Analisis Performansi Deteksi Objek Pada Metode *Complex* YOLOv4 Untuk *Autonomous Driving*

1st Hanaluthfina Nurhadiati
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
hanaluthfina@student.telkomuniversity.ac.id

2nd Suryo Adhi Wibowo
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
suryoadhiwibowo@telkomuniversity.ac.id

3rd Agus Pratondo
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
pratondo@student.telkomuniversity.ac.id

Abstrak*— Dalam beberapa tahun terakhir, deteksi objek 3 dimensi (3D) telah diimplementasikan secara luas dan membawa inovasi baru pada sains dan teknologi terkini. Salah satunya pada autonomous driving. Autonomous driving adalah sebutan bagi kendaaran yang dapat mengemudikan kendaraan tanpa kendali manusia. Algoritma deteksi objek merupakan peran utama dalam mengidentifikasi serta memprediksi objek disekitar kendaraan. Kekhawatiran keamanan dan kebutuhan akan estimasi yang akurat secara real-time menyebabkan munculnya sistem deteksi menggunakan Light Detection and Ranging (LiDAR). Tugas Akhir ini menganalisis pengaruh pada modifikasi hyperparameter algoritma yang digunakan untuk meningkatkan performansi deteksi objek pada autonomous driving. Algoritma deteksi objek yang digunakan yaitu Complex YOLOv4. Data input pada metode Complex YOLOv4 berupa 3D point cloud dari LiDar. Hasil keluaran dari penelitian ini berupa model modifikasi terhadap konfigurasi jaringan Complex YOLOv4 dengan nilai performansi terbaik. Dalam Tugas Akhir ini digunakan KITTI Vision Benchmark sebagai dataset training. Skema pengujian Tugas Akhir ini berfokus pada kinerja dua hyperparameter yang dipakai yaitu epoch dan network size. Skema pengujian dengan performansi terbaik didapatkan pada skema III dengan nilai mAP sebesar 58.3%. berdasarkan hasil mAP tersebut, modifikasi terhadap ukuran network size dan penggunaan jumlah epoch yang tinggi dapat mempengaruhi performansi dan kinerja deteksi objek untuk autonomous driving.

***Kata kunci*— Complex YOLOv4, Object Detection, Autonomous Driving, Epoch, Network Size**

I. PENDAHULUAN

Dalam beberapa tahun terakhir, deteksi objek 3 dimensi (3D) telah diimplementasikan secara luas dan membawa inovasi baru pada sains dan teknologi terkini. Salah satunya pada *autonomous driving*. *Autonomous driving* adalah sebutan bagi kendaaran yang dapat mengemudikan kendaraan tanpa kendali manusia. Algoritma deteksi objek berperan untuk identifikasi dan prediksi objek disekitar kendaraan [1]. Kekhawatiran keamanan dan kebutuhan akan estimasi yang akurat secara *real-time* menyebabkan munculnya sistem deteksi menggunakan *Light Detection and Ranging* (LiDAR) [2]. Oleh karena itu, untuk mendukung terciptanya deteksi objek 3D pada *autonomous driving* yang akurat dan *real-time* telah dilakukan beberapa penelitian.

Pada penelitian sebelumnya terdapat beberapa masalah dalam mencapai hasil deteksi yang efesien. Sebagai contoh, metode *Sparsely Embedded Convolutional Detection* (SECOND) [3] oleh Yan Yan dan kawan-kawan. Metode SECOND hanya menggunakan data dari LiDar saja. Metode ini mampu secara *real-time* mendeteksi objek dalam kelas *car*, *pedestrian*, dan *cyclist*. Namun, terjadi beberapa kegagalan dalam mendeteksi kelas *car* secara akurat ketika objek berada jauh dari LiDar atau dalam keadaan lalu lintas yang padat. Metode ini juga menunjukkan kinerja yang lebih rendah pada kelas *pedestrian*, *cyclist*, dan pada deteksi *Bird-Eye-View* (BEV). Pada contoh lain, terdapat metode 3D YOLO [4] oleh Ezeddin Al Hakim. Metode ini menggunakan data dari LiDar saja dan memiliki kinerja yang baik secara *real-time* dalam mendeteksi objek pada kelas yang sama seperti pada metode SECOND. Tetapi, metode 3D YOLO memiliki nilai *Average Precision* (AP) yang belum mampu mencapai hasil maksimal secara *real-time* untuk *autonomous driving*.

Berdasarkan dari permasalahan pada metode-metode tesebut, pada Tugas Akhir ini akan dilakukan pengujian performansi deteksi objek pada metode Complex YOLOv4 untuk *autonomous driving* dengan beberapa skema. Skema pengujian akan dilakukan dengan memodifikasi *hyperparameter* Complex YOLOv4. Dengan harapan mendapatkan hasil performansi deteksi objek secara *real-time* dan akurasi yang lebih baik.

II. KAJIAN TEORI

A. *Autonomous Driving*

Autonomous driving merupakan sebutan bagi kendaaran yang dapat mengemudikan kendaraan tanpa kendali manusia. Hal utama yang harus dimiliki *autonomous driving* yaitu dapat mendeteksi kendaraan, pejalan kaki, dan objek lain di sekitar kendaraan secara akurat dan cepat. Kemampuan deteksi tersebut berguna untuk menghindari terjadinya kecelakaan lalu lintas [1].

B. *Point Cloud*

Pemprosesan *point cloud* menjadi semakin penting untuk *autonomous driving*. *Point cloud* merupakan sekumpulan titik yang mewakili sebuah objek atau ruang dalam 3D. Titik-titik tersebut mewakili koordinat geometris X, Y, dan Z dari hasil deteksi objek. Tahap *point cloud preprocessing* dilakukan sebelum proses pencarian *feature*, rekontruksi permukaan objek 3D, dan visualisasi data. *Point cloud*

dihasilkan menggunakan *laser velodyne* (HDL64) [8] dan *Light Detection and Ranging* (LiDar). Setiap titik yang dihasilkan akan ditransformasikan menjadi *Red Green Blue* (RGB)-*maps*. RGB-*maps* divisualisasikan dari *point cloud* 3D menjadi *grid* 2 dimensi (2D) dengan resolusi *grid*. pada YOLOv4, *point cloud preprocessing* bekerja dengan mengurangi ukuran resolusi *grid* agar kesalahan perhitungan yang dihasilkan kecil dan resolusinya tinggi. Proses deteksi objek 3D pada metode YOLOv4 mengutamakan efesiensi.

C. RGB-*maps*

Red Green Blue (RGB)-*maps* memiliki 3 *layer* warna primer berupa *red*, *green*, dan *blue*. *Layer-layer* tersebut memiliki sejumlah informasi untuk menetukan model warna RGB, hal ini disebut sebagai *triplet* RGB. Triplet RGB antara lain adalah nilai *pixel*, *channel*, dan *filter* yang digunakan pada sebuah citra. *Triplet* RGB dapat dikur dengan berbagai cara. Pada umumnya, komputer menggunakan nilai integer 8-bit mulai dari 0 hingga 255 sebagai representasi pada setiap warna RGB. Saat *point cloud* mendeteksi sebuah objek, objek tersebut akan ditransformasikan menjadi RGB-*maps*. Hal ini bertujuan untuk menghasilkan rekontruksi 3D berbasis citra yang dapat mengatasi perbedaan pencahayaan pada citra dan sebagai model deteksi objek 3D yang dapat bekerja secara *real-time*.

D. *Convolutional Neural Networks*

Convolutional Neural Network (CNN) pada dasarnya adalah struktur klasifikasi untuk mengklasifikasikan gambar ke dalam kelas berkategori [9]. CNN pada umumya melibatkan 3 lapisan utama dalam mengekstraksi sebuah gambar, yaitu convolutional layer, subsampling atau *pooling layer*, dan *fully connected layer* [10].

E. *You Only Look Once*

Kemampuan deteksi objek untuk *autonomous driving* yang akurat dan cepat merupakan hal utama untuk menghindari terjadinya kecelakaan lalu lintas. *You Only Look Once* (YOLO) merupakan salah satu metode yang digunakan untuk mendeteksi objek secara *real-time*. Seiring berkembangnya teknologi, terdapat beberapa versi terbaru dari metode YOLO untuk meningkatkan kualitasnya. Pada dasarnya versi dari YOLO antara lain adalah YOLOv1, YOLOv2, YOLOv3, dan YOLOv4. Diantara versi tersebut, YOLOv4 merupakan metode yang memiliki tingkat performansi deteksi dan kecepatan deteksi yang paling unggul. Namun, untuk tingkat presisinya belum sempurna [13]. Pada Tugas Akhir ini digunakan metode Complex YOLOv4 yang menggunakan pendekatan algoritma YOLOv4 untuk menghasilkan tingkat performansi yang lebih baik.

F. Complex YOLOv4

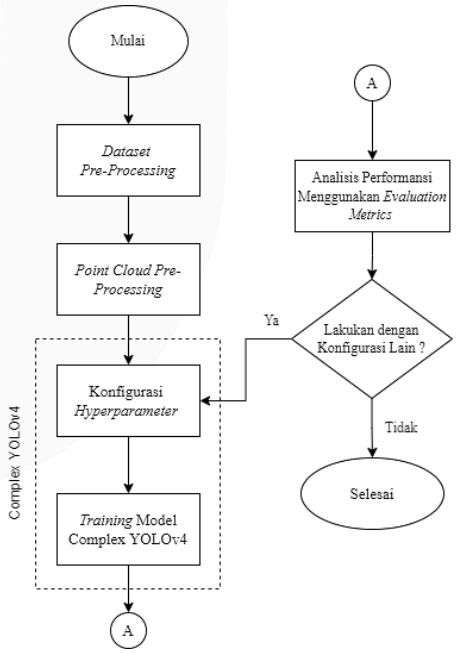
Complex YOLOv4 merupakan metode yang menggunakan pendekatan Complex YOLOv4 dengan algoritma YOLOv4. Metode ini memiliki tingkat presisi yang tinggi dan efektif dalam mendeteksi objek. Dalam mendeteksi objek 3 dimensi (3D) yang berorientasi multi-kelas saat beroperasi secara *real-time*, Complex YOLOv4 menggunakan LiDar dan sudut kompleks yang berhubungan dengan regresi. Maka dari itu, metode ini dinamakan Complex YOLOv4 [8]. Dalam menciptakan orientasi yang akurat pada *Bounding Box* (Bbox) digunakan *Euler Region Proposal* (E-RPN) digunakan untuk memperkirakan orientasi objek berdasarkan nilai imajiner dan riil. Oleh karena itu, prediksi Bbox hanya dilakukan dalam satu *inference*. E-RPN merupakan bagian dari network dan menggunakan keluaran dari lapisan terakhir CNN untuk memprediksi Bbox [12].

G. *Hyperparameter*

Dalam mengoptimalkan kinerja pembelajaran dasar *machine learning* terdapat 2 jenis parameter. Parameter model dan *hyperparameter*. Parameter model dapat diinisialisasikan dan diperbarui melalui proses *training* model, misalnya saat menetukan nilai *neuron* dalam *neural network*. Sedangkan *hyperparameter* tidak dapat diinisialisasikan secara langsung melalui proses pembelajaran data. *Hyperparameter* harus ditentukan sebelum sebuah model akan di *training*. Konfigurasi arsitektur model yang ideal dan optimal dapat dilakukan dengan mengatur komponen pada *hyperparameter* [14]. Salah satu komponen pada *hyperparameter* yaitu *batch size*, *epoch*, dan *network size*.

III. METODE

Pada Tugas Akhir ini akan dilakukan analisis performansi deteksi objek pada metode Complex YOLOv4 untuk *autonomous driving*. Penelitian ini dilakukan percobaan modifikasi pada 3 *hyperparameter* metode Complex YOLOv4, diantaranya adalah *epoch* dan *network size*. Hasil dari beberapa skema pengujian akan dianalisis dengan mengukur tingkat nilai *evaluation metrics* masing-masing model. Secara umum diagram alir sistem Tugas Akhir ini memiliki alur kerja seperti pada Gambar 1.



GAMBAR 1
DESAIN ALIR SISTEM

A. *Dataset Pre-Processing*

Pada Tugas Akhir ini menggunakan dataset dari KITTI Vision Benchmark dengan kelas deteksi 3D yang dibatasi berupa *car*, *pedestrian*, dan *cyclist*. *Dataset* dari KITTI Vision Benchmark merupakan *dataset open-sorce*. Pada *dataset* yang digunakan terdapat 5.481 data sampel untuk

training dan 5.518 data sampel untuk *testing*. *File* pada data sampel *training* berisi *image*, *label*, *calib*, dan *velodyne*. Pada *file* data sampel *testing* berisi *image*, *calib*, dan *velodyne*.

B. *Point Cloud Pre-Processing*

Point cloud yang dihasilkan *laser velodyne* (HDL64) [8] dan *Light Detection and Ranging* (LiDar) selanjutnya akan dikonversi menjadi RGB-*maps*. *Point cloud* mengandung data RGB yang memberikan warna pada setiap titik-titik yang mewakili sebuah objek atau ruang dalam 3D, sehingga dapat membentuk model 3D yang menyerupai objek yang didapatkan oleh *laser* pada LiDAR. Titik-titik tersebut mewakili koordinat geometris X, Y, dan Z dari hasil deteksi objek. RGB-*maps* dikodekan dengan ketinggian (*height*), intensitas (*intensity*), dan kepadatan (*density*). *Point cloud* diproyeksikan pada *grid* 2D. Ukuran *grid* yang digunakan yaitu n = 1.204 dan m = 512 dengan resolusi *cell* dari setiap *grid* 8.

C. Konfigurasi *Hyperparameter*

Pada Tugas akhir ini menggunakan skema konfigurasi *hyperparameter* diantaranya berupa *epoch* dan *network size*. Konfigurasi *hyperparameter* dilakukan untuk memperoleh skema pengujian terbaik dalam mendeteksi objek 3D secara *real-time* dan akurat. Tugas akhir ini menggunakan konfigurasi *batch size* dengan nilai 4 untuk semua skema pengujian serta konfigurasi *epoch* dengan jumlah 10, 20 dan 30. Sedangkan untuk konfigurasi *network size* ukuran yang digunakan antara lain adalah 512 × 512 dan 256 × 256. Ukuran konfigurasi pada ketiga *hyperparamter* ini merupakan batas maksimum ukuran yang dapat didukung oleh Google Colab Pro.

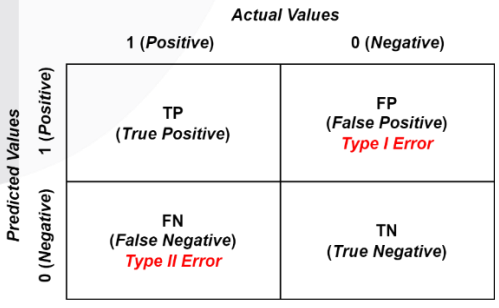
D. *Training* Model Complex YOLOv4

Pada tahap ini dilakukan proses *training* model Complex YOLOv4 menggunakan Google Colab Pro. Proses ini dilakukan untuk mengetahui kinerja dari setiap skema pengujian. Tahap pertama dilakukan *mount* to Google Drive yang bertujuan untuk mengakses *file* yang tersimpan pada Google Drive. Setelah dilakukan *mount* to Google Drive, tahap selanjutnya adalah melakukan perintah Git Clone dan *Install Requirements* untuk membuat salinan repositori dalam Google Drive dan meng*install* beberapa persyaratan untuk menjalankan proses *training* skema pengujian menggunakan metode Complex YOLOv4. Lalu dilakukan proses *training* berdasarkan skema pengujian yang digunakan dan dilakukan evaluasi hasil performansi menggunakan *evaluation metrics*. Berdasarkan skema pengujian yang telah dirancang, proses *training* model dilakukan sebanyak enam kali percobaan. Setelah dilakukan proses *training* model, selanjutnya akan dilakukan analisis performansi menggunakan *evaluation metrics*. Rincian modifikasi pada konfigurasi Complex YOLOv4 diantaranya sebagai berikut:

1. Complex YOLOv4 Orisinal: *Train* model orisinal ini dilakukan tanpa modifikasi pada bagian *hyperparameter*. Sehingga dijadikan acuan untuk perbandingan performansi dengan model skema lainnya.
2. Skema I: *Train* model skema I dimodifikasi pada bagian *hyperparameter epoch* dan *network size* dengan jumlah *epoch* 10 dan ukuran *network size* 512 × 512. Skema I bertujuan untuk mengetahui pengaruh performansi model.
3. Skema II: *Train* model skema II dimodifikasi pada bagian *hyperparameter epoch* dan *network size* dengan jumlah *epoch* 20 dan ukuran *network size* 512 × 512. Skema II bertujuan untuk mengetahui pengaruh performansi model.
4. Skema III: *Train* model skema III dimodifikasi pada bagian *hyperparameter epoch* dan *network size* dengan jumlah *epoch* 30 dan ukuran *network size* 512 × 512. Skema III bertujuan untuk mengetahui pengaruh performansi model.
5. Skema IV: *Train* model skema IV dimodifikasi pada bagian *hyperparameter epoch* dan *network size* dengan jumlah *epoch* 10 dan ukuran *network size* 256 × 256. Skema IV bertujuan untuk mengetahui pengaruh performansi model.
6. Skema V: *Train* model skema I dimodifikasi pada bagian *hyperparameter epoch* dan *network size* dengan jumlah *epoch* 10 dan ukuran *network size* 256 × 256. Skema V bertujuan untuk mengetahui pengaruh performansi model.
7. Skema VI: *Train* model skema I dimodifikasi pada bagian *hyperparameter epoch* dan *network size* dengan jumlah *epoch* 30 dan ukuran *network size* 256 × 256. Skema VI bertujuan untuk mengetahui pengaruh performansi model.

E. Paramater Performansi

Dalam Tugas Akhir ini digunakan *evaluation metrics* sebagai cara untuk mengukur seberapa baik model *machine learning* yang sudah dirancang. Pada Tugas Akhir ini *evaluation metrics* yang digunakan antara lain adalah *Precision*, *Recall*, *Average Precision* (AP), F1 *Score*, dan *Mean Average Precision* (mAP). Selain itu, terdapat *confusion matrix* atau *error matrix*. *Confusion matrix* [16] memberikan informasi perbandingan dari hasil klasifikasi yang dilakukan oleh model dengan hasil klasifikasi sebenarnya. *Confusion matrix* berbentuk tabel yang menggambarkan kinerja sebuah model seperti pada Gambar 2.



GAMBAR 2
CONFUSION MATRIX

Precision merupakan perbandingan antara *True Positive* (TP) dengan banyaknya data yang diprediksi *True Positive* (TP) dan *False Positive* (FP) seperti pada persamaan 1. *Precision* mereprentasikan tingkat presisi antara data yang diinginkan dengan hasil prediksi yang dihasilkan oleh model.

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \quad (1)$$

Recall merupakan perbandingan antara prediksi *True Positive* (TP) dengan banyaknya data yang diprediksi *True Positive* (TP) dan *False Negative* (FN) seperti pada persamaan 2. *Recall* merepresentasikan tingkat keberhasilan model dalam menemukan kembali sebuah informasi.

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \quad (2)$$

Average Precision (AP) adalah cara untuk merangkum nilai *precision* dan *recall* menjadi satu nilai yang mewakili rata-rata semua *precision*. AP dihitung seperti pada persamaan 3. Menggunakan *loop* yang melewati semua *precision* dan *recall*, perbedaan *recall* saat ini dan berikutnya akan dihitung dan kemudian dikalikan dengan *precision* saat ini. Dimana, n adalah nilai dari *thresholds*.

$$\mathrm{AP} = \sum_{k=0}^{k=n-1} [Recall(k) - Recall(k-1)] \times Precision(k)] \quad (3)$$

F1 *Score* merupakan perbandingan rata-rata *precision* dan *recall* seperti pada persamaan 4. Pada dasarnya, F1 *Score* memiliki nilai yang baik ketika model yang digunakan mempunyai nilai *precision* dan *recall* yang baik. Nilai terbaik F1 *Score* adalah (1.0) dan nilai terburuknya adalah (0.0). F1 *Score* pada umumnya digunakan ketika sebuah dataset memiliki masalah imbalanced yaitu jumlah data *False Negatif* (FN) dan *False Positif* (FP) yang tidak seimbang.

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

Mean Average Precision (mAP) merupakan nilai rata-rata *Average Precision* (AP) dari suatu objek pada setiap kelas. Perhitungan mAP dapat dilakukan dengan menggunakan persamaan seperti pada persamaan 5. Berdasarkan persamaan 3.5, N merupakan jumlah keseluruhan kelas yang akan di training, i merupakan iterasi yang dimulai dari nilai 1, dan AP adalah *Average Precision*.

$$mAP = \frac{1}{\mathrm{N}} \sum_{k=1}^{k=n} \mathrm{AP_k} \times 100\% \quad (5)$$

IV. HASIL DAN PEMBAHASAN

Berdasarkan hasil pengujian nilai *evaluation metrics* yang diantaranya adalah *Precision*, *Recall*, *Average Precision* (AP), *F1 Score*, dan *Mean Average Precision* (mAP) model yang akan penulis jadikan acuan sebagai model acuan adalah model yang telah melalui proses *training*. Hal ini dikarenakan hasil *evaluation metrics* yang didapatkan merupakan hasil modifikasi dan optimasi dari proses *training* model Complex YOLOv4 orisinal. Pada setiap *training* yang dilakukan nilai *batch size* yang digunakan adalah 4 serta jumlah *epoch* yang digunakan adalah 10, 20, dan 30. Selain itu, akan dilakukan modifikasi konfigurasi *network size* pada setiap *epoch* dengan ukuran 512 × 512 dan 256 × 256. Nilai mAP yang mendekati 100% adalah yang terbaik.

A. Analisis Nilai mAP pada Model Orisinal dengan *Epoch*

Pengujian pertama dengan model Complex YOLOv4 orisinal dilakukan dengan menggunakan tiga jumlah *epoch*, yaitu 10, 20, dan 30. *Epoch* digunakan untuk menunjukkan berapa kali sebuah sampel telah dijalankan selama proses *training*. Seperti pada Tabel 1, saat jumlah *epoch* ditambah semakin besar dimulai dari 10 hingga 30, maka terjadi peningkatan secara *linear* dengan kenaikan 16.3% pada *epoch* 20 dan 5% pada *epoch* 30. *Epoch* 30 mendapatkan nilai mAP terbaik dibandingkan dengan *epoch* 10 dan 20 yaitu sebesar 57.1%. Jika jumlah *dataset* yang digunakan dalam proses *training* lebih banyak, penggunaan *epoch* dengan jumlah besar akan menghasilkan nilai akurasi yang tinggi dan lebih baik dibandingkan dengan jumlah *epoch* yang kecil [17].

TABEL 1
HASIL MAP MODEL ORISINAL

No.	*Epoch*	mAP
1.	10	35.8%
2.	20	52.1%
3.	s	57.1%

B. Analisis Nilai mAP pada Model Orisinal dengan *Network Size*

Pada pengujian ini dilakukan dengan struktur yang sama seperti model Complex YOLOv4 orisinal namun terdapat modifikasi pada konfigurasi *network size* dengan dua ukuran 512 × 512 dan 256 × 256. Penyesuaian ukuran *network size* dipengaruhi oleh kapasitas memori GPU untuk mencegah terjadinya kelebihan beban. Seperti pada Tabel 2, pengujian ini dilakukan dengan 3 jumlah *epoch* 10,20, dan 30. Saat jumlah *epoch* ditambah semakin besar dimulai dari 10 hingga 30, maka terjadi peningkatan secara *linear*. Akan tetapi, pada *epoch* 30 *network size* yang berukuran kecil yaitu 256 × 256 mengalami penurunan. Hal ini disebabkan tidak semua *feature* kompatibel satu sama lain dan beberapa kombinasi *feature* sering kali mengurangi akurasi sebuah model saat digunakan bersama. Mendeteksi beberapa objek dengan berbagai ukuran dan lokasi yang tepat memerlukan ukuran *network size* yang lebih besar dalam jaringan deteksi objek. Lebih lanjut, untuk peningkatan yang dialami pada *network size* 512 × 512 saat jumlah *epoch* 30 memiliki nilai mAP lebih besar 1.2% daripada model orisinal. Hal ini disebabkan oleh penggunaan ukuran *network size* yang di *training* dengan *feature* yang meningkatkan akurasi dalam deteksi objek dapat berdampak negatif pada akurasi detektor di beberapa jaringan deteksi objek.

TABEL 2
HASIL MAP HASIL MODIFIKASI PADA KONFIGURASI NETWORK SIZE

Skema	*Epoch*	*Network Size*	mAP
I	10	512 × 512	36.9%
II	20	512 × 512	51.2%
III	30	512 × 512	58.3%
IV	10	256 × 256	37.1%
V	20	256 × 256	54.1%
VI	30	256 × 256	55.9%

C. Analisis Nilai *Precision* pada Model Orisinal dan Skema I s.d VI

Pada analisis ini dibandingkan nilai *precision* pada model Complex YOLOv4 orisinal dan skema I s.d VI dengan tiga jumlah *epoch* mulai dari 10, 20, dan 30. Nilai *precision* untuk setiap *epoch* dapat dilihat pada Tabel 3. Kelas *car* mengungguli nilai *precision* diantara kelas *pedestrian* dan *cyclist*. Hal ini disebabkan oleh adanya ketidakseimbangan *dataset* atau *imbalance dataset* pada KITTI Vision Benchmark. Berdasarkan jumlah *label* setiap kelas, kelas *car* memiliki *label* terbanyak diantara kelas lainnya seperti pada Tabel 4. Hal tersebut menyebabkan akurasi dominan terhadap kelas *car*. Berdasarkan perbandingan nilai *precision* setiap kelas pada Tabel 3, saat jumlah *epoch* ditambah semakin besar dimulai dari 10 hingga 30, maka akan terjadi peningkatan secara *linear*. Hal ini disebabkan oleh ukuran *epoch* yang lebih besar membuat model yang di *training* memiliki akurasi yang semakin tinggi dan lebih baik serta menghasilkan nilai *precision* yang lebih tinggi juga. Pada kelas *car* skema VI mendapatkan nilai *precision* tertinggi sebesar 0.600, serta pada kelas *pedestrian* nilai *precision* tertinggi didapatkan oleh skema III dan skema VI dengan nilai 0.231. Sedangkan nilai *precision* tertinggi pada kelas *cyclist* dimiliki oleh skema III dengan nilai 0.247.

TABEL 3
HASIL PRECISION PADA MODEL ORISINAL DAN SKEMA I S.D VI

Skema	*Epoch*	*Precision*		
		Car	*Pedestrian*	*Cyclist*
Orisinal	10	0.405	0.120	0.059
	20	0.462	0.160	0.148
	30	0.595	0.216	0.190
I	10	0.377	0.125	0.059
II	20	0.484	0.144	0.137
III	30	0.594	0.231	0.247
IV	10	0.393	0.111	0.074
V	20	0.459	0.160	0.142
VI	30	0.600	0.231	0.211

TABEL 4
JUMLAH LABEL PADA SETIAP KELAS OBJEK

Kelas	*Labels*
Car	4.110
Pedestrian	1.096
Cyclist	275

D. Analisis Nilai *Recall* pada Model Orisinal dan Skema I s.d VI

Pada analisis ini dibandingkan nilai *recall* pada model Complex YOLOv4 orisinal dan skema I s.d VI dengan tiga jumlah *epoch* mulai dari 10, 20, dan 30. Nilai *recall* untuk setiap *epoch* dapat dilihat pada Tabel 5. Berdasarkan perbandingan nilai *recall* setiap kelas pada Tabel 5, pada kelas *car*, model orisinal pada *epoch* 30 mendapatkan nilai *recall* tertinggi sebesar 0.969, serta pada kelas *pedestrian* nilai *recall* tertinggi didapatkan oleh skema II dengan nilai 0.843. Sedangkan nilai *recall* tertinggi pada kelas *cyclist* dimiliki oleh skema III dengan nilai 0.699. Nilai *recall* yang tergolong rendah dapat disebabkan banyaknya gangguan pada gambar *dataset* seperti banyaknya gambar yang gelap ataupun terlalu terang sehingga hal tersebut dikenali sebagai objek dan mempengaruhi hasil deteksi objek.

TABEL 5
HASIL RECALL PADA MODEL ORISINAL DAN SKEMA I S.D VI

Skema	*Epoch*	*Recall*		
		Car	*Pedestrian*	*Cyclist*
Orisinal	10	0.914	0.691	0.386
	20	0.962	0.823	0.646
	30	0.969	0.837	0.676
I	10	0.915	0.747	0.409
II	20	0.957	0.843	0.650
III	30	0.968	0.817	0.699
IV	10	0.914	0.767	0.462
V	20	0.962	0.836	0.656
VI	30	0.966	0.837	0.686

E. Analisis Nilai AP pada Model Orisinal dan Skema I s.d VI

Pada analisis ini dibandingkan nilai *Average Precision* (AP) pada model Complex YOLOv4 orisinal dan skema I s.d VI dengan tiga jumlah *epoch* mulai dari 10, 20, dan 30. Nilai AP untuk setiap *epoch* dapat dilihat pada Tabel 6. Berdasarkan perbandingan nilai AP setiap kelas pada Tabel 6, terjadi peningkatan nilai AP secara *linear*. Skema III mendapatkan nilai AP tertinggi pada setiap kelas objek dengan nilai sebesar 0.929, 0.449, dan 0.371. Karena adanya ketidakseimbangan *dataset* atau *imbalance dataset* pada KITTI Vision Benchmark, pengujian model akan mempelajari *feature* pada kelas objek *car* secara keseluruhan dibandingkan dengan kelas objek yang seharusnya.

TABEL 6
HASIL AP PADA MODEL ORISINAL DAN SKEMA I S.D VI

Skema	*Epoch*	*AP*		
		Car	*Pedestrian*	*Cyclist*
Orisinal	10	0.799	0.247	0.029
	20	0.895	0.392	0.277
	30	0.926	0.446	0.340
I	10	0.796	0.279	0.032
II	20	0.893	0.428	0.216
III	30	0.929	0.449	0.371
IV	10	0.813	0.247	0.053
V	20	0.896	0.446	0.280
VI	30	0.918	0.424	0.334

F. Analisis Nilai F1 *Score* pada Model Orisinal dan Skema I s.d VI

Pada analisis ini dibandingkan nilai F1 *Score* pada model Complex YOLOv4 orisinal dan skema I s.d VI dengan tiga jumlah *epoch* mulai dari 10, 20, dan 30. Nilai F1 *Score* untuk setiap *epoch* dapat dilihat pada Tabel 7. Berdasarkan perbandingan nilai F1 *Score* setiap kelas pada Tabel 7,

terdapat peningkatan nilai F1 *Score* secara linear pada setiap kelas objek. Dapat disimpulkan nilai dari hasil *training* untuk proses uji setiap model skema berjalan dengan baik dan sudah memiliki keandalan yang baik. Pada kelas *car* dan *pedestrian* nilai F1 *Score* tertinggi dimiliki oleh skema VI dengan nilai 0.740 dan 0.362. Sedangkan pada kelas *cyclist* didapatkan nilai tertinggi pada skema III dengan nilai sebesar 0.365.

TABEL 7
HASIL F1 SCORE PADA MODEL ORISINAL DAN SKEMA I S.D VI

Skema	*Epoch*	*F1 Score*		
		Car	*Pedestrian*	*Cyclist*
Orisinal	10	0.561	0.205	0.103
	20	0.624	0.269	0.241
	30	0.737	0.344	0.297
I	10	0.534	0.214	0.103
II	20	0.643	0.246	0.227
III	30	0.737	0.360	0.365
IV	10	0.549	0.195	0.127
V	20	0.621	0.269	0.233
VI	30	0.740	0.362	0.323

V. KESIMPULAN

Pengujian dan analisis modifikasi *hyperparameter* pada algoritma metode Complex YOLOv4 telah berhasil dilakukan. Modifikasi *hyperparameter* juga telah berhasil diimplementasikan dan mendapatkan performansi yang lebih baik dibandingkan dengan model Complex YOLOv4 orisinal. Performansi terbaik berdasarkan nilai mAP dari setiap skema pengujian yang telah dilakukan didapatkan pada model skema III dengan nilai mAP sebesar 58.3%. Adapun beberapa kesimpulan dari hasil pengujian dan analisis adalah Pada penelitian ini telah dilakukan modifikasi dua *hyperparameter* yang dipakai yaitu *epoch* dan *network size* yaitu ppada algoritma deteksi objek untuk autonomous driving dengan metode Complex YOLOv4, terjadi peningkatan performansi pada konfigurasi modifikasi dibandingkan konfigurasi model orisinal khususnya pada pengujian skema III dengan ukuran *network size* 512 × 512 dan jumlah *epoch* 30, melalui pengujian dengan *training dataset*, didapatkan hasil bahwa setiap konfigurasi memiliki keunggulan pada masing-masing pengujian, dan terdapat ketidakseimbangan *dataset* atau *imbalance dataset* pada KITTI Vision Benchmark. Berdasarkan jumlah *label* setiap kelas, kelas *car* memiliki *label* terbanyak diantara kelas lainnya. Hal ini menyebabkan akurasi dominan terhadap kelas *car*.

REFERENSI

[1] R. Wang et al., "A Real-Time Object Detector for Autonomous Vehicles Based on YOLOv4," Computational Intelligence and Neuroscience, vol. 2021, 2021, doi: 10.1155/2021/9218137.

[2] Y. Li and J. Ibanez-Guzman, "Lidar for Autonomous Driving: The principles, challenges, and trends for automotive lidar and perception systems," Apr. 2020, doi: 10.1109/MSP.2020.2973615.

[3] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," Sensors (Switzerland), vol. 18, no. 10, Oct. 2018, doi: 10.3390/s18103337.

[4] E. al Hakim, "3D YOLO: End-to-End 3D Object Detection Using Point Clouds," 2018.

[5] Y. Zhou and O. Tuzel, "VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection," Nov. 2017, [Online]. Available: http://arxiv.org/abs/1711.06396

[6] "How Autonomous Vehicles Sensors Fusion Helps Avoid Deaths | Intellias Blog." https://intellias.com/sensor-fusion-autonomous-cars-helps-avoid-deaths-road/ (accessed Jun. 29, 2022).

[7] "How Autonomous Cars Map The Environment – SWS Website." https://www.smallworldsocial.com/how-autonomous-cars-map-the-environment/ (accessed Jun. 29, 2022).

[8] M. Simon, S. Milz, K. Amende, and H.-M. Gross, "Complex-YOLO: Real-time 3D Object Detection on Point Clouds," Mar. 2018, [Online]. Available: http://arxiv.org/abs/1803.06199

[9] T. Okuyama, T. Gonsalves, and J. Upadhay, "Autonomous Driving System based on Deep Q Learning," 2018. doi: 10.1109/ICoIAS.2018.8494053.

[10] Z. Yi, "Evaluation and Implementation of Convolutional Neural Networks in Image Recognition," in Journal of Physics: Conference Series, Oct. 2018, vol. 1087, no. 6. doi: 10.1088/1742-6596/1087/6/062018.

[11] IEEE Staff, 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)(. IEEE, 2017.

[12] W. Wang and Y. Yang, "Development of convolutional neural network and its application in image classification: a survey," Optical Engineering, vol. 58, no. 04, p. 1, Apr. 2019, doi: 10.1117/1.oe.58.4.040901.

[13] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," Apr. 2020, [Online]. Available: http://arxiv.org/abs/2004.10934

[14] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," Neurocomputing, vol. 415, pp. 295–316, Nov. 2020, doi: 10.1016/j.neucom.2020.07.061.

[15] P. M. Radiuk, "Impact of Training Set Batch Size on the Performance of Convolutional Neural Networks for Diverse Datasets," Information Technology and Management Science, vol. 20, no. 1, pp. 20–24, Jan. 2017, doi: 10.1515/itms-2017-0003.

[16] H. Dalianis, "Evaluation Metrics and Evaluation," in Clinical Text Mining, Springer International Publishing, 2018, pp. 45–53. doi: 10.1007/978-3-319-78503-5_6.

[17] M. Resa Arif Yudianto, H. al Fatta, and Kusrini, "ANALISIS PENGARUH TINGKAT AKURASI KLASIFIKASI CITRA WAYANG DENGAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK," Daerah Istimewa Yogyakarta, Dec. 2020. doi: 10.36294/jurti.v4i2.1319.