Eksploitasi Fitur Untuk Peningkatan Kinerja Deteksi Objek Berbasis Pada Pesawat Tanpa Awak

1st Nurul Jannah
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
nuruljn@student.telkomuniversity.ac.id

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

3rd Thomhert Suprapto Siadari
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
thomhert@telkomuniversity.ac.id

Abstrak

Teknologi kecerdasan buatan atau *Artificial Intelligence* (AI) berkembang pesat seiring dengan perkembangan zaman. Salah satu bagian pada AI yaitu *computer vision* khususnya *Object Detection* menjadi teknologi yang terus dikembangkan. *Object Detection* dapat diimplementasikan pada berbagai sistem, salah satunya *Object Detection for Unmanned Aerial Vehicles* (UAV). Peningkatan performansi kinerja *object detection* pada UAV merupakan tantangan bagi para peneliti yang disebabkan oleh beberapa masalah seperti, ketidakseimbangan data, perubahan skala, prediksi yang akurat, keterbatasan memori dan komputasi untuk aplikasi *real-time*. Pada Tugas Akhir ini dilakukan eksploitasi fitur terhadap algoritma YOLOv5 dalam meningkatkan performansi object detection berbasis pada pesawat tanpa awak. Algoritma ini terdiri dari bagian *backbone*, *neck*, dan *head*. YOLOv5 mengadaptasi CSP *network* sebagai *backbone* dan PANet sebagai *neck*. Beberapa analisis yang dilakukan pada Tugas Akhir ini adalah pengaruh eksploitasi pada *backbone* yaitu *freeze backbone*, eksploitasi pada layer *concat* dan *freeze backbone*, eksploitasi dengan penambahan layer pada bagian *neck* dan *freeze backbone*, dan penggabungan ketiga eksploitasi tersebut. Model dengan performansi terbaik didapatkan pada model penggabungan tiga eksploitasi dengan nilai mAP@.5 sebesar 22,4%. Berdasarkan hasil mAP tersebut, eksploitasi fitur dengan dilakukan *freeze backbone* dan penambahan layer pada bagian *neck* juga modifikasi pada layer *concat* dapat mempengaruhi performansi dan kinerja *object detection* berbasis pada pesawat tanpa awak.

Kata kunci: Eksploitasi, *Object Detection*, UAV, YOLOv5

Abstract

Artificial intelligence (AI) technology is developing rapidly along with the times. One part of AI, namely computer vision, especially Object Detection, is a technology that continues to be developed. Object Detection can be implemented on various systems, one of which is Object Detection for Unmanned Aerial Vehicles (UAV). Improving the performance of UAV-Based Object Detection is a challenge for researchers caused by several problems such as data imbalance, scale variants, accurate predictions, memory and computation limitations for real-time applications. In this final project, feature exploitation of the YOLOv5 algorithm is carried out in improving the performance of UAV-based object detection. This algorithm consists of the backbone, neck, and head. YOLOv5 used the CSP network as the backbone and PANet as the neck. Some of the analyzed carried out in this Final Project are the effect of exploitation on the backbone, namely freeze backbone, exploitation of the concat and freeze backbone layers, exploitation by added layers to the neck and freeze backbone, and merged the three exploits. The model with the best performance is the combination of three exploitation models with mAP@.5 value of 22.4%. Based on the results of the mAP, the exploitation of features by set freeze backbone and added a layer on the neck as well as modified to the concat layer can affect the performance and performance of object detection based on drones.

Keywords : Exploitation, Object Detection, UAV, YOLOv5

I. PENDAHULUAN

Teknologi kecerdasan buatan atau *Artificial Intelligence* (AI) berkembang pesat seiring dengan perkembangan zaman. Salah satu bagian pada AI yaitu *computer vision* khususnya Deteksi Objek menjadi teknologi yang terus dikembangkan dengan berbagai fitur dalam mendukung kebutuhan industri dan mempermudah manusia dalam menjalankan pekerjaan tertentu. Deteksi objek dapat diimplementasikan pada berbagai sistem, seperti *smart surveillance system* [6], *Autonomous Driving* [7], *Object Detection for Unmanned Aerial Vehicles* (UAV) [8]. Penggunaan deteksi objek pada UAV merupakan teknologi yang berkembang pesat saat ini. Deteksi objek pada UAV dapat diimplementasikan pada berbagai bidang seperti agrikultur, fotografi udara, pengiriman barang, keamanan dan pengawasan, dan pencarian dan penyelamatan. Namun, deteksi objek pada UAV merupakan tantangan bagi para peneliti dalam menyelesaikan beberapa masalah seperti, ketidakseimbangan data, prediksi yang akurat, keterbatasan memori dan komputasi untuk aplikasi *real-time*. Oleh karena itu, masalah tersebut menjadi perhatian bagi para peneliti untuk mengoptimalkan deteksi objek pada UAV.

Beberapa penelitian mampu menyelesaikan masalah diatas dengan menggunakan metode yang berbeda-beda. Pada penelitian Zhang yang menggunakan model SlimYOLOV3 dan VisDrone2018-DET sebagai dataset menunjukkan bahwa model tersebut mampu mencapai akurasi deteksi yang sebanding dengan YOLOv3 dengan FLOP yang jauh lebih sedikit dan berjalan lebih

cepat dibandingkan dengan YOLOv3 orisinal, dan model ini dapat digunakan untuk aplikasi UAV real-time [9]. Namun, penelitian tersebut tidak mengatasi ketidakseimbangan dataset sehingga skor mAP (*Mean Average Precision*) kelas objek yang dominan lebih tinggi dibandingkan dengan kelas lainnya [9]. Hal tersebut terjadi dikarenakan pada penelitian tersebut menggunakan *one step detector* yaitu SlimYOLOV3 dan ketidakseimbangan dataset terjadi karena *one step detector* secara penuh mengambil sampel wilayah dari seluruh gambar, sedangkan *two step detector* menghindari masalah ini dengan menggunakan mekanisme (RPN) [1]. Jannesari mengemukakan arsitektur *Deep Feature Pyramid Network* (DFPN) dan *loss function* yang sudah dimodifikasi untuk mengatasi ketidakseimbangan dataset VisDrone-2018 serta mencapai deteksi objek secara *real-time* [1]. Berdasarkan hasil evaluasi mAP pada kedua penelitian tersebut, menunjukkan bahwa DFPN dengan jaringan dasar mobilenet memiliki skor yang lebih tinggi dengan nilai 29,2 dibandingkan dengan SlimYOLOV3 yaitu 23,9. Namun, penelitian Zhu et al dengan menggunakan dataset yang sama dan TPH-YOLOv5 sebagai objek detektor menghasilkan map yang lebih tinggi yaitu dengan nilai 39,18 [10]. Berdasarkan nilai mAP tersebut, model TPH-YOLOv5 mampu mengatasi masalah deteksi pada objek skala kecil, objek yang rapat, dan objek blur [10].

Sementara itu, saat ini terdapat beberapa algoritma deteksi objek yang digunakan para peneliti diantaranya Faster R-CNN, YOLO, SSD, dll. Berdasarkan hasil evaluasi pada penelitian Zhang et al, SlimYOLOv3- SPP3-90 menghasilkan mAP dengan nilai 23,9 [9]. Sedangkan pada penelitian Zhu et al, deteksi objek pada UAV menggunakan metode YOLOv5 dengan mengintegrasikan *Transformer Prediction Heads* (TPH) dan menambah *prediction head* pada arsitektur YOLOv5 dapat menunjukkan hasil mAP yang lebih tinggi [10]. Oleh karena itu, berdasarkan kemiripan lingkungan sistem dan analisis pada evaluasi parameter tersebut, pada Tugas Akhir ini penulis menggunakan model YOLOv5 dengan modifikasi pada arsitektur dengan penambahan pada detection head serta modifikasi pada bagian neck dalam meningkatkan akurasi deteksi pada dataset VisDrone.

II. KAJIAN TEORI

A. *Object Detection*

Object Detection adalah suatu proses menemukan dan mengklasifikasikan objek dalam suatu gambar dan menampilkan *bounding box* pada objek tersebut untuk menunjukkan tingkat konfidensi eksistensi [11]. Pada proses pendeteksian suatu objek, terdapat dua metode dalam menyelesaian tugas tersebut diantaranya *one-stage* dan *two-stage detectors* [12]. *One-stage-detector* menggunakan jaringan tunggal *feed-forward fully convolutional* yang secara langsung menyediakan *bounding box* dan klasifikasi objek, sedangkan *two-stage-detector* terdiri dari proses *region proposal* dan tahap klasifikasi [12]. Beberapa algoritma *Object Detection* yang menggunakan *one-stage-detector* adalah YOLO dan SSD [13].

1. *UAV-Based Object Detection*

Pada penelitian tugas akhir ini, *Object Detection* diujikan pada *aerial images* yang diambil menggunakan pesawat tanpa awak (UAV). *Object detection* pada UAV memiliki beberapa tantangan tersendiri seperti oklusi, deteksi pada objek berukuran kecil, variasi skala, dsb. Jenis object detection ini dapat diimplementasikan ke dalam UAV dan dapat dimanfaatkan ke berbagai hal seperti keamanan udara, pengiriman barang, fotografi udara, dan pencarian.

B. *Convolutional Neural Network*

Convolutional Neural Network (CNN) adalah salah satu jenis algoritma deep learning yang digunakan untuk memproses suatu data yang memiliki pola seperti citra [14]. Arsitektur pada CNN terdiri dari convolution layers, pooling layers, dan fully connected layers. CNN terinspirasi oleh proses biologis dimana pola konektivitas antar neuron terinspirasi oleh organisasi korteks visual hewan. Sebuah neuron seperti fungsi, dibutuhkan beberapa input yang dikalikan dengan weights tertentu dan mengembalikan output [15].

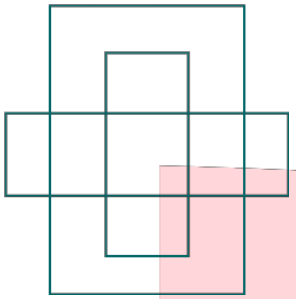
C. YOLO

You Only Look Once (YOLO) adalah algoritma *object detection* yang menggunakan *convolutional neural network* dan darknet sebagai jaringan dasar untuk training dan inferensi [16]. YOLO memiliki 24 lapisan konvolusi dengan dua *fully connected layers*. Detektor ini juga menggunakan 1 × 1 *reduction layers* diikuti dengan 3 × 3 *convolutional layers*. YOLO melatih *convolutional layers* pada tugas klasifikasi ImageNet dengan setengah resolusi (224 × 224) dan kemudian menggandakan resolusi tersebut untuk *object detection*. Terdapat beberapa versi dari perkembangan YOLO berdasarkan dari penelitian-penelitian lainnya, diantaranya YOLOv2 [16], YOLOv3 [18], YOLOv4 [19], dan YOLOv5 [2].

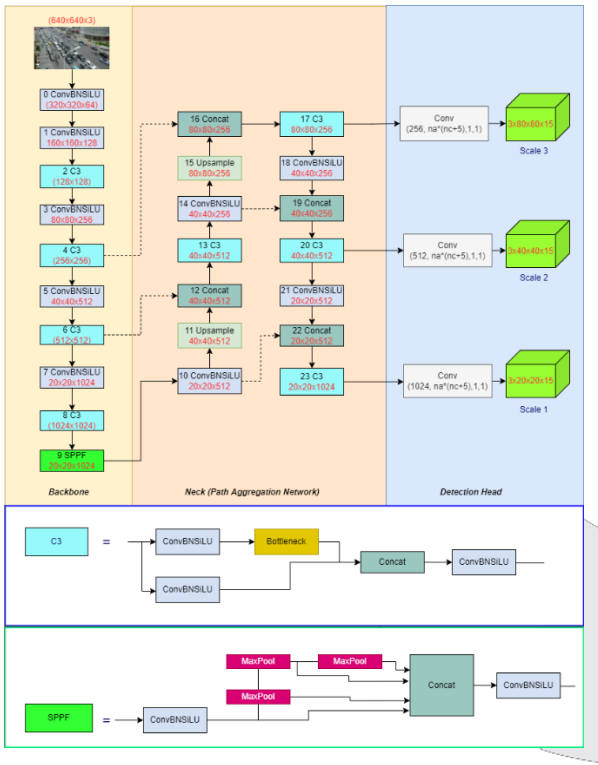
1. YOLOv5

YOLOv5 merupakan versi kelima algoritma Object Detection dari YOLO (You Only Look Once). Arsitektur YOLOv5 mengadaptasi dari model YOLOv4 [19], detail arsitektur dapat dilihat pada gambar 2.1. Berdasarkan file .yaml model YOLOv5 memiliki struktur model sebagai berikut *Backbone*, *Neck*, dan *Head*. Pada YOLOv5 terdapat C3 atau CSPBottleneck dengan tiga konvolusi sebagai *feature extractor* atau *backbone* yang berfungsi untuk memelihara fitur melalui propagasi, mendorong jaringan untuk menggunakan kembali fitur, dan membantu mempertahankan butiran halus fitur untuk meneruskan ke lapisan yang lebih dalam dengan lebih efisien. Sebelum fitur diteruskan ke bagian *neck*, fitur menjalankan proses pada layer SPPF. SPPF (*Spatial Pyramid Pooling Fast*) berfungsi meningkatkan bidang reseptif dan memisahkan fitur yang paling penting, serta terdapat proses *max pooling*. Selanjutnya fitur diproses oleh PANet sebagai *neck* yang menghasilkan *feature pyramids*. *Feature pyramids* membantu model dalam menghasilkan penskalaan objek dengan baik, juga

membantu untuk identifikasi objek yang sama dengan ukuran dan skala yang berbeda. *Detection Head* pada YOLOv5 digunakan untuk deteksi bagian terakhir. Bagian *head* ini menerapkan *anchor boxes* pada fitur dan menghasilkan output vektor terakhir dengan probabilitas kelas, skor objektivitas, dan *bounding boxes* [20].



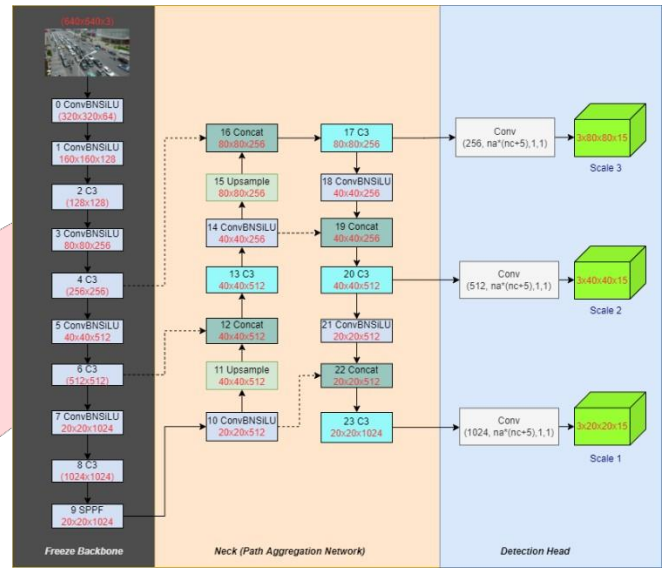
GAMBAR 2.2 Anchor Boxes.



GAMBAR 2.1. Arsitektur YOLOv5

D. Freeze Backbone

Freeze Backbone adalah cara yang efektif untuk melatih kembali model dengan cepat pada data baru tanpa harus melatih ulang seluruh *network*. Sebagai gantinya, sebagian dari bobot awal dibekukan di tempatnya, dan bobot lainnya digunakan untuk menghitung *loss* dan diperbarui oleh *optimizer*. Hal ini membutuhkan daya yang lebih sedikit daripada *training* normal dan memungkinkan waktu *training* yang lebih cepat, meskipun hal ini juga dapat mengakibatkan pengurangan akurasi pada hasil *training* [3]. *Freeze backbone* pada YOLOv5 akan dibekukan dengan mengatur gradiennya ke nol sebelum training dimulai. Model *backbone* yang dibekukan dapat dilihat pada gambar 2.3, lapisan 0-9 akan dibekukan sebelum *training* dimulai. Performansi berdasarkan nilai mAP model *training* dengan *freeze backbone* mimiliki akurasi yang tidak jauh berbeda dengan *trained* model *default*.



GAMBAR 2.3. Flowchart *Freeze Backbone* YOLOv5.

E. Evaluasi Model

1. *Mean Average Precision*

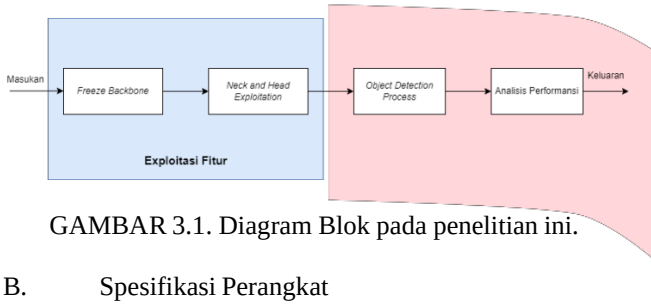
Mean Average Precision (mAP) adalah metrik evaluasi yang menyatakan nilai rata-rata AP (*Average Precision*) dari setiap kelas pada suatu dataset [22]. Metrik ini membandingkan *ground-truth* dengan *bounding box* yang terdeteksi dan mengembalikan sebuah nilai. Berdasarkan ambang pada IoU, nilai mAP dapat dikalkulasikan berdasarkan nilai tertentu seperti evaluasi metrik dataset Pascal VOC dan COCO. Dataset pascal VOC menggunakan ambang IoU lebih dari 0.5 yang dianggap benar pada hasil prediksi [23]. Sedangkan COCO menggunakan IoU 0.5 sampai dengan 0.95 dengan *step* 0.05, dan juga terdapat kalkulasi mAP pada IoU 50 dan 75 [24]. MAP@0.5 adalah nilai mAP dengan ambang IoU 0.5 yang didapatkan dari nilai rata-rata AP setiap kategori atau kelas dari semua gambar pada ambang IoU tersebut. Perhatikan formula 2.1, variabel C pada formula tersebut merupakan jumlah dari seluruh kelas yang diujikan dan variabel i adalah iterasi. Konsep dari mAP dapat dikalkulasikan sebagai berikut:

$$mA\diamond\diamond = \frac{1}{C}\sum_{i=1}^{C} AP_i \quad (2.1)$$

III. METODE

A. Diagram Blok Penelitian

Diagram blok penelitian tugas akhir ini menjadi panduan bagi penulis dalam merancang sistem deteksi objek berbasis UAV dalam meningkatkan performansi pada model YOLOv5. Pada penelitian ini dilakukan percobaan eksploitasi pada arisitektur YOLOv5, diantaranya yaitu *freeze backbone*, mengubah proses pada bagian *neck* dan penambahan layer serta *detection head*. Hasil dari beberapa percobaan dibandingkan dalam peningkatan performansi dengan mengukur tingkat nilai mAP masing-masing model. Diagram blok dapat dilihat pada gambar 3.1.



GAMBAR 3.1. Diagram Blok pada penelitian ini.

B. Spesifikasi Perangkat

Perangkat keras yang digunakan pada penelitian ini yaitu Google Colab. Google Colab (colab) adalah produk dari Google Research yang memiliki spesifikasi yang tinggi sehingga mendukung untuk program *machine learning*. Colab adalah *executable document* yang bisa digunakan untuk menulis, menyimpan, serta membagikan program yang telah ditulis melalui Google Drive. Bahasa pemrograman yang digunakan pada Colab adalah python. Spesifikasi colab dapat dilihat pada tabel 3.1.

TABEL 3.1. Spesifikasi Google Colab.

GPU	Nvidia T4
GPU *Memory*	16 GB
RAM	32 GB
Runtime	24 hours
Disk Space	358 GB

C. Dataset

Dataset adalah kumpulan data yang disediakan untuk diujikan dan dianalisa pada suatu sistem. Pada *object detection*, dataset yang diperlukan adalah data yang berisikan gambar beserta anotasi. Dataset dapat ditemukan melalui internet dan terdapat berbagai macam jenis dataset. Jenis dataset yang digunakan pada penelitian ini yaitu dataset yang terdiri dari *aerial images*. Dataset *aerial images* yang digunakan pada penelitian ini adalah VisDrone-DET2019 [25].

Dataset tersebut dibuat oleh tim AISKYEYE di Laboratorium *Machine Learning* and *Data Mining*, Tianjin University, China [5]. Dataset VisDrone-DET2019 menggunakan data yang sama dengan VisDrone-DET2018, yaitu 8.599 gambar yang ditangkap oleh *drone* di tempat dan ketinggian yang berbeda. Selain itu, lebih dari 540 ribu target *bounding box* yang dianotasi dengan sepuluh kategori [25].

Dataset VisDrone terbagi menjadi tiga pengelompokan berdasarkan *train*, *valid*, dan *test set*. Data *train* digunakan saat training pada algoritma *Object Detection*. Algoritma yang telah dilatih diujikan dengan data *valid* untuk mendapatkan performansi sementara dan jika tidak sesuai dengan target maka *training* model dapat diulang sehingga mendapatkan performansi yang diinginkan. Data test digunakan untuk mengetahui akurasi akhir pada model yang sudah dilatih. Data citra *test* tidak sama dengan data *train*, sehingga model dapat menguji data tersebut tanpa mengandalkan data *train*. Pembagian data berdasarkan *train*, *valid*, dan *test* dapat dilihat pada tabel 3.2.

TABEL 3.2 Pembagian Dataset VisDrone Berdasarkan *Train*, *Valid*, dan *Test*.

Pembagian	Jumlah Citra
Train	6471
Valid	548
Test	1610

E. *Train* Model

Setelah menentukan dataset yang digunakan, selanjutkan dilakukan *train* model. Pada proses ini, model menggunakan data *train* untuk dapat mendeteksi objek-objek pada dataset VisDrone-DET2019. Pada penelitian ini, terdapat beberapa percobaan dalam eksploitasi model yaitu train model YOLOv5 orisinal, eksploitasi I, eksploitasi II, eksploitasi III, dan eksploitasi IV. Beberapa eksploitasi dan analisis yang dilakukan pada penelitian ini dapat diuraikan sebagai berikut:

1. YOLOv5 Orisinal: *Train* model YOLOv5 dengan tanpa eksploitasi pada bagian arsitektur sehingga dijadikan acuan untuk perbandingan performansi dengan model eksploitasi lainnya.
2. Eksploitasi I: *Train* model YOLOv5 dengan *Freeze Backbone* untuk mengetahui pengaruh performansi antara YOLOv5 orisinal dengan model tersebut.
3. Eksploitasi II: *Train* model dengan modifikasi pada proses layer *concat* untuk mengetahui pengaruh peningkatan performansi pada eksploitasi tersebut.
4. Eksploitasi III: *Train* model dengan penambahan layer pada bagian *neck* untuk mengetahui pengaruh pertambahan layer serta *detection head* pada performansi model.
5. Eksploitasi IV: *Train* model dengan dilakukan kombinasi eksploitasi II dan III untuk mengetahui seberapa besar peningkatan terhadap kedua eksploitasi tersebut.

Semua percobaan pada penelitian ini menggunakan model YOLOv5 seri l atau *large* dan tidak menggunakan *pretrained weight* model atau *training from scratch*. Epoch yang diatur pada seluruh percobaan adalah sebanyak 50. Serta ukuran *input image* yang diproses yaitu 640x640. Semua model eksploitasi dilatih dengan *freeze backbone* untuk mengurangi beban komputasi saat training model.

1. YOLOv5 Orisinal

Struktur model YOLOv5 Orisinal terdiri dari tiga *prediction head* dan 24 *stage*. Pada bagian *detection head*, terdapat tiga ukuran *feature map* yang berbeda dengan nilai berturut-turut adalah 80x80, 40x40, dan 20x20. Hasil performansi model ini dibandingkan dengan model eksploitasi lainnya.

2. Eksploitasi I

Percobaan selanjutnya adalah *train* model dengan eksploitasi struktur model pada backbone. Pada percobaan ini, dilakukan *freeze* pada *backbone* dengan mengatur nilai 0 pada gradien model *backbone* YOLOv5l orisinal. Pada struktur model ini terdapat tiga *prediction head* dan 24 *stage* sama seperti struktur model YOLOv5 Orisinal.

3. Eksploitasi II

Percobaan selanjutnya adalah train model dengan eksploitasi struktur model pada layer *concat* yang menggabungkan proses *upsample* pada bagian *neck* dengan proses C3 pada bagian *backbone*. Perbedaan eksploitasi II dengan I terdapat pada layer *concat* khususnya *stage* 12 dan 16, pada YOLOv5 orisinal layer tersebut digabungkan dengan layer C3, namun pada eksploitasi ini diubah dengan layer conv. Pada struktur model ini memiliki tiga *prediction head* dan 24 *stage* yang sama dengan struktur model YOLOv5 Orisinal.

4. Eksploitasi III

Percobaan ketiga adalah *train* model dengan eksploitasi struktur model bagian *neck* dan *head*. Pada bagian *head* model ini ditambahkan satu *detection head* serta penambahan layer sehingga terdapat empat *detection head* dengan ukuran *feature map* yang lebih besar yaitu 320x320 (scale 4). Pada percobaan ini terdapat 35 stage.

5. Eksploitasi IV

Percobaan keempat adalah *train* model dengan eksploitasi struktur model pada layer *concat* serta penambahan layer pada bagian *neck* dan *head*. Pada bagian *head* model ini terdapat empat *detection head* dengan ukuran *feature map* yang seperti model eksploitasi III. Pada percobaan ini juga terdapat 35 *stage* serta adanya pengubahan proses pada layer *concat* yang sama seperti model eksploitasi II.

IV. HASIL DAN PEMBAHASAN

A. Analisis Nilai mAP

1. Analisis pengaruh Performansi terhadap Model dengan *Freeze Backbone*

Pada percobaan pertama, dilakukan train pada model YOLOv5 orisinal dan eksploitasi I. Model eksploitasi I adalah model dengan struktur yang sama seperti YOLOv5 Orisinal dan dilakukan *freeze* pada *backbone*. Performansi model eksploitasi I tersebut menunjukkan hasil yang jauh berbeda dengan YOLOv5 orisinal. Persentase nilai mAP@.5 model YOLOv5 orisinal dan eksploitasi I berturut-turut adalah 26,4% dan 8,1%. Pada hal ini model YOLOv5 orisinal jauh mengungguli performansi model eksploitasi I dengan selisih 18,3%. Hal tersebut dikarenakan gradien pada proses *backbone* model eksploitasi I diatur ke 0 sehingga nilai *weight* pada *backbone* tidak diperbarui selama *training*, sedangkan model YOLOv5 orisinal menjalankan algoritma dan mengambil nilai *weight* dari seluruh proses pada *network* termasuk *backbone*.

2. Analisis Pengaruh Performansi Terhadap Pengubahan pada Layer *Concat*

Pada analisis ini dibandingkan performansi model eksploitasi I dan eksploitasi II. Model eksploitasi II adalah model dengan struktur yang sama seperti YOLOv5 Orisinal namun terdapat pengubahan pada layer *concat* khususnya *stage* 12 dan 16. Pengubahan yang dilakukan yaitu mengganti proses layer *concat* C3 dengan conv pada *backbone*. Persentase nilai mAP@.5 model eksploitasi I dan eksploitasi II berturut-turut adalah 8,1% dan 10,3%. Performansi model eksploitasi II mengungguli model eksploitasi I. Hal tersebut menunjukkan bahwa pengubahan proses pada layer *concat* mempengaruhi nilai mAP pada model. Pada struktur model YOLOv5 orisinal, proses *concat* menggabungkan *featur map* dari layer sebelumnya dan proses C3 (CSPbottleneck dengan tiga konvolosi), sedangkan eksploitasi II ini dilakukan pengubahan proses yang digabungkan dengan conv (standar konvolusi). Hal yang menyebabkan peningkatan performansi pada pengubahan layer *concat* adalah *feature map* yang dihasilkan berbeda dengan proses C3, karena proses C3 terdapat proses bottleneck dan tiga konvolusi yang menyebabkan *feature map* dieksploitasi lebih banyak sehingga menyebabkan informasi penting pada fitur tersebut dapat berkurang. Sehingga pada percobaan ini, dengan diubahnya proses pada layer *concat* khususnya stage 12 dan 16 persentase peningkatan performansi didapatkan sebesar 2,2%.

3. Analisis Pengaruh Performansi Penambahan Layer pada *Detection Head*

Pada analisis ini dibandingkan performansi model eksploitasi I dan eksploitasi III. Model eksploitasi III adalah model dengan struktur yang berbeda dengan YOLOv5 Orisinal, terdapat penambahan layer serta skala *output detection head* yang lebih besar. Perbedaan yang signifikan pada model eksploitasi I dan III adalah jumlah *detection head*. Nilai mAP pada model eksploitasi I dan III berturut-turut adalah 8,1% dan 20,9%. Pada percobaan ini model dengan layer yang lebih banyak dan ditambahkan *output detection head* dengan skala yang berbeda dapat berpengaruh pada performansi model, sehingga model tersebut mengungguli model eksploitasi I. Hal tersebut disebabkan oleh dengan adanya penambahan layer pada *network*, *output detection head* juga bertambah dengan skala yang lebih besar pada penelitian ini, yaitu dengan ukuran *feature map* 320x320. Dengan ukuran *feature map* yang lebih besar, informasi terhadap objek kecil lebih memudahkan untuk diproses dibandingkan dengan ukuran *feature map* yang lebih kecil. Sehingga, pada percobaan ini dengan pertambahan layer dan *output detection head*

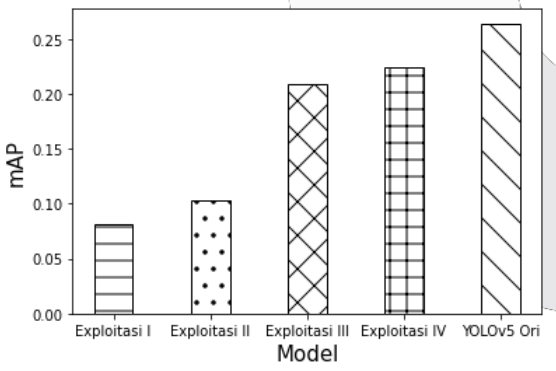
dengan skala yang lebih besar maka dapat meningkatkan performansi model.

4. Analisis Pengaruh Performansi Terhadap Pengubahan Layer *Concat* dan Penambahan Layer

Pada analisis ini dibandingkan performansi model eksploitasi I dengan eksploitasi IV. eksploitasi IV menggunakan struktur model yang dikombinasikan antara model dengan pengubahan layer pada *concat* dan penambahan layer serta *detection head*. Performansi pada model eksploitasi IV mengungguli eksploitasi I dengan persentase nilai mAP berturut-turut adalah 22,4% dan 8,1%. Pada model eksploitasi IV menunjukkan bahwa dengan adanya pengubahan terhadap proses concat yang dikombinasikan dengan penambahan layer serta output detection head maka terjadi peningkatan performansi. Hal tersebut dapat disebabkan oleh feature yang dihasilkan lebih baik dikarenakan adanya pengubahan fitur yang diproses pada bagian neck sehingga model dapat memproses fitur dengan efektif dibandingkan model eksploitasi I. Dengan adanya penambahan detection head dengan ukuran feature map yang lebih besar dapat memudahkan model dalam mendeteksi objek kecil dikarenakan ukuran yang lebih besar lebih sensitif dalam memproses lokalisasi dan menentukan objek kecil.

5. Analisis Pengaruh Eksploitasi Terhadap Model YOLOv5 Orisinal

Pada analisis ini dibandingkan performansi semua model yang telah dilatih dan diujikan. Pada hal ini, semua model eksploitasi diatur *freeze backbone* sehingga untuk performansi mAP lebih rendah dibandingkan model YOLOv5 Orisinal yang dilatih pada seluruh *network*. Perbandingan nilai mAP@.5 pada model dapat dilihat pada gambar 4.1.



GAMBAR 4.1. Perbandingan nilai mAP@.5 pada Semua Model yang Telah Dilatih.

Berdasarkan gambar 4.1, YOLOv5 orisinal memiliki nilai mAP paling tinggi dibandingkan model eksploitasi lainnya. Hal tersebut disebabkan oleh pengaruh dari *freeze backbone* yang dapat mengurangi performansi model, karena pada hal ini model YOLOv5 orisinal dilatih dengan seluruh *network* termasuk *backbone*. Sedangkan pada model eksploitasi seluruhnya diatur *freeze backbone* yang nilai *weight*-nya tidak diperbarui selama *training* dan menggunakan nilai *weight* dengan inisiasi awal. Namun, pada penelitian ini berdasarkan pada model eksploitasi I s.d IV menunjukkan bahwa dengan adanya peningkatan performansi pada model yang diatur *freeze backbone*. Dengan adanya pengubahan proses pada layer *concat* dan juga penambahan pada layer serta *output detection head* dengan skala yang lebih besar maka mAP pada model dengan *freeze backbone* dapat meningkat. Jika diperhatikan pada gambar 4.1, performansi model eksploitasi I dan YOLOv5 memiliki selisih nilai mAP sebesar 18,3%, yang menunjukkan bahwa model yang dilatih dengan seluruh *network* berpotensi memiliki performansi yang jauh lebih tinggi atau dapat dikalkulasikan 4x lebih besar akan bertambah nilai performansinya pada kasus ini. Sehingga, model eksploitasi IV yang memiliki nilai mAP 22,4% dengan struktur model yang diatur dengan *freeze backbone* dan kombinasi eksploitasi II dan III dapat berpotensi memiliki nilai mAP yang lebih tinggi apabila dilatih pada keseluruhan *network* yaitu sebesar 89,6%.

B. Analisis Nilai mAP Setiap kategori

Pada analisis ini dibandingkan nilai mAP setiap kategori model YOLOv5 Orisinal dan empat model eksploitasi pada batas ambang IoU 0.5. Kategori *car* mengungguli nilai mAP diantara yang lainnya pada semua model, hal ini disebabkan oleh adanya ketidakseimbangan dataset pada VisDrone-DET2019. Berdasarkan jumlah label setiap kategori, kategori car memiliki label terbanyak diantara kategori lainnya. Hal tersebut menyebabkan akurasi dominan terhadap kategori *car*.

TABEL 4.2. Nilai mAP@.5 untuk Semua Model pada 10 Kategori.

Kategori	mAP@.5				
	YOLOv5 Ori	Eksploitasi I	Eksploitasi II	Eksploitasi III	Eksploitasi IV
Pedestrian	0.26	0.05	0.08	0.19	0.24
People	0.16	0.04	0.05	0.11	0.14
Bicycle	0.07	0.03	0.02	0.06	0.08
Car	0.69	0.30	0.36	0.62	0.64
Van	0.29	0.05	0.07	0.25	0.25
Truck	0.24	0.05	0.09	0.17	0.18
Tricycle	0.10	0.03	0.03	0.06	0.08
Awning-tricycle	0.11	0.02	0.03	0.09	0.09
Bus	0.50	0.22	0.24	0.41	0.47
Motor	0.23	0.04	0.07	0.14	0.18

Berdasarkan perbandingan nilai AP setiap model untuk setiap kategori pada tabel 4.2, model YOLOv5 Orisinal memiliki nilai AP tertinggi pada kategori semua kategori, hal ini disebabkan oleh performansi model

dengan dilatih pada seluruh *network* dapat lebih tinggi dibandingkan model eksploitasi yang dilatih hanya sebagian *network*. Perbandingan performansi model eksploitasi II dengan eksploitasi I yaitu pengaruh performansi pada pengubahan layer *concat* atau model eksploitasi II meningkat pada semua kategori kecuali *bicycle*. Pada model eksploitasi III juga mengungguli model eksploitasi I dan II, yang mana pada model ini dilakukan penambahan layer dan *detection head* dengan skala *feature map* yang lebih besar. Dengan penggabungan kedua eksploitasi tersebut atau eksploitasi IV, nilai mAP seluruh kategori mengungguli model eksploitasi I s.d III. Hal tersebut menunjukkan dengan adanya eksploitasi pada bagian layer *concat* dan penambahan layer serta *detection head* semakin meningkat akurasi daripada tanpa eksploitasi. Namun, semua model pada tugas akhir ini belum mampu menyelesaikan masalah pada ketidakseimbangan dataset, sehingga untuk nilai mAP pada setiap kategori terdapat selisih yang signifikan yang menyebabkan kategori yang tidak dominan sulit mendeteksi objek dengan optimal.

V. KESIMPULAN

Eksploitasi fitur pada algoritma model YOLOv5 telah berhasil dilakukan pengujian dan analisis. Berdasarkan performansi model terhadap model eksploitasi dengan *freeze backbone*, didapatkan performansi terbaik pada model eksploitasi IV dengan nilai mAP sebesar 22,4%. Pada penelitian ini telah dilakukan eksploitasi fitur pada algoritma deteksi objek berbasis UAV yaitu pada model YOLOv5 dan eksploitasi terhadap *freeze backbone* dan pengubahan proses pada layer *concat* serta penambahan layer pada detection head arsitektur YOLOv5. Pengaruh performansi *freeze backbone* pada model dapat berkurang dikarenakan model tidak dilatih pada seluruh *network* sehingga terdapat nilai *weight* pada *backbone* yang tidak diperbarui selama *training* berlangsung. Eksploitasi pada pengubahan layer *concat* dengan mengganti proses C3 ke conv dapat meningkatkan performansi model dengan persentase 2,2%. Eksploitasi fitur pada arsitektur YOLOv5 dengan penambahan *detection head* dapat mempengaruhi tingkat performansi suatu model, khususnya model eksploitasi III memiliki nilai mAP yang lebih baik dibandingkan dengan model eksploitasi I dan II. Model eksploitasi IV dengan kombinasi eksploitasi II dan III dapat meningkatkan performansi. Hal ini menunjukkan bahwa model dengan *freeze backbone* dan eksploitasi pada layer concat serta penambahan layer juga *detection head* dapat meningkatkan performansi deteksi objek.

Referensi

[1] S. Vaddi, C. Kumar, and A. Jannesari, “Efficient object detection model for real-time UAV applications,” CoRR, vol. abs/1906.00786, 2019. [Online]. Available: http://arxiv.org/abs/1906.00786

[2] G. Jocher, A. Stoken, A. Chaurasia, J. Borovec, NanoCode012, TaoXie, Y. Kwon, K. Michael, L. Changyu, J. Fang, A. V, Laughing, tkianai, yxNONG, P. Skalski, A. Hogan, J. Nadar, imyhxy, L. Mammana, AlexWang1900, C. Fati, D. Montes, J. Hajek, L. Diaconu, M. T. Minh, Marc, albinxavi, fatih, oleg, and wanghaoyang0106, “ultralytics/YOLOv5: v6.0 - YOLOv5n ’Nano’ models, Roboflow integration, TensorFlow export, OpenCV DNN support,” Oct. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5563715

[3] “Transfer Learning with Frozen Layers,” https://docs.ultralytics.com/tutorials/ transfer-learning-froze-layers/, accessed: 2022-07-06.

[4] R. Padilla, S. L. Netto, and E. A. B. da Silva, “A survey on performance metrics for object-detection algorithms,” 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), pp. 237–242, 2020.

[5] P. Zhu, L. Wen, D. Du, X. Bian, H. Fan, Q. Hu, and H. Ling, “Detection and tracking meet drones challenge,” IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1–1, 2021.

[6] C.-J. Yang, T. Chou, F.-A. Chang, C. Ssu-Yuan, and J.-I. Guo, “A smart surveillance system with multiple people detection, tracking, and behavior analysis,” in 2016 International Symposium on VLSI Design, Automation and Test (VLSI-DAT), 2016, pp. 1–4.

[7] J. Choi, D. Chun, H. Kim, and H.-J. Lee, “Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving,” in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 502–511.

[8] J. Lee, J. Wang, D. Crandall, S. Sabanovi ˇ c, and G. Fox, “Real-time, ´ cloud-based object detection for unmanned aerial vehicles,” in 2017 First IEEE International Conference on Robotic Computing (IRC), 2017, pp. 36–43.

[9] P. Zhang, Y. Zhong, and X. Li, “Slimyolov3: Narrower, faster and better for real-time uav applications,” in 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), 2019, pp. 37–45.

[10] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, “Tph-yolov5: Improved yolov5 based on transformer prediction head for object detection on drone-captured scenarios,” ArXiv, vol. abs/2108.11539, 2021.

[11] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, “Dive into deep learning,” 2021.

[12] M. Carranza-Garc´ıa, J. Torres-Mateo, P. Lara-Ben´ıtez, and J. Garc´ıa-Gutierrez, “On the performance of one-stage and two-stage ´ object detectors in autonomous vehicles using camera data,” Remote. Sens., vol. 13, p. 89, 2021.

[13] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in ECCV, 2016.

[14] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," Insights into Imaging, vol. 9, pp. 611 – 629, 2018.

[15] "Convolutional Neural Network (CNN)," https://medium.com/@raycad. seedotech/convolutional-neural-network-cnn-8d1908c010ab, accessed: 2022-07-13.

[16] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779–788, 2016.

[17] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6517–6525, 2017.

[18] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," ArXiv, vol. abs/1804.02767, 2018.

[19] A. Bochkovskiy, C.-Y. Wang, and H.-y. Liao, "Yolov4: Optimal speed and accuracy of object detection," ArXiv, vol. abs/2004.10934, 2020.

[20] D. Thuan, D. Thuan, "Evolution of yolo algorithm and yolov5: The state-of-the-art object detention algorithm," https://www.theseus.fi/handle/10024/452552, accessed: 2021-11-12.

[21] D. Lema, O. Pedrayes, R. Usamentiaga, F. D. Garcia, and A. Alonso, "Cost-performance evaluation of a recognition service of livestock activity using aerial images," Remote Sensing, vol. 13, p. 2318, 06 2021.

[22] R. Padilla, W. Lobato Passos, T. Dias, S. Netto, and E. da Silva, "A comparative analysis of object detection metrics with a companion open-source toolkit," Electronics, vol. 10, pp. 279–306, 01 2021.

[23] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," International Journal of Computer Vision, vol. 88, pp. 303–338, 06 2010.

[24] "Metrics," https://cocodataset.org/#detection-eval, accessed: 2021-12-08.

[25] D. Du, P. Zhu, L. Wen, X. Bian, H. Lin, Q. Hu, T. Peng, J. Zheng, X. Wang, Y. Zhang, L. Bo, H. Shi, R. Zhu, A. Kumar, A. Li, A. Zinollayev, A. Askergaliyev, A. Schumann, B. Mao, B. Lee, C. Liu, C. Chen, C. Pan, C. Huo, D. Yu, D. Cong, D. Zeng, D. R. Pailla, D. Li, D. Wang, D. Cho, D. Zhang, F. Bai, G. Jose, G. Gao, G. Liu, H. Xiong, H. Qi, H. Wang, H. Qiu, H. Li, H. Lu, I. Kim, J. Kim, J. Shen, J. Lee, J. Ge, J. Xu, J. Zhou, J. Meier, J. W. Choi, J. Hu, J. Zhang, J. Huang, K. Huang, K. Wang, L. Sommer, L. Jin, L. Zhang, L. Huang, L. Sun, L. Steinmann, M. Jia, N. Xu, P. Zhang, Q. Chen, Q. Lv, Q. Liu, Q. Cheng, S. S. Chennamsetty, S. Chen, S. Wei, S. S. S. Kruthiventi, S. Hong, S. Kang, T. Wu, T. Feng, V. A. Kollerathu, W. Li, W. Dai, W. Qin, W. Wang, X. Wang, X. Chen, X. Chen, X. Sun, X. Zhang, X. Zhao, X. Zhang, X. Zhang, X. Chen, X. Wei, X. Zhang, Y. Li, Y. Chen, Y. H. Toh, Y. Zhang, Y. Zhu, Y. Zhong, Z. Wang, Z. Wang, Z. Song, and Z. Liu, "Visdrone-det2019: The vision meets drone object detection in image challenge results," in 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), 2019, pp. 213–226