

**EKSPLOITASI PERFORMA MODEL *EFFICIENTDET* BERBASIS BIAYA
KOMPUTASI RENDAH UNTUK DETEKSI OBJEK *UNMANNED AERIAL
VEHICLE***

***PERFORMANCE EXPLOITATION IN LOW COMPUTING COST
MODEL-BASED EFFICIENTDET FOR UAV OBJECT DETECTION***

TUGAS AKHIR

Disusun sebagai syarat mata kuliah Tugas Akhir pada Program Studi S1 Teknik
Telekomunikasi

Oleh

IGA NARENDRA PRAMAWIJAYA

110118XXXX



**Telkom
University**

**FAKULTAS TEKNIK ELEKTRO
UNIVERSITAS TELKOM
BANDUNG
2022**

LEMBAR PENGESAHAN

TUGAS AKHIR

**EKSPLORASI PERFORMA MODEL *EFFICIENTDET* BERBASIS BIAYA
KOMPUTASI RENDAH UNTUK DETEKSI OBJEK *UNMANNED AERIAL***

VEHICLE

***PERFORMANCE EXPLOITATION IN LOW COMPUTING COST MODEL-BASED
EFFICIENTDET FOR UAV OBJECT DETECTION***

Telah disetujui dan disahkan sebagai Tugas Akhir

Program S1 Teknik Telekomunikasi

Fakultas Teknik Elektro

Universitas Telkom

Bandung

Disusun oleh:

Iga Narendra Pramawijaya

110118XXXX

Bandung, 2 Agustus 2022

Menyetujui,

Pembimbing I

tanggal sidang: 8/8/2022

Suryo Adhi Wibowo, S.T., M.T., Ph.D. Dr. Koredianto Usman, S.T., M.Sc.

10870003

02750053

Pembimbing II

tanggal sidang: 8/8/2022

LEMBAR PERNYATAAN ORISINALITAS

Nama : Iga Narendra Pramawijaya

NIM : 110118

Alamat :

No. Telepon :

Email : iga.narendra@gmail.com

Menyatakan bahwa Tugas Akhir ini merupakan karya orisinal saya sendiri, dengan judul :

EKSPLOITASI PERFORMA MODEL *EFFICIENTDET* BERBASIS BIAYA KOMPUTASI RENDAH UNTUK DETEKSI OBJEK *UNMANNED AERIAL VEHICLE*

Atas pernyataan ini, saya siap menanggung resiko/sanksi yang dijatuhkan kepada saya apabila kemudian ditemukan adanya pelanggaran terhadap kejujuran akademik atau etika keilmuan dalam karya ini, atau ditemukan bukti yang menunjukkan ketidakaslian karya ini.

Bandung, 2 Agustus 2022



Iga Narendra Pramawijaya

110118

ABSTRAK

Kendaraan udara tanpa awak atau *Unmanned Aerial Vehicles* (UAV) dilengkapi kamera dengan resolusi tinggi. Kamera pada UAV dapat mengambil foto atau citra UAV. Penelitian sebelumnya merangkum model deteksi objek citra UAV. Berbagai model tersebut memakan beban komputasi tinggi. Model jaringan syaraf tiruan yang tidak dilatih menggunakan citra UAV akan menghasilkan performa yang buruk. Model deteksi objek yang dilatih kembali menggunakan citra UAV akan memakan beban komputasi yang tinggi. Model pengenalan objek *EfficientDet* lebih ringan secara komputasi. Maka dari itu penulis ingin meneliti peforma model pengenalan objek beban komputasi rendah *EfficientDet* pada citra UAV.

Dalam tugas akhir ini, sebuah sistem deteksi objek pada citra UAV yang dilatih menggunakan komputer konvensional akan dirancang untuk mendeteksi 10 kelas objek dengan menggunakan model *EfficientDet* versi D0. Setelah data didapatkan maka akan dilakukan *preprocessing* berupa konversi anotasi. Selanjutnya akan dilakukan proses pelatihan model. Setiap pelatihan, model akan diuji sehingga menghasilkan nilai validasi. Nilai validasi pada pelatihan terakhir yang akan dianalisa sebagai patokan performa.

Tugas akhir ini menggunakan himpunan data VisDrone yang terdiri atas 6471 gambar data latih dan 548 gambar data uji. Seluruh gambar berukuran 960×540 piksel. Tolak ukur performa model deteksi objek adalah nilai *average precision* (AP) area besar, AP nilai IoU 0,5; 0,75, dan *average recall* (AR) dengan maksimal deteksi 1, 10, dan 100. Model *EfficientDet d0* unggul melalui AR maksimal deteksi 1 dibandingkan penelitian sebelumnya dengan skor 2,1%.

Kata Kunci : *EfficientDet*, Citra UAV, Beban Komputasi Rendah.

ABSTRACT

Unmanned Aerial Vehicles are equipped with high-resolution cameras. The camera on the UAV can take UAV photos. Previous research novelized object detection models for UAV images, but available models took high computational cost. Artificial neural network models not trained using the UAV image will result in poor performance. When object detection models retrained using UAV images, it will take up a high computational cost. EfficientDet object detection model is significantly lower in computing cost. Therefore, the author wants to examine the performance of the low computing cost model-based EfficientDet on UAV images.

In this thesis, an object detection system on UAV imagery is trained using conventional computers will be designed to detect 10 classes object using the D0 version of the EfficientDet model. After the data is obtained then preprocessing will be carried out in the form of annotation conversion. Next, model training process will be carried out several times. In each training, the model will be tested so as to produce a validation value. The last validation value will be analyzed as a benchmark for performance.

This thesis uses the VisDrone dataset consisting of 6471 images as training data and 548 images of test data with 960×540 pixels resolution. In this thesis, large area average precision (AP) value, AP IoU value 0.5, 0.75, and average recall (AR) with maximum detection of 1, 10, and 100 will be benchmarks indicating model performance. As a result, EfficientDet d0 model surpassed TridentNet model according to latest research with ARmax1 value of 2.1%.

Key Word : *EfficientDet, UAV Images, Low Computing Cost.*

KATA PENGANTAR

Puji dan syukur kepada Tuhan yang Maha Esa atas segala kasih dan anugerah yang telah dilimpahkan kepada penulis, sehingga Tugas Akhir yang berjudul **Deteksi Objek pada Citra Unmanned Aerial Vehicle menggunakan EfficientDet.** Tugas akhir ini disusun sebagai syarat untuk menempuh gelar sarjana dan menyelesaikan pendidikan pada program studi S1 Teknik Telkomunikasi Universitas Telkom, Bandung.

Pada penulisan tugas akhir ini, penulis menyadari bahwa masih terdapat banyak kekurangan karena keterbatasan ilmu yang dimiliki. Segala kritik dan saran yang bersiat membangun sangat diterima guna memperbaiki penulisan tugas akhir ini.

Sebagai penutup, penulis memohon maaf atas segala kesalahan yang penulis lakukan baik sengaja maupun yang tidak disengaja saat menyelesaikan Tugas Akhir ini. Semoga Tugas Akhir ini dapat bermanfaat bagi penulis, pembaca, dan semua kalangan di dunia pendidikan.

Bandung, 2 Agustus 2022

Iga Narendra Pramawijaya

UCAPAN TERIMA KASIH

Dalam penyusunan Tugas Akhir ini, Penulis ingin memberikan ucapan terima kasih yang sebesar-besarnya kepada seluruh pihak yang terlibat dalam pembuatan Tugas Akhir ini. Melalui ucapan terima kasih ini, penulis ingin mengucapkan terima kasih kepada:

- **Tuhan Yang Maha Esa, Ida Sang Hyang Widhi Wasa** senantiasa memberikan kelancaran, kekuatan, petunjuk, serta pengampunan tiada akhir untuk penulis.
- Bapak **Prof. Adiwijaya** selaku Rektor Telkom University beserta jajarannya.
- Kaprodi S1 Teknik Telekomunikasi sekaligus Dosen Wali penulis, ibu **Dr. Leanna Vidya Yovita, S.T., M.T.** yang telah memberikan tuntunan akademis selama masa studi penulis dan akan senantiasa menjadi inspirasi.
- Bapak **Dr. Koredianto Usman, S.T., M.Sc.** dan bapak **Suryo Adhi Wibowo, S.T., M.T., Ph.D.** selaku pembimbing II dan pembimbing I yang tiada bosan memberikan arahan teknis dan akademik untuk penulis.
- Seluruh keluarga besar di Subang, Denpasar, Makassar, Klungkung, Gianyar, dan Bandung, khususnya ayah penulis, **I Gusti Agung Mayun Pariwijaya, S.E.**, dan ibu penulis, **Dr. Ni Made Swasti Wulanyani, S.Psi, M.Erg, Psi.** yang telah memberikan dukungan penuh kepada penulis sampai saat ini dan seterusnya, serta telah menjadi inspirasi besar penulis dalam bidang akademik dan finansial.
- Rekan seperjuangan tugas akhir, **Zachary, Mutiarahmi, Jannah, Felix, Melani, Bangga, Alfan, Andi Wahyu, Maulin, Ivy, Denta, Dewi, Ibrohim,**

Zahra, Ivena, Fidela, Farhan, dan Tsabit yang tiada henti berbagi tawa dan motivasi sepanjang penyusunan tugas akhir.

- **Laboratorium IMV dan i-Smile** yang telah memberikan fasilitas penunjang bagi penulis untuk menyelesaikan tugas akhir ini.
- **Keluarga Besar PukulEnam** selaku tim profesional penulis yang senantiasa membantu kelancaran usaha penulis, juga jajaran *Co-Founder* seperti Putu Gede Arya Karna Sampalan, I Nyoman Satiya Nanjaya Sadha, S.Ked., Vi-sakha Vidyadevi Wiguna, S.Ked., dan Abiyyu Didar Haq, S.Ked. yang telah sangat memotivasi penulis untuk menyelesaikan tugas akhir dengan cepat dan tepat.
- Semua insan sesama penyitas gangguan mental, karena telah bertahan hidup walaupun pernah mencoba mengakhiri hidup, dan karena tetap yakin bahwa harapan serta hari esok yang lebih cerah masih ada dan dapat diperjuangkan.
- Serta pihak-pihak lain yang tidak dapat disebutkan satu-persatu.

Bandung, 18 Juli 2022

Iga Narendra Pramawijaya

DAFTAR ISI

LEMBAR PENGESAHAN

LEMBAR PERNYATAAN ORISINALITAS

ABSTRAK

iv

KATA PENGANTAR

vi

UCAPAN TERIMA KASIH

vii

DAFTAR ISI

ix

DAFTAR GAMBAR

xi

DAFTAR SINGKATAN

xv

I PENDAHULUAN

1

1.1	Latar Belakang Masalah	1
1.2	Rumusan Masalah	2
1.3	Tujuan dan Manfaat	2
1.4	Batasan Masalah	2
1.5	Metode Penelitian	3
1.6	Sistematika Penulisan	4

DAFTAR LAMPIRAN

1

II DASAR TEORI

5

2.1	Citra digital dengan kanal warna RGB	5
2.2	Citra <i>Unmanned Aerial Vehicle</i> (UAV)	6

2.3	Jaringan Syaraf Tiruan (<i>Artifical Neural Network</i>)	7
2.4	Deteksi objek	9
2.5	<i>EfficientDet</i>	10
2.6	Fungsi optimasi	12
2.6.1	Adam	12
2.6.2	<i>Stochastic Gradient Descent</i>	16
III PERANCANGAN SISTEM		18
3.1	Rancangan Sistem	18
3.1.1	Akuisisi Data	20
3.1.2	<i>Preprocessing</i>	20
3.1.3	Pelatihan Model	20
3.1.4	Analisis Hasil	21
3.2	Performansi Sistem	21
3.2.1	<i>Average Precision</i> (AP) dan <i>Average Recall</i> (AR)	22
3.3	Sistematika Data	22
IV ANALISIS SIMULASI SISTEM		24
4.1	Analisis dan Hasil Pengukuran <i>Average Precision</i>	24
4.1.1	Hasil <i>Average Precision</i>	24
4.1.2	Hasil <i>Average Recall</i>	29
4.2	Pembahasan	33
V KESIMPULAN DAN SARAN		34
5.1	Kesimpulan	34
5.2	Saran	34
DAFTAR PUSTAKA		35
LAMPIRAN		

DAFTAR GAMBAR

2.1	Contoh warna yang dihasilkan bila sebuah titik memiliki nilai RGB (205,100,50) (warna yang dihasilkan mungkin kurang akurat karena keterbatasan percetakan)	5
2.2	Contoh gambar pada himpunan data <i>VisDrone</i>	7
2.3	Cara kerja salah satu model deteksi objek (YOLO)	10
2.4	Grafik kinerja dan FLOP dari berbagai model deteksi objek termasuk <i>EfficientDet</i>	11
2.5	Bagian dari model <i>EfficientDet</i> yang akan di eksplorasi	12
3.1	Diagram blok rancangan sistem.	18
3.2	Diagram alir sistem.	19
3.3	Kelas-kelas dalam himpunan data <i>Visdrone</i>	20
4.1	Skor AP50 pada fungsi optimasi SGD dan Adam setelah dilatih 10 sampai 50 <i>epoch</i>	24
4.2	<i>intesection over union</i> dari kotak deteksi menutupi 50 persen area deteksi asli.	25
4.3	Skor AP75 pada fungsi optimasi SGD dan Adam setelah dilatih 10 sampai 50 <i>epoch</i>	26
4.4	<i>intesection over union</i> kotak deteksi prediksi sebesar 75 persen dari kotak deteksi asli.	26
4.5	Skor AP pada fungsi optimasi SGD dan Adam setelah dilatih 10 sampai 50 <i>epoch</i>	27
4.6	Contoh hasil inferensi	28

4.7	ARmax1 pada fungsi optimasi SGD dan Adam setelah dilatih 10 sampai 50 <i>epoch</i>	29
4.8	ARmax10 pada fungsi optimasi SGD dan Adam setelah dilatih 10 sampai 50 <i>epoch</i>	30
4.9	ARmax100 pada fungsi optimasi SGD dan Adam setelah dilatih 10 sampai 50 <i>epoch</i>	32
0.1	Bukti Persetujuan Pembimbing 1 (Suryo Adhi Wibowo, S.T., M.T., Ph.D)	
0.2	Bukti Persetujuan Pembimbing 2 (Dr. Koredianto Usman, S.T., M.Sc)	
0.3	Rincian AP50 pada fungsi optimasi SGD dan Adam setelah dilatih 10 <i>epoch</i>	
0.4	Rincian AP50 pada fungsi optimasi SGD dan Adam setelah dilatih 20 <i>epoch</i>	
0.5	Rincian AP50 pada fungsi optimasi SGD dan Adam setelah dilatih 30 <i>epoch</i>	
0.6	Rincian AP50 pada fungsi optimasi SGD dan Adam setelah dilatih 50 <i>epoch</i>	
0.7	Rincian AP75 pada fungsi optimasi SGD dan Adam setelah dilatih 10 <i>epoch</i>	
0.8	Rincian AP75 pada fungsi optimasi SGD dan Adam setelah dilatih 20 <i>epoch</i>	
0.9	Rincian AP75 pada fungsi optimasi SGD dan Adam setelah dilatih 30 <i>epoch</i>	
0.10	Rincian AP75 pada fungsi optimasi SGD dan Adam setelah dilatih 40 <i>epoch</i>	
0.11	Rincian AP75 pada fungsi optimasi SGD dan Adam setelah dilatih 50 <i>epoch</i>	

0.12	Rincian AP area besar pada fungsi optimasi SGD dan Adam setelah dilatih 10 <i>epoch</i>
0.13	Rincian AP area besar pada fungsi optimasi SGD dan Adam setelah dilatih 20 <i>epoch</i>
0.14	Rincian AP area besar pada fungsi optimasi SGD dan Adam setelah dilatih 30 <i>epoch</i>
0.15	Rincian AP area besar pada fungsi optimasi SGD dan Adam setelah dilatih 40 <i>epoch</i>
0.16	Rincian AP area besar pada fungsi optimasi SGD dan Adam setelah dilatih 50 <i>epoch</i>
0.17	Rincian ARmax1 pada fungsi optimasi SGD dan Adam setelah dilatih 10 <i>epoch</i>
0.18	Rincian ARmax1 pada fungsi optimasi SGD dan Adam setelah dilatih 20 <i>epoch</i>
0.19	Rincian ARmax1 pada fungsi optimasi SGD dan Adam setelah dilatih 30 <i>epoch</i>
0.20	Rincian ARmax1 pada fungsi optimasi SGD dan Adam setelah dilatih 40 <i>epoch</i>
0.21	Rincian ARmax1 pada fungsi optimasi SGD dan Adam setelah dilatih 50 <i>epoch</i>
0.22	Rincian ARmax10 pada fungsi optimasi SGD dan Adam setelah dilatih 10 <i>epoch</i>
0.23	Rincian ARmax10 pada fungsi optimasi SGD dan Adam setelah dilatih 20 <i>epoch</i>
0.24	Rincian ARmax10 pada fungsi optimasi SGD dan Adam setelah dilatih 30 <i>epoch</i>
0.25	Rincian ARmax10 pada fungsi optimasi SGD dan Adam setelah dilatih 40 <i>epoch</i>

0.26 Rincian ARmax10 pada fungsi optimasi SGD dan Adam setelah dilatih 50 <i>epoch</i>
0.27 Rincian ARmax100 pada fungsi optimasi SGD dan Adam setelah dilatih 10 <i>epoch</i>
0.28 Rincian ARmax100 pada fungsi optimasi SGD dan Adam setelah dilatih 20 <i>epoch</i>
0.29 Rincian ARmax100 pada fungsi optimasi SGD dan Adam setelah dilatih 30 <i>epoch</i>
0.30 Rincian ARmax100 pada fungsi optimasi SGD dan Adam setelah dilatih 40 <i>epoch</i>
0.31 Rincian ARmax100 pada fungsi optimasi SGD dan Adam setelah dilatih 50 <i>epoch</i>

DAFTAR SINGKATAN

UAV Unmanned Aerial Vehicle

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Kendaraan udara tanpa awak, atau *Unmanned Aerial Vehicles* (UAV) telah dilengkapi dengan berbagai instrumentasi. Berbagai instrumen tersebut digunakan untuk navigasi kendaraan itu sendiri, atau untuk pengambilan data dari jarak jauh. Pengambilan data menggunakan UAV sudah semakin sering dilakukan seiring perkembangan teknologi pada UAV dan meningkatnya kebutuhan pengambilan data dari tempat yang sulit dijangkau.

Salah satu instrumen yang populer digunakan pada UAV adalah kamera dengan resolusi tinggi. Kamera pada UAV ini dapat mengambil foto atau video yang disebut citra UAV. Citra UAV dapat diimplementasikan dalam berbagai hal. Pemantauan dan pengindraan jarak jauh [1] [2], mitigasi, dan pemetaan pasca bencana merupakan macam implementasi yang sering melibatkan citra dari UAV [3] [4]. Citra UAV memiliki berbagai karakteristik diantaranya (i) *ultra-high spatial resolution* [5], (ii) dipengaruhi oleh kondisi cuaca, dan (iii) dipengaruhi oleh ketinggian.

Model jaringan syaraf tiruan dapat membantu implementasi penggunaan citra UAV [6]. Model jaringan syaraf tiruan dapat dibentuk untuk mengenali objek-objek. Dari berbagai model pengenalan objek yang tersedia, *EfficientDet* lebih ringan secara komputasi dibandingkan model pengenalan objek yang lain [7]. Model jaringan syaraf tiruan khususnya pengenalan objek kendaraan dan manusia yang tersedia tidak dilatih menggunakan citra UAV, maka model yang tersedia tidak dihadapkan dengan berbagai kondisi unik dari karakteristik citra UAV, dan akan menghasilkan prediksi yang buruk. Pada penelitian sebelumnya, [8] Zhu, dkk. merangkum berbagai model deteksi yang telah dilatih kembali menggunakan citra UAV,

namun berbagai model tersebut memakan beban komputasi yang tinggi. Bila model deteksi objek yang sudah ada dilatih kembali menggunakan citra UAV, maka akan memakan beban komputasi yang tinggi karena tingginya resolusi dari citra UAV. Walau pada *EfficientDet* beban komputasi dapat ditekan, penelitian sebelumnya menggunakan sistem komputasi performa tinggi yang tidak terjangkau. Hal ini memungkinkan terjadinya penurunan performa saat model dilatih menggunakan sistem komputer konvensional. Maka dari itu penulis ingin meneliti peforma model pengenalan objek dengan beban komputasi rendah *EfficientDet* pada citra UAV.

1.2 Rumusan Masalah

Adapun permasalahan yang terjadi yaitu belum ada penelitian mengenai performansi model deteksi objek dengan sistem komputer konvensional menggunakan *Efficientdet* pada citra UAV.

1.3 Tujuan dan Manfaat

Tujuan dari Tugas Akhir ini adalah menimplementasi dan mananlisis performansi model deteksi objek *Efficientdet* pada citra UAV. Adapun manfaat dalam Tugas Akhir ini adalah:

1. Meningkatkan performa model deteksi objek rendah komputasi *Efficientdet* pada citra UAV
2. Dapat mengimplementasikan model deteksi objek *Efficientdet* untuk menge- nali berbagai objek pada citra UAV.

1.4 Batasan Masalah

Batasan masalah untuk membatasi penelitian ini adalah sebagai berikut:

1. Simulasi dan *retrain* menggunakan komputer jinjing (*laptop*) pribadi dengan spesifikasi:

- GPU Nvidia GeForce GTX 1660 Ti *with Max-Q Design* dengan memori sebesar 4 GB GDDR6 terdedikasi
- Prosesor 2,6 GHz (4,5 GHz turbo) Intel Core i7-9750H
- RAM 16 GB, DDR4 2933 MHz *dual channel*

2. Objek yang dideteksi dalam penelitian ini meliputi:

- pejalan kaki
- manusia
- sepeda
- mobil
- minibus
- bus
- truk
- sepeda motor
- sepeda motor roda tiga, dan
- sepeda roda tiga

3. Himpunan data yang digunakan dalam penelitian ini adalah *VisDrone* versi 2019 [8].

1.5 Metode Penelitian

Metode penelitian yang diterapkan dalam penyelesaian Tugas Akhir ini dengan melakukan simulasi, perhitungan *average recall* (AR) dan perhitungan *average precision* (AP) yang menunjukkan ketepatan model mendeteksi objek pada berbagai gambar yang diujicobakan. Semakin tinggi AP dan AR, semakin baik model deteksi objek.

1.6 Sistematika Penulisan

- **BAB I PENDAHULUAN**

Bab ini membahas motivasi utama penelitian ini, penelitian-penelitian lain terkait yang sudah pernah dilakukan sebelumnya, serta tujuan dari penelitian ini.

- **BAB II DASAR TEORI**

Bab ini membahas landasan teori dan literatur yang digunakan dalam proses penelitian analisis performansi dan implementasi model deteksi objek *EfficientDet* pada citra UAV.

- **BAB III PERANCANGAN SISTEM**

Bab ini berisi tahapan-tahapan yang dilakukan dalam proses penelitian berupa diagram alir penelitian, parameter yang menjadi referensi penetian, dan desain rancangan setiap skenario.

- **BAB IV ANALISIS SIMULASI SISTEM**

Bab ini berisi pembahasan hasil dari nilai *Average Recall (AR)* dan *Average Precision (AP)* setiap variasi skenario. Pada bab ini juga disertakan grafik untuk mempermudah proses analisis.

- **BAB V KESIMPULAN DAN SARAN**

Bab ini berisi kesimpulan dan saran Tugas Akhir untuk pengembangan selanjutnya.

BAB II

DASAR TEORI

2.1 Citra digital dengan kanal warna RGB

Dalam penelitian ini, citra UAV yang diteliti berbentuk citra digital dengan kanal warna merah, hijau, biru (*Red, Green, Blue* atau *RGB*). Citra digital dengan kanal warna *RGB* merupakan larik (*array*) dua dimensi (matriks) yang pada masing-masing elemennya memiliki tiga nilai warna, yaitu merah, hijau, dan biru. Setiap nilai warna merepresentasikan tingkat kekuatan warna pada titik tersebut. Contohnya bila pada sebuah titik memiliki nilai (205, 100, 50), berarti merah mendominasi warna pada titik tersebut, lalu diikuti warna hijau dan biru. Dapat dilihat pada Gambar 2.1, warna yang dihasilkan didominasi merah. Warna yang dihasilkan pada Gambar 2.1 tidak sepenuhnya merah, melainkan jingga. Hal ini disebabkan karena walaupun warna merah mendominasi dengan nilai 205, warna-warna lain (hijau dan biru) masih memiliki nilai dan memberikan kontribusi dalam campuran warna. Nilai warna citra pada penelitian ini bernilai 24 bit sehingga memiliki rentang 0 sampai 255 untuk masing-masing warna. Sehingga titik dengan nilai warna (0,0,0) berwarna hitam dan titik bernilai (255,255,255) berwarna putih.



Gambar 2.1. Contoh warna yang dihasilkan bila sebuah titik memiliki nilai *RGB* (205,100,50) (warna yang dihasilkan mungkin kurang akurat karena keterbatasan percetakan)

Suatu titik pada matriks citra disebut piksel, sehingga dapat dikatakan bahwa citra digital merupakan gabungan dari banyak piksel yang berbentuk matriks dua dimensi, dengan kedalaman kanal warna yang bervariasi. Keuntungan menggunakan skema RGB dibanding skema warna lain adalah setiap piksel bisa memiliki warna yang memiliki warna bervariasi dan dinamis namun dapat mudah diinterpretasi secara numerik, sehingga mempermudah pengiriman data citra melalui *pipeline* algoritma deteksi objek. Kekurangan yang umum dari skema RGB adalah kesulitan untuk *scaling* khususnya untuk citra yang tersusun atas piksel [9].

2.2 Citra *Unmanned Aerial Vehicle* (UAV)

Citra UAV dalam penelitian ini adalah gambar digital dengan kanal warna RGB yang diambil menggunakan kamera resolusi tinggi yang terdapat pada kendaraan udara tanpa awak (*Unmanned Aerial Vehicle* atau UAV). Pengambilan data menggunakan UAV sudah semakin sering dilakukan seiring perkembangan teknologi pada UAV dan meningkatnya kebutuhan pengambilan data dari tempat yang sulit dijangkau. Citra UAV dapat diimplementasikan dalam berbagai hal. Pemantauan dan pengindraan jarak jauh [1] [2], mitigasi, dan pemetaan pasca bencana merupakan macam implementasi yang sering melibatkan citra dari UAV [3] [4]. Citra UAV memiliki berbagai karakteristik diantaranya (i) *ultra-high spatial resolution* [5], (ii) dipengaruhi oleh kondisi cuaca, dan (iii) dipengaruhi oleh ketinggian. Pada penelitian ini, penulis akan menggunakan himpunan data *VisDrone* [8]. Himpunan data ini sebenarnya merupakan sebuah kompetisi untuk peneliti menunjukkan hasil terbaik mereka membuat atau memodifikasi model pengenalan objek untuk citra UAV. Penulis akan melihat dan membandingkan hasil dari penelitian ini dengan publikasi hasil kompetisi pada tahun 2019 [10]. Gambar 2.2 merupakan contoh gambar-gambar yang tersedia pada himpunan data *VisDrone*.



Gambar 2.2. Contoh gambar pada himpunan data *VisDrone*

2.3 Jaringan Syaraf Tiruan (*Artifical Neural Network*)

Jaringan syaraf tiruan adalah salah satu model pembelajaran mesin yang terinspirasi dari cara kerja otak manusia, dimana ribuan dari miliaran sel syaraf (neuron) yang saling terhubung memproses data secara paralel. Para peneliti menirukan neuron dengan membuat *node* titik pemrosesan data, dan memberikan pembebanan di masing-masing hubungan antar *node*, dan bias di masing-masing titik pemrosesan. Biasanya beban (*weight*) menggunakan operasi perkalian, dan bias menggunakan operasi penjumlahan. Jaringan syaraf tiruan terdiri atas lapisan *input*, lapisan tersembunyi (*hidden layer*) setebal satu sampai tiga lapis, dan lapisan *output*. Keluaran dari neuron i adalah h_i dengan persamaan:

$$h_i = \sigma\left(\sum_{j=1}^N V_{ij}x_j + T_i^{hid}\right), \quad (2.1)$$

- dimana $\sigma(\cdot)$ disebut fungsi aktivasi atau fungsi transfer, N adalah jumlah *input* neuron, V_{ij} adalah beban, x_j adalah *input* menuju neuron, dan T_i^{hid} adalah *threshold* dari neuron-neuron yang tersembunyi. Fungsi dari fungsi transfer adalah agar *neural network* dapat mengerti dan menyelesaikan masalah yang non-linier (yang merupakan kelebihan dari model *perceptron*) dan untuk memberikan nilai permulaan kepada neuron sehingga jaringan tidak terpengaruh oleh neuron yang divergen

[11]. Jaringan syaraf tiruan juga dapat disebut *perceptron* multi lapis (*multilayer perceptron*) selama memiliki sifat *fully connected* [12]. Agar menghasilkan hasil prediksi yang optimal, nilai beban dan bias harus selalu disesuaikan. Penyesuaian itu didapat melalui pemrosesan secara berulang (rekursif) yang Watt, dkk [13] rumuskan sebagai berikut:

- Pilih jenis fungsi aktifasi $\sigma(\cdot)$
- Hitung persamaan 2.1
- Bentuk hasil dari persamaan 2.1 dan fungsi $\sigma()$ akan memicu nilai beban dan bias yang lebih baik, dengan melihat seberapa jauh beda hasil prediksi dengan hasil sebenarnya.

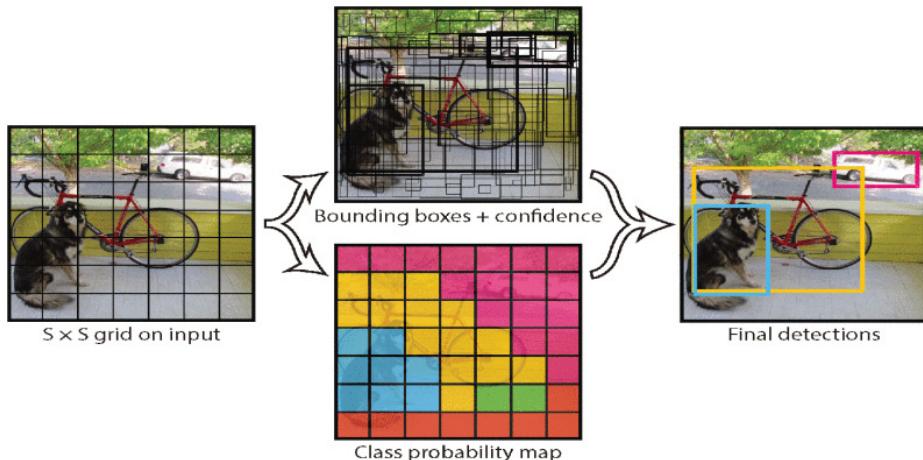
Jaringan syaraf tiruan sudah banyak diterapkan dalam berbagai jenis pengolahan informasi. Informasi yang diolah menggunakan jaringan syaraf tiruan biasanya yang memiliki pola dan kompleksitas tertentu seperti (i) data dalam bentuk tabel yang nantinya menghasilkan prediksi berupa angka (disebut sebagai model regresi) (ii) data (dapat berupa gambar, tabel, dan lainnya) yang nantinya menghasilkan prediksi mengelompokkan masukan ke dalam kelas tertentu (disebut model klasifikasi), (iii) data yang mengandung parameter waktu (disebut data *time series*) serta pola dan nantinya dapat memprediksi seluruh data masukan walaupun data masukan hanya sebagian kecil (disebut *model forecasting*), dan (iv) data berupa kalimat-kalimat yang dapat dicari maknanya (disebut *dataset* kalimat), yang biasanya untuk keperluan pemrosesan bahasa natual (*natural language processing*). Pada penelitian ini penulis akan menggunakan model klasifikasi yang tidak hanya mengelompokkan citra masukan, melainkan memberikan lokasi detil mengenai keberadaan objek di dalam citra. Selanjutnya hal ini disebut deteksi objek.

2.4 Deteksi objek

Deteksi objek adalah sebuah bagian dari pengenalan citra yang lebih kompleks, dimana model tidak hanya menklasifikasikan gambar, namun juga memperkirakan secara tepat kelas dan lokasi dari objek yang terdapat pada sebuah gambar [14]. Tantangan ini sudah mulai dicobakan sejak tahun 1998 melalui penelitian Papageorgiou, dkk. Pada saat itu, dikembangkan teknik deteksi representasi *wavelet* yang dibuat himpunan bagian dan direpresentasi sebagai kelas, serta dibentuk menyerupai model masukan *support vector machine* [15]. Jaringan syaraf tiruan dengan pembelajaran yang dalam (*deep neural network*, yang selanjutnya akan disebut *deep learning*) telah menunjukkan hasil yang luar biasa dalam tantangan klasifikasi, dengan jaringan yang diberi nama *ImageNet* [16]. Hal ini menunjukkan potensi *deep learning* menyelesaikan tantangan selanjutnya yaitu deteksi objek. *ImageNet* secara tidak langsung menjadi populer sebagai *baseline* keberhasilan atau seberapa canggih sebuah model *deep learning* untuk gambar. Szegedy, dkk. dalam penelitian mereka [14] membuktikan bahwa *deep learning* tidak hanya mampu mempelajari fitur-fitur dari sebuah gambar untuk keperluan klasifikasi, namun juga mampu menangkap informasi geometrik yang kuat. Informasi geometrik diperlukan untuk memastikan keberadaan sebuah objek dalam gambar. Hal ini meningkatkan potensi *deep learning* untuk keperluan deteksi objek.

Melihat potensi ini, banyak peneliti tertarik mengembangkan sistem *deep learning* untuk pengenalan objek. Menurut Pathak, dkk. [17] sampai tahun 2018 sudah ada 14 kerangka kerja (*frameworks*), 6 layanan berbasis komputasi awan, dan banyak sekali model *deep learning* yang ditujukan sebagai detektor objek. Tidak terhitungnya jumlah model dikarenakan setiap peneliti atau bahkan insiyur dapat menyesuaikan lapisan-lapisan sesuai kebutuhan mereka. Ketika lapisan sudah ada yang berbeda, itu dapat disebut sebagai model *deep learning* yang baru. Kelemahan dari berbagai model yang sudah tersedia populer atau dikembangkan secara individu sebagian besar datang dari harga komputasi. Pada umumnya model *deep learning*

untuk deteksi objek mahal untuk dilatih karena membutuhkan *Floating Point Operations* (FLOP) yang tinggi. Hal ini dapat diatasi dengan *pretrained model*, yaitu model yang sudah dilatih sebelumnya menggunakan komputer yang sangat tinggi kemampuan komputasinya (*mainframe*, superkomputer, dan sejenisnya) menggunakan sebuah himpunan data tertentu (contoh: MS-COCO, himpunan data untuk ImageNet, CIFAR-100). Contoh jaringan yang memiliki akurasi tinggi namun cukup berat untuk dilatih kembali adalah YOLO (*You Only Look Once*)[18] yang cara kerjanya digambarkan pada Gambar 2.3. *Pretrained model* dapat digunakan langsung untuk aplikasi, karena sudah memiliki beban dan bias di masing-masing neuronnya. *Pretrained model* perlu dilatih kembali bila aplikasi model berbeda dengan jenis himpunan data pada proses *training* sebelumnya. Hal ini kembali mengungkit masalah beban komputasi yang tinggi, walaupun tidak setinggi sebelumnya.

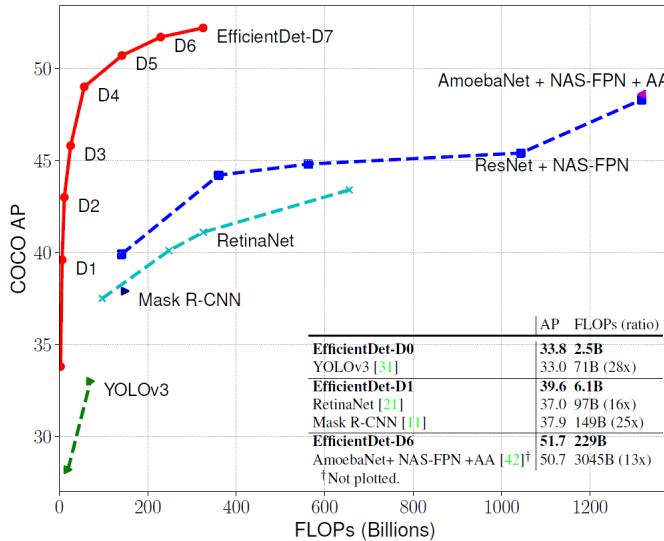


Gambar 2.3. Cara kerja salah satu model deteksi objek (YOLO)

2.5 EfficientDet

EfficientDet adalah salah satu model deteksi objek yang dikembangkan oleh Tan, dkk. yang memiliki keunggulan di bagian efisiensi model [7]. Model ini begitu menarik karena versi D7 nya dapat meraih akurasi yang mutakhir (*state-of-the-art*) sebesar 52.2% presisi rerata (*Average Precision (AP)*) pada himpunan data COCO *test-dev* dengan hanya 325 juta FLOP. Model *EfficientDet* berukuran 4-9 kali le-

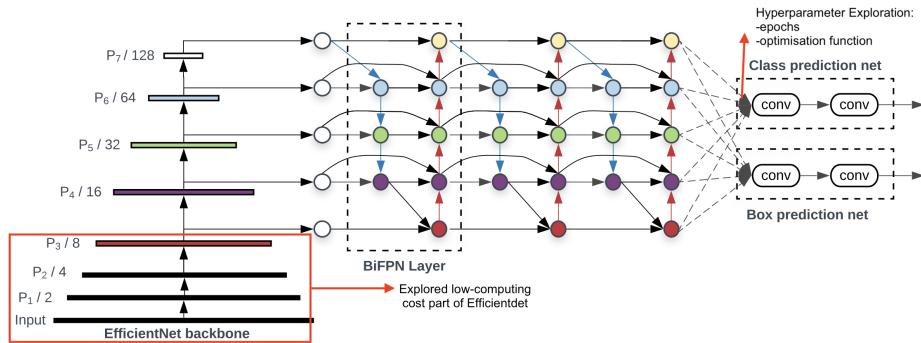
bih kecil dibanding pendahulunya. Walaupun 325 juta FLOP terlihat besar, model ini jumlah FLOP nya 13-42 kali lebih rendah dibandingkan dengan model-model deteksi objek *state-of-the-art* pendahulunya, salah satunya *AmoebaNet-based NAS-FPN detector*[7]. Gambar 2.4 menggambarkan hubungan performansi pada sumbu Y terhadap FLOP pada sumbu X [7].



Gambar 2.4. Grafik kinerja dan FLOP dari berbagai model deteksi objek termasuk *EfficientDet*

Model ini menggunakan jaringan piramida dua arah untuk melakukan ekstraksi ciri. Model piramida ini terdiri atas 8 tingkatan, dimana piramida dua arah (Bi-FPN) yang diadaptasi dari Chen, dkk.[19] diterapkan dari tingkatan ketiga sampai ketujuh. Jaringan piramida ini diperuntukan untuk membuat peta fitur yang lebih kecil namun lebih kaya akan informasi. Satu tingkatan piramida menuju lainnya dihubungkan menggunakan operasi konvolusi, sehingga ketika berpindah tingkat-an, matriks peta fitur menjadi lebih kecil dimensinya, namun semakin padat informasinya. Selain itu, saat bergerak dari satu tingkatan menuju yang lainnya diberi pembebanan, artinya saat proses pembelajaran, jaringan *EfficientDet* akan mempelajari tingkatan mana yang bobotnya tinggi dan rendah. Sehingga bisa meningkatkan presisi melalui pemberian bobot yang lebih tinggi kepada tingkatan yang menghasilkan peta fitur yang representatif terhadap hasil prediksi. Pada penelitian ini, akan

di eksplorasi model *Efficientdet* yang berbasis biaya komputasi rendah. Semakin rendah tingkatan dalam jaringan piramida, semakin rendah pula biaya komputasi yang dibutuhkan. Maka dari itu penelitian ini akan menggunakan *Efficientdet* versi terendah yaitu d0. Melalui penelitian ini akan diketahui pengaruh fungsi optimasi Adam [20] dan *Stochastic Gradient Descend* atau SGD [21]. Rincian bagian dari model yang akan di eksplorasi tertera pada gambar 2.5.



Gambar 2.5. Bagian dari model *EfficientDet* yang akan di eksplorasi

2.6 Fungsi optimasi

Fungsi optimasi berfungsi untuk menyesuaikan kembali parameter-parameter dalam model seperti bias, dan *weight* sehingga *loss* dapat dibuat sekecil-kecilnya dan pada akhirnya meningkatkan kinerja model. Dalam penelitian ini, digunakan dua macam fungsi optimasi. Adam [20] dan *Stochastic Gradient Descend* atau SGD [21].

2.6.1 Adam

Nama fungsi optimasi Adam diambil dari algoritma bagaimana fungsi ini bekerja. Nama Adam diturunkan dari '*adaptive moment estimation*' yang memiliki arti 'prakiraan momen secara adaptif'. Adam diperkenalkan oleh Diederik Kingma dan Jimmy Ba pada tahun 2015. Seperti pendahulunya, Adam adalah algoritma optimasi tingkat satu (*first-order*) berbasis gradien. Adam memiliki berbagai keunggulan yang cocok dengan penelitian ini seperti penggunaan memori yang rendah dan efisi-

en secara komputasi. Perbedaan Adam dengan algoritma optimasi berbasis gradien lainnya adalah *learning rate* yang adaptif untuk parameter yang berbeda. Algoritma Adam seperti menggabungkan keuntungan dari dua fungsi optimasi berbasis gradien lainnya, yaitu *Adaptive Gradient Algorithm* dan *Root Mean Square Propagation* [22]. *Adaptive Gradient Algorithm* atau juga disebut propagasi momentum dapat meningkatkan kinerja model khususnya pada model pengolahan citra dengan mengambil rata-rata tertimbang eksponensial (*exponentially weighted average*) dari gradien pada iterasi sebelumnya ke dalam perhitungan. Gradien merupakan pembagian antara fungsi turunan *loss* dan fungsi turunan *weight* pada iterasi tertentu. Perhitungan propagasi momentum untuk mengubah *learning rate* menjadi adaptif dapat dijelaskan melalui persamaan:

$$W_{i+1} = W_i - \alpha m_i \quad (2.2)$$

nilai m_i ditentukan melalui persamaan:

$$m_i = \beta m_{i-1} + (1 - \beta) \left[\frac{\delta L}{\delta W_i} \right] \quad (2.3)$$

dimana:

- m_i adalah agregat dari gradien pada iterasi i (iterasi saat ini, pada awal iterasi $m_i = 0$),
- m_{i-1} adalah agregat gradien pada iterasi sebelumnya,
- W_i adalah nilai *weight* pada iterasi sekarang,
- W_{i+1} adalah nilai *weight* pada iterasi mendatang yang akan dihitung,
- α adalah *learning rate*,
- δL adalah turunan dari *loss function*,

- δW_i adalah turunan dari *weight* pada iterasi i, dan
- β disebut 'momentum', yaitu konstanta yang menentukan kecepatan perubahan *learning rate*, biasanya momentum bernilai 0,9.

Root Mean Square Propagation atau RMSProp bertujuan untuk memutakhirkan algoritma propagasi momentum. Berdasarkan persamaan yang sama dengan persamaan 2.2, RMSProp mengambil nilai rata-rata bergerak eksponensial (*exponential moving average*). Secara sederhana, RMSProp menambah operasi aritmatika dengan tujuan untuk mengutamakan hasil rerata dari *weight* terakhir. Hal ini bertujuan untuk meminimalisir kemungkinan *learning rate* tidak berubah signifikan dari satu iterasi ke iterasi lain. *learning rate* yang stagnan disebabkan oleh nilai m_i yang terlalu rendah pada persamaan 2.3. *Learning rate* yang kurang adaptif akan membuat proses pelatihan lebih mudah terjebak dalam nilai *loss* minimum lokal. Algoritma RMSProp memodifikasi persamaan 2.2 menjadi:

$$W_{i+1} = W_i - \frac{\alpha_i}{\sqrt{v_i + \epsilon}} * [\frac{\delta L}{\delta W_i}] \quad (2.4)$$

v_i adalah penjumlahan kuadrat (*exponential moving average*) dari semua gradien pada iterasi sebelumnya ($\sum_a^{i-1} \frac{\delta L}{\delta W_a}$) atau dapat dijabarkan sebagai:

$$v_i = \beta v_{i-1} + (1 - \beta) * [\frac{\delta L}{\delta W_i}]^2 \quad (2.5)$$

dimana:

- W_i adalah *weight* pada iterasi i (saat ini),
- W_{i+1} adalah *weight* pada iterasi i+1 (iterasi yang akan datang),
- α_i adalah *learning rate* pada iterasi i,
- δL adalah turunan dari *loss function*,

- δW_i adalah turunan dari *weight* pada iterasi i,
- v_i adalah penjumlahan kuadrat dari semua gradien pada iterasi sebelumnya,
- ϵ adalah konstanta positif yang bernilai 10^{-8} , berfungsi mencegah α_i dibagi nol, dan
- β disebut 'momentum', yaitu konstanta yang menentukan kecepatan perubahan *learning rate*, biasanya momentum bernilai 0,9.

Adam menggabungkan keunggulan dari dua algoritma di atas. Adam mempercepat laju penurunan gradien selayaknya algoritma momentum. Algoritma Adam mengadaptasi cara algoritma RMSProp untuk mengontrol penurunan gradien sehingga osilasi dapat ditekan ketika mencapai minimum global. Namun algoritma Adam dapat mengambil langkah yang cukup besar (*learning rate*) sehingga tidak terjebak minimum lokal. Sehingga, penggabungan dari algoritma momentum dan RMSProp akan efektif untuk membuat algoritma optimasi gradien yang cepat namun dapat dikontrol dengan baik. Mengambil dari persamaan 2.3 dan 2.5, bila inisialisasi nilai m_i dan v_i adalah nol, maka nilai β untuk persamaan 2.3 (β_1) dan 2.5 (β_2) akan mendekati 1. Hal ini membuat kedua fungsi menjadi bias mendekati nilai 0. Dalam implementasinya, Adam menerapkan perhitungan koreksi bias untuk menanggulangi masalah ini. Implementasi persamaan 2.3 dan 2.5 dalam optimasi Adam dijelaskan melalui persamaan:

$$\hat{m}_i = \frac{m_i}{1 - \beta_1^i}, \hat{v}_i = \frac{v_i}{1 - \beta_2^i} \quad (2.6)$$

Dengan memasukkan persamaan 2.6 ke persamaan awal 2.4, maka didapat persamaan baru:

$$W_{i+1} = W_i - \hat{m}_i \left(\frac{\alpha_i}{\sqrt{\hat{v}_i + \epsilon}} \right) \quad (2.7)$$

Dalam publikasi mereka, Diederik Kingma dan Jimmy Ba menyatakan detil fungsi optimasi Adam dalam bentuk algoritmik sebagai berikut [20]:

```

Require:  $\alpha$ : Stepsize
Require:  $\beta_1, \beta_2 \in [0, 1]$ : Exponential decay rates for the moment estimates
Require:  $f(\theta)$ : Stochastic objective function with parameters  $\theta$ 
Require:  $\theta_0$ : Initial parameter vector
 $m_0 \leftarrow 0$  (Initialize 1st moment vector)
 $v_0 \leftarrow 0$  (Initialize 2nd moment vector)
 $t \leftarrow 0$  (Initialize timestep)
while  $\theta_t$  not converged do
     $t \leftarrow t + 1$ 
     $g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$  (Get gradients w.r.t. stochastic objective at timestep  $t$ )
     $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$  (Update biased first moment estimate)
     $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$  (Update biased second raw moment estimate)
     $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$  (Compute bias-corrected first moment estimate)
     $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$  (Compute bias-corrected second raw moment estimate)
     $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$  (Update parameters)
end while
return  $\theta_t$  (Resulting parameters)

```

2.6.2 Stochastic Gradient Descent

(SGD) Algoritma-algoritma yang dijelaskan sebelumnya merupakan pembaharuan dari SGD, atau berlandaskan pada penurunan gradien. Kata '*stochastic*' berarti sistem atau proses yang terkait dengan probabilitas acak. Oleh karena itu, dalam SGD, beberapa sampel dipilih secara acak, bukan seluruh kumpulan data untuk setiap iterasi. Dalam *Gradient Descent* biasa, ada istilah yang disebut "*batch*" yang menunjukkan jumlah total sampel dari dataset yang digunakan untuk menghitung gradien pada setiap iterasi. Dalam optimasi *Gradient Descent* yang biasa, seperti *Batch Gradient Descent*, satu bagian *batch* dianggap sebagai keseluruhan dataset. Meskipun menggunakan seluruh dataset sangat berguna untuk mencapai minimum dengan cara yang mengurangi osilasi dan tidak terlalu acak, masalah muncul ketika dataset menjadi semakin besar. Masalah ini diselesaikan dengan SGD. Dalam SGD, hanya menggunakan satu sampel, yaitu ukuran *batch*satu, untuk melakukan setiap iterasi. Sampel diacak dan dipilih setiap melakukan iterasi. Dalam SGD, karena hanya satu sampel dari kumpulan data yang dipilih secara acak untuk setiap iterasi, jalur yang diambil oleh fungsi SGD untuk mencapai *loss* minimum biasanya ber-*osilasi* lebih keras (seperti berjalur zig-zag) daripada algoritma *Gradient Descent*

biasa. Namun osilasi yang tinggi tidak menjadi masalah berarti karena SGD memiliki waktu yang lebih singkat dalam menemukan *global minima* [23]. Karena SGD berosilasi lebih tinggi dibanding *Gradient Descent* lain, terkadang SGD memerlukan iterasi yang lebih tinggi untuk menemukan *global minima*, dan lebih rawan untuk terjebak di *local minima*. Algoritma SGD dirincikan sebagai berikut:

Require: Learning rate ϵ_k .
Require: Initial parameter $\boldsymbol{\theta}$

while stopping criterion not met **do**

Sample a minibatch of m examples from the training set $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ with corresponding targets $\mathbf{y}^{(i)}$.

Compute gradient estimate: $\hat{\mathbf{g}} \leftarrow +\frac{1}{m} \nabla_{\boldsymbol{\theta}} \sum_i L(f(\mathbf{x}^{(i)}; \boldsymbol{\theta}), \mathbf{y}^{(i)})$

Apply update: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \epsilon \hat{\mathbf{g}}$

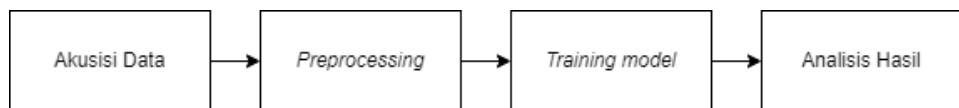
end while

BAB III

PERANCANGAN SISTEM

3.1 Rancangan Sistem

Pada tugas akhir ini, sistem dirancang untuk mendekripsi objek-objek pada citra UAV. Sistem menggunakan model jaringan syaraf tiruan *EfficientDet-D0*. Gambaran secara umum rancangan sistem dapat dilihat pada Gambar 3.1.



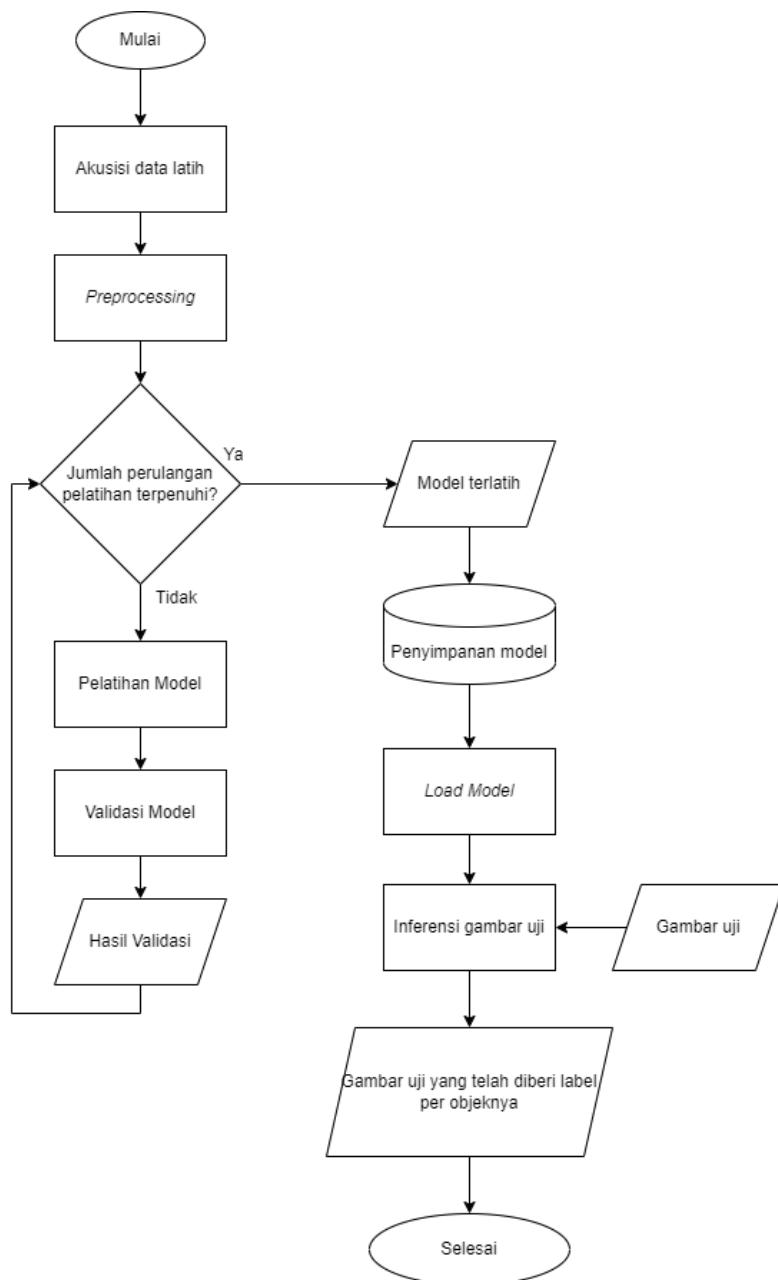
Gambar 3.1. Diagram blok rancangan sistem.

Secara keseluruhan, ada empat tahapan pada sistem, yaitu:

1. Akuisisi data, yaitu proses mendapatkan citra-citra UAV dari dataset VisDrome.
2. *Preprocessing*, yaitu tahap untuk mengolah dan menyesuaikan dataset sehingga dapat dilatih menggunakan *EfficientDet*.
3. *Training Model*, merupakan proses pelatihan model untuk mengenali objek-objek pada citra UAV.
4. Analisis Hasil, merupakan tahap untuk mengukur performa deteksi menggunakan parameter *average precision* dan *average recall*.

Dalam prosesnya, tugas akhir ini menggunakan satu proses linear dari pelatihan dan validasi. Tidak seperti tugas akhir lain yang proses pelatihan dan validasinya terpisah, pada tugas akhir ini validasi langsung dilakukan setelah pelatihan berakhir. Proses validasi akan menghasilkan nilai yang menjadi tolak ukur kinerja dari model

deteksi objek. Proses pelatihan dan validasi dilakukan berulang. Nilai validasi yang akan ditinjau adalah nilai validasi terakhir yang dihasilkan model. Setelah perulangan proses latih dan validasi, model yang sudah dilatih akan disimpan lalu dilakukan proses inferensi. Proses inferensi artinya memberi label objek-objek yang terdeteksi model pada citra. Proses pelatihan dan validasi dapat dilihat pada Gambar 3.2.



Gambar 3.2. Diagram alir sistem.

3.1.1 Akuisisi Data

Data yang digunakan dalam tugas akhir adalah himpunan data VisDrone yang tersedia secara umum. Citra yang tersedia menggunakan format .jpg dan mempunyai ukuran 960×540 piksel. Terdapat 10 kelas objek yang dapat dideteksi. 10 kelas tersebut adalah pejalan kaki, manusia, sepeda, mobil, minibus, bus, truk, sepeda motor, sepeda motor roda tiga, dan sepeda roda tiga. Contoh bagaimana masing-masing kelas terlihat pada citra tertera pada gambar 3.3 Jumlah gambar yang digunakan sebagai data latih sebanyak 6471 gambar. Data validasi yang digunakan sebanyak 548 gambar.



Gambar 3.3. Kelas-kelas dalam himpunan data *Visdrone*

3.1.2 Preprocessing

Proses *preprocessing* pada tugas akhir ini hanya terdiri dari satu tahap, yaitu mengubah format anotasi (catatan mengenai lokasi berbagai objek pada masing-masing gambar) menjadi *tfrecord* dari format PASCAL VOC. Konversi ini diperlukan karena model deteksi *EfficientDet* pada tugas akhir ini menggunakan bantuan kerangka kerja *automl* yang dapat membaca format anotasi *tfrecord* secara bawaan.

3.1.3 Pelatihan Model

Model *EfficientDet-D0* dilatih menggunakan data VisDrone dengan perubahan anotasinya menjadi *tfrecord*. Dalam satu iterasi pelatihan, model memproses se-

banyak 5696 gambar yang dipilih secara acak dari total 6471 gambar sebagai data latih. Selanjutnya, iterasi pelatihan akan disebut dengan *epoch*. Satu *epoch* diraih dengan 6500 proses kalkulasi atau yang selanjutnya disebut dengan *step*. Setelah dilatih, model akan langsung dinilai performanya menggunakan data validasi. Sebagai upaya peningkatan performa. Selanjutnya model akan di latih kembali dengan 5696 dari 6471 gambar yang diacak kembali. Setelah melalui beberapa proses perulangan latih-validasi, proses latih akan dihentikan, model disimpan untuk inferensi, dan skor validasi terakhir akan disimpan sebagai tolak ukur performa.

3.1.4 Analisis Hasil

Setelah proses pelatihan terakhir, kita akan mendapatkan skor validasi. Skor validasi ini terdiri atas *average precision* dan *average recall*. Dua parameter ini akan menjadi tolak ukur performa model. Analisis dilakukan dengan menggunakan bantuan diagram batang atau kurva dengan parameter *average precision* atau *average recall* pada sumbu tegak, dan jenis fungsi optimasi serta jumlah *epochs* pada sumbu datar.

3.2 Performansi Sistem

Seperti model deteksi objek lainnya, pada tugas akhir ini, performansi sistem diukur menggunakan *average precision* (AP) dan *average recall* (AR). Hasil skor validasi terdiri atas 6 macam AP dan 6 macam AR. Berbagai macam AP dan AR ini dipengaruhi area deteksi (kecil, sedang, besar), *Intersection over Union* (IoU), dan jumlah deteksi objek maksimum. Pada tugas akhir ini, mengikuti beberapa penelitian yang menggunakan himpunan data serupa, akan hanya diambil 6 nilai, yaitu AP untuk area besar, AP untuk nilai IoU 0.5, AP untuk nilai IoU 0.75, AR dengan maksimal deteksi 1, 10, dan 100.

3.2.1 Average Precision (AP) dan Average Recall (AR)

Average precision atau presisi rata-rata artinya sebuah nilai yang menunjukkan seberapa tepat sebuah model mendekripsi objek dalam menentukan objek. Nilai AP diadpat dari sebuah fungsi yang membandingkan kotak penanda objek pada anotasi *tfrecord* dengan kotak penanda objek yang dihasilkan model menggunakan IoU. Nilai rata-rata AP didapat melalui persamaan:

$$\frac{1}{n} \sum_{k=1}^{k=n} AP_k, \quad (3.1)$$

- dimana n adalah jumlah kelas dan AP_k adalah AP dari masing-masing kelas. Setelah dibandingkan, akan didapat nilai IoU dan selanjutnya ditentukan apakah model dinyatakan berhasil atau gagal dalam menentukan lokasi objek melalui suatu am-bang batas nilai IoU. *Average recall* artinya sebuah nilai yang menunjukkan seberapa akurat sebuah model mendekripsi objek dalam melakukan deteksi. Metode yang digunakan hampir sama dengan saat mencari AP, namun AR lebih berfokus pada jumlah objek yang terdeteksi tepat (*True Positive, False Negative*), dan AP berfokus pada ketepatan model saat objek diberi label tertentu (*True Positive, False Positive*).

Nilai AR rata-rata didapat melalui persamaan:

$$\frac{1}{n} \sum_{k=1}^{k=n} AR_k, \quad (3.2)$$

- dimana n adalah jumlah kelas dan AR_k adalah AR dari masing-masing kelas objek yang di deteksi.

3.3 Sistematika Data

Berbagai gambar yang ada pada himpunan data *VisDrone* diambil dengan latar belakang urban. Seluruh citra diambil di daerah perkotaan, dengan berbagai ken-daraan, jalan raya, lahan parkir, bangunan, fasilitas publik, taman, kolam renang,

lapangan, dan lain sebagainya fasilitas yang umumnya dapat ditemukan di sebuah kota. Waktu saat pengambilan gambar bervariasi dari pagi hingga malam hari. Seluruh citra merupakan citra UAV yang diambil dari ketinggian. Tidak hanya waktu, kondisi cuaca saat citra diambil juga bervariasi dari cerah sampai hujan yang cukup mengurangi jarak pandang. Berbagai citra ini di sampel dari 14 kota berbeda di seluruh kawasan Republik Rakyat Cina. Himpunan data ini diambil dan dikelola oleh tim AISKEYEYE, Laboratorium Pembelajaran Mesin dan Penambangan Data , Universitas Tianjin, Republik Rakyat Cina.

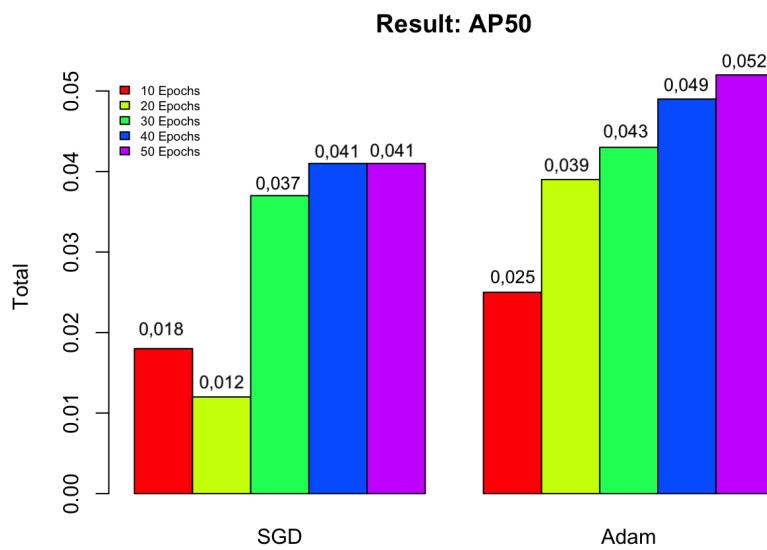
BAB IV

ANALISIS SIMULASI SISTEM

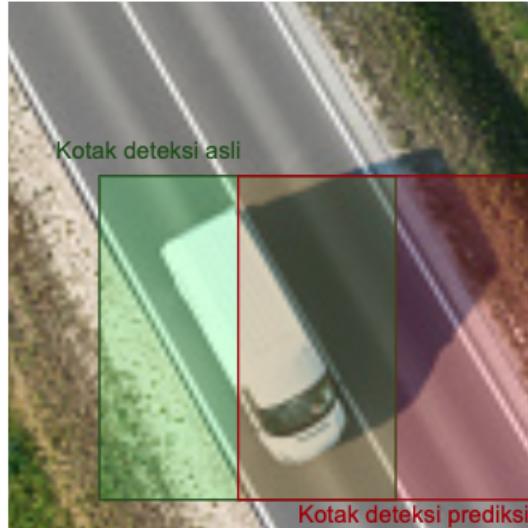
4.1 Analisis dan Hasil Pengukuran *Average Precision*

Model *efficientdet D0* dilatih sebanyak 10 sampai 50 iterasi terlebih dahulu. Performa pada 10 iterasi dilakukan sebagai batas terendah untuk menentukan kecocokan model pada data dan dapat ditentukan dalam waktu yang lebih singkat. Model diukur *average precision*-nya menggunakan data *test* yang disediakan *Videosdrone*, sebanyak 548 citra. Proses pengujian ini menganalisis dua macam fungsi optimasi, yaitu *stochastic gradient descend* atau SGD[21] dan Adam [20]. 10 *epochs* ekuivalen dengan kurang lebih 56 ribu *steps*.

4.1.1 Hasil *Average Precission*



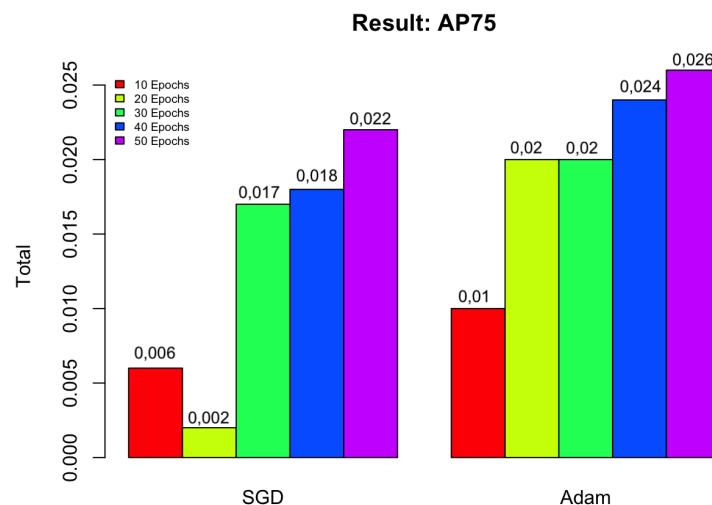
Gambar 4.1. Skor AP50 pada fungsi optimasi SGD dan Adam setelah dilatih 10 sampai 50 *epoch*



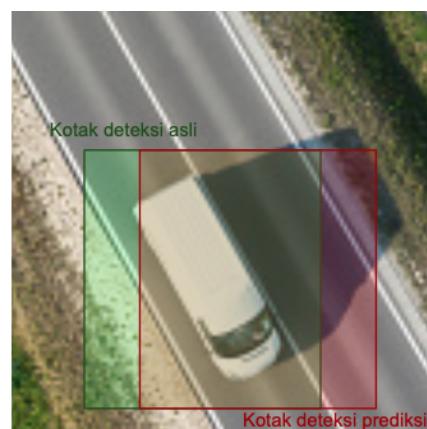
Gambar 4.2. *intesection over union* dari kotak deteksi menutupi 50 persen area deteksi asli.

AP50 adalah nilai presisi rata-rata dari model saat mengenali objek bila *intesection over union* (IoU) dari kotak deteksi menutupi 50 persen atau lebih area deteksi asli. Ilustrasi *intesection over union* untuk pengukuran AP50 tertera pada gambar 4.2. AP50 dari model yang dilatih menggunakan *stochastic gradient descend* sebanyak 10 iterasi menghasilkan 0,018 atau 1,8%. Sementara model yang dilatih menggunakan fungsi optimasi Adam dengan jumlah iterasi yang sama memberikan hasil yang sedikit lebih baik, yaitu di angka 2,5% atau 0,025. Semakin banyak iterasi pelatihan, skor AP50 akan meningkat sampai tertingginya 0,041 untuk fungsi optimasi SGD dan 0,052 untuk fungsi optimasi Adam. Rincian kenaikan skor AP50 tertera pada gambar 4.1. Fungsi optimasi Adam unggul secara general dibandingkan SGD. Keunggulan tersebut dikarenakan pada kasus ini, kemunculan objek pada gambar tidak selalu pada satu tempat yang sama atau berdekatan. Tidak hanya lokasi pada gambar, ukuran objek target deteksi pada tiap gambar juga bervariasi. Keanekaragaman pada gambar ini membuat Adam sebagai fungsi optimasi dengan *learning rate* yang adaptif unggul untuk mencari loss terkecil dibandingkan dengan fungsi SGD yang akan lebih mudah terjebak pada loss minimum lokal (lokal minima) [24][25]. Namun dengan keunggulan Adam tersebut, skor ini belum cukup tinggi untuk setara dengan model pada penelitian sebelumnya. Hal ini disebabk-

an oleh banyak faktor, salah satunya iterasi pelatihan yang masih tergolong rendah dibandingkan dengan model pada penelitian sebelumnya. Berbagai model pada penelitian sebelumnya dilatih dengan iterasi pelatihan yang tinggi, lebih dari 100 kali pelatihan. Pola Kenaikan pada 10 *epoch* pertama menunjukkan kenaikan yang beriringan, menunjukkan bahwa model yang digunakan cocok dengan data. Selanjutnya untuk melihat kemampuan model dalam tingkat presisi yang lebih tinggi, standar IoU ditingkatkan ke 75 persen. Bila kotak deteksi prediksi hanya mampu menutupi kurang dari 75 persen kotak deteksi asli, maka hasil deteksi tidak dihitung.

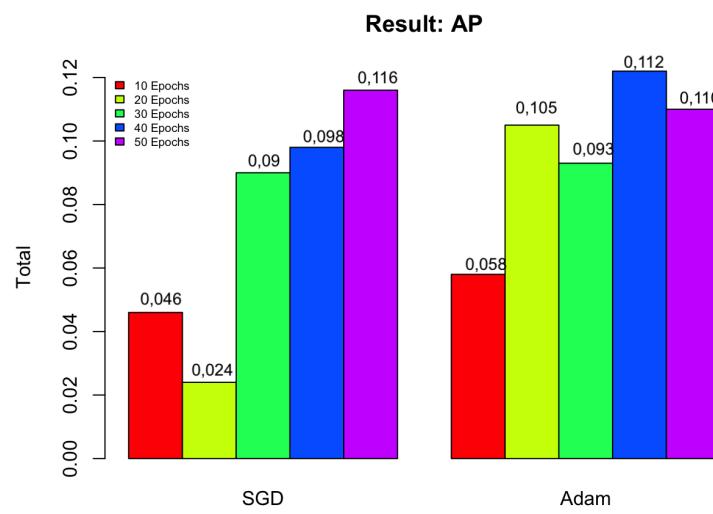


Gambar 4.3. Skor AP75 pada fungsi optimasi SGD dan Adam setelah dilatih 10 sampai 50 *epoch*

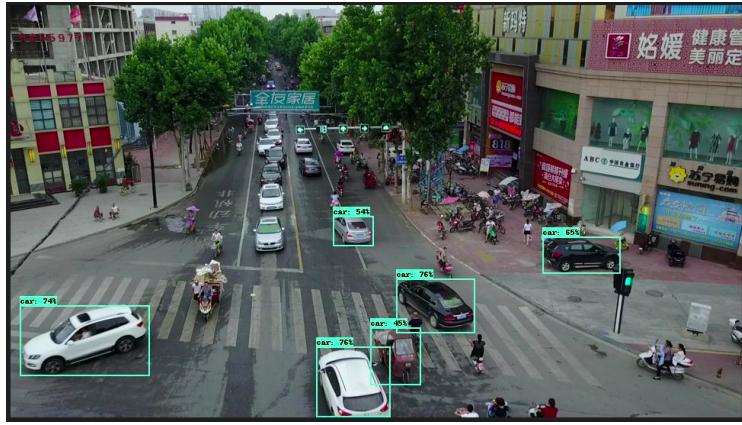


Gambar 4.4. *intesection over union* kotak deteksi prediksi sebesar 75 persen dari kotak deteksi asli.

AP75 adalah nilai presisi rata-rata dari model saat mengenali objek bila *intersection over union* dari kotak deteksi menutupi 75 persen atau lebih area deteksi asli. Ilustrasi *intersection over union* untuk pengukuran AP75 tertera pada gambar 4.4. Untuk nilai AP75, keduanya fungsi optimasi menunjukkan skor yang lebih rendah lagi. Pada 10 iterasi pelatihan, AP75 untuk fungsi optimasi Adam hanya sebesar 0,01 atau 1% dan *stochastic gradient descend* bahkan lebih kecil lagi, sebesar 0,006 atau 0,6%. Hasil AP75 menunjukkan peningkatan saat dilatih sebanyak 50 iterasi. Optimasi Adam pada 50 iterasi mendapat skor 0,026 dan unggul tipis dari fungsi optimasi SGD dengan skor 0,022. Rincian kenaikan skor AP75 tertera pada gambar 4.3. Rendahnya skor AP75 ini disebabkan karena standar minimum terhitungnya deteksi menjadi semakin tinggi, sehingga jumlah deteksi yang berhasil dinilai akan berkurang. Selayaknya bila kita meningkatkan *passing grade* pada suatu ujian, akan semakin sedikit peserta ujian yang lolos dari *passing grade*. Selanjutnya, *average precision* kembali diukur, namun kali ini standarnya adalah menggunakan objek yang ukurannya relatif lebih besar pada citra. Selanjutnya akan disebut AP untuk area besar.



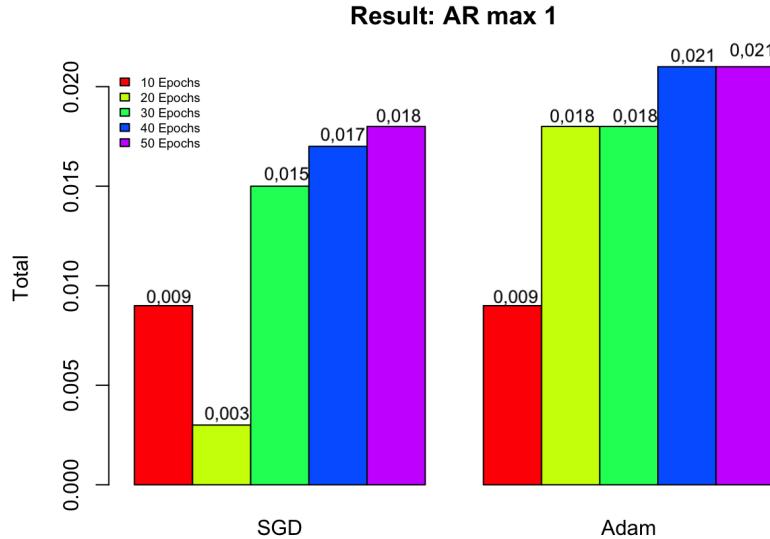
Gambar 4.5. Skor AP pada fungsi optimasi SGD dan Adam setelah dilatih 10 sampai 50 epoch



Gambar 4.6. Contoh hasil inferensi

Average precision kali ini adalah AP untuk area besar. AP area besar diukur dengan cara menghitung presisi deteksi rata-rata objek yang relatif lebih besar dibanding objek lain pada citra, seperti pada gambar 4.6. AP jenis ini dipilih karena menunjukkan performa model mendeteksi objek yang jaraknya relatif dekat. Pengukuran deteksi objek jarak dekat merupakan hal yang penting karena model harus dapat mendeteksi objek yang mendekat dalam waktu yang singkat dan tepat untuk menghindari tabrakan bila UAV terbang secara mandiri. AP untuk area besar menunjukkan hasil yang lebih baik dari kedua jenis fungsi optimasi. Model yang dilatih menggunakan fungsi optimasi Adam selama 10 iterasi mendapat skor 0,058 atau 5,8%. Model dengan *stochastic gradient descend* mendapat skor 0,046 atau 4,6% setelah dilatih selama 10 iterasi. Puncak skor untuk fungsi optimasi Adam diraih setelah melatih model selama 40 iterasi, yaitu sebesar 0,122 atau 12,2%. Puncak skor untuk fungsi optimasi SGD diraih setelah 50 iterasi pelatihan, dengan skor sebesar 0,116 atau 11,6%. Model dengan fungsi optimasi Adam pada 50 iterasi mendapat skor lebih rendah dibandingkan dengan 40 iterasi, namun memiliki waktu inferensi yang lebih cepat. Dapat disimpulkan dari iterasi ke 41 ke atas, model dengan fungsi optimasi Adam mengadaptasi *learning rate*-nya untuk mendeteksi lebih cepat dengan sedikit mengorbankan presisi. Rincian kenaikan skor AP untuk area besar tertera pada gambar 4.5.

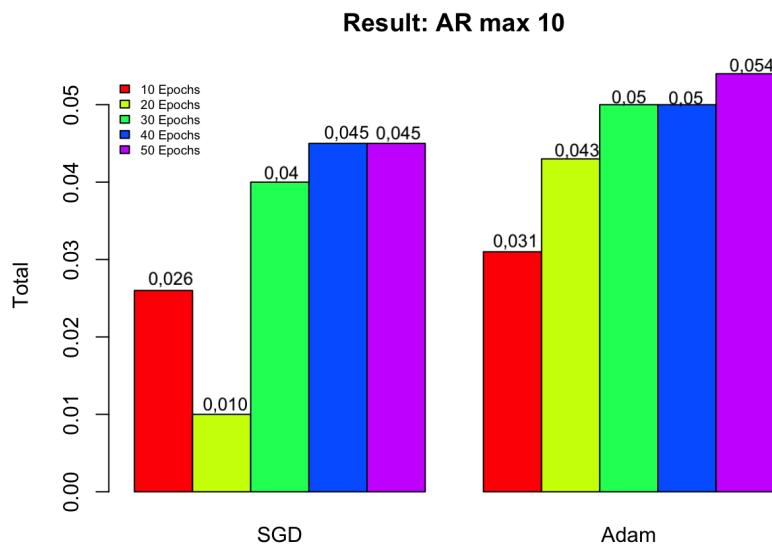
4.1.2 Hasil Average Recall



Gambar 4.7. ARmax1 pada fungsi optimasi SGD dan Adam setelah dilatih 10 sampai 50 epoch

Average recall atau AR mengukur seberapa baik model untuk membedakan jenis objek satu dengan yang lainnya. Semakin tinggi nilai *Average recall* maka deteksi akan semakin baik, serta semakin cepat membedakan jenis-jenis objek. AR yang diukur ada tiga jenis. AR jenis pertama diukur bila model hanya befokus pada satu objek dalam citra, disebut ARmax1. Selanjutnya ambang jumlah deteksi ditingkatkan menjadi 10 dan 100 objek yang disebut ARmax10 dan ARmax100. Rincian kenaikan skor ARmax1 tertera pada gambar 4.7. Dapat dilihat pada gambar 4.7, ARmax1 untuk kedua fungsi optimasi baik SGD maupun Adam mengalami tren peningkatan. Ketika dilatih sebanyak 10 iterasi, kedua fungsi optimasi mendapat skor 0,009 atau 0,9%. Untuk 10 iterasi pertama eksplorasi *Efficientdet* berbasis biaya komputasi rendah sudah mampu mengungguli beberapa model pada penelitian sebelumnya, namun belum menjadi model terbaik dari segi ARmax1. Hasil terbaik dari kedua fungsi optimasi didapat setelah model dilatih sebanyak 50 iterasi. Fungsi optimasi SGD dengan skor 0,018 atau 1,8% dan Adam dengan skor

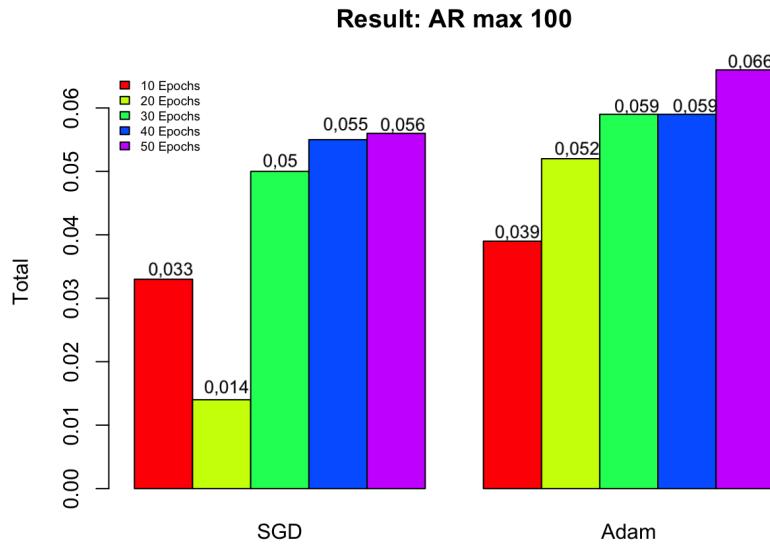
0,021 (2,1%). Pada penelitian sebelumnya, TridentNet menjadi yang paling unggul dari ARmax1 dengan skor 1,17% [10]. Eksplorasi *Efficientdet* berbiaya komputasi rendah dapat mengungguli hasil TridentNet hanya dengan 20 iterasi menggunakan fungsi optimasi Adam dengan skor sebesar 1,8%. Sementara itu, model yang dilatih dengan optimasi SGD dapat mengungguli TridentNet setelah 50 iterasi saja. Ini merupakan sebuah keunggulan mengingat berbagai model pada penelitian sebelumnya berbiaya komputasi tinggi dan dilatih selama ratusan iterasi.



Gambar 4.8. ARmax10 pada fungsi optimasi SGD dan Adam setelah dilatih 10 sampai 50 epoch

Ambang deteksi ditingkatkan menjadi 10 objek per citra. Model dengan kedua fungsi optimasi kembali menunjukkan tren peningkatan. Untuk model dengan fungsi optimasi SGD, nilai terendah berada di 0,01 atau 1% saat dilatih sebanyak 20 iterasi dan terus meningkat sampai iterasi pelatihan ke 40 dengan skor 0,045 atau 4,5%. Dengan nilai ini, model eksplorasi mampu mengungguli model pada penelitian sebelumnya seperti *CenterNet-Hourglass* dan *EnDet* [10]. Walaupun unggul dari beberapa model pada penelitian sebelumnya, skor pada model eksplorasi belum cukup untuk menjadi yang terbaik atau *state-of-the-art* pada dataset ini. Iterasi 20 memiliki skor yang lebih buruk dibanding iterasi 10, ini dikarenakan model

mengorbankan kualitas deteksi untuk mempercepat deteksi objek pada citra atau selanjutnya disebut dengan inferensi. Mengorbankan skor ARmax10 untuk mempercepat waktu inferensi juga terjadi dari iterasi 40 ke 50 pada model yang dioptimasi fungsi SGD. Selanjutnya model eksplorasi menunjukkan peningkatan ketika dilatih menggunakan fungsi optimasi Adam. Nilai terendah pada eksplorasi ini didapat setelah model dilatih sebanyak 10 iterasi dengan skor 0,031 atau 3,1%. Lalu nilai ARmax10 pada model dengan fungsi optimasi Adam meningkat sampai 0,054 atau 5,4% setelah dilatih sebanyak 50 iterasi. Pola yang sama dengan model SGD, model dengan optimasi Adam juga sempat menghasilkan skor yang sama (0,05) pada iterasi 30 dan 40. Ini disebabkan karena saat memasuki iterasi ke 31, model berusaha untuk mengurangi waktu inferensi, namun penurunan skor yang terjadi sangat minim. Minimnya pengurangan skor disebabkan karena fungsi optimasi Adam yang adaptif dengan menyesuaikan *learning rate* saat model berusaha mengurangi waktu inferensi, sehingga waktu inferensi yang lebih rendah dapat tercapai dengan pengurangan performa yang minim [25]. Dengan skor maksimal 0,054 ini, model eksplorasi berhasil mengungguli beberapa model pada penelitian sebelumnya seperti *FS-Retinanet* dan *SGE-cascade R-CNN* hanya dengan 50 iterasi saja [10]. Rincian hasil masing-masing iterasi pada kedua fungsi optimasi dapat dilihat pada gambar 4.8.



Gambar 4.9. ARmax100 pada fungsi optimasi SGD dan Adam setelah dilatih 10 sampai 50 epoch

Pengukuran terakhir yaitu dengan meningkatkan kembali ambang deteksi menjadi 100 objek per citra atau selanjutnya disebut dengan ARmax100. Kedua model kembali menunjukkan tren peningkatan, namun pada ARmax100 kelemahan eksplorasi terlihat. Untuk model dengan fungsi optimasi SGD, nilai terendah berada di 0,014 atau 1,4% saat dilatih sebanyak 20 iterasi dan terus meningkat sampai iterasi pelatihan ke 50 dengan skor 0,056 atau 5,6%. Sementara itu model eksplorasi yang dilatih menggunakan fungsi optimasi Adam mendapat skor terendah sebesar 0,039 atau 3,9% setelah 10 iterasi pelatihan. Model dengan fungsi optimasi Adam terus meningkat sampai 0,066 atau 6,6% pada iterasi ke 50. Kedua hasil ini tidak dapat mengungguli model manapun pada penelitian sebelumnya [10]. Kelemahan ini timbul karena terbatasnya kemampuan komputasi dan jumlah iterasi pelatihan yang relatif sedikit dibandingkan dengan ratusan iterasi pelatihan berbagai model pada penelitian sebelumnya. Model yang dilatih dengan iterasi yang relatif sedikit akan kesulitan untuk mendeteksi lebih dari 10 objek dan membedakan jenis dari objek tersebut. Hal ini dikarenakan untuk mendeteksi banyak objek dalam satu citra dan secara cermat membedakan jenis objeknya dibutuhkan waktu dan biaya

komputasi yang tinggi. Selain itu, karena citra UAV yang digunakan memiliki karakteristik dipengaruhi jarak, objek yang semakin jauh akan terlihat semakin kecil dan mempengaruhi kualitas deteksi. Ditambah dengan karakteristik citra UAV yang memiliki resolusi tinggi, objek yang relatif jauh dari titik pengambilan gambar akan tetap terlihat, namun dibutuhkan biaya komputasi yang lebih tinggi untuk mendeteksi objek tersebut secara mendetail. Rincian skor dari masing-masing iterasi pada kedua model dapat dilihat pada gambar 4.9.

4.2 Pembahasan

Model *EfficientDet D0* yang dilatih dalam sistem komputer pribadi konvensional yang terbatas mampu unggul dari performa *average recall* bila dibandingkan dengan penelitian sebelumnya [10]. Perlu diketahui juga bahwa berbagai model pada penelitian sebelumnya dilatih dalam ratusan iterasi. Model *EfficientDet D0* mampu unggul hanya dengan 20 iterasi atau lebih. Nilai *average recall* penting untuk sebuah model deteksi objek khususnya pada implementasi UAV karena model akan mendeteksi lebih banyak objek dan mengacu pada lokalisasi akurasi [26]. Ketika sebuah model memiliki *recall* yang tinggi namun *precision* yang rendah, model akan mengklasifikasi lebih banyak objek atau sampel dengan tepat, namun akan banyak juga sampel yang teridentifikasi sebagai kelas lain (atau *false positive*) [27]. Dalam kasus deteksi objek pada UAV, memprioritaskan nilai *average recall* diperlukan khususnya pada kasus inferensi langsung saat UAV bergerak dan beroperasi. Prioritas ini dibutuhkan karena untuk UAV akan lebih penting untuk mendeteksi keberadaan objek terlebih dahulu. Prioritas deteksi keberadaan objek yang berarti memprioritaskan nilai *average recall* dan mentoleransi rendahnya nilai *average precision* dapat diterima pada kasus implementasi UAV. Kecepatan deteksi juga menjadi hal penting dalam model deteksi objek pada UAV. Hosang, dkk. dalam artikelnya juga menyebutkan pada model deteksi dengan waktu proses yang lebih rendah, nilai kualitas deteksinya dapat berkurang dan itu dapat ditoleransi [26].

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil simulasi dan analisis dapat ditarik kesimpulan bahwa model *EfficientDet d0* dapat menjadi yang paling unggul atau *state-of-the-art model* dari segi ARmax1. Model eksplorasi dengan fungsi optimasi Adam dapat unggul model dari penelitian sebelumnya dalam 20 iterasi atau lebih walaupun dalam kompleksitas dan kemampuan komputasi perangkat keras yang terbatas. Model *EfficientDet d0* cocok digunakan sebagai model deteksi objek berbiaya komputasi rendah untuk implementasi UAV karena memiliki nilai *recall* yang tinggi. Rendahnya biaya komputasi yang diperlukan akan mempermudah proses pelatihan, meningkatkan kecepatan deteksi pada UAV dan skalabilitas deteksi objek di masa mendatang.

5.2 Saran

Saran yang dapat diterapkan pada penelitian selanjutnya yang berkaitan dengan topik Tugas Akhir adalah terdapat potensi untuk meningkatkan hasil deteksi dari model. Tidak hanya untuk meningkatkan *recall*, namun juga untuk meningkatkan *precision*. Penelitian selanjutnya dapat meningkatkan reliabilitas model secara keseluruhan dengan menerapkan *preprocessing* yang lebih banyak untuk menyederhanakan citra masukan, serta modifikasi lapisan *pooling* pada model untuk meningkatkan kapabilitas model bila dihadapkan dengan citra resolusi tinggi dari UAV.

DAFTAR PUSTAKA

- [1] J. Wu, G. Yang, X. Yang, B. Xu, L. Han, and Y. Zhu, “Automatic counting of in situ rice seedlings from uav images based on a deep fully convolutional neural network,” *Remote Sensing*, vol. 11, no. 6, 2019. [Online]. Available: <https://www.mdpi.com/2072-4292/11/6/691>
- [2] S. Oleire-Oltmanns, I. Marzolff, K. D. Peter, and J. B. Ries, “Unmanned aerial vehicle (uav) for monitoring soil erosion in morocco,” *Remote Sensing*, vol. 4, no. 11, pp. 3390–3416, 2012. [Online]. Available: <https://www.mdpi.com/2072-4292/4/11/3390>
- [3] K. G. Panda, S. Das, D. Sen, and W. Arif, “Design and deployment of uav-aided post-disaster emergency network,” *IEEE Access*, vol. 7, pp. 102 985–102 999, 2019.
- [4] J. M. M. Neto, R. A. da Paixão, L. R. L. Rodrigues, E. M. Moreira, J. C. J. dos Santos, and P. F. F. Rosa, “A surveillance task for a uav in a natural disaster scenario,” in *2012 IEEE International Symposium on Industrial Electronics*, 2012, pp. 1516–1522.
- [5] G.-H. Kwak and N.-W. Park, “Impact of texture information on crop classification with machine learning and uav images,” *Applied Sciences*, vol. 9, no. 4, 2019. [Online]. Available: <https://www.mdpi.com/2076-3417/9/4/643>
- [6] C. Kyrou, G. Plastiras, T. Theocharides, S. I. Venieris, and C.-S. Bouganis, “Dronet: Efficient convolutional neural network detector for real-time uav applications,” in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2018, pp. 967–972.

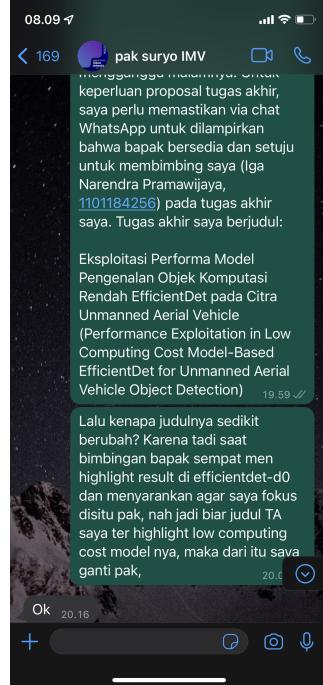
- [7] M. Tan, R. Pang, and Q. V. Le, “Efficientdet: Scalable and efficient object detection,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 781–10 790.
- [8] P. Zhu, L. Wen, D. Du, X. Bian, H. Fan, Q. Hu, and H. Ling, “Detection and tracking meet drones challenge,” 2021.
- [9] T. McREYNOLDS and D. BLYTHE, “Chapter 19 - illustration and artistic techniques,” in *Advanced Graphics Programming Using OpenGL*, ser. The Morgan Kaufmann Series in Computer Graphics, T. McREYNOLDS and D. BLYTHE, Eds. San Francisco: Morgan Kaufmann, 2005, pp. 501–530. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/B9781558606593500214>
- [10] 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.

- [11] S.-C. Wang, “Artificial neural network,” in *Interdisciplinary computing in java programming*. Springer, 2003, pp. 81–100.
- [12] A. V. Joshi, *Machine learning and artificial intelligence*. Springer, 2020.
- [13] J. Watt, R. Borhani, and A. K. Katsaggelos, *Machine learning refined: foundations, algorithms, and applications*. Cambridge University Press, 2020.
- [14] C. Szegedy, A. Toshev, and D. Erhan, “Deep neural networks for object detection,” 2013.
- [15] C. P. Papageorgiou, M. Oren, and T. Poggio, “A general framework for object detection,” in *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*. IEEE, 1998, pp. 555–562.
- [16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.
- [17] A. R. Pathak, M. Pandey, and S. Rautaray, “Application of deep learning for object detection,” *Procedia Computer Science*, vol. 132, pp. 1706–1717, 2018, international Conference on Computational Intelligence and Data Science. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1877050918308767>
- [18] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, “Object detection with deep learning: A review,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, 2019.
- [19] J. Chen, H. Mai, L. Luo, X. Chen, and K. Wu, “Effective feature fusion network in bifpn for small object detection,” in *2021 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2021, pp. 699–703.

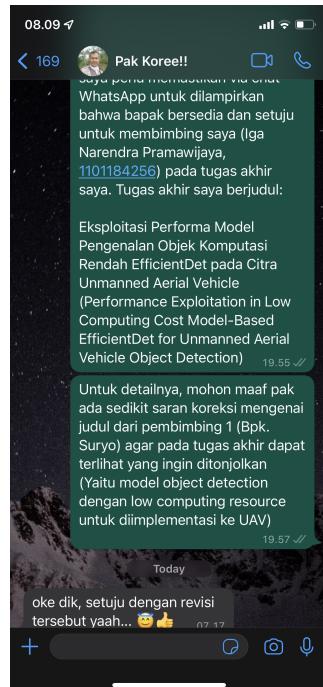
- [20] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [21] C. Lemaréchal, “Cauchy and the gradient method,” 2012.
- [22] J. Brownlee, *Better deep learning: train faster, reduce overfitting, and make better predictions.* Machine Learning Mastery, 2018.
- [23] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning.* MIT press, 2016.
- [24] A. Lydia and S. Francis, “Adagrad - an optimizer for stochastic gradient descent,” vol. Volume 6, pp. 566–568, 05 2019.
- [25] J. Zhang, S. P. Karimireddy, A. Veit, S. Kim, S. J. Reddi, S. Kumar, and S. Sra, “Why are adaptive methods good for attention models?” 2019. [Online]. Available: <https://arxiv.org/abs/1912.03194>
- [26] J. H. Hosang, R. Benenson, P. Dollár, and B. Schiele, “What makes for effective detection proposals?” *CoRR*, vol. abs/1502.05082, 2015. [Online]. Available: <http://arxiv.org/abs/1502.05082>
- [27] A. Gad, “Evaluating object detection models using mean average precision,” Mar 2022. [Online]. Available: <https://www.kdnuggets.com/2021/03/evaluating-object-detection-models-using-mean-average-precision.html>

LAMPIRAN

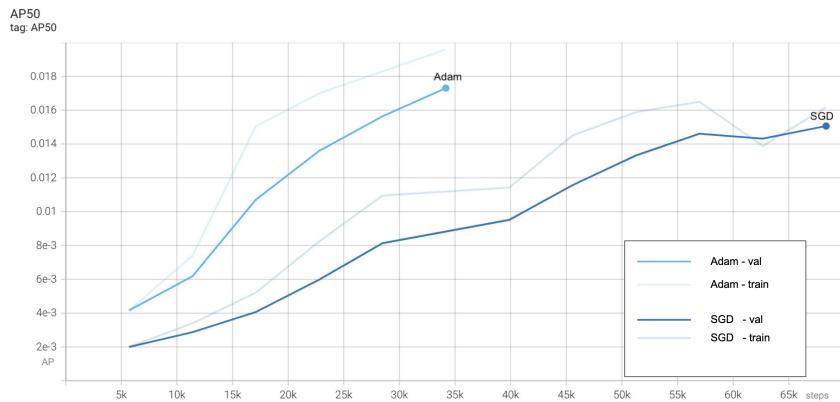
Bukti persetujuan pembimbing



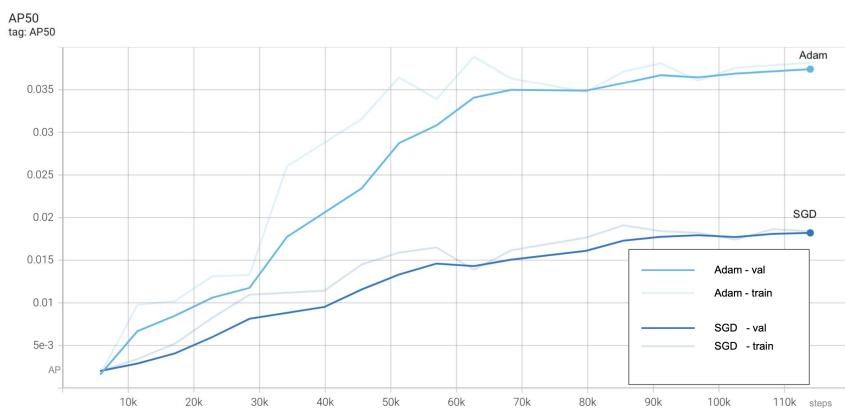
Gambar 0.1. Bukti Persetujuan Pembimbing 1 (Suryo Adhi Wibowo, S.T., M.T., Ph.D)



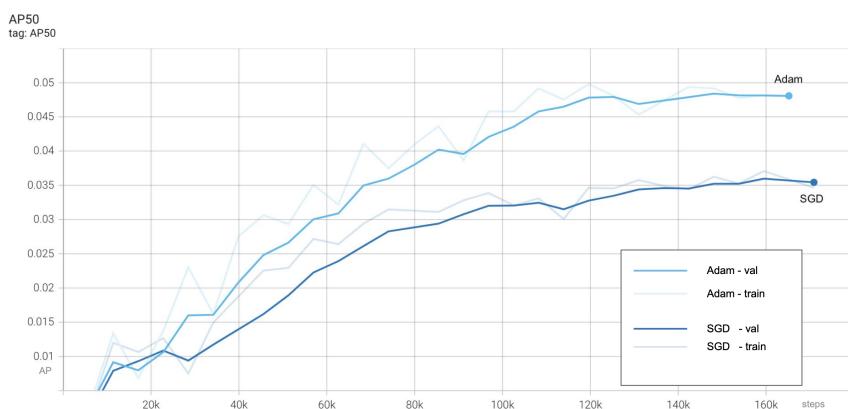
Gambar 0.2. Bukti Persetujuan Pembimbing 2 (Dr. Koredianto Usman, S.T., M.Sc)



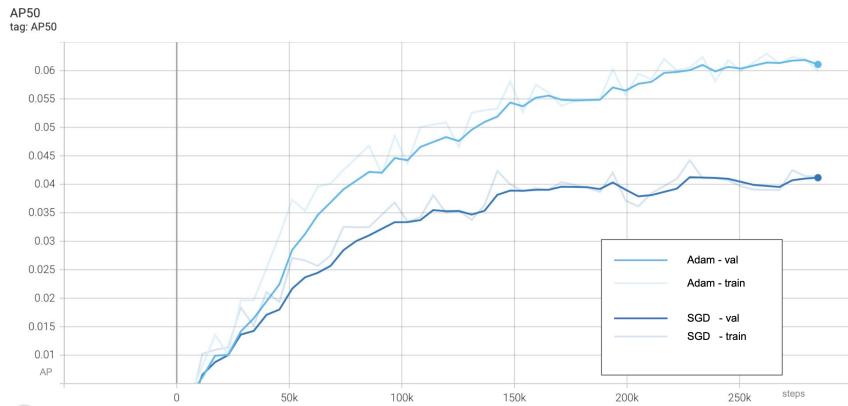
Gambar 0.3. Rincian AP50 pada fungsi optimasi SGD dan Adam setelah dilatih 10 epoch



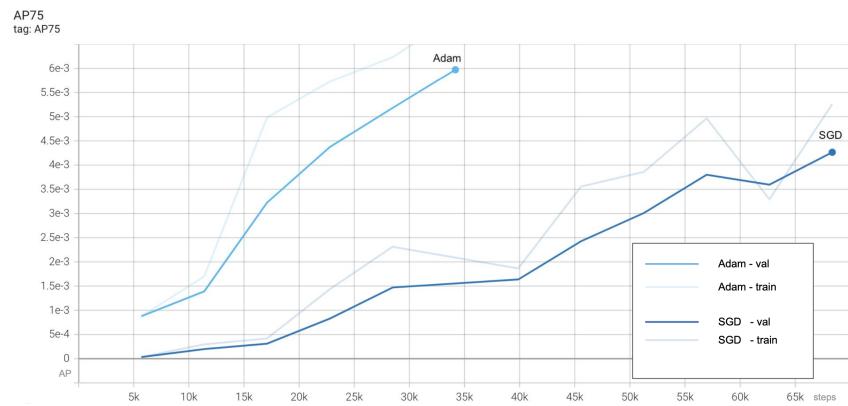
Gambar 0.4. Rincian AP50 pada fungsi optimasi SGD dan Adam setelah dilatih 20 epoch



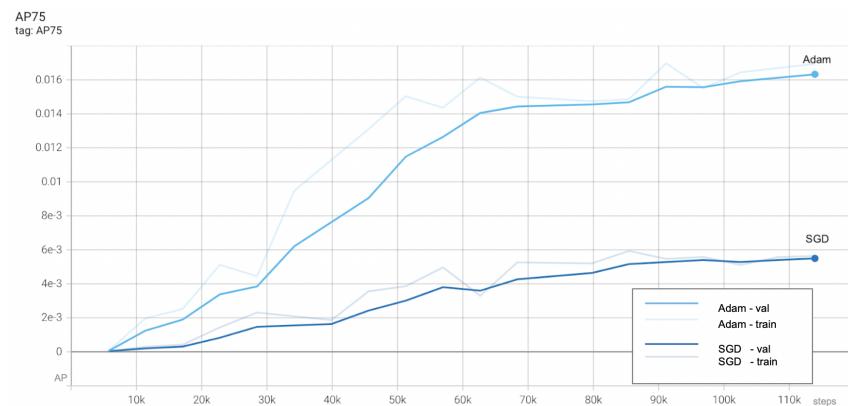
Gambar 0.5. Rincian AP50 pada fungsi optimasi SGD dan Adam setelah dilatih 30 epoch



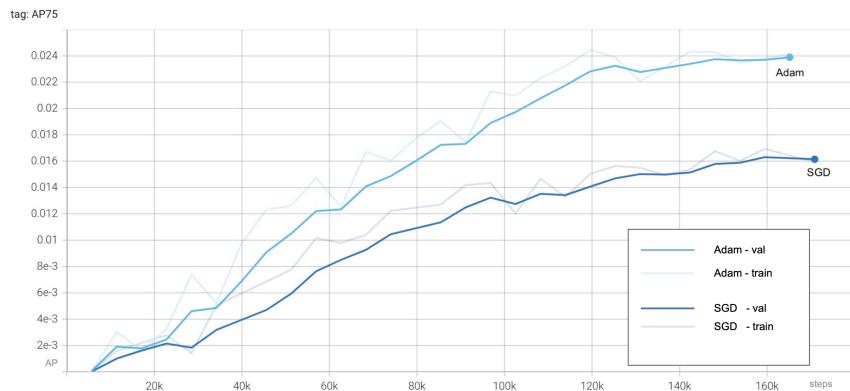
Gambar 0.6. Rincian AP50 pada fungsi optimasi SGD dan Adam setelah dilatih 50 epoch



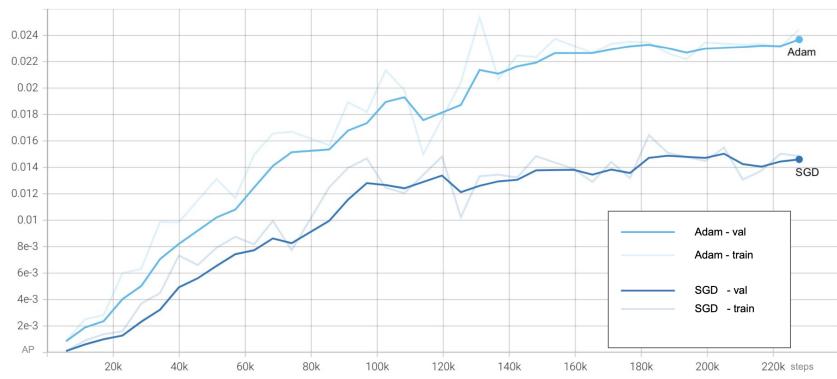
Gambar 0.7. Rincian AP75 pada fungsi optimasi SGD dan Adam setelah dilatih 10 epoch



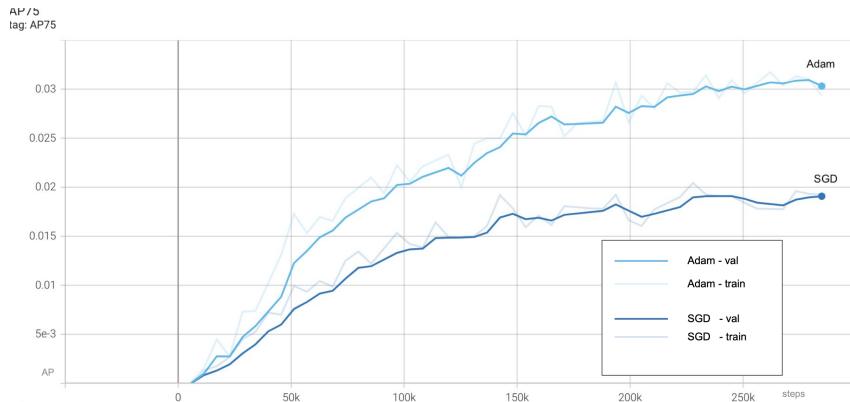
Gambar 0.8. Rincian AP75 pada fungsi optimasi SGD dan Adam setelah dilatih 20 epoch



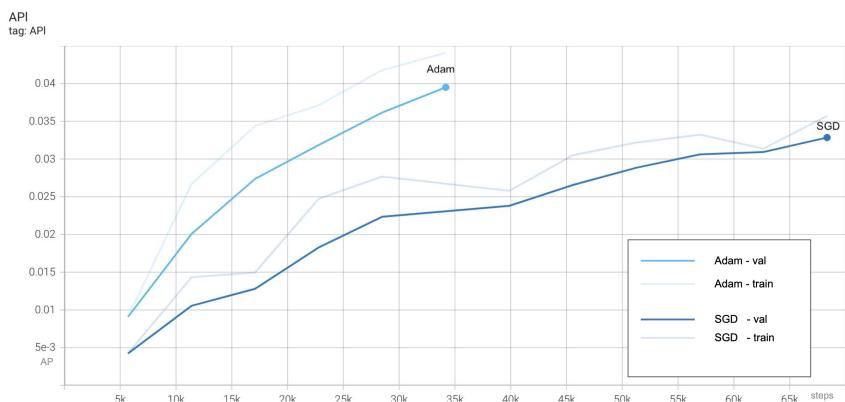
Gambar 0.9. Rincian AP75 pada fungsi optimasi SGD dan Adam setelah dilatih 30 epoch



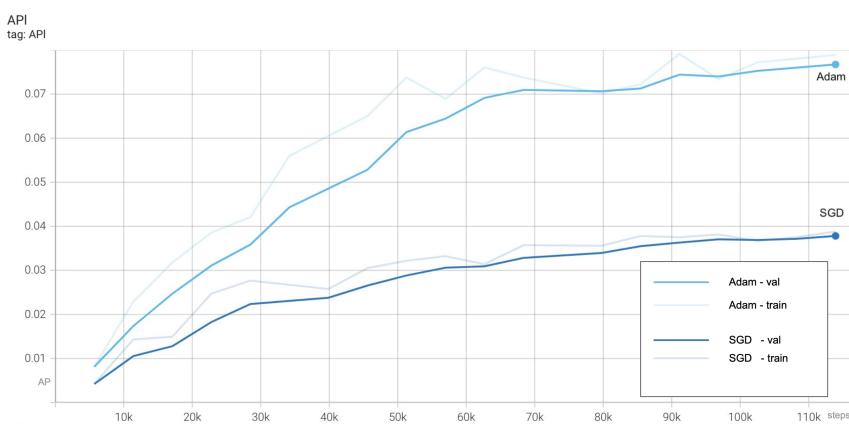
Gambar 0.10. Rincian AP75 pada fungsi optimasi SGD dan Adam setelah dilatih 40 epoch



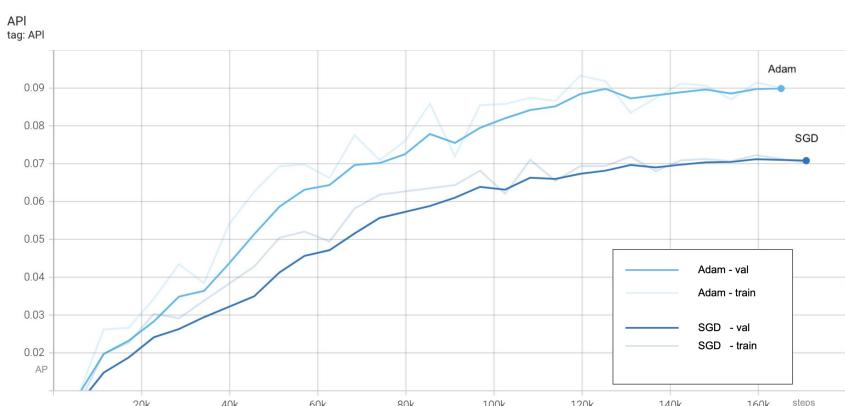
Gambar 0.11. Rincian AP75 pada fungsi optimasi SGD dan Adam setelah dilatih 50 epoch



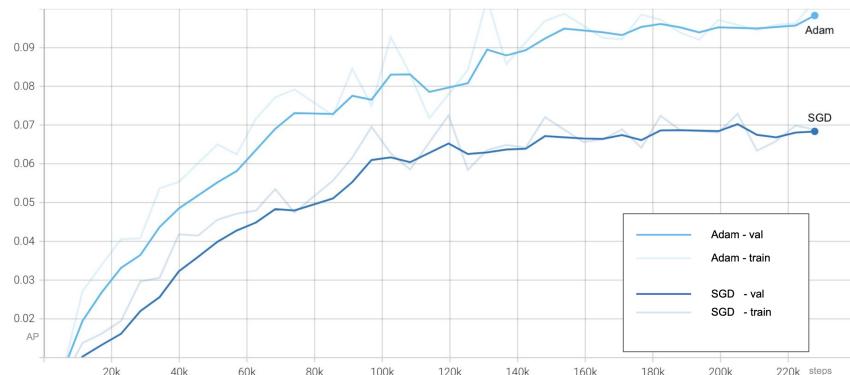
Gambar 0.12. Rincian AP area besar pada fungsi optimasi SGD dan Adam setelah dilatih 10 epoch



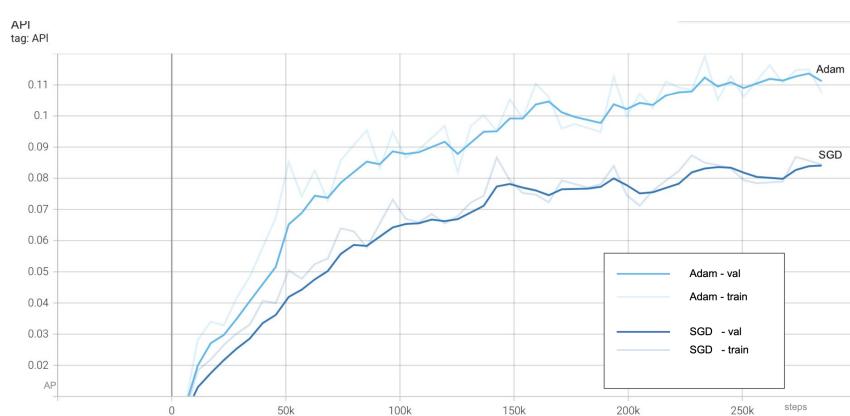
Gambar 0.13. Rincian AP area besar pada fungsi optimasi SGD dan Adam setelah dilatih 20 epoch



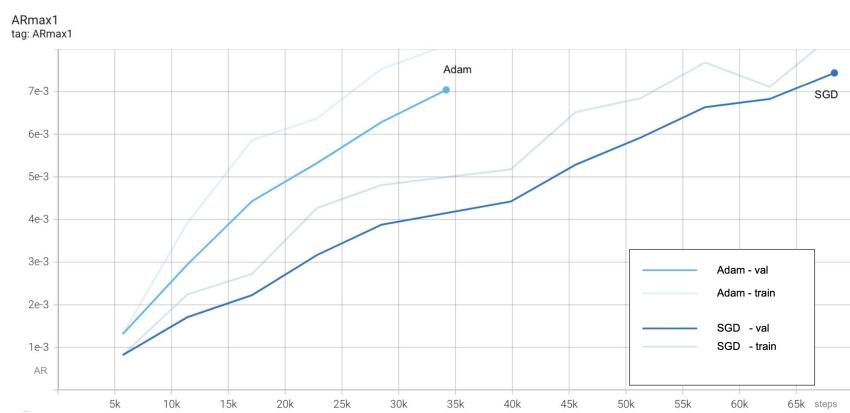
Gambar 0.14. Rincian AP area besar pada fungsi optimasi SGD dan Adam setelah dilatih 30 epoch



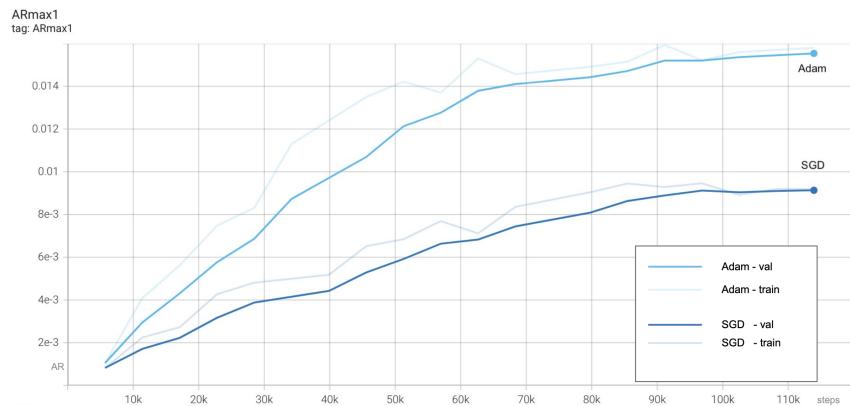
Gambar 0.15. Rincian AP area besar pada fungsi optimasi SGD dan Adam setelah dilatih 40 epoch



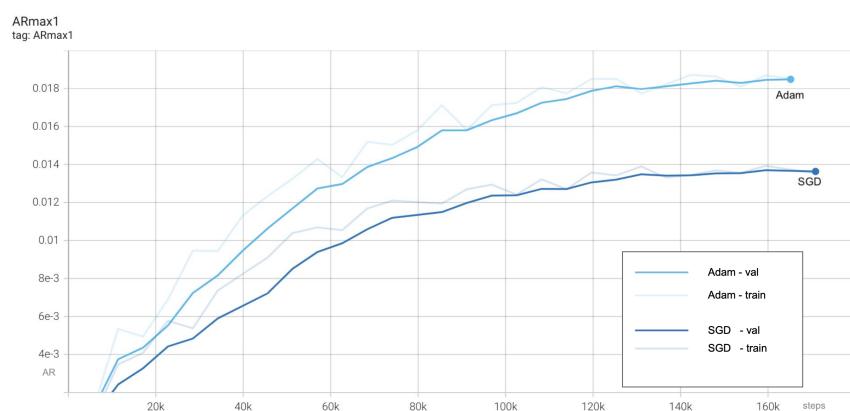
Gambar 0.16. Rincian AP area besar pada fungsi optimasi SGD dan Adam setelah dilatih 50 epoch



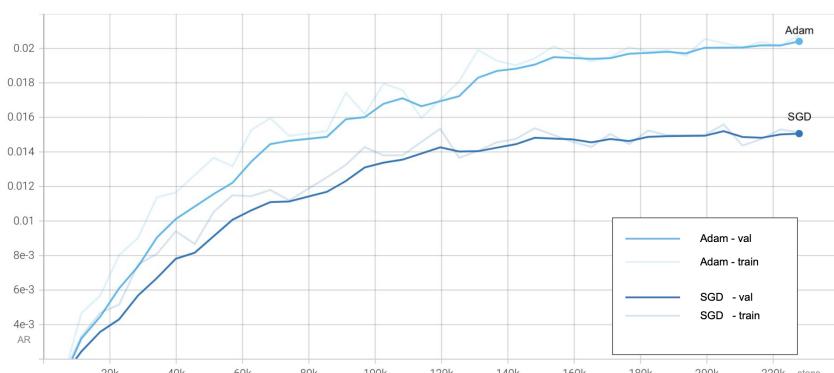
Gambar 0.17. Rincian ARmax1 pada fungsi optimasi SGD dan Adam setelah dilatih 10 epoch



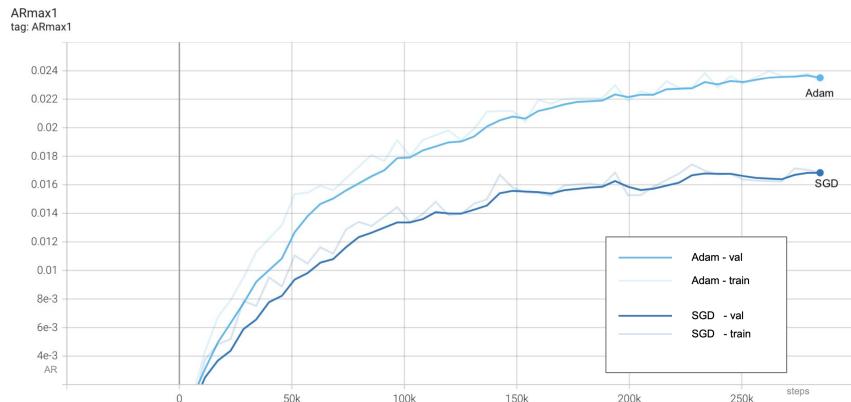
Gambar 0.18. Rincian ARmax1 pada fungsi optimasi SGD dan Adam setelah dilatih 20 epoch



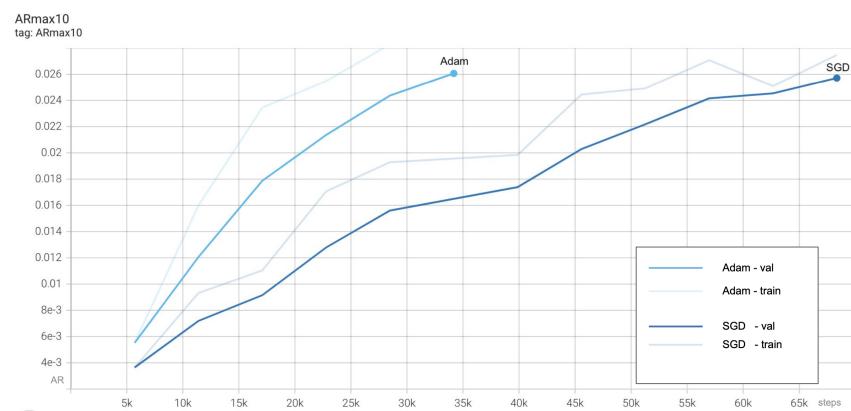
Gambar 0.19. Rincian ARmax1 pada fungsi optimasi SGD dan Adam setelah dilatih 30 epoch



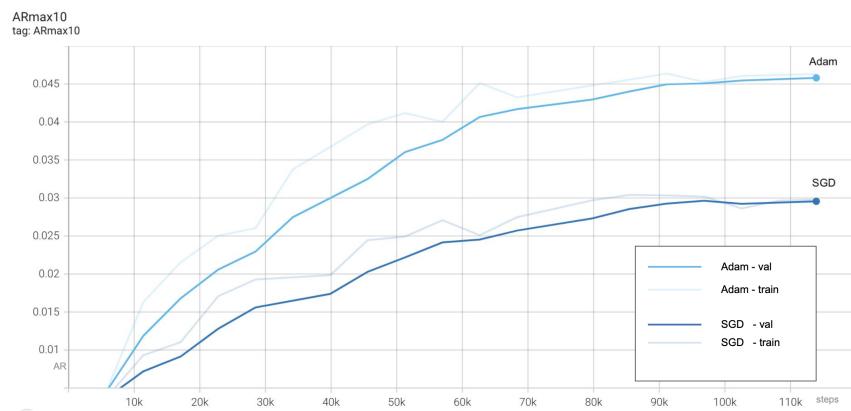
Gambar 0.20. Rincian ARmax1 pada fungsi optimasi SGD dan Adam setelah dilatih 40 epoch



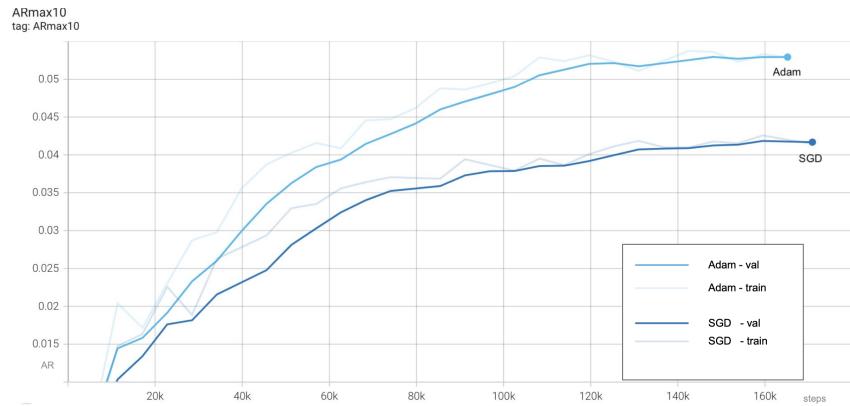
Gambar 0.21. Rincian ARmax1 pada fungsi optimasi SGD dan Adam setelah dilatih 50 epoch



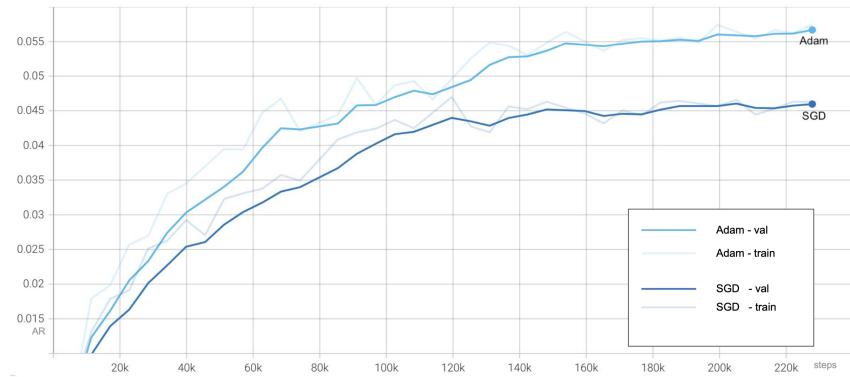
Gambar 0.22. Rincian ARmax10 pada fungsi optimasi SGD dan Adam setelah dilatih 10 epoch



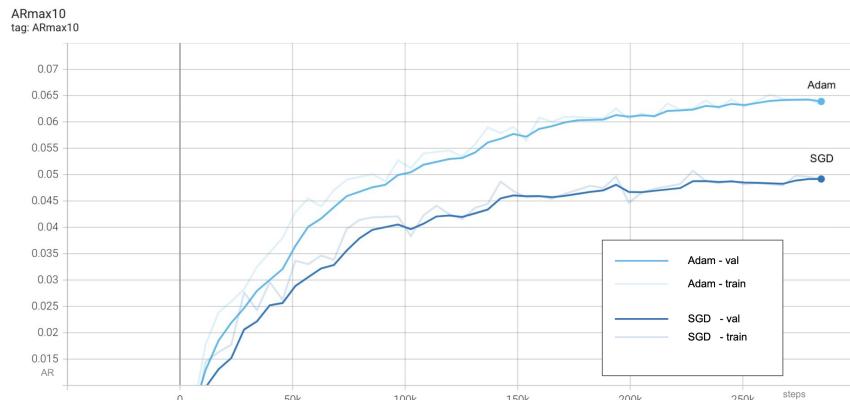
Gambar 0.23. Rincian ARmax10 pada fungsi optimasi SGD dan Adam setelah dilatih 20 epoch



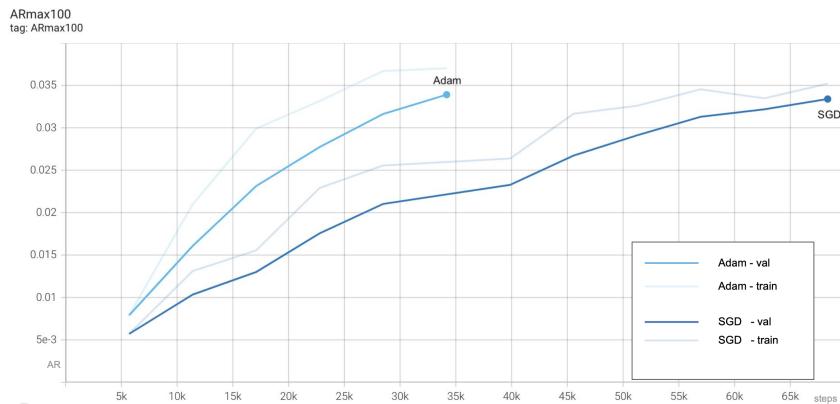
Gambar 0.24. Rincian ARmax10 pada fungsi optimasi SGD dan Adam setelah dilatih 30 epoch



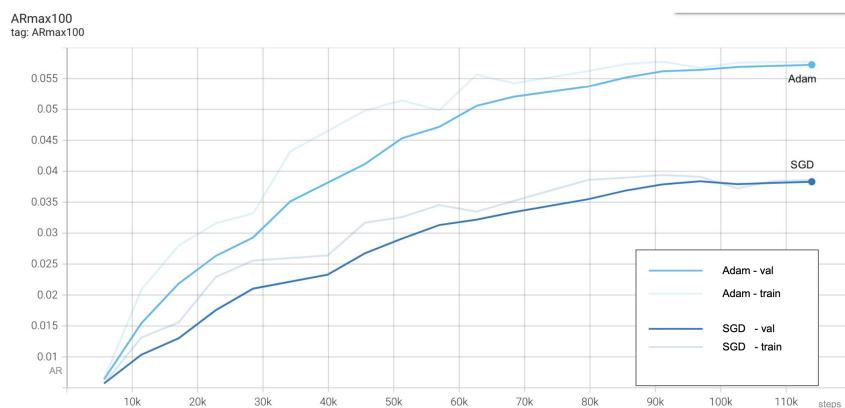
Gambar 0.25. Rincian ARmax10 pada fungsi optimasi SGD dan Adam setelah dilatih 40 epoch



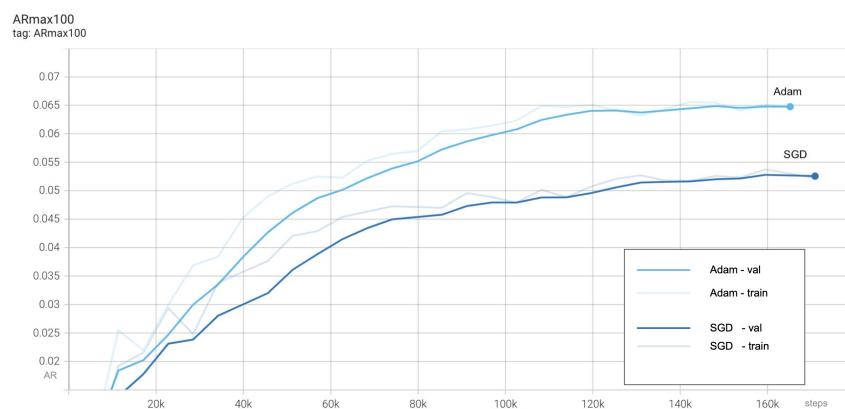
Gambar 0.26. Rincian ARmax10 pada fungsi optimasi SGD dan Adam setelah dilatih 50 epoch



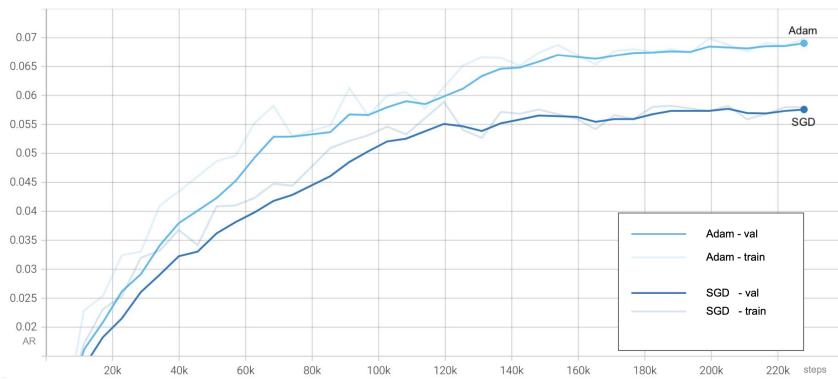
Gambar 0.27. Rincian ARmax100 pada fungsi optimasi SGD dan Adam setelah dilatih 10 epoch



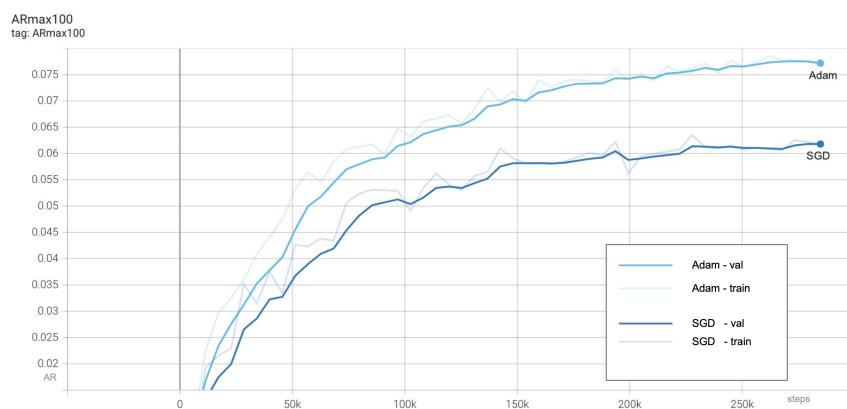
Gambar 0.28. Rincian ARmax100 pada fungsi optimasi SGD dan Adam setelah dilatih 20 epoch



Gambar 0.29. Rincian ARmax100 pada fungsi optimasi SGD dan Adam setelah dilatih 30 epoch



Gambar 0.30. Rincian ARmax100 pada fungsi optimasi SGD dan Adam setelah dilatih 40 epoch



Gambar 0.31. Rincian ARmax100 pada fungsi optimasi SGD dan Adam setelah dilatih 50 epoch