Analisis Performansi Complex-Yolov4 Menggunakan Pendekatan Upsampling Untuk Objek Deteksi 3-Dimensi (3d) Pada Autonomous Driving

1st Dena Ananda
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
bintangsandyp@student.telkomuniversity.ac.id

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

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

Abstrak

Penggunaan teknologi LiDAR dengan metode *Complex*-YOLOv4 pada pengujian dengan *dataset* uji KITTI (*Karlsruhe Institute of Technology and Toyota Technological Institute*) menjadi metode yang baik pada teknologi objek deteksi 3D secara *real-time*. Namun, hasil pendeteksian untuk kelas pejalan kaki dan pesepeda pada model ini masih kurang akurat dan perlu ditingkatkan. Pada Tugas Akhir ini dilakukan analisis pada backbone CSPDarknet-53 dari model Complex-YOLOv4 dengan menggunakan pendekatan *upsample* diantaranya yaitu *bicubic*, *bilinear*, dan *nearest*. Parameter yang digunakan untuk perbandingan uji performansi diantaranya *precision*, *recall*, *average precision* (AP), *f1-score*, dan *mean average precision* (mAP). Hasil dari tiap pendekatan *upsample* akan dilakukan perbandingan dengan *upsample expand* (*upsample* orisinal). Dari hasil evaluasi, *upsample nearest* menghasilkan mAP 88,7%. *Upsample bilinear* menghasilkan mAP 88,5%. Pendekatan *upsample* terbaik didapatkan dengan menggunakan *upsample bicubic*. *Upsample bicubic* menghasilkan mAP terbaik yaitu 88,9% dan lebih tinggi 0.2% dari *upsample* orisinal yang digunakan pada *Complex*-YOLOv4.

Kata kunci : autonomous driving, Complex-YOLOv4, LiDAR, object detection, point cloud.

Abstract

The use of LiDAR technology with the Complex-YOLOv4 method in testing with the KITTI test dataset (Karlsruhe Institute of Technology and Toyota Technological Institute) is a good method for real-time 3D object detection technology. However, the detection results for the class of pedestrians and cyclists in this model are still less accurate and need to be improved. In this final project, an analysis of the CSPDarknet-53 backbone from the Complex-YOLOv4 model is carried out using an upsample approach including bicubic, bilinear, and nearest. The parameters used for comparison of performance tests include precision, recall, average precision (AP), f1-score, and mean average precision (mAP). The results of each upsample approach will be compared with the upsample expand (original upsample). From the evaluation results, the nearest upsample resulted in an mAP of 88.7%. Bilinear upsample yielded 88.5% mAP. The best upsample approach is obtained by using the bicubic upsample. The bicubic upsample produced the best mAP, which was 88.9% and 0.2% higher than the original upsample used in Complex-YOLOv4.

Keywords: autonomous driving, Complex-YOLOv4, LiDAR, object detection, point cloud.

I. PENDAHULUAN

Pada sebuah penelitian menyatakan bahwa dalam beberapa tahun terakhir, salah satu tantangan dalam deteksi objek 3D adalah mengurangi latensi *inference* (beban komputasi) dan memori tanpa mengorbankan akurasi deteksi berbasis LiDAR untuk *autonomous driving* [1]. Oleh karena itu, masalah tersebut menjadi perhatian bagi para peneliti untuk mengoptimalkan deteksi objek 3D dengan tetap mempertahankan tingkat akurasi. Beberapa penelitian deteksi objek 3D untuk *autonomous driving* telah dilakukan.

Frustum-based Networks [3] telah menunjukkan performansi tinggi di deretan KITTI (*Karlsruhe Institute of Technology and Toyota Technological Institute*) *Benchmark*. Model ini memiliki dua kelemahan yaitu pertama adalah akurasi model sangat tergantung pada gambar kamera dan *Convolutional Neural Network* (CNN) yang digunakan, sehingga tidak mungkin untuk menerapkan pendekatan hanya meggunakan data dari LiDAR serta model saat di-*inference* berjalan dengan kecepatan yang terlalu rendah pada sekitar 7 *frame per second* (fps) menggunakan GPU NVIDIA GTX 1080i sehingga sangat tidak efisien diterapkan secara *real-time*. Adapun penelitian yang hanya menggunakan data dari LiDAR saja yaitu dari Yin

Zhou dan kawan-kawanya [4]. Selain itu, ada juga Chen X dan kawan-kawannya menghasilkan suatu model *Multi-View 3D Object Detection Network* (MV3D) [5]. Kelebihan dari model ini yaitu waktu komputasi yang cepat menggunakan NVIDIA GTX 1080i GPU, namun sayangnya terdapat kekurangan yaitu kebutuhan input sensor sekunder (kamera).

Berdasarkan permasalahan diatas, maka pada Tugas Akhir ini diusulkan untuk melakukan pengembangan pada metode yang sudah ada yaitu *YOLO-Complex* [2] dengan analisis menggunakan pendekatan upsample pada bagian *backbone CSPDarknet-53* untuk mengetahui metode ini dapat bekerja dengan baik secara *real-time*. Skema pengujian akan dilakukan setelah melakukan analisis. Dengan ini diharapkan Tugas Akhir ini dapat digunakan sebagai referensi pada penelitian selanjutnya.

II. KAJIAN TEORI

A. *Light Detection and Ranging* (LiDAR)

Light Detection and Ranging (LiDAR) merupakan salah satu teknologi yang mengandalkan sensor dan radar. Pemanfaatan LiDAR ini sering digunakan dalam bidang geomorfologi, peraba jarak jauh, geologi dan geografi [6]. Sebutan lain untuk LiDAR adalah ALSM (*Airborne Laser Swath Mapping*). Terdapat beberapa metode pemindaian yang digunakan oleh LiDAR yaitu azimuth and elevation, dual oscillating plane mirrors, dual axis scanner, dan polygonal mirrors [8].

LiDAR akan mengirimkan cahaya *infrared* pada objek disekitarnya, lalu objek tersebut akan dipantulkan kembali pada LiDAR. Waktu yang dibutuhkan untuk memancarkan cahaya dan menangkap kembali cahaya digunakan sebagai pengukuran jarak oleh LiDAR. Data yang dihasilkan saat LiDAR memperoleh informasi tentang objek di lingkunagnnya yaitu berupa *point clouds* [7].

B. *Convolutional Neural Network* (CNN)

Convolutional Neural Network (CNN) merupakan jenis *neural network* untuk mendeteksi dan mengenali objek pada sebuah gambar atau video. CNN terdiri dari *neuron* yang memiliki *weight, bias dan activation function* [9]. Secara umum, CNN terbagi menjadi tiga layer, diantaranya *convolutional layer, pooling layer dan fully-connected layer* [10]. CNN bekerja dengan operasi konvolusi.

C. You Only Look Once (YOLO)

You Only Look Once (YOLO) mempunyai algoritma *convolutional neural network* (CNN) untuk mendeteksi objek secara real-time. Algoritma ini sering digunakan dibandingkan dengan model lain karena kecepatan dan akurasinya yang lebih unggul. YOLO dapat mendeteksi objek lebih cepat dibandingkan dengan *RCNN* yang membutuhkan waktu cukup lama karena menggunakan banyak network untuk mendeteksi satu citra [12]. Algoritma pada YOLO menggunakan 3 teknik yaitu *residual blocks, bounding box regression*, dan *Intersection over Union* (IoU).

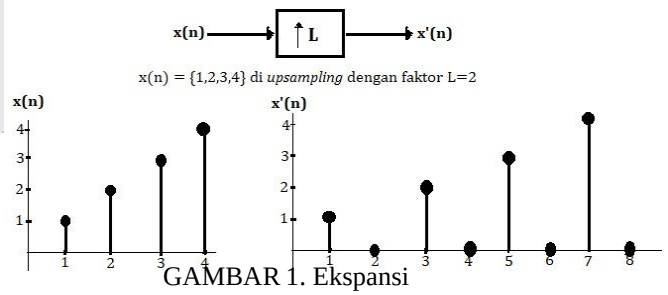
YOLO menggunakan *bounding box regression* tunggal untuk memprediksi tinggi, lebar, pusat, dan kelas objek [13]. *Bounding box* adalah garis yang menyoroti objek dalam gambar. IoU dalam deteksi objek menggambarkan bagaimana kotak tumpang tindih atau beririsan. Tujuan dari IoU ini yaitu untuk mengevaluasi area 2 *bounding box* yang saling beririsan yaitu *bounding box* hasil prediksi dan *bounding box ground truth* (kebenaran)

D. *Python*

Python adalah bahasa pemrograman sederhana yang dapat digunakan untuk berbagai macam program dan dapat digunakan di berbagai *platform*. Bahasa pemrograman ini populer digunakan untuk *data science, machine learning, deep learning* dan *Internet of Things* (IoT). *Python* memiliki *library* dan *framework* yang sesuai dengan kebutuhan dari *deep learning*, serta memiliki ribuan modul yang disebut *Python Package Index* (PyPI) [14]. Sintaks yang dimiliki *python* sederhana dan mudah dipelajari sehingga nantinya bisa mengurangi biaya *maintenance* sebuah program [15].

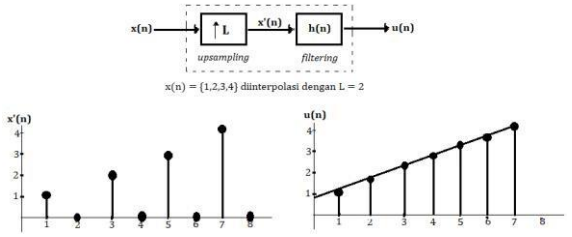
E. *Upsampling*

Pada Tugas Akhir ini dilakukan pengujian dengan menggunakan pendekatan *upsample* pada bagian *backbone* sistem yaitu *darknet*. Proses upsample ini bertujuan agar tidak terdapat data yang hilang serta hasil lebih *smooth* untuk digunakan *secara real-time*. Penggunaan *upsample* ini dapat menggunakan *PyTorch*. *PyTorch* jauh lebih mudah dipelajari daripada *machine learning* library lainnya [16] dan PyTorch mendukung grafik komputasi yang dinamis [17]. Pendekatan *upsample* yang digunakan pada Tugas Akhir ini terdapat 2 cara diantaranya ekspansi dan interpolasi.



GAMBAR 1. Ekspansi

Pada Gambar 1 merupakan contoh dari penerapan upsampling dengan metode ekspansi pada proses pengolahan sinyal digital. Upsample dengan metode ekspansi ini merupakan suatu urutan sinyal asli x[n] yang dipisahkan atau diselipkan dengan nilai 0 (zero filling) sebanyak L-1, dimana L merupakan faktor upsampling. Perangkat yang melakukan proses upsampling disebut upsampler (interpolator).

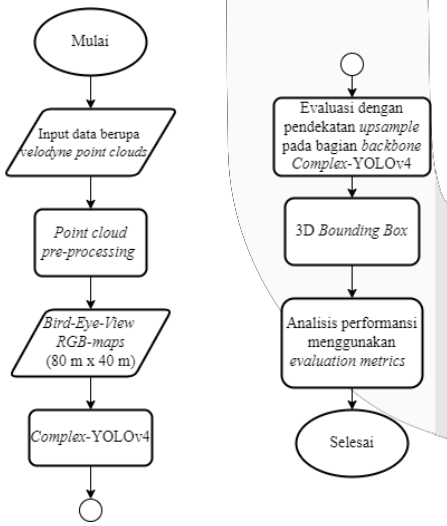


GAMBAR 2. Interpolasi

Pada gambar 2 merupakan contoh dari penerapan *upsample* interpolasi. *Upsample* interpolasi berfungsi untuk menghaluskan diskontinuitas hasil ekspansi dengan filter yang menggantikan nilai 0. Pada interpolasi ini terjadi proses *upsampling* dan penambahan filter *anti-imaging Low Pass Filter* (LPF) dengan *bandwidth* sebesar (π)/L dimana L merupakan faktor sampling. *Anti- imaging filter* ini berfungsi untuk menghindari *multiple image* di output spektrum.

III. METODE

Pada Tugas Akhir ini dilakukan analisis performa dengan pendekatan *upsampling* untuk deteksi objek 3 dimensi dengan data masukan berupa *Birds-Eye-View RGB-Maps*. Skema dari Tugas Akhir ini menggunakan arsitektur *Complex-YOLOv4*. Secara umum Tugas Akhir ini memiliki alur kerja seperti pada Gambar 3.



GAMBAR 3. Diagram Alir Sistem

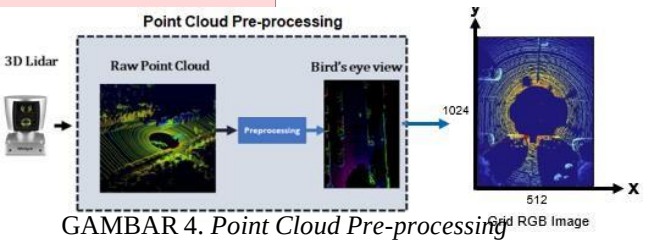
A. *Dataset Point Clouds*

Pada Tugas Akhir ini menggunakan KITTI 3D *object detection dataset* berupa *point cloud*. Terdapat 7158 data sampel velodyne untuk *testing* dan 7481 data sampel untuk *training*. *Point cloud* tersebut merupakan kumpulan titik data dalam ruang yang dihasilkan oleh pemindaian laser LiDAR. *Point cloud* mengandung data RGB yang memberikan warna pada setiap titik, sehingga dapat membentuk model 3D yang menyerupai objek yang dipindai oleh laser pada LiDAR.

B. *Point Cloud Pre-processing*

Point cloud 3D yang dihasilkan oleh pemindaian laser velodyne HDL64 dikonversi menjadi gambar *bird-eye-view RGB*-map dengan area cakupan 80×40 meter. Untuk mentedeksi objek 2D dengan detector 2D seperti YOLO, *point cloud* dikompresi ke data gambar sebelum dimasukan dalam CNN.

Setiap titik dipetakan atau diproyeksikan dalam sel grid citra RGB. Dimana RGB ini akan memuat semua informasi dari *point cloud*. Layer merah berisi informasi *density map* (kepadatan), layer hijau berisi informasi *height map* (ketinggian), dan terakhir layer biru yang berisi informasi *intensity map* (intensitas). Alur kerja *point cloud pre-processing* dapat dilihat pada Gambar 4.



GAMBAR 4. *Point Cloud Pre-processing*

C. *Bird-Eye-View RGB-Map*

Gambar RGB-*maps* ini mempunyai 3 fitur channel yaitu merah (*red*), hijau (*green*) dan biru (*blue*) dengan lambang zr, zg, zb dimana $z_{r,g,b} \in \mathbb{R}^{m \times n}$. *Point cloud* pada tiap fitur *channel* ini dihitung $P \in \mathbb{R}^3$ dalam ruang lingkup area Ω. Posisi lidar 1.73m [18] untuk menutupi area diatas tanah dengan ketinggian sekitar 3m dengan truk sebagai objek tertinggi.

D. *Complex-YOLOv4*

Complex-YOLOv4 menggunakan arsitektur *Convolutional Neural Network* (CNN) serta menggunakan regresi sudut kompleks (*complex angle regression*) dan *Euler-Region-Proposal* (E-RPN) untuk mendeteksi multi-kelas yang berorientasi secara akurat pada objek 3D saat beroperasi secara *real-time*. *Euler Region Proposal Network* (E-RPN) adalah bagian dari jaringan dan menggunakan output dari lapisan konvolusi terakhir untuk memprediksi semua kotak pembatas. Untuk mendapatkan orientasi yang akurat, pada E-RPN ini menambahkan perhitungan sudut kompleks sehingga versi YOLO ini bernama *Complex*-YOLOv4.

Dengan adanya sudut kompleks ini, E-RPN memperkirakan objek berdasarkan pecahan imajiner dan nyata yang langsung tertanam ke dalam jaringan. *Complex angle regression* atau regresi sudut kompleks ini merupakan sudut yang digunakan untuk menghitung sudut orientasi dari tiap objek. Regresi sudut kompleks ini menggunakan parameter

kompleks yaitu t_{im} dan t_{re} dengan sudut nya yaitu $\arctan_2(t_{im},t_{re})$. Untuk lebih jelas, persamaan complex angle regression dapat dilihat pada persamaan 1.

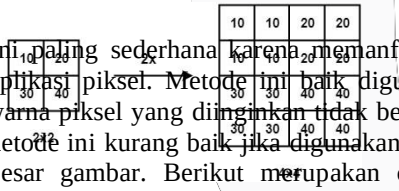
$$\theta_{\emptyset} = \arg(|\theta|e^{i\emptyset}) = \arctan_2(\theta_{Im}, \theta_{Re}). \quad (1)$$

E. Pendekatan *Upsample*

Pada interpolasi ini terdapat beberapa pendekatan *upsampling* yang digunakan pada Tugas Akhir ini diantaranya yaitu mode *bicubic, nearest, dan bilinear*. Penggunaan interpolasi *bicubic* lebih halus, dapat mempertajam dan memperbesar gambar digital. Interpolasi *bicubic* memproses kotak 4x4 (16 piksel). Interpolasi *bicubic* merupakan metode interpolasi yang lebih canggih dan hasilnya lebih halus dibandingkan dengan metode lainnya. Berikut merupakan persamaan untuk permukaan interpolasi *bicubic* dapat dilihat pada persamaan 2.

$$f(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij} x^i y^j \quad (2)$$

Interpolasi *nearest* merupakan interpolasi tercepat karena algoritma dari interpolasi ini hanya memberikan nilai piksel baru dengan nilai piksel yang sama dengan tetangganya. Dengan kata lain, metode ini paling sederhana karena memanfaatkan teknik replikasi piksel. Metode ini baik digunakan apabila warna piksel yang diinginkan tidak berubah, namun metode ini kurang baik jika digunakan untuk memperbesar gambar. Berikut merupakan contoh dari aplikasi interpolasi *nearest* pada Gambar 5.



GAMBAR 5. Interpolasi *Nearest*

Interpolasi bilinear merupakan dasar resampling dalam dunia *computer vision* dan *image processing*. Interpolasi bilinear ini dilakukan dengan menggunakan interpolasi *linier* terlebih dahulu dalam satu arah, kemudian dilanjutkan pada arah lain. Berbanding terbalik dengan *upsample nearest*, *upsample bilinear* ini mengurangi beberapa distorsi visual yang disebabkan oleh pengubahan ukuran gambar ke faktor *zoom*, yang akan membuat beberapa piksel tampak lebih besar daripada yang lain dalam gambar yang diubah ukurannya. Berikut merupakan persamaan dari interpolasi *bilinear* dapat dilihat pada persamaan 3.

$$f(x,y) = \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}).$$

Prediksi *bounding box* ini berdasarkan parameter regresi serta sudut kompleks untuk orientasi kotak pembatasnya. Transisi dari 2 dimensi menjadi 3 dimensi dilakukan dengan menambahkan ketinggian yang telah ditentukan untuk setiap kelas.

G. Parameter Perfromansi

Precision menggambarkan tingkat ketepatan/keakuratan antara data yang diinginkan dengan hasil prediksi yang diberikan oleh model. *Precision* merupakan perbandingan antara *True Positive* (TP) dengan banyaknya data yang diprediksi positif [19]. Berikut merupakan perhitungan *precision* untuk *multi-class* pada persamaan 4.

$$Precision = \frac{\sum_{i=1}^{l} TP_i}{\sum_{i=1}^{l} (TP_i + FP_i)}. \quad (4)$$

Recall adalah perbandingan antara *True Positive* (TP) dengan banyaknya data yang sebenarnya positif [19]. *Recall* menggambarkan keberhasilan model dalam menemukan kembali sebuah informasi. Berikut merupakan perhitungan *recall* untuk *multi-class* pada persamaan 5.

$$Recall = \frac{\sum_{i=1}^{l} TP_i}{\sum_{i=1}^{l} (TP_i + FN_i)}. \quad (5)$$

F1-Score merupakan rata-rata harmonik antara *precision* dan *recall* [19]. Secara representasi, jika *F1-Score* mempunyai skor yang baik maka model mempunyai *precision* dan *recall* yang baik. Jika dataset memiliki jumlah data *false negatif* dan *false positif* yang tidak mendekati, maka sebaiknya gunakan *F1-Score* sebagai acuan [19]. Berikut merupakan persamaan dari *f1-Score* dapat dilihat pada persamaan 6.

$$F1\text{-}Score = 2 * \frac{Precision * Recall}{Precision + Recall}.$$

(6)

Average precision membandingkan kotak *ground truth* dengan kotak pada objek yang terdeteksi dan mengembalikan skor. Semakin skornya tinggi, maka semakin akurat pula model dalam mendeteksi objek. Menggunakan *loop* yang melewati semua *precision/recall*, perbedaan antara *recall* saat ini dan berikutnya dihitung dan kemudian dikalikan dengan presisi saat ini [19]. Berikut

$$\text{1} \qquad \text{2}$$

(3)

F. *3D Bounding Box*

3D bounding box ini merupakan garis pembatas untuk menggambarkan lokasi objek secara 3 dimensi. merupakan persamaan dari *average precision* dapat dilihat pada persamaan 7.

$$\text{(7)}$$

Untuk menghitung mAP, mulailah dengan menghitung AP untuk setiap kelas. Rata-rata AP untuk semua kelas adalah *mean Average Precision* (mAP). Adapun rumus untuk menghitung *mean average precision* adalah seperti pada persamaan 8.

$$AP = \frac{1}{N} \sum_{i=1}^{N} AP_i \quad (8)$$

IV. HASIL DAN PEMBAHASAN

A. *Upsample* pada Parameter *Precision* untuk Setiap Kelas

Nilai tertinggi parameter *precision* diperoleh pada hasil pendekatan dengan upsample *bilinear* untuk kelas mobil sebesar 91,3%. Pada kelas pejalan kaki dengan nilai precision tertinggi didapatkan dengan pendekatan upsample bilinear yaitu 70,2%. Nilai *precision* pada kelas pejalan kaki sangat kecil dibandingkan dengan nilai *precision* kelas lain karena objeknya yang kecil sehingga *upsample* dengan nilai *precision* tertinggi pada kelas pejalan kaki diperoleh dengan menggunakan *upsample bilinear*. Hal ini disebabkan karena *upsample bilinear* ini jauh lebih unggul digunakan untuk memperbesar objek yang kecil. Dan pada kelas pesepeda tertinggi diperoleh oleh *upsample nearest* dan *bicubic* sebesar 80,0%. Hasil *precision* dengan pendekatan *upsample* untuk tiap kelas dapat dilihat pada Tabel 1.

TABEL 1. Hasil *Precision*

	Nearest	***Bicubic***	***Bilinear***
Car	91,1%	90,8%	91,3%
Ped	69,6%	69,9%	70,2%
Cyc	80,0%	80,0%	78,7%

B. *Upsample* pada Parameter *Recall* untuk Setiap Kelas

Dari hasil analisis, nilai parameter *recall* lebih tinggi dibandingkan dengan hasil dari parameter *precision*. Hal ini menunjukan bahwa sistem memiliki nilai *false negative* yang lebih rendah dibandingkan dengan nilai *false positive*. Pada kelas pejalan kaki dan pesepeda didapatkan nilai tertinggi recall dengan menggunakan pendekatan *upsample nearest*, hal ini menunjukan bahwa *upsample* ini baik digunakan pada objek kecil seperti pesepeda dan pejalan kaki. Hasil *recall* dengan pendekatan *upsample* untuk tiap kelas dapat dilihat pada Tabel 2.

TABEL 2. Hasil *Recall*

	Nearest	***Bicubic***	***Bilinear***
Car	97,5%	97,6%	97,4%
Ped	93,0%	93,0%	92,4%
Cyc	93,7%	93,7%	93,4%

C. *Upsample* pada Parameter *F1-Score* untuk Setiap Kelas

Hasil *f1-Score* dengan pendekatan *upsample* untuk tiap kelas dapat dilihat pada Tabel 3. Jika dilihat pada Tabel 3 dapat disimpulkan bahwa untuk kelas mobil dan pesepeda memperoleh nilai *precision* dan *recall* yang baik dibuktikan dengan nilai *f1-score* yang tinggi, namun untuk kelas pejalan kaki tidak sebaik kelas mobil dan pesepeda dan dapat dikategorikan cukup. Jika dilihat secara keseluruhan untuk tiap kelas, sistem ini memiliki rata-rata nilai *f1-score* 86,7%, sehingga bisa dikategorikan baik karena mendekati nilai 1.

TABEL 3. Hasil *F1-Score*

	Nearest	***Bicubic***	***Bilinear***
Car	94,2%	94,1%	94,2%
Ped	79,6%	79,8%	79,8%
Cyc	86,3%	86,3%	85,4%

D. *Upsample* pada Parameter *Average Precision* untuk Setiap Kelas

Nilai tertinggi parameter *average precision* diperoleh pada hasil pendekatan dengan *upsample bicubic* untuk kelas mobil sebesar 96,9%. Pada kelas pejalan kaki dengan nilai *average precision* tertinggi didapatkan oleh pendekatan *upsample bicubic* yaitu 78,7%. Dan selanjutnya, pada kelas pesepeda mendapatkan nilai tertinggi pada *upsample bicubic* sebesar 91,1%. Dari tiap kelas didapatkan nilai tertinggi dengan menggunakan pendekatan *upsample bicubic*. Hasil *average precision* dengan pendekatan *upsample* untuk tiap kelas dapat dilihat pada Tabel 4.

TABEL 4. Hasil *Average Precision*

	Nearest	***Bicubic***	***Bilinear***
Car	96,8%	96,9%	96,8%
Ped	78,4%	78,7%	78,2%
Cyc	90,9%	91,1%	90,7%

E. *Upsample* pada Parameter *Mean Average Precision* untuk Setiap Kelas

Pada *upsample nearest* didapatkan nilai *mean average precision* sebesar 88,7%, untuk *upsample bicubic* dengan nilai 88,9%, dan untuk *upsample bilinear* sebesar 88,5%. Dari ketiga nilai parameter *mean average precision* ini, *upsample bicubic* memiliki nilai tertinggi yaitu lebih tinggi 0.2% dibandingkan dengan *upsample expand* (*upsample* orisinal). Karena penggunaan *upsample bicubic* dominan lebih unggul untuk tiap parameter pada setiap kelasnya, sehingga nilai parameter *mean average precision* untuk *upsample bicubic* lebih tinggi.

TABEL 5. Hasil *Mean Average Precision*

	Mean Average Precision	
Upsample	***Nearest***	88,7%
	Bicubic	88,9%
	Bilinear	88,5%

V. KESIMPULAN

Pendekatan upsample menggunakan 3 pendekatan yaitu nearest, bicubic, dan bilinear berhasil dilakukan pengujian, analisis dan berhasil diimplementasikan dengan tujuan untuk menemukan pendekatan upsample terbaik yang dapat digunakan oleh *Complex*-YOLOv4. Adapun beberapa

kesimpulan yang didapat dari hasil pengujian dan analisis diantaranya:

A. Nilai parameter *precision* tertinggi didapatkan dengan menggunakan pendekatan *upsample bilinear* pada kelas mobil dengan nilai 91,3%.
B. Nilai parameter *recall* tertingi didapatkan dengan menggunakan pendekatan *upsample bicubic* pada kelas mobil dengan nilai 97,6%.
C. Nilai parameter *f1-score* tertingi didapatkan dengan menggunakan pendekatan *upsample nearest* dan *bilinear* pada kelas mobil dengan nilai 94,2%.
D. Nilai parameter *average precision* tertinggi didapatkan dengan menggunakan pendekatan *upsample bicubic* pada kelas mobil dengan nilai 96,9%.
E. Pada nilai parameter *mean average precision* diperoleh nilai 88,9% dan lebih tinggi 0,2% dari hasil *mean average precision* orisinal *Complex*-YOLOv4.

REFERENSI

[1] Bhat, Manoj, Steve Han, and Fatih Porikli."Fast Polar Attentive 3D Object Detection on LiDAR Point Clouds." (2021).

[2] Simon, Martin, et al. "Complexer-yolo: Real-time 3d object detection and tracking on semantic point clouds." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. 2019.

[3] Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum Pointnets for 3D Object Detection From RGB-D Data. CoRR abs/1711.08488 (2017).

[4] Zhou, Y., Tuzel, O.: Voxelnet: End-to-End Learning for Point Cloud Based 3D Object Detection. CoRR abs/1711.06396 (2017).

[5] Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-View 3D Object Detection Network For Autonomous Driving. CoRR abs/1611.07759 (2016).

[6] Duggin, M. (1993). A Review of: "Introduction to Remote Sensing". International Journal of Remote Sensing, 14(3), 600–600.

[7] Wang, Ruisheng; Peethambaran, Jiju; Chen, Dong (2018). LiDAR Point Clouds to 3-D Urban Models: A Review. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 11(2), 606–627.

[8] LIDR is the Latest Game-Changing Advancement for Autonomous Vehicles (https://innovationatwork.ieee.org/lidr-is-the-latest-game-changing-advancement-for-autonomous-vehicles/, diakses pada 18 Desember 2022)

[9] S. Khan, H. Rahmani, S. A. A. Shah, and M. Bennamoun, A Guide to Convolutional Neural Networks for Computer Vision, Synth. Lect. Comput. Vis., vol. 8, no. 1, pp. 1207, 2018, doi: 10.2200/s00822ed1v01y201712cov015. [10] V. H. Phung and E. J. Rhee, "A deep learning approach for classification of cloud image patches on small datasets," J. Inf. Commun. Converg. Eng., vol. 16, no. 3, pp. 173–178, 2018, doi: 10.6109/jicce.2018.16.3.173.

[11] Yingge, Huo; Ali, Imran; Lee, Kang-Yoon (2020). [IEEE 2020 IEEE International Conference on Big Data and Smart Computing (BigComp) - Busan, Korea (South) (2020.2.19-2020.2.22)] 2020 IEEE International Conference on Big Data and Smart Computing (BigComp) - Deep Neural Networks on Chip - A Survey. 589–592.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 2016-Decem, pp. 779–788, 2016, doi: 10.1109/CVPR.2016.91.

[13] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 2019-June, pp. 658–666, 2019, doi: 10.1109/CVPR.2019.00075.

[14] C. S. Lent, "Learning to Program with MATLAB Building GUI Tools," John Wiley Sons Inc., p. 310, 2013.

[15] What is Python? Executive Summary — Python.org. https://www.python.org/doc/essays/blurb/ (accessed Feb. 18, 2022).

[16] Ketkar, Nikhil (2017). "Introduction to PyTorch". Deep Learning with Python. Apress, Berkeley, CA. pp. 195–208. doi:10.1007/978-1-4842-2766-4_12. ISBN 9781484227657.

[17] K. Tran, What is PyTorch?, 2020. https://towardsdatascience.com/what-ispytorch-a84e4559f0e3 (accessed Feb. 18, 2022).

[18] Geiger, A.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). CVPR '12, Washington, DC, USA, IEEE Computer Society (2012) 3354–3361.

[19] Dalianis H. (2018) Evaluation Metrics and Evaluation. In: Clinical Text Mining. Springer, Cham. https://doi.org/10.1007/978-3-319-78503-5_6.