PROPOSAL TUGAS AKHIR - EC224701

OPTIMISASI PENDETEKSIAN OBJEK KECIL YOLOv7 UNTUK MENDETEKSI OBJEK-OBJEK *AIRBORNE*

Dion Andreas Solang

NRP 0721 19 4000 0039

Dosen Pembimbing

Reza Fuad Rachmadi, S.T., M.T., Ph.D

NIP 19850403201212 1 001

Dr. I Ketut Eddy Purnama S.T., M.T.

NIP 19690730199512 1 001

Program Studi Strata 1 (S1) Teknik Komputer

Departemen Teknik Komputer

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya

2022

LEMBAR PENGESAHAN

OPTIMISASI PENDETEKSIAN OBJEK KECIL YOLOv7 UNTUK MENDETEKSI OBJEK-OBJEK *AIRBORNE*

PROPOSAL TUGAS AKHIR

Diajukan untuk memenuhi salah satu syarat
memperoleh gelar Sarjana Teknik pada
Program Studi S-1 Teknik Komputer
Departemen Teknik Komputer
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember

Oleh: **Dion Andreas Solang**
NRP. 0721 19 4000 0039

Disetujui oleh Tim Penguji Proposal Tugas Akhir:

Reza Fuad Rachmadi, S.T., M.T., Ph.D (Pembimbing)
NIP: 19850403201212 1 001

Dr. I Ketut Eddy Purnama S.T., M.T. (Ko-Pembimbing)
NIP: 19690730199512 1 001

TBA (Penguji I)
NIP: TBA

TBA (Penguji II)
NIP: TBA

TBA (Penguji III)
NIP: TBA

SURABAYA
Februari, 2023

OPTIMISASI PENDETEKSIAN OBJEK KECIL YOLOv7 UNTUK MENDETEKSI OBJEK-OBJEK *AIRBORNE*

Nama Mahasiswa / NRP : Dion Andreas Solang / 07211940000039
Departemen : Teknik Komputer FTEIC - ITS
Dosen Pembimbing : 1. Reza Fuad Rachmadi, S.T., M.T., Ph.D
2. Dr. I Ketut Eddy Purnama S.T., M.T.

Abstrak

Objek-objek *airborne* merupakan objek-objek yang akan terlihat sangat kecil pada kamera. YOLOv7 merupakan model pendeteksi objek *real time state of the art* yang dioptimisasi untuk pendeteksian objek umum. Oleh karena itu, untuk mendeteksi objek *airborne* dengan baik, perlu dilakukan modifikasi pada YOLOv7. Tujuan dari penelitian ini adalah untuk menemukan solusi modifikasi YOLOv7 yang dapat mengoptimisasi kemampuan pendeteksian objek kecil khususnya objek *airborne*. Modifikasi yang dilakukan terhadap YOLOv7 meliputi modifikasi arsitektur dan modifikasi *bag-of-freebies*. Modifikasi arsitektur meliputi modifikasi *neck* dan penambahan *layer head*. Modifikasi *bag-of-freebies* meliputi penambahan augmentasi mosaik dan rekalkulasi *anchor* aktif. Modifikasi-modifikasi ini akan dikombinasikan dan diuji performanya. Modifikasi yang menghasilkan model dengan skor mAP tertinggi pada dataset *airborne* akan dipilih sebagai solusi optimisasi pada penelitian ini.

Kata Kunci: ***Deteksi Objek Kecil, YOLOv7, Modifikasi Arsitektur, Modifikasi Bag-of-Freebies, Objek Airborne***

YOLOv7 SMALL OBJECT DETECTION OPTIMIZATION TO DETECT AIRBORNE OBJECTS

Student Name / NRP: Dion Andreas Solang / 07211940000039
Department : Teknik Komputer FTEIC - ITS
Advisors : 1. Reza Fuad Rachmadi, S.T., M.T., Ph.D
2. Dr. I Ketut Eddy Purnama S.T., M.T.

Abstract

Airborne objects appear very small on cameras. YOLOv7 is the state of the art real-time object detector optimized for general object detections. Thus, to detect airborne objects with YOLOv7, modifications are needed to be applied. The purpose of this research is to find a modification solution for YOLOv7 to optimize its small object detection capability especially for airborne objects. Modifications to be applied on YOLOv7 consists of architecture modifications and bag-of-freebies modifications. Architecture modifications consist of neck modification and head layer addition. Bag-of-freebies modifications consist of mosaic data augmentation dan active anchor recalculation. These modifications will be combined one with another and have their performance tested. Modifications that produces model with the highest mAP score on airborne objects dataset will be chosen as the optimization solution of this research.

Keywords: ***Small Object Detection, YOLOv7, Architecture Modification, Bag-of-Freebies Modification, Airborne Object***

DAFTAR ISI

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Seiring berkembangnya teknologi *autonomous vehicles*, terdapat banyak keinginan untuk mengaplikasikan teknologi tersebut di berbagai bidang. Salah satu aplikasi teknologi ini di bidang komersil adalah *Amazon Prime Air*. *Amazon Prime Air* memanfaatkan *Autonomous Aerial Vehicle* (AAV) untuk melakukan pengantaran barang dari warehouse ke rumah kostumer secara *autonomous* (Amazon 2022). Untuk melakukan hal ini, AAV yang digunakan harus mempunyai kemampuan penerbangan *autonomous* yang mumpuni.

Salah satu tantangan terbesar dari penerbangan autonomous adalah kemampuan *Sense and Avoid* (SAA) dari AAV. Meskipun di udara terdapat ruang gerak yang luas, tetap terdapat resiko AAV akan menabrak objek di udara. Objek - objek tersebut dapat berupa helikopter, pesawat, burung, misil, dan lain - lain. Objek - objek ini juga sering disebut dengan objek *airborne* (*Airborne Object Tracking Challenge* 2021).

Salah satu sensor yang dapat digunakan untuk melakukan SAA adalah kamera. Kamera memiliki bobot yang cukup ringan, sehingga dapat dibawah oleh AAV. Selain itu, kamera juga memiliki harga yang relatif lebih murah dibandingkan sensor - sensor seperti LiDAR atau Radar.

Dengan memilih kamera sebagai sensor, maka dibutuhkan suatu model computer vision untuk diaplikasikan pada kamera tersebut. Objek - objek *airborne* akan tampak sangat kecil pada kamera seperti yang dapat dilihat pada Gambar 1.1. Beberapa dataset kamera *airborne* yang memiliki resolusi 20482448 pixel, objeknya dapat berukuran 4 (0.00008% luas resolusi) hingga 1000 pixel (0.01% luas resolusi) sehingga terlihat sangat kecil (*Airborne Object Tracking Dataset* 2021). Oleh karena itu, dibutuhkan suatu model yang dapat mendeteksi objek - objek yang sangat kecil sehingga dapat mendeteksi objek *airborne*.

Gambar 1.1: Contoh Dataset Objek *Airborne*

YOLOv7 merupakan model state-of-the-art untuk melakukan pendeteksian objek secara

real-time. YOLOv7 memiliki akurasi tertinggi dari semua model pendeteksi objek dengan kecepatan deteksi 30 FPS (yang terpublikasi) pada GPU Nvidia V100. Terdapat versi scaled dari YOLOv7 yang memiliki jumlah parameter yang lebih kecil dan dapat diaplikasikan pada device edge computing (Wang, Bochkovskiy, et al. 2022). Oleh karena itu, YOLOv7 ini cocok untuk digunakan pada AAV di mana dibutuhkan suatu pendeteksi objek yang real-time.

1.2 Rumusan Masalah

YOLOv7 bukan merupakan model deteksi objek umum sehingga YOLOv7 tidak didesain untuk melakukan deteksi objek kecil seperti objek-objek *airborne*. Oleh karena itu, dibuatlah rumusan masalah seperti berikut:

- Apa solusi yang dapat diaplikasikan pada YOLOv7 agar kemampuan deteksi objek *airborne*-nya dapat dioptimalisasi?

1.3 Tujuan

Adapun tujuan dari tugas akhir ini adalah untuk menemukan solusi untuk mengoptimisasi kemampuan YOLOv7 mendeteksi objek airborne.

1.4 Batasan Masalah

Optimisasi kemampuan deteksi objek kecil hanya akan dilakukan dengan memodifikasi YOLOv7. Modifikasi yang diaplikasikan tidak boleh menyebabkan YOLOv7 untuk tidak dapat melakukan pendeteksian secara *real time*. Target pengaplikasian model ini adalah untuk AAV dengan *computational resource* yang terbatas sehingga model hasil modifikasi harus cukup ringan untuk hal tersebut.

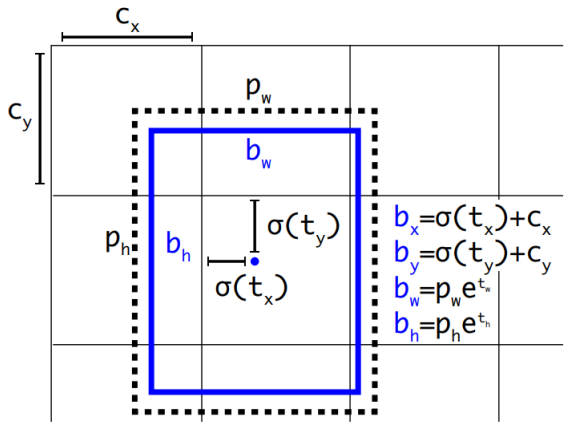
BAB 2 TINJAUAN PUSTAKA

2.1 Teori Dasar

2.1.1 Arsitektur Famili YOLO

Arsitektur famili YOLO pada dasarnya terbagi akan 3 bagian yaitu *head*, *neck*, dan *backbone*. Setiap bagian ini mempunyai fungsi masing-masing. Berikut adalah penjelasan fungsi dan cara kerja dari ketiga bagian tersebut.

2.1.1.1 *Layer Head* YOLO dan *Anchor Box*



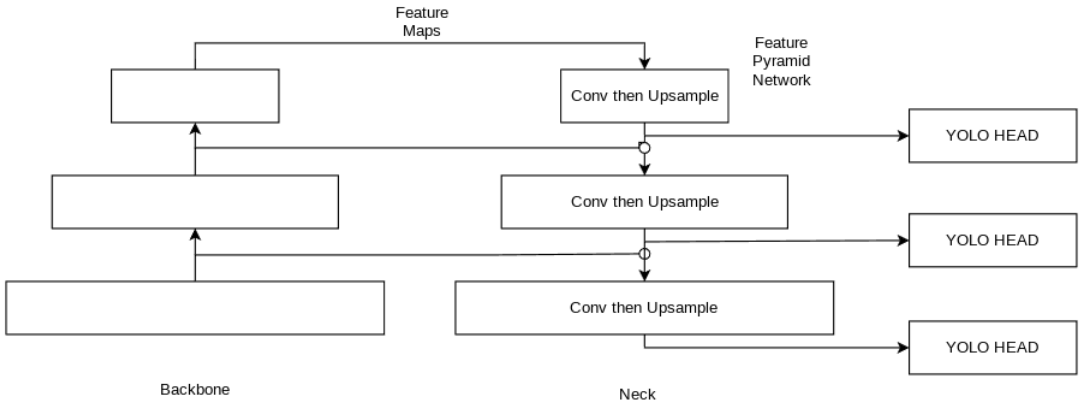
Gambar 2.1: Prediksi *Anchor Box* dan *offset* dari koordinat latis (Redmon and Farhadi 2018)

Arsitektur famili YOLO yang dipublikasikan setelah YOLOv2 terus menggunakan *anchor box* untuk melakukan deteksi (Redmon and Farhadi 2016, 2018; Bochkovskiy et al. 2020; Wang, Bochkovskiy, et al. 2020; Jocher et al. 2022; Wang, Yeh, et al. 2021; Wang, Bochkovskiy, et al. 2022). *Anchor boxes* merupakan beberapa *Bounding Box* yang telah terdefinisikan. Arsitektur YOLO akan memprediksi probabilitas *anchor box* berada pada suatu koordinat latis beserta dengan *offset anchor box* tersebut untuk menepatkan *anchor box* pada objek yang dideteksi. Penggunaan *anchor box* ini dapat meningkatkan akurasi deteksi karena *neural network* hanya perlu mencari titik tengah objek dan *error* dimensi *boudning box* dengan menggunakan *offset* (Redmon and Farhadi 2018). Hal ini lebih sederhana daripada mencari titik-titik *bounding box* secara independen sehingga lebih mudah untuk dipelajari oleh *neural network*.

Prediksi *bounding boxes* terjadi di bagian *head* dari arsitektur YOLO. Bagian *head* dari YOLO akan mengambil beberapa hasil *upsampling* yang terjadi pada *neck* YOLO, dan kemudian melakukan prediksi *anchor boxes* dari hasil tersebut. Hasil prediksi *Head* YOLO pada suatu tingkatan *upsampling* berupa tensor dengan ukuran $N \times N \times [A \times (4 + 1 + C)]$ dengan N sebagai dimensi hasil *upsampling*-nya, A sebagai jumlah *anchor boxes* untuk *scaling* tersebut, dan C sebagai jumlah kelas prediksi. Angka 4 merepresentasikan 4 *offset* b_x, b_y, b_w, b_h seperti pada Gambar 2.1 dan angka 1 merepresentasikan *objectness score* dari prediksi *bounding box*.

2.1.1.2 *Neck* YOLO

Neck dari YOLO merupakan *layer-layer* dimana *head* YOLO mengambil fitur untuk dilakukan deteksi *bounding box*. Pada YOLOv3 Redmon and Farhadi (2018), arsitektur *neck*



Gambar 2.2: *Feature Pyramid Network* pada *Neck* YOLO

dibuat menyerupai *Feature Pyramid Network* (FPN) seperti pada Gambar 2.2. Pada versi-versi YOLO selanjutnya, bentuk *neck* ini tidak banyak berubah dan pada dasarnya tetap mempertahankan bentuk *pyramid*-nya.

Penaikkan tingkatan *pyramid* dari FPN merupakan *upsampling* dari *feature map* yang dihasilkan *backbone*. Output tiap tingkatan pada FPN di *neck* inilah yang diinputkan pada *head* YOLO. Melakukan prediksi pada tingkatan *upsampling* yang berbeda-beda dapat membuat *neural network* mendapatkan lebih banyak informasi semantik dan informasi yang lebih detail sehingga dapat lebih akurat dalam mendeteksi objek besar maupun kecil.

2.1.1.3 *Backbone* YOLO

Backbone dari YOLO merupakan bagian yang mengekstrak fitur dari citra yang diinputkan. Hasil ekstraksi fitur ini akan diinputkan pada *neck* yang kemudian akan di*upsampling* olehnya. Model-model YOLO dapat menggunakan *feature extractor* dari model-model klasifikasi citra sebagai *backbone*-nya. Sebagai contoh, salah satu varian YOLO, YOLO-Z menggunakan DenseNet sebagai *backbone*-nya sedangkan arsitektur YOLO dasarnya, YOLOv5 menggunakan *backbone* YOLOv5v7.0 (Benjumea et al. 2021).

2.1.2 YOLOv7

YOLOv7 merupakan pendeteksi objek *real time* dengan skor akurasi tertinggi pada dataset COCO di tahun 2022. Pada YOLOv7, dilakukan beberapa perubahan untuk meningkatkan akurasi dan kecepatan deteksinya. Perubahan-perubahan tersebut dilakukan pada arsitekturnya dan pada *bag-of-freebies*-nya.

Perubahan arsitektur dilakukan pada *backbone*. YOLOv7 menggunakan *Extended Efficient Layer Aggregation Network* (E-ELAN) sebagai *backbone*, berbeda dengan leluhurnya YOLOv4 yang menggunakan CSP-Darknet. E-ELAN merupakan arsitektur *neural network* yang efisien karena E-ELAN didesain dengan mengontrol *gradient path* terpanjang yang terpendek. Karena efisiensinya, arsitektur E-ELAN ini dapat meningkatkan kecepatan deteksi dan akurasi. (Wang, Bochkovskiy, et al. 2022)

Bag-of-freebies merupakan kumpulan metode peningkatan akurasi yang tidak meningkatkan *cost inferrence* (Bochkovskiy et al. 2020). Pada YOLOv7, ditambahkan beberapa *bag-of-freebies* yang dapat dilatih seperti *re-parameterized convolution* dan *extra auxilary head* di tengah-tengah *neural network*. Selain kedua itu, YOLOv7 juga menambahkan *trainable bag-of-freebies* dari YOLOR seperti EMA, *Implicit Knowledge*, dan *conv-bn topology Batch Nor-*

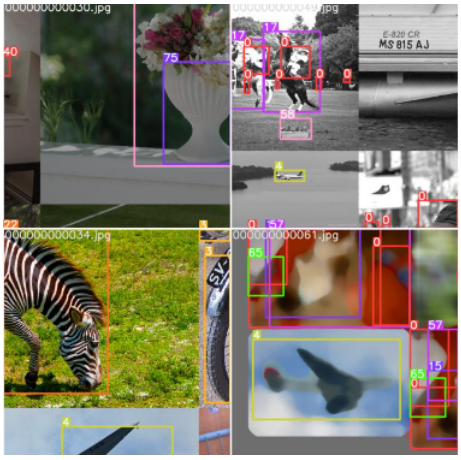
malization (Wang, Bochkovskiy, et al. 2022).

2.2 Rekalkulasi *Anchor*

Anchor box dari model-model *pre-trained* YOLO pada umumnya mengoptimisasi *anchor box* modelnya pada dataset COCO. Ukuran *anchor box* yang akan digunakan pada model YOLO dapat dikonfigurasikan agar lebih sesuai dengan dataset yang akan digunakan untuk melatih model YOLO. Penyesuaian ini dapat meningkatkan IoU(*Intersection Over Union*) prediksi model dengan *ground truth* sehingga meningkatkan akurasi.

Penyesuaian anchor box dapat dilakukan pada saat sebelum training atau pada saat training. Penyesuaian anchor box sebelum training dapat dilakukan dengan cara mengkonfigurasi secara manual tiap ukuran *anchor box* atau dengan menggunakan algoritma *clustering*. Penggunaan algoritma *clustering* akan lebih baik karena setiap ukuran *anchor box*-nya disesuaikan dengan pengelompokan-pengelompokan ukuran *bounding box* natural yang terdapat pada dataset. Untuk penyesuaian saat training, dapat digunakan algoritma dari Zhong et al. (2018). Algoritma ini akan mengoptimisasi ukuran-ukuran anchor bukan hanya berdasarkan dataset, namun berdasarkan kemampuan dari neural network pendeteksi objeknya. Untuk melakukan hal tersebut, algoritma ini akan memanfaatkan back propagation localization loss untuk rekalkulasi anchor.

2.3 Augmentasi Mosaik



Gambar 2.3: Contoh Augmentasi Data Mosaik (Jocher et al. 2022)

Augmentasi mosaik merupakan teknik augmentasi yang baru dikenalkan pada YOLOv4. Teknik augmentasi ini akan memilih 4 gambar dari dataset, memotong gambar-gambar tersebut dan menggabungkannya secara acak pada satu gambar seperti pada Gambar 2.3. Hasil dari penggabungan itu membuat gambar terlihat seperti mosaik. Teknik augmentasi ini mampu meningkatkan akurasi model (Bochkovskiy et al. 2020).

2.4 Penelitian Terkait

2.4.1 YOLO-Z

YOLO-Z merupakan arsitektur famili YOLO yang modifikasi dari YOLOv5 (Benjumea et al. 2021). Modifikasi-modifikasi yang dilakukan meliputi pergantian *backbone*, *neck*, dan jumlah *anchor Backbone* dari YOLOv5r5.0 menjadi DenseNet yang di-*downscale*. *Neck* dari YOLO-Z juga diganti dari PanNet menjadi FPN dan biFPN tergatung pada *scale* YOLO-Z yang digunakan.

Modifikasi pada YOLO-Z didesain untuk mendeteksi objek kecil untuk tujuan melakukan deteksi *cone* yang nampak jauh pada lintasan *autonomous racing* secara *real time* (lihat Gambar 2.4). Modifikasi-modifikasi dibuktikan dapat meningkatkan kemampuan pendeteksian objek kecil (Benjumea et al. 2021). Oleh karena itu, untuk meningkatkan kemampuan mendeteksi objek kecil YOLOv7, beberapa modifikasi yang dilakukan YOLO-Z pada YOLOv5 dapat diaplikasikan.

Gambar 2.4: Contoh Objek *Cone* yang Terlihat Jauh dari Kamera

2.4.2 exYOLO

exYOLO merupakan hasil modifikasi arsitektur YOLOv3 (Xiao 2021). Pada exYOLO, dilakukan modifikasi *neck* dengan menambahkan suatu *Receptive Field Block* sebelum penggabungan *feature map* yang akan diupsampling. Modifikasi-modifikasi ini membuat exYOLO memiliki skor mAP yang lebih tinggi daripada YOLOv3 pada dataset PASCAL VOC 2007.

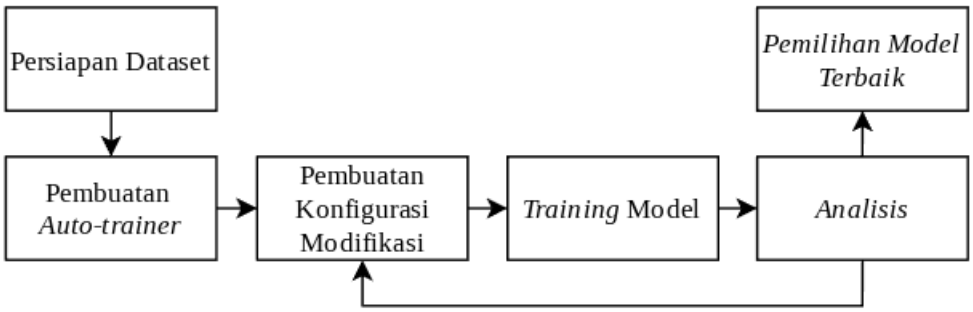
BAB 3 METODOLOGI

3.1 Metode Pencarian Solusi Optimisasi

Untuk mencari solusi optimisasi pendeteksian objek kecil YOLOv7 yang terbaik, akan dilakukan penambahan atau perubahan *bag-of-freebies*, *bag-of-specials*, dan arsitektur YOLOv7. Setiap modifikasi-modifikasi itu akan diaplikasikan secara independen dan kombinatif. Yang dimaksud dengan kombinatif adalah modifikasi-modifikasi akan dikombinasikan menjadi 1 model YOLOv7.

Setiap kombinasi modifikasi, independen atau tidak, akan diuji kemampuannya mendeteksi objek *airborne*. Solusi optimisasi terbaik akan ditentukan berdasarkan metrik AP_{50}. Subbab 3.2 akan membahas tentang kandidat modifikasi-modifikasi yang dapat dilakukan.

Tahapan pencarian solusi optimisasi sendiri dapat dibagi menjadi enam tahap. Tahap-tahap tersebut adalah Persiapan Dataset, Pembuatan *Auto-trainer*, Pembuatan Konfigurasi Modifikasi, *Training Model*, Analisis, dan Pemilihan Model Terbaik. Urutan pengerjaan dari tahap-tahap ini dapat dilihat pada Gambar 3.1.



Gambar 3.1: Urutan Pengerjaan Penelitian

Pada tahap persiapan dataset, akan dilakukan pengunduhan dataset dari *Airborne Object Tracking Dataset* (2021). Gambar-gambar pada dataset ini kemudian akan di-*sampling* dan didistribusikan menjadi dataset *training*, validasi, dan pengujian. Pada pendistribusian dataset ini juga akan dilakukan *balancing* antar kelas dan dataset positif negatif. *Balancing* dilakukan agar tidak ada kelas yang mendominasi pada dataset.

Selanjutnya, di tahap pembuatan *auto-trainer*, akan dilakukan pembuatan program yang dapat dengan otomatis membangun dan melatih *neural network* modifikasi YOLOv7. Pembuatan *auto-trainer* ini dilakukan agar proses-proses pengerjaan tahapan-tahapan selanjutnya menjadi dapat dilakukan dengan lebih mudah.

Setelah itu, akan dilakukan pembuatan konfigurasi-konfigurasi modifikasi. Konfigurasi modifikasi YOLOv7 akan dibuat agar dapat diinputkan pada *auto-trainer*. Konfigurasi modifikasi akan berisi kombinasi modifikasi-modifikasi yang ada pada subbab 3.2.

Tahapan selanjutnya adalah *Training* model. Pada tahap ini, konfigurasi-konfigurasi modifikasi pada tahapan sebelumnya akan dibangun dan kemudian dilatih. Hasil dari tahap ini adalah *weights neural network* modifikasi YOLOv7, histori pelatihannya, dan metrik-metriknya pada dataset uji.

Pada tahap analisis, hasil dari tahap sebelumnya akan dianalisis untuk mencari tahu performa model-model hasil modifikasi. Analisis juga dilakukan untuk menemukan *gap* kandidat modifikasi lain yang masih bisa dieksplorasi untuk meningkatkan kemampuan pendeteksian objek kecil YOLOv7. Ketika suatu kandidat modifikasi seperti itu ditemukan, maka akan dilakukan kembali pembuatan konfigurasi modifikasi untuk menguji kandidat modifikasi tersebut.

Tahapan terakhir adalah pemilihan model terbaik. Pada tahapan ini akan dipilih model yang memiliki performa terbaik dari antara model-model hasil modifikasi lainnya. Pemilihan model akan dilakukan dengan berdasarkan pada skor mAP tertinggi. Untuk mempertahankan solusi optimisasi yang dapat melakukan deteksi secara *real time*, model yang akan dipilih hanyalah model yang dapat melakukan deteksi dengan kecepatan minimum 10 FPS pada *onboard computing device* seperti Jetson TX2.

3.2 Kandidat Modifikasi

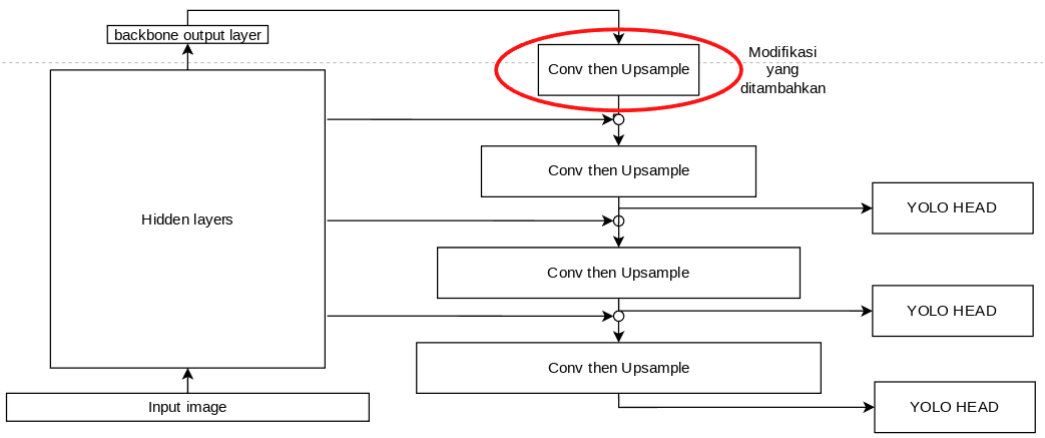
3.2.1 Rekalkulasi *Anchor On-Training*

Yang dimaksud dengan rekalkulasi *anchor* aktif adalah ketika ukuran-ukuran *anchor box* direkalkulasi pada saat training. Berbeda dengan *clustering* ukuran *anchor* sebelum *training* seperti pada YOLOv2 (Redmon and Farhadi 2016), ukuran-ukuran *anchor* akan di-*learning* bersama dengan pendeteksi objeknya. Untuk melakukan hal ini, digunakan algoritma optimisasi *anchor box* dari Zhong et al. (2018). Dengan menggunakan algoritma tersebut, ukuran-ukuran *anchor* akan teroptimisasi bukan hanya pada dataset, namun juga pada kemampuan YOLOv7.

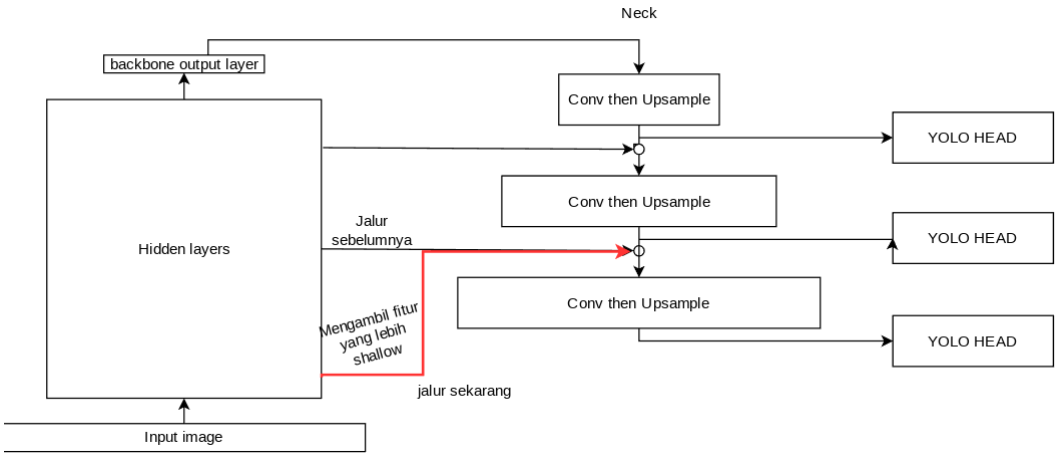
3.2.2 Augmentasi Mosaik

Berbeda dengan versi YOLO sebelumnya (Bochkovskiy et al. 2020; Jocher et al. 2022), YOLOv7 tidak menggunakan augmentasi mosaik. Padahal, dapat dilihat dari YOLOv4, augmentasi mosaik dapat meningkatkan akurasi deteksi. Oleh karena itu, akan dilakukan percobaan penambahan augmentasi mosaik pada YOLOv7 untuk melihat apakah augmentasi mosaik ini akan meningkatkan akurasinya atau tidak.

3.2.3 Modifikasi Neck



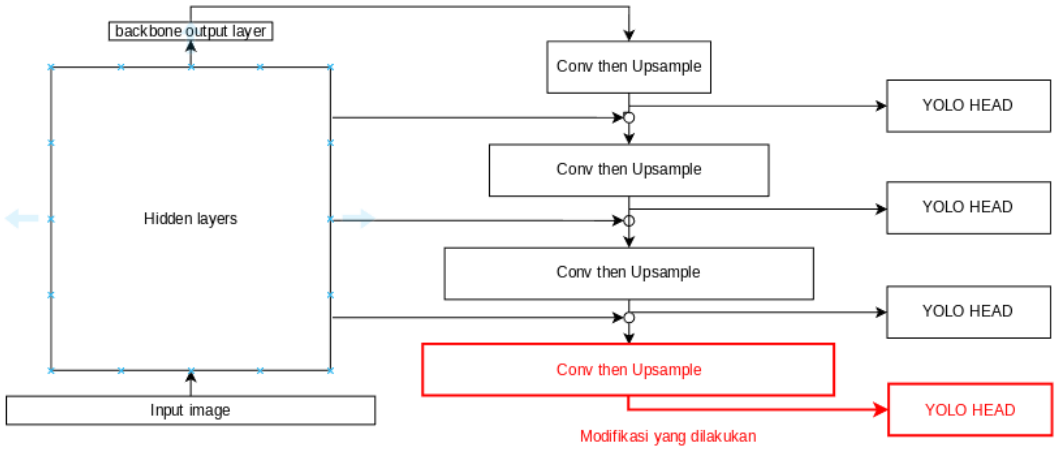
Gambar 3.2: Menambah *upsampling* pada *neck*



Gambar 3.3: Menggunakan *feature map* dari layer yang lebih di belakang

Seperti pada penelitian-penelitian terkait di subbab 2.4, modifikasi *neck* dapat dilakukan untuk meningkatkan akurasi pendeteksian objek kecil. Modifikasi *neck* dapat dilakukan dengan menambahkan layer upsampling seperti pada Gambar 3.2 atau dengan memindahkan sumber feature map untuk dari beberapa layer neck lebih jauh ke belakang seperti pada Gambar 3.3. Penambahan layer upsampling dapat membuat neural network untuk mendapatkan feature-map yang lebih detail sehingga dapat melakukan pendeteksian objek kecil dengan lebih baik. Pemindahan sumber feature map ke belakang dapat dilakukan untuk mengantisipasi fitur objek kecil yang dapat hilang ketika layer neural network semakin dalam. Dengan memindahkannya lebih ke belakang, YOLOv7 akan melakukan deteksi pada layer dengan abstraksi yang lebih rendah (sedikit informasi yang hilang).

3.2.4 Penambahan YOLO Head



Gambar 3.4: Penambahan *Layer* Head pada YOLO

Penambahan YOLO head dapat membuat YOLOv7 melakukan deteksi pada skala yang lebih tinggi. Hal ini akan berpengaruh pada kemampuan pendeteksian objek kecil. Dengan melakukan pendeteksian pada skala yang lebih beragam, YOLOv7 dapat mendeteksi objek yang besar maupun kecil. Perhatikan bahwa penambahan YOLO Head akan diikuti dengan penambahan *layer upsampling* pada *neck* seperti di Gambar 3.4.

3.3 Dataset

3.3.1 Sumber Dataset

Dataset untuk objek-objek *airborne* didapatkan dari *Airborne Object Tracking Dataset* (2021) dan dihost pada suatu server AWS S3 Bucket. Dataset ini berisi video-video monokromatik penerbangan UAV. Terdapat 4 kelas pada dataset ini yaitu pesawat, helikopter, burung, dan *other*. Distribusi dataset training dan uji dapat dilihat pada tabel 3.1 sedangkan distribusi kelas dari dataset dapat dilihat pada Tabel 3.2.

Tabel 3.1: Distribusi Dataset Training dan Test

Pembagian Dataset	Ukuran (TB)	Sekuen penerbangan UAV	Jumlah Gambar	Jumlah Label
Training	11,3	4154	4975765	2891891
Validation + Test	2.1	789	943852	496075
Total	13,4	4943	5919617	3387966

Tabel 3.2: Distribusi Kelas Dataset

Pembagian	Total Objek	Pesawat	Helikopter	Burung	*Other*
Training	2,89 juta	0,79 juta	1,22 juta	0,33 juta	0,54 juta
Validation + Test	0,50 juta	0,13 juta	0,17 juta	0,06 juta	0,14 juta
Total	3,39 juta	0,92 juta	1,39 juta	0,39 juta	0,69 juta

3.3.2 Sampling Dataset

Karena jumlah dataset pada *Airborne Object Tracking Dataset* (2021) berukuran sangat besar, hanya sebagian dari dataset tersebut akan digunakan untuk *training* dan *test*. Akan diambil total 60000 gambar dari dataset dengan pembagian sesuai dengan Tabel 3.3

Tabel 3.3: Distribusi Sampling Dataset

Pembagian	Total	Presentase				
	Gambar	Pesawat	Helikopter	Burung	*Other*	Negatif
Training	54000	23,75%	23,75%	23,75%	23,75%	5%
Validation	3000	20%	20%	20%	20%	20%
Test	3000	20%	20%	20%	20%	20%

3.4 Skema Training Model

Untuk melatih berbagai modifikasi YOLOv7, akan dibuat suatu *auto-trainer*. *Auto-trainer* ini akan menerima suatu *file* konfigurasi modifikasi YOLOv7, dan dengan otomatis membangun arsitektur YOLOv7 yang termodifikasi dan melatihnya. Setelah mendapatkan model modifikasi YOLOv7 yang sudah dilatih, *auto-trainer* akan menguji model tersebut dengan dataset uji. Metrik-metrik pengujian, grafik histori *training loss vs validation loss*, dan *weights* dari model kemudian akan dikirim ke user. Dengan membuat *auto-trainer* ini, proses pelatihan model dan pelaporan hasil menjadi terotomasi sehingga akan mempermudah proses penelitian.

3.5 Timeline Pelaksanaan Penelitian

Tabel 3.4: Tabel timeline

Kegiatan	Minggu															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Persiapan Dataset																
Pemb. *Auto-trainer*																
Pemb. Konfigurasi Modifikasi																
Training Model																
Analisis																
Pemb. Laporan																

[Halaman ini sengaja dikosongkan]

DAFTAR PUSTAKA

Airborne object tracking challenge. (2021). Retrieved September 1, 2022, from https://www.aicrowd.com/challenges/airborne-object-tracking-challenge

Airborne object tracking dataset. (2021). Retrieved September 1, 2022, from https://registry.opendata.aws/airborne-object-tracking

Amazon. (2022). *Amazon prime air prepares for drone deliveries*. Retrieved September 1, 2022, from https://www.aboutamazon.com/news/transportation/amazon-prime-air-prepares-for-drone-deliveries

Benjumea, A., Teeti, I., Cuzzolin, F., & Bradley, A. (2021). Yolo-z: Improving small object detection in yolov5 for autonomous vehicles.

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection.

Jocher, G., Chaurasia, A., Stoken, A., Borovec, J., NanoCode012, Kwon, Y., Michael, K., TaoXie, Fang, J., imyhxy, & et al. (2022). Ultralytics/yolov5: V7.0 - yolov5 sota realtime instance segmentation. https://doi.org/10.5281/zenodo.7347926

Redmon, J., & Farhadi, A. (2016). Yolo9000: Better, faster, stronger.

Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement.

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2020). Scaled-YOLOv4: Scaling cross stage partial network.

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors.

Wang, C.-Y., Yeh, I.-H., & Liao, H.-Y. M. (2021). You only learn one representation: Unified network for multiple tasks.

Xiao, J. (2021). exYOLO: A small object detector based on YOLOv3 object detector. *Procedia Computer Science*, *188*, 18–25. https://doi.org/10.1016/j.procs.2021.05.048

Zhong, Y., Wang, J., Peng, J., & Zhang, L. (2018). Anchor box optimization for object detection.