Implementasi SSD_Resnet50_v1 untuk Penghitung Kendaraan

Muhammad Nur Rizal [1], Radityo Adi Nugroho [2], Dodon T nugrahadi [3], Mohammad Reza Faisal [4], Friska Abadi [5]
[12345]Ilmu Komputer FMIPA ULM
Jl. A. Yani Km 36 Banjarbaru, Kalimantan Selatan
Email : J1f115217@mhs.ulm.ac.id

Abstract

Google has released the Tensorflow Object Detection API to facilitate deep learning application development using the Tensorflow Object Detection API. The TensorFlow Object Detection API is an open-source framework that can be used to develop, train, and deploy object detection models. In this study, the Tensorflow Object Detection API is implemented in a vehicle counter application with the SSD_Resnet50_v1 detection model. From the research that has been done, applications with the detection of the SSD_Resnet50_v1 model get an accuracy of 56.49% in calculating motor-type vehicles and 54.43% for car-type vehicles.

Kata Kunci : *SSD_Resnet50_v1, Vehicle Counting, Tensorflow Object Detection API*

Abstrak

Google telah merilis *Tensorflow Object Detection API* untuk mempermudah pengembangan aplikasi *Deep learning* dengan menggunakan *Tensorflow Object Detection API*. *TensorFlow Object Detection API* adalah *open source framework* yang dapat digunakan untuk mengembangkan, melatih, dan menggunakan model deteksi objek. Pada penelitian ini *Tensorflow Object Detection API* diimplementasikan pada aplikasi penghitung kendaraan dengan model deteksi *SSD_Resnet50_v1.* Dari penelitian yang telah dilakukan, aplikasi dengan model deteksi *SSD_Resnet50_v1* mendapatkan akurasi sebesar 56,49% dalam menghitung kendaraan berjenis motor dan 54,43% untuk kendaraan berjenis mobil.

Kata Kunci : *SSD_Resnet50_v1,* penghitung kendaraan, *Tensorflow Object Detection API*

1. PENDAHULUAN

Untuk mempermudah pengembangan aplikasi *Deep Learning*, google merilis open source framework yang bernama *Tensorflow Object Detection API*, framework *Tensorflow Object Detetion API* ini dapat dikembangkan, dilatih dan digunakan untuk model deteksi objek. Google sendiri telah menerapkan *Tensorflow Object Detection API* ke berbagai produknya seperti deteksi wajah, pencarian gambar, pengenalan plat nomor kendaraan pada *Google Streetview*,

Self Driving Car atau waymo, *Google Assistant* dan lain sebagainya. API yang tersedia telah dilatih dengan dataset Common Object In Context (COCO). API ini juga menyediakan lima model deteksi objek yang berbeda-beda dimana masing-masing mempunyai perbedaan dalam kecepatan dan akurasi dalam mendeteksi objek. "Model SSD dengan menggunakan MobileNet dapat berjalan dengan komputasi ringan, sehingga dapat dijalankan secara real time di perangkat seluler"[2]. Sedangkan "model Faster-RCNN lebih berat secara komputasi, tetapi menghasilkan pendeteksian yang jauh lebih akurat" [3].

Namun, kemampuan pendeteksiannya terbatas karena kurangnya sampel yang dianotasi dengan baik, terutama dalam pemandangan yang padat. Selain itu, karena daftar sumber data penginderaan jauh tersedia, eksploitasi informasi yang berguna secara efisien dari data multi-sumber untuk deteksi kendaraan yang lebih baik menjadi tantangan. Untuk mengatasi masalah di atas, kerangka kerja deteksi kendaraan fine-tuning aktif (Ms-AFt) multi-sumber diusulkan, yang mengintegrasikan pembelajaran transfer, segmentasi, dan klasifikasi aktif ke dalam kerangka kerja terpadu untuk pelabelan dan deteksi otomatis. MsAFt yang diusulkan menggunakan jaringan fine tuning untuk pertama-tama menghasilkan set pelatihan kendaraan dari dataset yang tidak berlabel. Untuk mengatasi keragaman kategori kendaraan, cabang segmentasi berbasis banyak sumber kemudian dirancang untuk membuat kumpulan objek kandidat tambahan. Pemisahan kendaraan berkualitas tinggi diwujudkan dengan jaringan klasifikasi yang dirancang dengan penuh perhatian. Akhirnya, ketiga cabang digabungkan untuk mencapai deteksi kendaraan. Hasil eksperimental ekstensif yang dilakukan pada dua dataset benchmark ISPRS terbuka, yaitu dataset desa Vaihingen dan kota Potsdam, menunjukkan keunggulan dan efektivitas Ms-AFt yang diusulkan untuk deteksi kendaraan. Selain itu, kemampuan generalisasi Ms-AFt dalam pemandangan penginderaan jauh yang padat diverifikasi lebih lanjut pada citra udara stereo dari lokasi perkemahan besar.

Dalam machine learning, model menangkap kecerdasan dari data menggunakan algoritme yang diimplementasikan pada framework seperti TensorFlow. Model belajar selama fase pelatihan; proses berulang di mana parameter disetel untuk meningkatkan akurasi prediksi. Repositori perangkat lunak digunakan untuk menyimpan artefak pengembangan model sehingga dapat dimodifikasi pada rilis berikutnya dan dibagikan di antara tim pengembangan. Masalah dalam menyimpan status project machine learning berbeda dengan praktik software standar. Model pembelajaran mesin setara dengan program biner yang dapat dieksekusi dan idealnya seseorang harus dapat membuat ulang model dari informasi yang disimpan dalam repositori proyek. Pembuatan ulang model menjadi perlu bila ada perubahan dalam tim pengembangan sebagai bagian dari transisi produk. Pelatihan ulang model

juga menjadi penting ketika ketidakakuratan dalam prediksi yang dibuat pada data baru meningkat melebihi batas yang dapat diterima. Saat sistem pengelolaan kode sumber tradisional digunakan untuk memelihara repositori project machine learning, kami menghadapi tantangan dalam menjaga informasi lengkap yang diperlukan untuk pengembangan model mulai dari status tersimpan. Makalah ini menyajikan hasil studi yang dilakukan untuk mengidentifikasi tantangan dalam memelihara project machine learning TensorFlow di repositori. Beberapa alat yang ada dibandingkan dan rekomendasi dibuat untuk meningkatkan kemudahan rekreasi model pembelajaran mesin dengan menyimpan informasi lengkap dalam repositori proyek yang dikelola dalam sistem kontrol kode sumber normal.

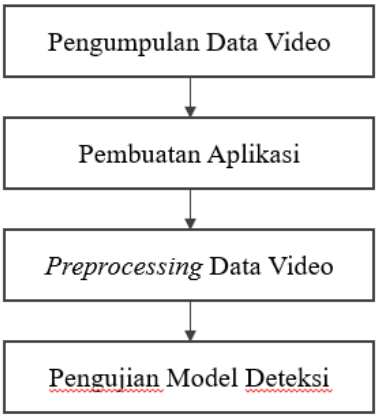
Memberi pengemudi informasi cuaca waktu nyata dan bantuan mengemudi selama cuaca buruk, termasuk kabut, sangat penting untuk mengemudi dengan aman. Fokus utama dari studi ini adalah mengembangkan metode deteksi kabut dalam kendaraan yang terjangkau, yang akan memberikan informasi cuaca tingkat lintasan yang akurat secara real-time. Studi ini menggunakan data video SHRP2 Naturalistic Driving Study (NDS) dan memanfaatkan beberapa teknik Deep Learning yang menjanjikan, termasuk Deep Neural Network (DNN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), dan Convolutional Neural Network (CNN). Pemrograman Python di library TensorFlow Machine Learning telah digunakan untuk melatih model Deep Learning. Analisis dilakukan pada dataset yang terdiri dari tiga kondisi cuaca yaitu cerah, kabut jauh dan dekat kabut. Selama proses pelatihan, dua pengoptimal, termasuk Adam dan Gradient Descent, telah digunakan. Sementara akurasi prediksi keseluruhan dari DNN, RNN, LSTM, dan CNN menggunakan pengoptimal Gradient Descent masing-masing ditemukan sekitar 85%, 77%, 84%, dan 97%; akurasi prediksi yang jauh lebih baik secara keseluruhan 88%, 91%, 93%, dan 98% untuk DNN, RNN, LSTM, dan CNN, masing-masing, diamati dengan mempertimbangkan Adam optimizer. Metode deteksi kabut yang diusulkan hanya membutuhkan satu kamera video untuk mendeteksi kondisi cuaca, dan oleh karena itu, dapat menjadi pilihan yang murah untuk dipasang di kendaraan pemeliharaan untuk mengumpulkan informasi cuaca tingkat lintasan secara real-time untuk memperluas serta memperbarui berbasis cuaca. Sistem Batas Kecepatan Variabel (VSL) dan Sistem Informasi Wisatawan Tingkat Lanjut (ATIS).

Namun data penggunaan model deteksi pada *Tensorflow Object Detection API* masih sedikit terutama pada model deteksi SSD_ResNet50_V1 karena ResNet-50 termasuk model deteksi yang beban komputasinya berat[4], akan tetapi model deteksi tersebut termasuk model detksi yang akurat, SSD (Single Shot Multibox Detector) sendiri merupakan algoritma yang memiliki kemampuan untuk mendeteksi banyak objek dalam satu gambar yang

dikembangkan oleh Liu et al. [3]. Model ini didesain untuk mendeteksi objek secara realtime, yang memiliki kemampuan untuk mendeteksi objek lebih cepat dan ResNet-50 merupakan jaringan paling umum untuk hal kinerja[4].

2. METODE PENELITIAN

Adapun prosedur penelitian ini yaitu sebagai berikut :
Dibawah ini merupakan alur dalam bentuk diagram terjadi dalam gambar 1.



Gambar 1. Alur penelitian

Adapun prosedur penelitian yang dilakukan pada penelitian ini adalah sebagai berikut :

2.1 Pengumpulan Data Video

Data video yang digunakan dalam penelitian ini berupa video dari kamera pengawas yang dipasang dekat jalan keluar Pasar Ulin Raya Banjarbaru untuk merekam kendaraan yang keluar dan masuk pada area Pasar Ulin Raya Banjarbaru sesuai dengan waktu aktifitas pasar serta dalam jangka waktu pengumpulan yang sudah ditentukan.

2.2 Pembuatan Aplikasi

Aplikasi yang dibuat untuk implementasi model deteksi SSD_Resnet50_V1 dibuat dengan bahasa pemrograman python, serta tambahan library OpenCV dan framework TensorFlow. Bahasa pemrograman python dipilih karena:

1. Bahasa pemrograman python mudah dibaca.
2. Ukuran file source code python yang cenderung kecil.
3. Program python dapat dijalankan hampir di semua sistem operasi komputer saat ini.
4. Library pada python yang berlimpah.
5. Python dapat terintegrasi dengan Bahasa pemrograman lainnya.

Library OpenCV dipilih karena open source serta merupakan library pemrosesan gambar yang paling luas penggunaannya yang dikembangkan oleh Intel. Dilengkapi oleh serangkaian fungsi yang menawarkan banyak algoritma umum untuk mewujudkan pemrosesan gambar dan komputasi visi computer, yang dapat digunakan untuk mencapai pemrosesan gambar yang kuat, dan untuk mengembangkan sistem aplikasi real-time.

Framework TensorFlow yang dipakai adalah Tensorflow Object Detection API yang dikembangkan oleh google. Framework ini selain open source juga dipakai untuk mempermudah pengembangan aplikasi Deep Learning dengan cara mengembangkan, melatih, dan menggunakan model deteksi objek. Google sendiri telah menerapkan Tensorflow Object Detection ke berbagai produknya seperti deteksi wajah, pencarian gambar, pengenalan plat nomor kendaraan pada google Streetview, Self Driving Car atau waymo, Google Assistant dan lain sebagainya.

2.3 *Preprocessing* Data Video

Preprocessing merupakan salah satu tahapan yang penting untuk data video pada proses deep learning. Data video yang digunakan dalam proses deep learning tidak selamanya dalam kondisi yang ideal untuk diproses. Terkadang pada data video tersebut terdapat berbagai permasalahan yang dapat menggangu dalam proses dan hasil dari proses deep learning itu sendiri seperti diantaranya adalah gambar video terlalu kecil atau samar, gambar video tertutupi oleh sebuah objek, ada objek yang tidak diinginkan pada data video karena kesalahan hardware, ataupun format video yang tidak sesuai dengan sistem. Oleh karenanya untuk mengatasi permasalahan tersebut dibutuhkan tahap preprocessing. Dalam penelitian ini preprocessing data video yang dilakukan adalah menghilangkan objek yang tidak diinginkan, mengurangi durasi data video yang tidak menampilkan apa-apa, melakukan perbesaran pada resolusi video, serta merubah format video agar bisa dibaca oleh sistem.

2.4 Pengujian Model Deteksi

Model deteksi SSD_Resnet50_V1 berbasis bahasa pemrograman python, serta library OpenCV dan framework TensorFlow akan diuji dengan menghitung kendaraan yang terdapat pada data video yang telah dikumpulkan dari kamera pengawas yang dipasang dekat jalan keluar Pasar Ulin Raya Banjarbaru dan sudah melewati tahap preprocessing seperti menghilangkan objek yang tidak diinginkan, mengurangi durasi data video yang tidak menampilkan apa-apa, melakukan perbesaran pada resolusi video, serta merubah format video agar bisa dibaca oleh sistem.

3. HASIL DAN PEMBAHASAN

3.1 Hasil

3.1.1 Pengumpulan Data Video

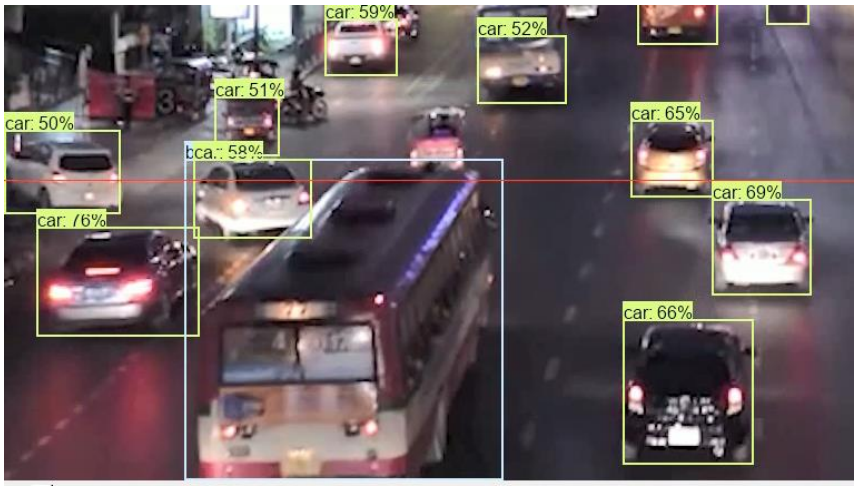
Data Video yang digunakan berupa video kendaraan keluar dari Kamera Pengawas yang dipasang dekat jalan keluar Pasar Ulin Raya Banjarbaru, letak pemasangan kamera telah disesuaikan dengan kebutuhan penelitian dan keamanan dari kamera pengawas tersebut. Data video yang digunakan pada penelitian juga disesuaikan dengan waktu aktif Pasar Ulin Raya Banjarbaru, yaitu dari pukul 06:00 pagi s/d 22:00 malam, dengan waktu pengumpulan selama 2 minggu pada bulan Oktober, dengan jumlah kendaraan seperti pada tabel 1.

Tabel 1. Jumlah Kendaraan

Tanggal	Jumlah Motor	Jumlah Mobil
05-Nov-19	3.163	411
06-Nov-19	2.758	318
07-Nov-19	2.686	265
08-Nov-19	2.955	380
09-Nov-19	4.496	526
10-Nov-19	3.161	450
11-Nov-19	2.527	308
12-Nov-19	2.636	295
13-Nov-19	2.203	284
14-Nov-19	2.665	276
22-Nov-19	2.833	278
23-Nov-19	3.561	405
24-Nov-19	2.376	292
03-Nov-19	2.792	296

3.1.2 Pembuatan Aplikasi

Aplikasi dibuat dengan bahasa pemrograman python, serta tambahan *library* OpenCV dan *Tensorflow Object Detection API, library* OpenCV digunakan untuk pemrosesan pertama aplikasi, yang bertujuan untuk mendapatkan *frame - frame* gambar dari video yang diproses pada aplikasi, *frame* gambar yang telah didapat akan diproses oleh *Tensorflow Object Detection API*, untuk mendeteksi objek yang ada pada *frame* gambar 1.



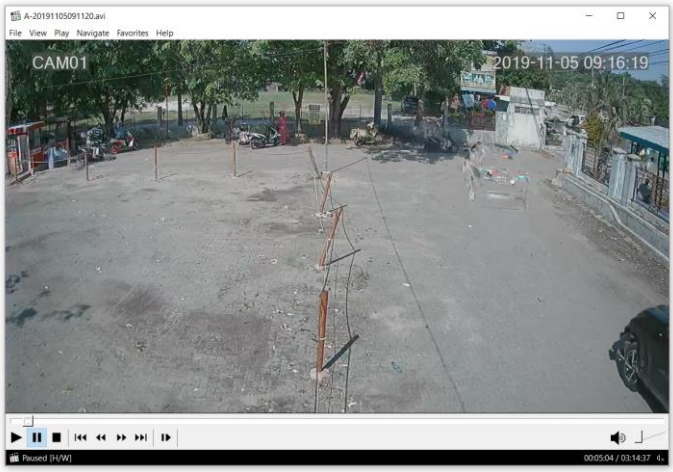
Gambar 1. Deteksi objek

Berikutnya fungsi untuk menghitung objek yang terdeteksi oleh *Tensorflow Object Detection API* yang berada pada titik hitung, karena yang perlu dihitung hanya kendaraan maka objek yang lain tidak akan dihitung walaupun berada pada titik hitung. Titik hitung divisualisasikan dengan garis, garis ini dibuat menggunakan OpenCV. Visualisasi ini bertujuan untuk memperjelas jika objek telah terhitung oleh aplikasi, garis dibuat berwana merah yang nantinya akan berubah menjadi hijau jika objek yang terdeteksi terhitung oleh aplikasi. Pada aplikasi juga akan ditampilkan jumlah objek yang telah terhitung, dapat dilihat pada gambar 2.

Gambar 2. Tampilan aplikasi

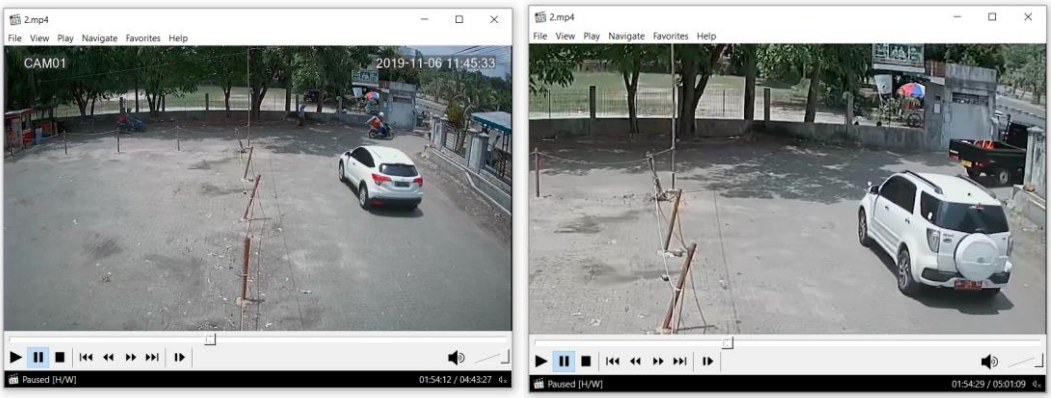
3.1.3 *Preprocessing Data video*

Data video yang didapat akan dikonversi untuk merubah format yang ada, agar dapat diedit pada aplikasi Adobe Premiere, dimana bagian-bagian video yang tidak digunakan akan dipotong seperti bagian video yang terdapat *unwanted object* pada gambar 3, juga video yang hanya menampilkan jalan keluar yang kosong akan dipotong.



Gambar 3. Unwanted object

Gambar video juga akan diperbesar agar objek pada video lebih jelas karena pemasangan kamera pengawas yang cukup jauh dari pintu keluar, data video yang telah melewati tahap *preprocessing* bisa dilihat pada gambar 4.



Gambar 4. Perbandingan data video

3.1.4 Pengujian Model Deteksi

Pengujian dilakukan dengan cara menghitung kendaraan yang keluar dari area Pasar Ulin Raya Banjarbaru dengan menggunakan aplikasi, dimana pada aplikasi telah diimplementasikan model deteksi SSD_Resnet50_v1 yaitu menghitung kendaraan yang melewati garis berwarna merah yang nantinya kan berubah menjadi hijau jika kendaraan yang terdeteksi terhitung oleh aplikasi serta menggunakan TensorFlow untuk mengenali objek tersebut berupa kendaraan atau bukan. Hasil dari pengujian terlihat seperti pada tabel 2.

Tabel 2. Hasil Pengujian

			Akurasi			
Tanggal	Jumlah Motor	Jumlah Mobil	Motor	Mobil	*Error* motor	*Error* mobil
05-Nov-19	786	439	24,94%	93,19%	-2.377	28

06-Nov-19	658	592	23,85%	53,71%	-2.100	274
07-Nov-19	926	596	34,47%	44,46%	-1.760	331
08-Nov-19	812	703	27,47%	54,05%	-2.143	323
09-Nov-19	1.920	3.427	42,70%	15,34%	-2.576	2.901
10-Nov-19	1.552	1.119	49,09%	40,21%	-1.609	669
11-Nov-19	1.366	901	54,05%	34,18%	-1.161	593
12-Nov-19	2.826	1.041	92,80%	28,33%	190	746
13-Nov-19	1.167	849	52,97%	33,45%	-1.036	565
14-Nov-19	1.756	991	65,89%	27,85%	-909	715
22-Nov-19	1.867	440	65,90%	63,18%	-966	162
23-Nov-19	3.043	374	85,45%	92,34%	-518	-31
24-Nov-19	1.991	319	83,79%	91,53%	-385	27
03-Nov-19	2.443	328	87,50%	90,24%	-349	32
	Rata-rata akurasi		56,49%	54,43%		

3.2 PEMBAHASAN

Hasil yang didapat dari pengujian adalah, aplikasi dengan model deteksi SSD_Resnet50_v1 mendapatkan akurasi sebesar 56,49% untuk kendaraan berjenis motor, dan 54,43% untuk mobil. Hasil akurasi terhitung rendah dikarenakan model deteksi kurang bisa mengenali objek kendaraan pada malam hari, sehingga banyak kendaraan yang tidak terdeteksi dan ada kendaraan berjenis motor yang didetksi sebagai mobil.

4. KESIMPULAN

Model yang dipakai dalam menghitung kendaraan pada aplikasi memiliki akurasi sebesar 56,49% untuk kendaraan berjenis motor dan 54,43% pada kendaraan berjenis mobil.

DAFTAR PUSTAKA

[1] He, K., Zhang, X., Ren, S., Sun, J. (2016). **Deep residual learning for image recognition.** CVPR 2016.

[2] Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., … Murphy, K. (2017). **Speed/accuracy trade-offs for modern convolutional object detectors**. Proceedings – 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017–Janua, 3296–3305. CVPR.2017.351

[3] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). **SSD: Single shot multibox detector. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, 9905 LNCS, 21–37. 978-3-319-46448-0_2

[4] Vijay, dkk. (2020). **MLPerf Inference Benchmark.** arXiv:1911.02549v2 [cs.LG]