SISTEM OTOMATISASI PENGHITUNGAN KENDARAAN (*VEHICLE COUNTING*) BERBASIS *CONVOLUTIONAL NEURAL NETWORK*

Skripsi

Untuk memenuhi sebagai persyaratan mencapai derajat Sarjana S-1

Program Studi Teknik Informatika

Disusun oleh:

Mochamad Ghifari 'Azmi

16650051

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI SUNAN KALIJAGA

YOGYAKARTA

2020

HALAMAN PENGESAHAN

KEMENTERIAN AGAMA
UNIVERSITAS ISLAM NEGERI SUNAN KALIJAGA
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Marsda Adisucipto Telp. (0274) 540971 Fax. (0274) 519739 Yogyakarta 55281

PENGESAHAN TUGAS AKHIR

Nomor : B-1734/Un.02/DST/PP.00.9/07/2020

Tugas Akhir dengan judul : SISTEM OTOMATISASI PENGHITUNGAN KENDARAAN (VEHICLE COUNTING) BERBASIS CONVOLUTIONAL NEURAL NETWORK

yang dipersiapkan dan disusun oleh:

Nama : MOCHAMAD GHIFARI 'AZMI
Nomor Induk Mahasiswa : 16650051
Telah diujikan pada : Senin, 20 Juli 2020
Nilai ujian Tugas Akhir : A

dinyatakan telah diterima oleh Fakultas Sains dan Teknologi UIN Sunan Kalijaga Yogyakarta

TIM UJIAN TUGAS AKHIR

Ketua Sidang/Penguji I

Nurochman, S.Kom., M.Kom
SIGNED

Valid ID: 5f3505325110

Penguji II

Muhammad Taufiq Nuruzzaman, S.T., M.Eng., Ph.D.
SIGNED

Valid ID: 5f33dcea6d710

Penguji III

Muhammad Didik Rohmad Wahyudi, S.T., M.T.
SIGNED

Valid ID: 5f34d7f4a8410

Yogyakarta, 20 Juli 2020
UIN Sunan Kalijaga
Plt. Dekan Fakultas Sains dan Teknologi

Dr. Murtono, M.Si.
SIGNED

Valid ID: 5f36098a47015

SURAT PERSETUJUAN SKRIPSI

SURAT PERSETUJUAN SKRIPSI/TUGAS AKHIR

Hal : Persetujuan Skripsi
Lamp :

Kepada
Yth. Dekan Fakultas Sains dan Teknologi
UIN Sunan Kalijaga Yogyakarta
di Yogyakarta

Assalamu'alaikum wr. wb.

Setelah membaca, meneliti, memberikan petunjuk dan mengoreksi serta mengadakan perbaikan seperlunya, maka kami selaku pembimbing berpendapat bahwa skripsi Saudara:

Nama : Mochamad Ghifari 'Azmi
NIM : 16650051
Judul Skripsi : "Sistem Otomatisasi Penghitungan Kendaraan (*Vehicle Counting*) Berbasis *Convolutional Neural Network*"

sudah dapat diajukan kembali kepada Program Studi Teknik Informatika Fakultas Sains dan Teknologi UIN Sunan Kalijaga Yogyakarta sebagai salah satu syarat untuk memperoleh gelar Sarjana Strata Satu dalam Program Studi Teknik Informatika

Dengan ini kami mengharap agar skripsi/tugas akhir Saudara tersebut di atas dapat segera dimunaqsyahkan. Atas perhatiannya kami ucapkan terima kasih.

Wassalamu'alaikum wr. wb.

Yogyakarta, 10 Juli 2020
Pembimbing

Nurochman, S.Kom., M.Kom
NIP. 19801223 200901 1 007

PERNYATAAN KEASLIAN SKRIPSI

PERNYATAAN KEASLIAN SKRIPSI

Saya yang bertanda tangan dibawah ini :

Nama : Mochamad Ghifari 'Azmi

NIM : 16650051

Jurusan : Teknik Informatika

Fakultas : Sains dan Teknologi

Menyatakan bahwa skripsi saya yang berjudul "**Sistem Otomatisasi Penghitungan Kendaraan (*Vehicle Counting*) Berbasis *Convolutional Neural Network***" merupakan hasil penelitian saya sendiri, tidak terdapat pada karya yang pernah di ajukan untuk memperoleh gelar sarjana di suatu perguruan tinggi, dan bukan plagiasi karya orang lain kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Kebumen, 13 Juli 2020
Yang Menyatakan

Mochamad Ghifari 'Azmi
NIM. 16650051

KATA PENGANTAR

Alhamdulillahi Robbil'alamin, segala puji hanya milik Allah SWT. Atas limpahan rahmat dan hidayah-Nya, penulis dapat menyelesaikan skripsi yang berjudul **Sistem Otomatisasi Penghitungan Kendaraan (*Vehicle Counting*) Berbasis *Convolutional Neural Network*** sebagai salah satu syarat untuk mendapatkan gelar sarjana program studi Teknik Informatika Universitas Islam Negeri Sunan Kalijaga Yogyakarta. Sholawat serta salam selalu tercurahkan kepada Rasulullah SAW, keluarga, sahabat, dan para pengikutnya.

Penulis menyadari bahwa penyusunan skripsi ini masih jauh dari kata sempurna. Oleh karena itu, penulis membuka diri untuk segala kritik dan saran yang membangun. Semoga penelitian ini kedepannya dapat memberikan manfaat dan sumbangan pemikiran bagi pembaca.

Pada penyusunan skripsi ini, penulis mengucapkan terimakasih yang tak terhingga kepada:

1. Bapak Prof. Dr. Phil. Al Makin, S.Ag., M.A. selaku Rektor UIN Sunan Kalijaga Yogyakarta.
2. Bapak Dr. H. Waryono, M.Ag., selaku Wakil Rektor Bidang Kemahasiswaan dan Kerjasama UIN Sunan Kalijaga Yogyakarta.
3. Bapak Dr. Murtono, M.Si., selaku Dekan Fakultas Sains dan Teknologi UIN Sunan Kalijaga Yogyakarta.
4. Bapak Sumarsono, S.T., M.Kom., selaku Ketua Program Studi S1 Teknik Informatika UIN Sunan Kalijaga Yogyakarta.

5. Bapak Nurochman, S.Kom., M.Kom., selaku dosen pembimbing skripsi yang telah dan selalu sabar membimbing, mengarahkan, dan memotivasi dalam penyusunan skripsi ini.
6. Bapak Muhammad Didik Rohmad Wahyudi, S.T., M.T.,selaku Dosen Pembimbing Akademik.
7. Dr. Agung Fatwanto, S.Si., M.Kom., Agus Mulyanto, S.Si., M.Kom., Aulia Faqih Rifa'i, M.Kom., M. Taufiq Nuruzzaman, S.T. M.Eng., Maria Ulfah Siregar, S.Kom. MIT., Ph.D., Nurochman, S.Kom., M.Kom., Rahmat Hidayat, S.Kom., M.Cs.,Usfita Kiftiyani, M.Sc.,selaku dosen pengampu mata kuliah program studi Teknik Informatika UIN Sunan Kalijaga Yogyakarta.
8. Seluruh staf dan karyawan Fakultas Sains dan Teknologi UIN Sunan Kalijaga Yogyakarta.
9. Kedua orang tua, Bapak Khumroni dan Ibu Umi Kulsum yang telah memberikan segalanya yang terbaik.
10. Seseorang yang selalu menemani dalam menghadapi lika-liku perjalanan kuliah, memberikan do'a, dukungan, motivasi serta semangat dari awal kuliah hingga akhir penyusunan skripsi.
11. Adik-adik saya Adib Al-Fikri, Alya Azzahra dan Aufa Ainun Yafi yang senantiasa memberikan dukungan, dan motivasi kepada saya.
12. Rekan skripsi seperjuangan saya Gilang Ari Saputra yang telah banyak memberikan informasi dan diskusi tentang skripsi ini.

13. Muhammad Dzulfikar Fauzi S.Kom., M.Cs selaku mentor terbaik dalam pengerjaan skripsi.
14. Teman-teman bertukar pikiran dalam segala hal, Tim ITTC, Lina, Nelly, Ardi, Ika, yang telah memberikan dukungan serta inspirasi dalam proses pengerjaan skripsi ini.
15. Teman-teman Teknik Informatika 2016 dan seluruh Keluarga Besar Teknik Informatika UIN Sunan Kalijaga yang tidak dapat penulis tuliskan satu per satu.
16. Teman-teman KKN 99 Tematik Pulau Raas, Khususnya Kelompok tiga Dusun Kranji Enji, Dana, Gilang, Hilman, Wahyu, Sintia, Isti, Ipeh dan Tiqoh, terimakasih telah berbagi keluh kesah, *sharing*, dan pengalaman saat serumah 2 bulan di Pulau Raas.
17. Semua pihak yang telah memberikan dorongan dan bantuan dalam penyusunan skripsi ini hingga selesai serta dalam menempuh studi yang tidak dapat disebutkan satu persatu.

Kebumen, 07 Juli 2020

Penulis

HALAMAN PERSEMBAHAN

Puji syukur kehadirat Allah SWT dan shalawat serta salam tercurahkan kepada Nabi Muhammad SAW

Skripsi ini saya persembahkan kepada:

Kedua orang tua saya Bapak Khumroni dan Ibu Umi Kulsum

serta ketiga adik-adik saya

Serta keluarga besar yang selalu memberi dukungan

baik dari segi moral ataupun moril dan do'a kepada saya

dan tak lupa almamaterku UIN Sunan Kalijaga Yogyakarta

Terimakasih untuk semuanya.

HALAMAN MOTTO

“

FOLLOW YOUR CURIOUSITY…

”

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR KODE SUMBER

Sistem Otomatisasi Penghitungan Kendaraan (*Vehicle Counting*) Berbasis *Convolutional Neural Network*

Mochamad Ghifari 'Azmi

16650051

INTISARI

Pertumbuhan pengguna kendaraan bermotor setiap tahun semakin meningkat. Seiring pertumbuhan pengguna kendaraan tersebut volume lalu lintas jalan juga ikut meningkat. Untuk mengatur lalu lintas yang baik tentunya dibutuhkan pemantauan arus lalu lintas. Pemantauan arus lalu lintas dapat dilihat dari jumlah kendaraan bermotor yang melewati suatu jalan. Namun proses penghitungan jumlah kendaraan bermotor dengan manual terbatas pada keakuratan penglihatan manusia. Saat ini hampir setiap persimpangan di Indonesia khususnya sudah terpasang kamera CCTV/ATCS, oleh karena itu penulis melihat kemungkinan pendayagunaan video dari kamera tersebut untuk diproses agar menghasilkan data pemantauan arus lalu lintas secara otomatis.

Penelitian ini diawali dengan pengumpulan data dari CCTV/ATCS Dinas Perhubungan Kabupaten Sukoharjo lalu setelah itu dilakukan *pre-processing* citra dengan *cropping*. Dilanjutkan dengan deteksi objek, *tracking*, dan *counting* atau penghitungan kendaraan. Penghitungan kendaraan pada video ini dilakukan menggunakan arsitektur *convolutional neural network*.

Hasil penelitian ini menyimpulkan bahwa jika ingin menghitung mobil baik itu truk, bus, minibus, sedan maupun *pickup* maka dengan COCO dataset cocok menggunakan kelas *truck, car*, dan *bus*. Sedangkan untuk menghitung motor kurang cocok, dan yang terakhir jika ingin menghitung semua kendaraan yang melintas dijalan baik itu sepeda, motor, truk, bus, *pickup* dan sedan COCO dataset juga masih kurang cocok diterapkan di wilayah Sukoharjo dan umumnya Indonesia.

Kata kunci: penghitungan kendaraan, *vehicle counting*, *convolutional neural network*, deteksi, *tracking*.

Convolutional Neural Network Based Vehicle Counting Automation System

Mochamad Ghifari 'Azmi

16650051

ABSTRACT

The growth of motor vehicle users is increasing every year. As vehicle users grow, road traffic volume also increases. To manage traffic properly, monitoring of traffic flow is certainly needed. Traffic flow monitoring can be seen from the number of motorized vehicles passing a road. But the process of counting the number of motorized vehicles manually is limited to the accuracy of human vision. Currently, almost every intersection in Indonesia, especially CCTV / ATCS cameras have been installed, therefore the author sees the possibility of utilizing the video from the camera to be processed in order to generate traffic flow monitoring data automatically.

This research was started by collecting data from CCTV / ATCS of the Department of Transportation of Sukoharjo Regency and after that the image pre-processing was done with cropping. Followed by object detection, tracking and counting or vehicle counting. The vehicle calculation in this video is done using the convolutional neural network architecture.

The results of this study conclude that if you want to count cars, be it trucks, buses, minibuses, sedans or pickups, the COCO dataset is suitable to use the truck, car and bus classes. Meanwhile, to calculate the motorbike is not suitable, and finally, if you want to count all the vehicles passing on the road, be it bicycles, motorbikes, trucks, buses, pickups and sedans, the COCO dataset is also not suitable for application in the Sukoharjo area and in general Indonesia.

Keywords: vehicle counting, convolutional neural network, detection, tracking.

BAB I

PENDAHULUAN

1.1. Latar Belakang

Dewasa ini pengguna kendaraan bermotor khususnya di Indonesia semakin meningkat, tentunya ini menjadikan jumlah kendaraan bermotor baik roda dua maupun roda empat di jalan raya juga semakin meningkat. Menurut Badan Pusat Statistik pada tahun 2018 jumlah kendaraan bermotor di Indonesia mencapai 146,86 juta, angka ini membuktikan bahwa jumlah kendaraan bermotor mengalami peningkatan 7% dari tahun sebelumnya. Kita ketahui bersama bahwasanya sekarang ini sudah banyak terpasang kamera CCTV (*Closed Circuit Television*) atau ATCS (*Area Traffic Control System*) pada jalan raya, baik itu di lampu lalu lintas dan di setiap persimpangan jalan. Ini dimaksudkan agar arus lalu lintas dapat terpantau oleh pihak Dinas Perhubungan. Selain itu kita sendiri juga dapat memantau arus lalu lintas melalui situs yang disediakan oleh pihak Dinas Perhubungan. Dari sana kita dapat mengetahui seberapa padat volume kendaraan yang melintas di suatu jalan. Salah satu daerah yang memiliki fasilitas tersebut adalah Kabupaten Sukoharjo dimana CCTV di daerah tersebut memiliki hasil video dengan kualitas yang cukup bagus dan terdapat berbagai macam kendaraan khas Indonesia yang terpantau dalam CCTV tersebut.

Umumnya untuk memperoleh data jumlah kendaraan yang melintasi suatu jalan raya dilakukan secara manual dengan menugaskan petugas untuk menghitung setiap kendaraan yang lewat pada suatu jalan dan dibagi pada

rentang waktu tertentu, namun metode tersebut masih memiliki beberapa kekurangan seperti tingkat akurasi data yang kurang maksimal karena keterbatasan manusia, membutuhkan waktu yang lama, dan membutuhkan sumber daya manusia yang banyak.

Adanya perkembangan ilmu pengetahuan dan teknologi, kini penghitungan jumlah kendaraan di jalan raya dapat dilakukan secara otomatis. Memanfaatkan video CCTV yang disediakan oleh pihak pemerintah (Dinas Perhubungan) penghitungan otomatis kendaraan dapat dilakukan dengan video *processing*, Kecerdasan buatan atau biasa kita sebut *Artificial Intelligence (AI)* dapat mengolah video menjadi suatu informasi yang bermanfaat. Kecerdasan buatan memiliki beberapa cabang ilmu, salah satunya yang saat ini sedang banyak diperbincangkan yaitu *Deep Learning*. *Deep Learning* sendiri mempunyai *neural network* yang terinspirasi dari otak manusia. Penghitungan jumlah kendaraan otomatis harus melewati beberapa tahapan yakni seperti deteksi dan *tracking*. Proses pendeteksian ini berfungsi membedakan kendaraan satu dengan yang lain ataupun dengan non kendaraan pada suatu citra dalam *frame* video, dan hasil dari pendeteksian tersebut akan digunakan untuk *tracking* dan selanjutnya dihitung (*counting*). Salah satu metode deteksi objek pada citra yaitu metode *YOLO* (*You Only Look Once*) yang mempunyai arsitektur *Convolutional Neural Network (CNN). Convolutional Neural Network* (*CNN*) menjadi salah satu arsitektur *Deep Learning* yang dapat digunakan untuk pendeteksian objek dalam kasus ini bisa digunakan untuk mendeteksi kendaraan. Berdasarkan uraian di atas, maka penulis menyusun

penelitian skripsi dengan judul Sistem Otomatis Untuk Penghitungan Kendaraan (*Vehicle Counting*) Berbasis *Convolutional Neural Network*.

1.2. Rumusan Masalah

Berdasarkan latar belakang di atas, maka dapat diperoleh rumusan masalah sebagai berikut:

1. Bagaimana merancang sebuah sistem yang akurat untuk penghitungan jumlah kendaraan?
2. Bagaimana menerapkan *convolutional neural network* untuk penghitungan kendaraan?

1.3. Batasan Masalah

Berdasarkan uraian permasalahan di atas, maka batasan masalah pada skripsi ini adalah:

1. Dalam penelitian ini data yang digunakan berasal dari CCTV Dinas Perhubungan Kabupaten Sukoharjo.
2. Pengambilan *sample* data dilakukan satu hari pada waktu tertentu masing-masing berdurasi 1 menit.
3. Dalam penelitian ini objek yang dihitung adalah kendaraan yang melintas pada jalan raya Kabupaten Sukoharjo.
4. Metode yang digunakan dalam penelitian ini adalah YOLO versi 3.
5. Program dibuat menggunakan bahasa pemrograman Python.

1.4. Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah:

1. Merancang sistem otomatis untuk penghitungan kendaraan (v*ehicle counting*) berbasis c*onvolutional neural network* agar penghitungan kendaraan menjadi efisien, mudah dan akurat.
2. Menerapkan *convolutional neural network* untuk penghitungan kendaraan.

1.5. Manfaat Penelitian

Manfaat dari penelitian ini adalah:

1. Memberikan kemudahan pada suatu pihak (dalam penelitian ini Dinas Perhubungan Kabupaten Sukoharjo) dalam mendeteksi dan menghitung jumlah kendaraan yang keluar masuk wilayah Sukoharjo secara otomatis.
2. Memberikan gambaran penerapan penghitungan kendaraan secara otomatis berbasis *convolutional neural network.*
3. Meninjau kecocokan YOLOv3 yang sudah di*training* dengan COCO dataset untuk kendaraan di Indonesia khususnya Kabupaten Sukoharjo.
4. Hasil penelitian ini dapat dijadikan data yang dapat diolah untuk berbagai kepentingan lalu lintas jalan Kabupaten Sukoharjo.

1.6. Keaslian Skripsi

Penelitian serupa yang berhubungan dengan penghitungan kendaraan sudah pernah dilakukan sebelumnya, namun terdapat perbedaan antara penelitian ini dengan penelitian sebelumnya yaitu pada data dan objek yang digunakan.

BAB V

PENUTUP

5.1. Kesimpulan

Berdasarkan penelitian yang telah dilakukan, maka dapat kita ambil kesimpulan bahwa:

1) Penulis berhasil membangun sistem otomatisasi kendaraan (*vehicle counting*) berbasis *Convolutional Neural Network* menggunakan bahasa pemrograman Python. Hal ini dapat ditunjukkan pada hasil pengujian akurasi pada BAB IV.
2) *Convolutional Neural Network* (*CNN*) sebagai arsitektur jaringan syaraf tiruan baik digunakan untuk penghitungan kendaraan pada jalan raya. Hal ini dibuktikan dengan hasil pengujian akurasi pada BAB IV.
3) Dalam pendeteksian sistem dengan kelas *car*, *bus* dan *truck* mampu mendapatkan hasil akurasi deteksi pada Video1 99,2%, Video2 100% dan pada Video3 100%. Dan juga menghasilkan produk *counting* masing-masing sebesar 96,5%, 90% dan 91,6%.
4) Kemudian dalam pendeteksian sistem dengan hanya kelas *motorbike* mampu mendapatkan hasil akurasi deteksi pada Video1 66,4%, Video2 71,4% dan pada Video3 70,8%. Dan juga menghasilkan produk *counting* masing-masing sebesar 43,1%, 48,2% dan 27,2%.
5) Kemudian dalam pendeteksian sistem dengan hanya kelas *motorbike* mampu mendapatkan hasil akurasi deteksi pada Video1 62,6%, Video2

73,2% dan pada Video3 73,7%. Dan juga menghasilkan produk *counting* masing-masing sebesar 64,9%, 58,5% dan 52,5%.

6) Berdasarkan hasil pengujian deteksi dan *counting* dapat disimpulkan bahwa jika kita ingin menghitung mobil baik itu truk, bus, minibus, sedan maupun *pickup* maka dengan COCO dataset cocok menggunakan kelas *truck, car,* dan *bus*. Namun jika ingin menghitung motor kurang cocok menggunakan COCO dataset. Dan yang terakhir jika ingin menghitung semua kendaraan yang melintas dijalan baik itu sepeda, motor, truk, bus, *pickup* dan sedan COCO dataset tersebut juga masih kurang cocok diterapkan di wilayah Sukoharjo dan umumnya Indonesia.

5.2. Saran

Dalam penelitian ini tentunya masih banyak ditemukan kekurangan, adapun saran untuk dijadikan perbaikan dalam penelitian selanjutnya yakni:

1) Pada penelitian selanjutnya dapat menggunakan metode deteksi yang berbeda selain YOLOv3 untuk memaksimalkan akurasi deteksi.
2) Penelitian selanjutnya data yang digunakan bisa langsung (*Live*) dari CCTV agar langsung dapat digunakan pada lalu lintas manapun.
3) Untuk instansi yang berminat membuat alat sistem otomatisasi kendaraan berbasis *Convolutional Neural Network* dapat menjadikan skripsi ini sebagai sumber referensi.

DAFTAR PUSTAKA

Al Kautsar, Havez Vazirani. Kusworo Adi. 2016. *Implementasi Object Tracking untuk Mendeteksi dan Menghitung Jumlah Kendaraan Secara Otomatis Menggunakan Metode Kalman Filter dan Gaussian Mixture Model.* Youngster Physics Journal Vol. 5, No.1, Januari 2016, Hal 13-20 ISSN:2302-7371.

Badan Pusat Statistik Indonesia. Diakses tanggal 12 Juni 2020 pukul 13.30 melalui https://www.bps.go.id/.

Dahria, Muhammad. 2008. *Kecerdasan Buatan (Artificial Intelligence).* Jurnal SAINTIKOM Vol. 5, No. 2 Agustus 2008.

C, S, Asha. A, V, Narasimhadhan. (2018). *Vehicle Counting for Traffic Management System using YOLO and Correlation Filter*. IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), Bangalore, 2018, pp. 1-6, doi: 10.1109/CONECCT.2018.8482380.

Jupiyandi, Sisco & Saniputra, Fadhil & Pratama, Yoga & Dharmawan, Muhammad & Cholissodin, Imam. (2019). *Pengembangan Deteksi Citra Mobil Untuk Mengetahui Jumlah Tempat Parkir Menggunakan Cuda Dan Modified Yolo*. Jurnal Teknologi Informasi dan Ilmu Komputer, Vol 6. hlm 413-419. 10.25126/jtiik.201961275.

Lazaro, Alvin & Buliali, Joko & Amaliah, Bilqis. (2017). *Deteksi Jenis Kendaraan di Jalan Menggunakan OpenCV*. Jurnal Teknik ITS. 6. 10.12962/j23373539.v6i2.23175.

Limantoro, Stephen, Ekaputra. Kristian, Yosi. Purwanto, Devi, Dwi. (2017). *Deteksi Pengendara Sepeda Motor Menggunakan Deep Convolutional Neural Networks*. Seminar Nasional Teknologi Informasi dan Komunikasi – SEMANTIKOM 2017

Swastika, Windra. Nur, Albert, Wahyudi. Kelana, Oesman, Hendra. (2019). *Monitoring Ruangan Untuk Deteksi Manusia Berbasis CNN Dengan Fitur Push Notification*. TEKNIKA, Volume 8 No.2, November 2019, ISSN: 2549-8037

Lou, Lu & Zhang, Qi & Liu, Chunfang & Sheng, Minglan & Zheng, Yu & Liu, Xuan. (2019). *Vehicles Detection of Traffic Flow Video Using Deep*

Learning. 2019 IEEE 8th Data Driven Control and Learning Systems Conference (DDCLS), 1012-1017. 10.1109/DDCLS.2019.8908873.

C., Rajesh Babu, and Anirudh G. (2018). *Vehicle Traffic Analysis Using Yolo*. Eurasian Journal of Analytical Chemistry 13 no. SP (2018): emEJAC181256.

Bas, Erhan & Tekalp, A. & Salman, F.. (2007). *Automatic Vehicle Counting from Video for Traffic Flow Analysis*. 2017 IEEE Intelligent Vehicles Symposium 392 - 397. 10.1109/IVS.2007.4290146.

https://id.wikipedia.org/wiki/Kendaraan. Diakses tanggal 19 Juni 2020

https://kbbi.web.id/kendaraan. Diakses tanggal 19 Juni 2020

Kementerian Perhubungan. *Undang-Undang Nomor 2 Tahun 2009-JDIH*. http://jdih.dephub.go.id/assets/uudocs/uu/uu_no.22_tahun_2009.pdf. Diakses tanggal 20 Juni 2020

Poerwadarminta, W.J.S. 1990. *Kamus Besar Bahasa Indonesia*. Balai Pustaka, Jakarta.

Andono, Pulung, Nurtanto. Sutojo, T. Muljono. (2017). *Pengolahan Citra Digital*. Yogyakarta: Penerbit Andi.

Putra, Darma. (2010). *Pengolahan Citra Digital*. Yogyakarta: Penerbit Andi.

Sena, Samuel. (2017). *Pengenalan Deep Learning Part 7: Convolutional Neural Network (CNN)*. https://medium.com/@samuelsena/pengenalan-deep-learning-part-7-convolutional-neural-network-*CNN*-b003b477dc94. Diakses tanggal 20 Juni 2020.

Sofia, Nadhifa. (2018). *Convolutional Neural Network*. https://medium.com/@nadhifasofia/1-convolutional-neural-network-convolutional-neural-network-merupakan-salah-satu-metode-machine-28189e17335b. Diakses tanggal 20 Juni 2020.

Redmon, J., Divvala, S., Girshick, R., dan Farhadi, A. 2015. *You Only Look Once: Unified, Real-Time Object Detection.* Diterima dari https://arxiv.org/abs/1506.02640

Redmon, Joseph & Farhadi, Ali. (2018). *YOLOv3: An Incremental Improvement*. Diterima dari https://arxiv.org/abs/1804.02767.

Rosebrock, Adrian. (2018). *YOLO Object Detection With OpenCV*.

https://www.pyimagesearch.com/2018/11/12/yolo-object-detection-with-opencv/. Diakses tanggal 27 Februari 2020.

Alfarisi, Haiqal Muhammad. (2020). *You Only Look Once (YOLO) Algoritma Deep Learning Object Detection Terbaik.* https://medium.com/@haiqalmuhamadalfarisi/you-only-look-once-yolo-algoritma-deep-learning-object-detection-terbaik-af9ed81de9e9. Diakses tanggal 20 Juni 2020.

Kathuria, Ayoosh. (2018). *What's New In YOLO v3?*. https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b. Diakses tanggal 20 Juni 2020.

Enterprise, Jubilee. 2019. *Python Untuk Programmer Pemula*. Jakarta: Elex Media Computindo.

Kadir, Abdul. 2019. *Langkah Mudah Pemrograman OpenCV & Python*. Jakarta: Elex Media Computindo

Teigens, Vasil. 2019. *Kecerdasan Umum Buatan*. Cambridge: Cambridge Stanford Books

Perez, C. 2019. *DEEP Learning Using Matlab. Neural Network APPLICATIONS*. Morrisville: Lulu Press, Inc.

Bathija, Akhansa & Sharma, Prof. Grishma. (2019). *Visual Object Detection and Tracking Using YOLO and SORT*. International Journal of Engineering Research & Technology (IJERT), Vol 8 Issue, ISSN: 2278-0181.

Bewley, Alex et al. (2016). *Simple Online and Realtime Tracking*. 2016 IEEE International Conference on Image Processing (ICIP): n. pag. Crossref. Web.

Nguk. (2007). *Waterfall Process Model*. https://tonyjustinus.wordpress.com/2007/. Diakses tanggal 20 Juni 2020.

Parker, R, James. 2019. *Game Development Using Python*. Dulles: Mercury Learning and Information.

Turker, Canbazoglu. (2018). *Introduction to Engineering: Engineering Fundamentals and Concepts*. Didapat dari https://books.google.co.id/books?id=8l5-DwAAQBAJ.

Nugroho, Kuncahyo Setyo. (2019). *Confusion Matrix Untuk Evaluasi Model Pada Supervised Learning*. https://medium.com/@ksnugroho/confusion-

matrix-untuk-evaluasi-model-pada-unsupervised-machine-learning-bc4b1ae9ae3f. Diakses tanggal 04 Juli 2020.

YOLO:Real-Time Object Detection. https://pjreddie.com/darknet/yolo/. Diakses tanggal 27 Februari 2020.

Zhao, X., Ni, Y., & Jia, H. 2017. *Modified Object Detection Method Based on YOLO. Computer Vision*, 233–244. doi:10.1007/978-981-10-7305-2_21

LAMPIRAN

Lampiran 1 *Source Code*

```
import numpy as np
import argparse
import imutils
import time
import cv2
import os
import glob
from sort import *


tracker = Sort()
memory = {}
line = [(0, 120), (253, 100)]
line2 = [(360,160), (810, 110)]
counterMasuk = 0
counterKeluar = 0
counted = False

# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-i", "--input",
	help="path to input video", default = "./pagi.mp4")
ap.add_argument("-o", "--output",
	help="path to output video", default = "./output/")
ap.add_argument("-y", "--yolo",
	help="base path to YOLO directory", default = "./yolo-obj")
ap.add_argument("-c", "--confidence", type=float, default=0.5,
	help="minimum probability to filter weak detections")
ap.add_argument("-t", "--threshold", type=float, default=0.3,
	help="threshold when applyong non-maxima suppression")
args = vars(ap.parse_args())

# Return true if line segments AB and CD intersect
def intersect(A,B,C,D):
	return ccw(A,C,D) != ccw(B,C,D) and ccw(A,B,C) != ccw(A,B,D)
	#return ccw(A,B,C) != ccw(B,C,D)

def ccw(A,B,C):
	return (C[1]-A[1]) * (B[0]-A[0]) > (B[1]-A[1]) * (C[0]-A[0])

# load the COCO class labels our YOLO model was trained on
labelsPath = os.path.sep.join([args["yolo"], "coco.names"])
LABELS = open(labelsPath).read().strip().split("\n")

# initialize a list of colors to represent each possible class label
np.random.seed(42)
COLORS = np.random.randint(0, 255, size=(200, 3),
	dtype="uint8")
```

```
# derive the paths to the YOLO weights and model configuration
weightsPath          =          os.path.sep.join([args["yolo"],
"yolov3.weights"])
configPath = os.path.sep.join([args["yolo"], "yolov3.cfg"])

# load our YOLO object detector trained on COCO dataset (80
classes)
# and determine only the *output* layer names that we need from
YOLO
print("[INFO] loading YOLO from disk...")
net = cv2.dnn.readNetFromDarknet(configPath, weightsPath)
ln = net.getLayerNames()
ln = [ln[i[0] - 1] for i in net.getUnconnectedOutLayers()]

# initialize the video stream, pointer to output video file,
and
# frame dimensions
vs = cv2.VideoCapture(args["input"])
writer = None
(W, H) = (None, None)

frameIndex = 0
clasku = ["car", "truck", "bus"]

# try to determine the total number of frames in the video file
try:
      prop = cv2.cv.CV_CAP_PROP_FRAME_COUNT if imutils.is_cv2()
\
            else cv2.CAP_PROP_FRAME_COUNT
      total = int(vs.get(prop))
      print("[INFO] {} total frames in video".format(total))

# an error occurred while trying to determine the total
# number of frames in the video file
except:
      print("[INFO] could not determine # of frames in video")
      print("[INFO] no approx. completion time can be provided")
      total = -1

# loop over frames from the video file stream
while True:
      # read the next frame from the file
      (grabbed, frame) = vs.read()

      #frame = imutils.resize(frame, width=500)
      #frame      = frame[320:720, 470:1280]

      # if the frame was not grabbed, then we have reached the
end
      # of the stream
      if not grabbed:
            break

      # if the frame dimensions are empty, grab them
      if W is None or H is None:
            (H, W) = frame.shape[:2]
```

```
# construct a blob from the input frame and then perform
a forward
# pass of the YOLO object detector, giving us our bounding
boxes
# and associated probabilities
blob = cv2.dnn.blobFromImage(frame, 1 / 255.0, (416, 416),
      swapRB=True, crop=False)
net.setInput(blob)
start = time.time()
layerOutputs = net.forward(ln)
end = time.time() #  initialize  our  lists  of  detected
bounding boxes, confidences,
# and class IDs, respectively
boxes = []
confidences = []
classIDs = []     # loop over each of the layer outputs
for output in layerOutputs:
      # loop over each of the detections
      for detection in output:
      #  extract  the  class  ID  and  confidence  (i.e.,
      probability)
      # of the current object detection
      scores = detection[5:]
      classID = np.argmax(scores)
      confidence = scores[classID]

      #  filter  out  weak  predictions  by  ensuring  the
      detected
      #  probability  is  greater  than  the  minimum
      probability
      if confidence > args["confidence"]:
            #  scale  the  bounding  box  coordinates  back
            relative to
            # the size of the image, keeping in mind that
            YOLO
            # actually returns the center (x, y)-coordinates
            of
            # the bounding box followed by the boxes' width
            and
            # height
            if LABELS[classID] in clasku:
                  box = detection[0:4] * np.array([W, H, W,
                  H])
                  (centerX,  centerY,  width,  height)  =
                  box.astype("int")

                  #  use  the  center  (x,  y)-coordinates  to
                  derive the top
                  # and and left corner of the bounding box
                  x = int(centerX - (width / 2))
                  y = int(centerY - (height / 2))

                  #  update  our  list  of  bounding  box
                  coordinates,
                  # confidences, and class IDs
                  boxes.append([x,      y,      int(width),
                  int(height)])
                  confidences.append(float(confidence))
```

```
                classIDs.append(classID)
            else:
                continue

    # apply non-maxima suppression to suppress weak,
    overlapping
    # bounding boxes
    idxs = cv2.dnn.NMSBoxes(boxes, confidences,
    args["confidence"], args["threshold"])

    dets = []
    if len(idxs) > 0:
        # loop over the indexes we are keeping
        for i in idxs.flatten():
            (x, y) = (boxes[i][0], boxes[i][1])
            (w, h) = (boxes[i][2], boxes[i][3])
            dets.append([x, y, x+w, y+h, confidences[i]])
    np.set_printoptions(formatter={'float': lambda x:
    "{0:0.3f}".format(x)})
    dets = np.asarray(dets)
    tracks = tracker.update(dets)
    boxes = []
    indexIDs = []
    c = []
    previous = memory.copy()
    memory = {}
    for track in tracks:
        boxes.append([track[0], track[1], track[2],
        track[3]])
        indexIDs.append(int(track[4]))
        memory[indexIDs[-1]] = boxes[-1]
    if len(boxes) > 0:
        i = int(0)
        for box in boxes:
            # extract the bounding box coordinates
            (x, y) = (int(box[0]), int(box[1]))
            (w, h) = (int(box[2]), int(box[3]))

            color = [int(c) for c in COLORS[indexIDs[i] %
            len(COLORS)]]
            cv2.rectangle(frame, (x, y), (w, h), color,
            2)

            if indexIDs[i] in previous:
                previous_box = previous[indexIDs[i]]
                (x2, y2) = (int(previous_box[0]),
                int(previous_box[1]))
                (w2, h2) = (int(previous_box[2]),
                int(previous_box[3]))
                p0 = (int(x + (w-x)/2), int(y + (h-y)/2))
                p1 = (int(x2 + (w2-x2)/2), int(y2 + (h2-
                y2)/2))
                cv2.circle(frame, (p0[0], p0[1]), 3,
                color, -1)
```

```
                    if intersect(p0, p1, line[0], line[1]):
                        counterMasuk += 1

                    if intersect(p0, p1, line2[0], line2[1]):
                        counterKeluar +=1
            i += 1

    # draw line
    cv2.line(frame, line[0], line[1], (0, 255, 255), 2)
    cv2.line(frame, line2[0], line2[1], (0, 255, 255), 2)

    # draw counter
    cv2.putText(frame, "Masuk : " + str(counterMasuk),
    (15,27), cv2.FONT_HERSHEY_DUPLEX, 1, (0, 255, 255), 2)
    cv2.putText(frame, "Keluar : " + str(counterKeluar),
    (240,27), cv2.FONT_HERSHEY_DUPLEX, 1, (0, 0, 255), 2)

    if writer is None:
        # initialize our video writer
        fourcc = cv2.VideoWriter_fourcc(*"MJPG")
        writer = cv2.VideoWriter(args["output"], fourcc, 30,
        (frame.shape[1], frame.shape[0]), True)

        # some information on processing single frame
        if total > 0:
            elap = (end - start)
            print("[INFO] single frame took {:.4f}
            seconds".format(elap))
            print("[INFO] estimated total time to finish:
            {:.4f}".format(elap * total))

        # write the output frame to disk
        writer.write(frame)
        # increase frame index
        frameIndex += 1

        if frameIndex >= total:
            print("[INFO] cleaning...")
            writer.release()
            vs.release()
            exit()

# release the file pointers
print("[INFO] cleaning up...")
writer.release()
vs.release()
exit()
```