstensi yang diperlukan, juga yang terpenting adalah menjalankan *training.*

3.3.1 Penentuan Tensorflow Zoo

Salah satu hal yang penting dalam pembuatan program pendeteksi api dalam Artikel ini adalah penentuan modul Tensorflow yang akan digunakan. Dalam penelitian yang dilakukan, penulis menggunakan 'ssd_mobilenet_v2_fpnlite_640x640' yang memiliki kecepatan 39 ms dan mAP 29,2.

3.3.2 Membuat label

Sebuah label dengan format ".pbtxt" memuat informasi kepada system dalam hal apa saja yang harus dideteksi. Dalam penelitian ini, hanya perlu satu ID untuk satu nama yang digunakan agar program bisa berlatih untuk mengetahui objek di dalam label tersebut.

3.3.3 Membuat file config

File "config" atau jika diperpanjang bisa dikatakan *configuration* adalah sebuah *file* informasi mengenai pengaturan sebelum memulai proses *training.* Hal itu termasuk penentuan modul Tensorflow dan label yang digunakan.

3.3.4 Melakukan training

Setelah semua hal di atas siap, maka hal terahir yang perlu dilakukan adalah melakukan *training.* Dalam hal ini, model diberikan *training* sebanyak 5000 kali atau bisa sampai mendapatkan nilai *lose* yang rendah. Nilai *lose* bisa didapatkan setelah melakukan evaluasi kemduain menggunakan *library* Tensorboard.

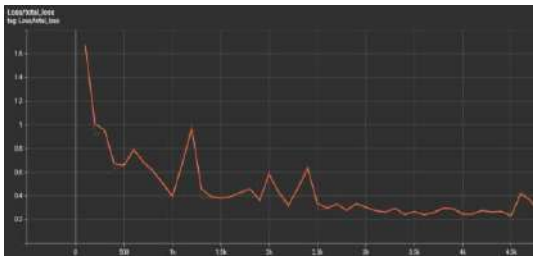
3.3.5 Menggunakan OpenCV

Setelah proses *training* selesai, maka tingga; melakukan penelitian dengan Open CV sebagai pembaca video secara *real time*.

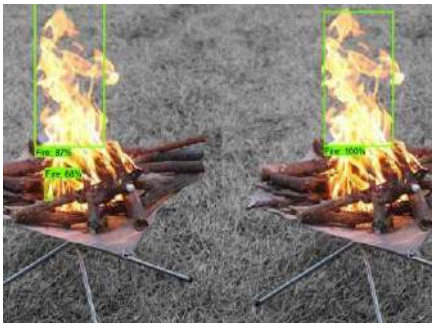
4. HASIL DAN PEMBAHASAN

4.1 Hasil

Setelah melakukan kegiatan *machine learnning* pada bab sebelumnya, akan dihasilkan sebuah grafik di mana dalam grafik tersebut menunjukkan nilai *loss.* Untuk mendapatkan grafik tersebut ada berbagai cara, salah satunya adalah dengan menuliskan perintah "tensorboard –logdir=." pada folder "Tensorflow\workspace\models\my_ssd_mobnet\train". Kemudian akan muncul beberapa grafik. Di antara grafik -grafik tersebut, yang terpenting adalah grafik dengan nama *total loss*, di mana di dalam grafik tersebut terdapat keseluruhan *loss* daripada modul yang sudah di-*training.* Semakin kecil *loss* maka akan semakin baik pula modelnya. Grafik *total loss* yang didapat dari percobaan serta hasil *train* dalam bentuk contoh gambar adalah sebagai berikut.



Gambar 4.1 Grafik nilai *loss*



Gambar 4.2 Hasil train pada Tensorboard

Setelah mengetahui grafik yang didapatkan dari hasil *Machine Learning*, selanjutnya yaitu melakukan pengujian terhadap api secara langsung, di mana dalam percobaan ini, api dinyalakan dalam jarak yang berbeda-beda. Dan yang dinilai adalah kecepatan waktu dalam mendeteksi setelah api dinyalakan. Hasil dari percobaan tersebut menggunakan kamera dengan spesifikasi 0.3 MP, hasil percobaan seperti yang tertera pada tabel IV-1 dan Tabel 4-2 di bawah ini.

Tabel 4-1 Hasil pengujian dengan cahaya 130 lux

No.	Jarak (cm)	Waktu respon (detik)
1	20	2
2	40	3
3	60	5
4	80	3
5	100	3

Tabel 4-2 Hasil pengujian dengan cahaya 2 lux

No.	Jarak (cm)	Waktu respon (detik)
1	20	4
2	40	2
3	60	2
4	80	2
5	100	2

4.2 Pembahasan

Dari hasil gambar IV.1 yang merupakan hasil dari *Machine Learning*, nilai *loss* semakin berkurang dengan semakin banyaknya *learning*. Penulis menggunakan 5000 kali *training* untuk model ini dan dihasilkanlah grafik seperti gambar IV.1 tersebut di atas. Besarnya nilai *loss* juga dapat dirubah dengan mengganti modul Tensorflow yang digunakan. Pada percobaan yang penulis lakukan, penulis menggunakan modul 'ssd_mobilenet_v2_fpnlite_640x640' yang memiliki kecepatan 39 ms dan mAP 29,2. Tentu model tersebut bisa diganti dengan model lain yang memiliki kecepatan dan mAP yang berbeda.

Pada awalnya, penulis menggunakan modul 'ssd_mobilenet_v2_fpnlite_320x320' yang memiliki kecepatan lebih tinggi namun dengan mAP yang lebih rendah, namun hasil yang didapat dari *training* dengan menggunakan modul tersebut kurang responsif untuk benda sejenis api.

Dari hasil pengujian yang telah dilakukan berdasarkan tabel Tabel 4-1 dan Tabel 4-2, masih ada hal yang janggal. Salah satunya adalah perubahan respon yang tidak linear pada data percobaan tersebut. Hal itu kemungkinan besar disebabkan karena

background dan pencahayaan yang berbeda dengan data *input* untuk keperluan *learning*. Dari data yang didapat juga bisa diidentifikasi jika perbedaan besar cahaya memengaruhi waktu deteksi. Cahaya yang lebih gelap akan menangkap lebih cepat api karena tak ada cahaya lain yang menjadi *background* dari api yang dinyalakan. Sementara ruangan dengan cahaya yang lebih terang membuat *background* lebih terang pula yang menjadikan api semakin tidak terlihat.

5. SIMPULAN

Kesimpulan yang didapatkan dari penelitian mengenai kamera pendeteksi api ini adalah: Dalam idealnya, pendeteksi api dengan kamera memiliki respon yang cepat dengan rata-rata dalam cahaya 130 lux sebesar 3,2 detik dan dalam cahaya 2 lux mampu mendeteksi dengan kecepatan 2,4 detik karena dengan hanya menampilkan gambar api yang bergerak sebentar saja, sensor tersebut langsung merespon. Namun perlu diperhatikan perihal *background*. *Background* dan koordinat api dalam *input* perlu diperhatikan untuk hasil yang lebih baik. Pencahayaan yang *background input* serta *background* output juga perlu diperhatikan.

Salah satu kelemahana tersebesar dari pendeteksi api menggunakan kamera adalah api yang hanya bisa dideteksi secara *visual*, dan jika api tersebut berada di balik benda lain, maka kamera tidak bisa menangkap adanya api. Oleh sebab itu, diperlukannya kamera dari beberapa sisi dengan sudut pandang yang berbeda agar bisa melihat suatu ruangan dari berbagai sisi dan bisa memperkecil adanya api yang tersembunyi dari suatu benda.

UCAPAN TERIMA KASIH

Penghargaan yang tinggi disampaikan pada Jurusan Teknik Mesin Politeknik Negeri Bandung yang telah memberikan bantuan dana penelitian Jurnal. Dan juga ucapan terima kasih kepada pihak-pihak yang terkait dalam penelitian ini.

DAFTAR PUSTAKA

[1] I. F. R. M. M. V. A. L. Yan Watuqulis Syafiudin, Dasar Pemrograman, Malang: Polinema Press, 2018.

[2] A. D. Janet Finlay, An Introduction to Artificial Intelegent, London, 1996.

[3] Wede, "Belajar Data Science : Apa yang dimaksud dengan Tensorflow dan Bagaimana Penggunaannya?," 18 November 2020. [Online]. [Accessed Mei 2022].

[4] S. M. Hanugra Aulia Sidharta, "Introduction to Open CV," 28 Oktober 2017. [Online]. [Accessed Mei 2022].

[5] N. Renotte, "Youtube," 9 April 2021. [Online]. Available: https://www.youtube.com/watch?v=yqkISICHH-U. [Accessed February 2022].