Deteksi Pelanggaran Parkir Pada Bahu Jalan Tol Dengan *Intelligent Transportation System* Menggunakan Algoritma Ssd

Parking Violation Detection On The Roadside Of Toll Roads With The Intelligent Transportation System Using Ssd Algorithm

1st Sandy Eka Putra
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
sandyekaputra@student.telkomuniversity.ac.id

2nd Casi Setianingsih
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
setiacasie@telkomuniversity.ac.id

3rd Ratna Astuti Nugrahaeni
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
ratnaan@telkomuniversity.ac.id

Abstrak

Seiring berjalannya waktu, pada beberapa jalan tol yang ada di Indonesia, ada beberapa pengguna mobil yang seringkali terlihat menggunakan bahu jalan sebagai tempat parkir sementara maupun untuk mendahului yang sebenarnya tidak boleh digunakan. Dengan adanya sistem *Intelligent Transportation System* (ITS) dan menggunakan salah satu Algoritma Deep Learning yaitu *Single Shot Detector* (SSD), maka diharapkan akan terciptanya metode untuk mendeteksi kendaraan yang melanggar di bahu jalan di jalan tol. Cara kerja sistemnya adalah kamera yang telah dipasang di atas jembatan penyebarang jalan tol yang telah dipasang algoritma SSD akan mendeteksi setiap mobil yang lewat atau berhenti secara *real-time*, yang nantinya jika kamera tersebut mendeteksi mobil yang melanggar pada bahu jalan tol, maka sistem akan mengirimkan sebuah pesan notifikasi kepada petugas jalan tol melalui bot Telegram. Tujuan dari sistem ini adalah *output* dari kamera dapat mendeteksi kendaraan, lalu informasi yang didapat dari sistem tersebut dapat disampaikan kepada petugas jalan tol melalui bot Telegram. Hasil konfigurasi performansi terbaik pada penelitian ini adalah dataset dengan rasio 80%:20%, *learning rate* 0.08, *epochs* 150 dan *batch size* 6.

Kata Kunci: *Intelligent Trasnportation System*, SSD

Abstract

As time goes by, when toll roads exist in Indonesia, there are several car drivers who are often seen using the shoulder of the road as a parking space or for passing other cars which is probihited. With the Intelligent Transportation System and using one of the Deep Learning Algorithms, namely Single Shot Detector (SSD), it is hoped that a method will be created to detect vehicles that violate the road on toll roads. The way this system works is that a camera that is installed on a toll road crossing bridge that has been properly installed SSD will detect every car that passes or stops in real time, which later if the camera detects a car that violates the shoulder of the toll road, the system sends a notification message to toll road officials via Telegram bot. The purpose of this system is that the output from the camera can detect cars accurately, then the information obtained from the system can be delivered to toll road officials via the Telegram bot. The results of the best performance configuration in this research are datasets with a ratio of 80%:20%, 0.08 learning rate, 150 epochs and 6 batch size.

Keywords: Intelligent Trasnportation System, SSD

I. PENDAHULUAN

Pada zaman sekarang, di Indonesia telah banyak jalan tol yang sudah dibuat, akan tetapi walaupun sudah banyak jalan tol tetap saja kemacetan selalu terjadi. Salah satu penyebab kemacetan di jalan tol yaitu banyaknya oknum-oknum yang masih menyalahgunakan bahu jalan sebagai tempat parkir ataupun untuk mendahului. Padahal, tujuan dibuatnya bahu jalan yaitu untuk *emergency stop* dimana orang-orang dapat berhenti menggunakan bahu jalan ketika keadaaan darurat saja, bukan untuk tempat parkir biasa atau untuk mendahului kendaraan lain. Jika satu mobil saja yang parkir mungkin tidak terlalu berpengaruh, tetapi jika ada beberapa kendaraan yang parkir di tempat yang bukan seharusnya, sudah pasti jalan tol pun akan semakin padat dan dapat berakhir dengan kemacetan total dan mungkin dapat menimbulkan kecelakaan.

Menurut PP no 15 tahun 2005 Pasal 41, bahu jalan tol tidak boleh digunakan untuk mendahului kendaraan lain [1]. Pada kenyataannya, masih banyak pengemudi yang melanggar aturan jalan tol seperti mendahului kendaraan lain menggunakan bahu jalan. Oleh karena itu diperlukan sebuah solusi untuk mengurangi pelanggaran lalu lintas pada jalan tol. *Intelligent Transportation System* adalah sistem yang menerapkan berbagai teknologi pada transportasi untuk berbagai macam tujuan, salah satunya untuk memonitoring kondisi jalan tol.

Oleh karena itu, penelitian ini akan difokuskan pada perancangan sistem *Intelligent Transportation System* yang dapat digunakan untuk mendeteksi kendaraan yang berada pada bahu jalan. Sistem ini direncanakan akan menggunakan algoritma SSD sebagai solusi pelanggaran lalu lintas pada jalan tol.

II. KAJIAN TEORI

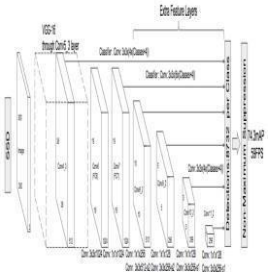
A. Intelligent Transportation System

Intelligent Transportation System (ITS) adalah sistem yang mengimplementasikan berbagai teknologi pada transportasi untuk menjadikan sistem tersebut lebih aman, lebih efisien, dan lebih ramah lingkungan tanpa mengubah infrastruktur yang ada secara fisik [2].

B. Single Shot Detection

Single Shot Detector (SSD) adalah salah satu metode yang digunakan untuk mengidentifikasi atau mendeteksi objek dalam suatu gambar, dan merupakan salah satu algoritma pendeteksi objek yang paling populer karena mudah diterapkan dan memiliki akurasi yang baik relatif terhadap perhitungan yang diperlukan. Metode *Single Shot Detector* (SSD) termasuk dalam *object detection* yang *real time* [3]. Cara kerja pada metode SSD yaitu dengan menentukan *bounding boxes* suatu objek yang telah ditentukan untuk dibandingkan dengan *bounding boxes* pada SSD, lalu hasilnya akan diklasifikasikan dalam bentuk *class*.

Berikut ini adalah gambaran dari arsitektur SSD:



Gambar 1 SSD *Architecture* [4]

C. Tensorflow

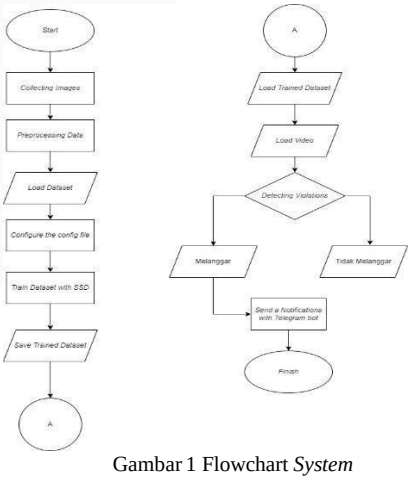
Tensorflow adalah *platform opensource machine learning* untuk mengembangkan model dari *training* data dalam waktu yang singkat dengan memanfaatkan sumber daya komputasi seperti CPU dan GPU [5]. Tensorflow telah diterima secara luas oleh komunitas *machine learning* dalam waktu yang singkat karena mudah digunakan.

D. Bot Telegram

Bot Telegram adalah *machine user*, dibuat untuk memiliki fungsi yang sama dengan *person user*, tetapi dikendalikan oleh mesin dan dapat memiliki kecerdasan buatan. Bot Telegram dapat diprogram untuk tujuan tertentu, contohnya mencari video, gambar, dan bahkan berinteraksi dengan Bot Telegram lainnya [6].

III. METODE

A. Perancangan Sistem



Gambar 1 Flowchart *System*

Pada *flowchart* diatas menunjukan langkah-langkah dari mulai *collecting dataset*, *labeling images*, *load dataset*, konfigurasi file *config*, *training dataset* dengan SSD, dan meyimpan hasil *training*.

Selanjutnya *load trained dataset* dan *load* video yang akan di uji. Setelah itu sistem akan mendeteksi

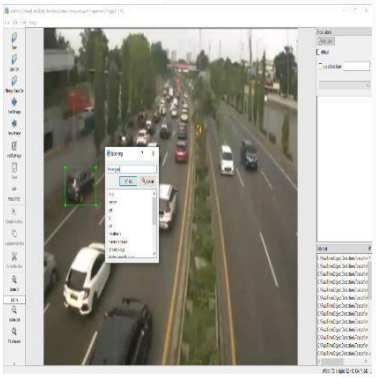
2 *class*, yaitu "Melanggar" dan "tidak melanggar". Jika sistem mendeteksi pelanggaran, maka sistem akan mengirim notifikasi ke bot Telegram, tetapi jika tidak sistem akan terus mendeteksi hingga durasi video selesai.

B. *Collecting Images*

Gambar 2 HK TOLL *Apps*

Pada tahap ini penulis mengambil 100 dataset yang terdiri dari 50 dataset pelanggaran dan 50 dataset yang tidak melanggar. Pengambilan dataset diambil melalui aplikasi "HK TOLL Apps" dengan cara mengambil screenshot satu per-satu pada CCTV ruas tol JORR di KM 30. Dataset yang diambil berbentuk JPG

C. *Preprocessing Dataset*



Gambar 3 *Label Image*

Pada tahap ini dataset yang telah didapat harus dilabelkan terlebih dahulu menggunakan *LabelImage*. Cara memberi labelnya yaitu dengan memberi *reactbox* pada mobil yang melanggar dan yang tidak melanggar. Jika melanggar maka *class*-nya akan diberi nama "Melanggar" dan jika tidak melanggar *class*-nya akan diberi nama "tidak melanggar". Hasil dari *labeling* adalah sebuah file bertentuk XML. Selanjutnya file XML dari dataset tersebut di *load* ke sistem dengan cara mengatur *path* pada sistem.

D. Konfigurasi Sistem

Setelah mengatur *path*, "label_map.txt" harus dibuat terlebih dahulu untuk meng-anotasikan *class* pada dataset yang sudah dilabelkan. Selanjutnya "tfrecord" akan dibuat dari folder *train* dan *test* menggunakan *script* "generate_tfrecord.py" dan anotasi yang sudah ada. Fungsi dari tfrecord ini adalah untuk menyimpan dataset yang sudah berbentuk XML.

Sebelum *training* dilakukan, diperlukan penyesuaian *hyperparameter* yang terdapat pada *configuration file* yang bernama "pipeline.config". Di antaranya adalah *steps*, *learning rate*, *epochs*, dan *batch size*.

Nilai *steps* yang digunakan adalah 40000 agar mendapatkan hasil yang baik. Untuk *learning rate* yang digunakan adalah 0.08 agar proses *training* berjalan optimal. Nilai *epochs* yang digunakan adalah 1, dan *batch size* yang digunakan adalah 4.

E. *Training*

Proses satu kali *training* dapat memakan waktu sebanyak ±4 jam. Hasil dari *training* data tersebut berbentuk *checkpoint* yang berfungsi menyimpan hasil *training*. Lalu *checkpoint* tersebut di *load* ke sistem untuk deteksi pelanggaran.

Gambar 4 *Checkpoint*

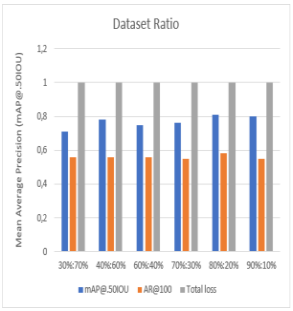
IV. HASIL DAN PEMBAHASAN

A. Pengujian rasio dataset

Dalam pengujian ini, 6 sampel telah dicoba untuk mendapatkan rasio terbaik dalam penelitian ini.

Dataset Ratio	mAP@.50IOU	AR@100	Total loss
30%:70%	71%	56%	±1
40%:60%	78%	56%	±1
60%:40%	75%	56%	±1
70%:30%	76%	55%	±1
80%:20%	81%	58%	±1
90%:10%	80%	55%	±1

Tabel 1 Hasil *training* berbeda rasio dataset



Gambar 5 Grafik performansi rasio dataset

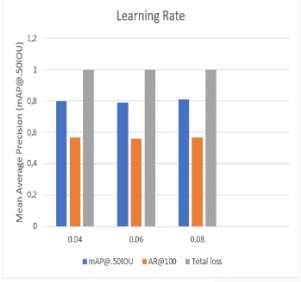
Dari keenam rasio tersebut, nilai tertinggi didapat dengan menggunakan rasio *train* 80% dan *test* 20% dengan nilai mAP sebesar 81%. Dapat diambil kesimpulan bahwa dengan dataset dengan jumlah yang cukup banyak pada folder *train* dan dataset dengan jumlah yang tidak terlalu sedikit pada folder *test* dapat menghasilkan nilai mAP yang tertinggi. Maka dari itu, rasio tersebut akan digunakan untuk menentukan *learning rate* terbaik.

B. Pengujian *learning rate*

Pada pengujian selanjutnya, 3 sampel telah dicoba untuk mencari *learning rate* terbaik dalam penelitian ini.

Tabel 2 Hasil *training* berbeda *learning rate*

Learning Rate	mAP@.50IOU	AR@100	Total loss
0.04	80%	57%	±1
0.06	79%	56%	±1
0.08	81%	57%	±1



Gambar 6 Grafik performansi learning rate

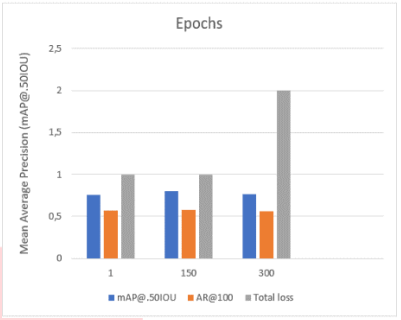
Dari ketiga learning rate tersebut, nilai tertinggi didapat dengan menggunakan learning rate 0.08 dengan nilai mAP sebesar 81%. Dapat diambil kesimpulan bahwa dengan learning rate dengan nilai terbesar pada model SSD ini dapat menghasilkan nilai mAP yang tertinggi. Maka dari itu, learning rate tersebut akan digunakan untuk menentukan epochs terbaik.

C. Pengujian *epochs*

Pada pengujian selanjutnya, 3 sampel telah dicoba untuk mencari *epochs* terbaik dalam penelitian ini.

Tabel 3 Hasil *training* berbeda *epochs*

Epochs	mAP@.50IOU	AR@100	Total loss
1	76%	57%	±1
150	80%	58%	±1
300	77%	56%	±2



Gambar 7 Grafik performansi *epochs*

Dari ketiga *epochs* tersebut, nilai tertinggi didapat dengan menggunakan *epochs* 150 dengan nilai mAP sebesar 80%. Dapat diambil kesimpulan bahwa dengan jumlah *epochs* 150 dapat menghasilkan nilai mAP yang optimum. Maka dari itu, *epochs* tersebut akan digunakan untuk menentukan *batch size* terbaik.

D. Pengujian *batch size*

Pada pengujian selanjutnya, 3 sampel telah dicoba untuk mencari *batch size* terbaik dalam penelitian ini.

Tabel 4 Hasil *training* berbeda *batch size*

Batch Size	mAP@.50IOU	AR@100	Total loss
4	76%	59%	±2
6	81%	60%	±1
8	80%	59%	±2

Gambar 8 Grafik performansi *batch size*

Dari ketiga batch size tersebut, dapat disimpulkan bahwa nilai tertinggi didapat dengan menggunakan batch size 6 dengan mAP sebesar 81%

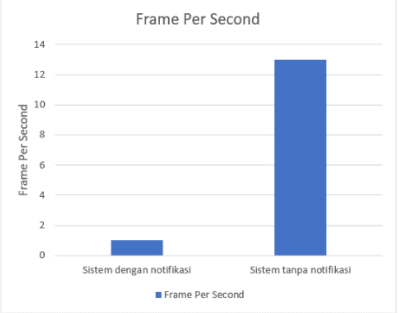
E. Pengujian *Frame Per Second*

Selanjutnya adalah pengujian *frame per second*. *Frame per second* adalah satuan untuk mengukur *frame* atau adegan gambar dalam 1 detik. Hasil

frame per second yang didapat pada sistem yang terhubung dengan bot Telegram dan yang tidak dengan bot Telegram sangatlah berbeda. Berikut adalah hasil pengujian FPS:

Tabel 5 Hasil pengujian *frame per second*

	Frame Per Second
Sistem dengan notifikasi	1
Sistem tanpa notifikasi	13



Gambar 9 Grafik pengujian *frame per second*

Dari grafik 9 dapat disimpulkan bahwa sistem dengan notifikasi bot Telegram hanya mendapat 1 FPS. Hal ini dikarenakan fitur notifikasi bot Telegram dapat memperlambat kinerja sistem. Sedangkan sistem yang tanpa menggunakan notifikasi bot Telegam mendapatkan lebih dari 10 FPS dikarenakan tidak adanya fitur tambahan yang dapat memperlambat kinerja sistem tersebut.

V. KESIMPULAN

Berdasarkan hasil pengujian, berikut merupakan kesimpulan yang dapat diambil Sistem dapat mendeteksi adanya pelanggaran pada bahu jalan tol menggunakan algoritma SSD dengan hasil konfigurasi *training* terbaik yang didapatkan pada penelitian ini adalah dengan rasio dataset 80%:20%, *learning rate* 0.08, *epochs* 150, dan *batch size* 6. Sistem dapat mendeteksi dan memberikan notifikasi ke petugas jalan tol melalui bot Telegram dengan FPS yang didapat sebesar 1. Beberapa saran yang dapat digunakan untuk mengembangkan sistem pada Tugas Akhir ini adalah sebagai berikut: Menggunakan variasi *hyperparameter* lainnya agar dapat meningkatkan performansinya. Membutuhkan perangkat keras yang lebih baik agar proses *training* dapat lebih cepat.

REFERENSI

[1] Peraturan Pemerintah (PP) tentang Jalan Tol, Pemerintah Pusat, March, 2005. Accessed on: January. 21, 2022. [Online]. Available: https://peraturan.bpk.go.id/Home/Details/49351/pp-no-15-tahun-2005.

[2] L. Qi, “Research on Intelligent Transportation System Technologies And Applications”, presented at Workshop on Power Electronics and Intelligent Transportation System, pp. 529-531, 2008.

[3] S. Fuady, Nehru, and G. Anggraeni, Deteksi Objek Menggunakan Metode Single Shot Multibox Detector Pada Alat Bantu Tongkat Tunanetra Berbasis Kamera, Journal of Electrical Power Control An®®d Automation, Vol. 3, No. 2, pp. 39-43, 2020

[4] M. Amarnah, “Vehicle Detection”, Available: https://github.com/mohammedamarnah/vehicle-detection. [Diakses 11 Juni 2021, 20:57 WIB]

[5] SunTec Business Solutions Pvt Ltd, "Project repositories for machine learning with TensorFlow," Procedia Computer Science , pp. 188-196, 2020.

[6] J. O. Carlos de and D. H. Santos, "Chatting with Arduino Platform through Telegram Bot," IEEE International Symposium on Consumer Electronics, 2016.