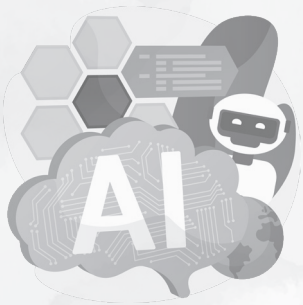


CHAPTER 18

DETEKSI OBJEK DAN PENGUKURAN PANJANG SERTA BERAT IKAN MENGGUNAKAN YOLOV3-RESNET18

Suryadiputra Liawatimena, Edi Abdurahman, Agung Trisetyarso, & Antoni Wibowo

Universitas Bina Nusantara[1]

ABSTRAK

Upaya menjaga kelestarian usaha perikanan tangkap terutama yang menghasilkan devisa, yaitu ikan tuna, tongkol dan cakalang, perlu dilakukan untuk menghindari kondisi *overfishing* yang akan berakibat pada kepunahan. Kontribusi penelitian dan novelti yang dicapai adalah percepatan pada proses yang terdapat pada algoritma Maxpool. Sistem yang dikembangkan diharapkan dapat mengenali empat jenis ikan dalam penelitian, yaitu cakalang (*Katsuwonus pelamis*), tongkol (*Euthynnus affinis*), lemadang (*Coryphaena hippurus*), dan cumi-cumi (*Loligo chinensis*), serta mendeteksi panjang ikan dan bobotnya dengan menggunakan kamera yang sebelumnya diukur dan ditimbang secara manual dengan latar belakang berbeda-beda dan jarak yang statis. Penelitian juga melakukan modifikasi *maxpool layer* sehingga dapat meningkatkan kinerjanya. Model yang dilatih dapat melakukan pendeteksian obyek dengan tingkat akurasi 89,55%. Penelitian menghasilkan eror estimasi panjang sebesar 2,59% dan berat sebesar 15,67%, serta percepatan *maxpool* 221,01 ms (27,99%).

Kata kunci: visi komputer, pengenalan obyek, *maxpool layer*, *overfishing*

[1] Korespondensi: Bina Nusantara University, Jl. K. H. Syahdan No. 9, Kemanggisan, Palmerah, Jakarta 11480, Indonesia; {suryadi; edia}@binus.ac.id; {atrisetyarso; anwibowo}@binus.edu

S. Liawatimena, E. Abdurahman, A. Trisetyarso, & A. Wibowo
Universitas Bina Nusantara, e-mail: suryadi@korika.id

S. Liawatimena, E. Abdurahman, A. Trisetyarso, and A. Wibowo, "Deteksi objek dan pengukuran panjang serta berat ikan menggunakan YOLOv3-ResNet18," in *Prosiding Use Cases Artificial Intelligence Indonesia: Embracing Collaboration for Research and Industrial Innovation in Artificial Intelligence*, B. R. Trilaksono, H. Riza, A. Jarin, N. D. S. Darmayanti, and S. Liawatimena, Eds. Jakarta: Penerbit BRIN, Februari 2023, ch. 18, pp. 195-223, doi: 10.55981/brin.668.c554
ISBN: 978-623-8052-49-3, E-ISBN: 978-623-8052-50-9

A. PENDAHULUAN

1. Latar Belakang

Ikan tuna, cakalang, dan tongkol (TCT) merupakan komoditas dengan nilai ekspor terbesar kedua dari hasil perikanan Indonesia setelah udang sebagaimana terlihat pada Tabel 1. Pada tahun 2018, TCT telah menyumbang devisa sebesar 713,9 juta USD atau 14,69% dari total nilai ekspor hasil perikanan. Sementara itu, dari sisi volume, ekspor TCT Indonesia pada tahun 2018 sebesar 168,4 ribu ton atau 14,96% dari total volume ekspor hasil perikanan [1]. Namun, hal penting yang perlu menjadi perhatian bersama adalah karena berdasarkan data yang dikeluarkan Indian Ocean Tuna Commission (IOTC), penangkapan tuna di hampir seluruh wilayah perairan Indonesia sudah dinyatakan melebihi batas *overfishing* dan sudah berjalan dari 2010 hingga 2014. Dua dari tiga jenis tuna yang ada, yakni madidihang dan cakalang, dinyatakan dapat punah dalam waktu tiga hingga sepuluh tahun jika tidak segera dilakukan pembatasan penangkapan. Wilayah perairan yang selalu menjadi target penangkapan tuna sebagian besar ada di Samudra Hindia, Samudra Pasifik, Laut Jawa, dan Laut Sulawesi [2].

Tabel 1. Nilai dan Volume Ekspor Produk Kelautan dan Perikanan Tahun 2018

KOMODITAS	NILAI (USD)	% NILAI	VOLUME (KG)	% VOLUME
Udang	1,742,119,193	35,84%	197,433,608	17,53%
Tuna-Tongkol-Cakalang	713,919,147	14,69%	168,433,759	14,96%
Cumi-Sotong-Gurita	554,594,192	11,41%	152,108,581	13,51%
Rajungan-Kepiting	472,962,123	9,73%	27,791,618	2,47%
Rumput Laut	291,837,226	6,00%	212,961,523	18,91%
Komoditas Lainya	1,085,479,049	22,33%	367,349,488	32,62%
Grand Total	4,860,910,931	100,00%	1,126,078,577	100,00%

Sumber: BPS diolah Ditjen PDS, 2019 [1]

Kementerian Kelautan dan Perikanan (KKP) melaksanakan tugas dan fungsi terkait kebijakan kelautan dan perikanan, salah satunya adalah pelaporan statistik kelautan dan perikanan yang diatur sesuai dengan Undang-Undang Republik Indonesia Nomor 16 Tahun 1997 Tentang Statistik (Indonesia, 1997) [3]. Masalah yang dihadapi adalah jumlah petugas pencacah yang sangat terbatas di setiap pangkalan pendaratan ikan (PPI) serta waktu pengumpulan dan pengompilasian data sehingga terjadi keterlambatan hingga satu tahun untuk menyerahkan dan keakuratan data yang dikumpulkan.

Berbagai penelitian untuk mengetahui berat ikan masih secara *sampling* dan ditimbang secara manual yang tentunya akan memakan waktu lama dan tidak akurat [4], beberapa di antaranya melakukan pengukuran ikan menggunakan kamera digital

[5][6][7][8][9][10][11]. Namun, kelemahan penelitian tersebut adalah menggunakan *setting* laboratorium dengan latar belakang yang sama dan jarak antara kamera dan ikan yang tetap. Tantangan yang dihadapi adalah penggunaan kamera digital yang dipasang pada kapal nelayan dan *convolutional neural network* (CNN) untuk dapat diujicobakan di lapangan. Sistem yang sudah dapat mengenali jenis ikan, panjang ikan dan beratnya diperlukan untuk mengidentifikasi pergerakan ikan yang sedang menggelepar di geladak dan sedang meluncur menuju ke lubang palka. Seiring perkembangan CNN dengan jumlah lapisan (*layer*) yang semakin banyak, maka pengenalan membutuhkan waktu proses yang semakin lama. Untuk mempercepat proses *training* dan inferensi CNN, diperlukan upaya untuk meningkatkan kinerja pada lapisan tersebut. MaxPool *layer* adalah salah satu komponen utama pada arsitektur CNN yang menjadi fokus penelitian. Menurut Habib dan Qureshi, cara melakukan optimisasi dan akselerasi CNN adalah dengan menggunakan *mixed pooling*, *stochastic pooling*, *spectral pooling*, dan *ordinal pooling* [12]. Namun, semuanya melakukan perhitungan satu sel demi satu sel secara berurutan untuk menghasilkan luaran (*output*).

Kemajuan teknologi kecerdasan artifisial berkembang semakin pesat dalam dekade terakhir, salah satunya adalah *machine learning* (ML). CNN adalah bagian dari ML yang baru-baru ini mendapat perhatian kalangan peneliti karena didorong oleh kemajuan secara drastis *graphics processing unit* (GPU) yang dikeluarkan oleh perusahaan NVIDIA® [13]. Menurut Baldominos, Saez, dan Isasi, akurasi untuk mengenali angka dan gambar oleh *deep learning* sudah melewati kemampuan manusia [14]. CNN dapat dimanfaatkan untuk mengenali objek berdasarkan sejumlah besar gambar sebagai masukan ke dalamnya. *Dataset modified national institute of standards and technology* (MNIST) berupa digit tulisan tangan diperkenalkan pada tahun 1988 oleh Lecun, Bottou, Bengio, dan Haffneer yang telah banyak digunakan untuk memvalidasi algoritma visi komputer sebagai tolok ukur pengujian arsitektur dan pendekatan jaringan saraf konvolusional yang berbeda. Hasil terbaik yang diperoleh untuk *dataset* ini melibatkan tingkat kesalahan pengujian sebesar 0,21% dan menggunakan jaringan saraf konvolusional [15].

Dengan mengumpulkan gambar (*image*) jenis ikan yang ingin dikenali, seperti cakalang (*Katsuwonus pelamis*), tongkol (*Euthynnus affinis*), lemadang (*Coryphaena hippurus*), dan cumi-cumi (*Loligo chinensis*), serta melakukan augmentasi (perubahan sudut, skala, kecerahan, dan lainnya) sebagai usaha melipatgandakan jumlah gambar lebih banyak lagi, pelatihan sebuah CNN dapat dilakukan dengan akurasi validasi mencapai 99,63% [15]. Pengenalan jenis ikan menggunakan CNN dan algoritma juga dikembangkan untuk mengonversi panjang ikan berdasarkan *bounding box* yang terdeteksi menjadi berat dengan menggunakan struktur ukuran ikan cakalang [16].

Oleh karena itu, penulis tertarik untuk melakukan penelitian pengenalan dan pengukuran berat ikan laut menggunakan algoritma YOLOv3-ResNet18 sebagai disertasi untuk membantu pengumpulan data hasil tangkapan secara otomatis.

2. Rumusan Permasalahan

Berdasarkan latar belakang yang telah dikemukakan pada bagian sebelumnya, dapat identifikasi rumusan permasalahan penelitian ini, yaitu bagaimana cara mengenali jenis ikan, mengukur panjang ikan, dan mengonversi data tersebut menjadi bobot sebagai langkah persiapan menolong petugas pencacah di setiap pangkalan pendaratan ikan (PPI). Permasalahan dibagi menjadi tiga pertanyaan penelitian sebagai berikut.

a. Bagaimana mengakselerasi algoritma Maxpool *layer*?
b. Bagaimana cara mendeteksi objek (*object detection*) ikan dari sebuah gambar dengan latar belakang berbeda dan jarak yang statis?
c. Bagaimana mendapatkan panjang objek yang sebenarnya dari hasil pendeteksian?

3. Tujuan Penelitian

Tujuan penelitian yang dicapai adalah sebagai berikut.

a. Melakukan analisis algoritma Maxpool *layer* untuk kebutuhan perubahan.
b. Mempelajari pendeteksian objek menggunakan YOLOv3-ResNet18 untuk mengenali cakalang, tongkol, ikan lemadang, dan cumi-cumi dengan latar belakang berbeda dan jarak yang statis.
c. Membangun sistem yang dapat mengenali jenis serta mengukur panjang dan berat ikan yang sebenarnya dari hasil pendeteksian objek.

4. Ruang Lingkup Penelitian

Penelitian ini bertujuan untujk mengenali cakalang, tongkol, ikan lemadang, dan cumi-cumi menggunakan ModelArts sebagai CNN dari gambar dengan latar belakang berbeda dan jarak (kamera dengan objek) statis. Penempatan kamera perpendiklular terhadap objek. Perhitungan menggunakan CPU dan tidak mengeksplorasi kemampuan paralel pada GPU. Penelitian disertasi mengasumsikan orientasi badan ikan adalah horizontal.

5. Kontribusi Penelitian

Kontribusi penelitian terbagi ke dalam dua aspek, yaitu aspek keilmuan dan kontribusi praktis. Aspek keilmuan dan novelti berkontribusi terhadap percepatan proses yang terdapat pada algoritma Maxpool untuk membuat sistem yang dapat mengenali serta mendeteksi panjang dan bobot cakalang, tongkol, ikan lemadang, dan cumi-cumi menggunakan kamera yang sebelumnya dilakukan secara manual dengan latar belakang berbeda-beda dan jarak statis.

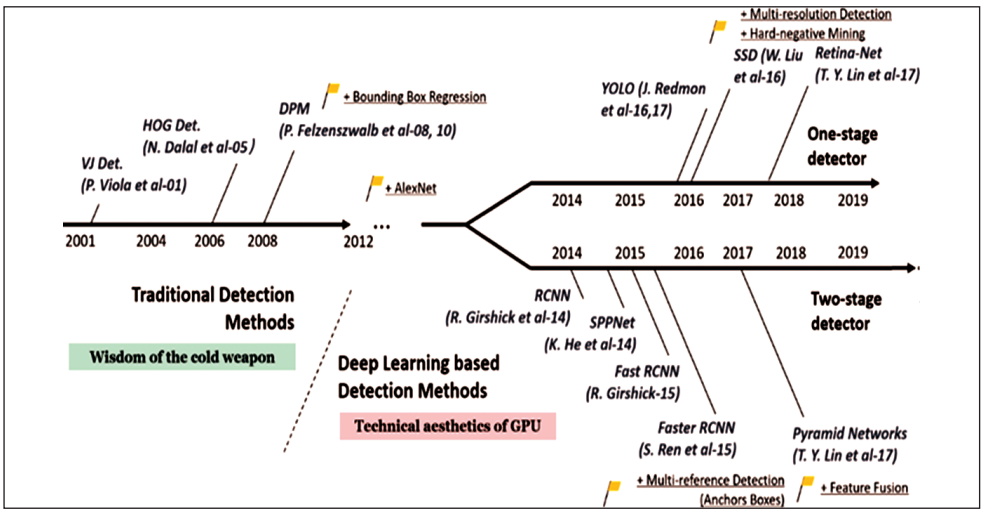
Sementara itu, aspek kedua, yaitu kontribusi praktis mengungkapkan bahwa metode ini dapat dilanjutkan untuk dapat diterapkan sebagai alat bantu bagi petugas pencacah di PPI untuk mengenali hasil tangkapan dan bobotnya.

B. STUDI LITERATUR

1. Visi Komputer (*Computer Vision*)

Menurut Zou dkk., deteksi objek, sebagai salah satu masalah paling mendasar dan menantang dalam visi komputer, telah menerima perhatian besar dalam beberapa tahun terakhir, seperti tonggak sejarah detektor objek (*object detector*), *dataset*, metrik, blok bangunan dasar sistem deteksi, dan teknik percepatan [17]. Pendeteksian objek adalah tugas visi komputer yang penting dan berkaitan dengan deteksi *instance* objek visual dari kelas tertentu (seperti manusia, hewan, mobil, dan bangunan) dalam gambar digital. Tujuan deteksi objek adalah untuk mengembangkan model dan teknik komputasi yang menyediakan salah satu bagian paling dasar dari informasi yang dibutuhkan oleh visi komputer.

Salah satu masalah mendasar dari visi komputer adalah mendeteksi objek seperti segmentasi contoh (*instance segmentation*) [18][19][20][21], keterangan gambar [22][23][24], dan pelacakan objek [25]. Dari sudut pandang aplikasi, deteksi objek dapat dikelompokkan menjadi dua topik penelitian, yaitu secara umum dan secara spesifik. Secara umum, deteksi objek bertujuan untuk mengeksplorasi metode mendeteksi berbagai jenis objek di bawah kerangka kerja terpadu untuk menyimulasikan visi manusia dan kognisi. Secara spesifik, deteksi objek mengacu pada deteksi di bawah skenario aplikasi spesifik, seperti deteksi pejalan kaki, wajah, dan teks. Dalam beberapa tahun terakhir, pesatnya perkembangan teknik pembelajaran mendalam (*deep learning*) [26] telah membawa semangat baru dalam penelitian deteksi objek, mengarah ke terobosan luar biasa, dan mendorongnya maju penelitian dengan perhatian yang belum pernah terjadi sebelumnya. Deteksi objek kini telah banyak digunakan dalam banyak aplikasi di dunia nyata, seperti pengemudian secara otonom, penglihatan robot, dan pengawasan video.



Gambar 1. Peta Jalan dan Tonggak Sejarah Deteksi Objek [17]

Meskipun orang selalu bertanya mengenai apa saja kesulitan dan tantangan dalam deteksi obyek, sebenarnya pertanyaan ini tidak mudah dijawab, bahkan mungkin terlalu digeneralisasi. Oleh karena tugas deteksi objek memiliki tujuan dan kendala yang sama sekali berbeda, kesulitannya mungkin berbeda satu sama lain. Selain beberapa tantangan umum dalam tugas penglihatan komputer lainnya, seperti objek di bawah sudut pandang yang berbeda, iluminasi, dan variasi antarkelas, tantangan dalam pendeteksian objek tidak terbatas pada rotasi objek dan perubahan skala (misalnya objek kecil), lokalisasi objek yang akurat, deteksi objek padat dan tumpang tindih, serta percepatan proses deteksi [17].

Dalam dua dekade terakhir, secara luas telah diterima bahwa kemajuan deteksi objek umumnya telah melewati dua periode sejarah, yaitu periode deteksi objek tradisional (sebelum 2014) dan periode deteksi berbasis pembelajaran yang mendalam (*deep learning*) (setelah 2014), seperti yang terlihat pada Gambar 1.

2. Penelitian Terdahulu

Penelitian ini dilakukan dengan mempelajari berbagai penelitian sebelumnya, seperti pengukuran panjang ikan dengan teori optik dan teknik pemrosesan gambar menggunakan kamera digital delapan megapiksel dengan jarak tetap dan latar belakang sama [5][8]. Anumudu dan Mojekwu melakukan pengenalan jenis (spesies) ikan berbentuk mirip menggunakan analisis morfometrik, seperti *truss network measurement*, analisis gambar dengan univariat, bivariat, dan multivariat, serta *principal component analysis* (PCA) dengan jarak tetap dan latar belakang sama [6]. Jamaluddin dkk. (2015) melakukan pengukuran panjang ikan dengan pendekatan pengukuran nonkontak menggunakan kamera USB, sensor jarak ultrasonik, dan mikrokontroler dengan jarak berbeda dan latar belakang sama [7]. Jin dkk. Mencari hubungan berat dan panjang ikan dan faktor kondisi Fulton cakalang [27], sedangkan Li dkk. berusaha mengenali ikan dengan menggunakan CNN [28]. Hao, Yu, dan Li melakukan pengukuran panjang ikan dengan visi komputer dengan kamera tunggal dan kamera stereo, pengukuran jarak dan perhitungan luas penampang ikan dengan jarak dan latar belakang sama [9]. Penelitian berikutnya melakukan pengukuran dan estimasi berat dan ukuran ikan dengan pengolahan gambar penetasan ikan di bawah air dengan jarak berbeda, tetapi dengan latar belakang sama [10]. Islamadina dkk. mengukur badan ikan berupa estimasi panjang, lebar, dan tinggi dengan kamera tunggal untuk jarak dan latar belakang sama [11]. Beberapa penelitian melakukan deteksi ikan melalu gambar, yaitu pendeteksian ikan menggunakan model CNN dengan segmentasi citra untuk mengenali ikan dari air laut yang buram [29] serta pendeteksian yang menghasilkan tiga model dengan kerangka deteksi objek untuk mendeteksi target spesies, ikan yang penting secara ekologis dari video yang diolah per bingkai (*frame*) [30]. Terakhir, Habib dan Qureshi melakukan optimalisasi dan percepatan jaringan saraf *convolutional* dengan melakukan perubahan pada *convolution* dan *maxpool layer*, di mana usaha perhitungan *maxpool layer* empat sel secara berurutan satu persatu [12].

Berdasarkan penelitian di atas, dapat disimpulkan bahwa visi komputer dapat digunakan untuk mengenali jenis ikan dan mengukur panjang ikan menggunakan kamera, baik tunggal maupun stereo, dengan latar belakang yang sama dan usaha untuk mengubah *convolution* dan *maxpool layer* secara berurutan satu per satu. Kontribusi penelitian ini pada latar belakang yang berbeda-beda, menggunakan dataset ikan sendiri (terdiri dari 4.000 foto), adanya estimasi bobot ikan berdasarkan panjang ikan yang terdeteksi berdasarkan ukuran dari *bounding box*nya, dan berupaya untuk melakukan peningkatan kecepatan *maxpool layer*.

3. Kecerdasan Artifisial

Kecerdasan artifisial (*aritificial intelligence*, AI) adalah studi dan implementasi teknologi, di mana suatu mesin melakukan hal yang membutuhkan kecerdasan manusia [31]. Beberapa hal, seperti mengidentifikasi gambar, menerjemahkan bahasa, dan analisis data hanya dapat dilakukan oleh kecerdasan manusia, tetapi, dengan kecerdasan artifisial, hal tersebut sekarang dapat ditangani oleh mesin. Oleh karena itu, mesin dengan kemampuan kecerdasan artifisial merupakan sebuah sistem yang dapat meniru salah satu bentuk dari kecerdasan manusia untuk menyelesaikan sebuah masalah serta memberikan sebuah terobosan baru di dalam dunia teknologi.

Menurut Entwistle, kecerdasan artifisial adalah ilmu yang berfokus untuk menciptakan suatu mesin pintar dengan tujuan memaksimalkan data-data di sekitarnya untuk menyelesaikan permasalahan layaknya manusia [32]. Pada era modern ini, kecerdasan artifisial digunakan pada berbagai bidang, antara lain *game playing*, *speech recognition*, *understanding natural language*, *computer vision*, *expert system*, dan *heuristic classification*. *Machine learning* (ML) adalah salah satu cabang ilmu kecerdasan artifisial yang memiliki tujuan mempelajari algoritma untuk meningkatkan performa secara sendiri lewat pengalaman belajar dari setiap iterasi. ML sebagai salah satu cabang dari kecerdasan artifisial meniru salah satu bentuk kecerdasan pada manusia melalui belajar.

Akan sangat berguna jika komputer dapat belajar dari pengalaman dan secara otomatis meningkatkan efisiensi program mereka sendiri selama eksekusi. Sebagai contoh, ketika tikus melihat sebuah makanan, ia akan mencoba memakannya sedikit dan akan menghabiskannya tergantung dari rasa serta efek yang timbul setelah memakannya. Hal ini merupakan faktor psikologis mahluk hidup. Jika makanan tersebut memberikan efek negatif, makanan tersebut tidak akan dihabiskan. Menurut Shalev-Shwartz dan Ben-David, peristiwa tersebut belajar dari pengalaman sebelumnya [33]. Sistem ML melakukan tugasnya secara spesifik layaknya yang dilakukan oleh manusia. Tahapan pertama, sistem akan melewati tahapan pelatihan dengan data latihan yang *supervised*, *unsupervised*, *supervised and unsupervised*, *self-learning*, atau *reinforcement*. Setelah dilatih, terciptalah sebuah model yang dapat digunakan untuk mendapatkan sebuah hasil dari semua data yang dimasukkan. *Deep learning* dengan menggunakan model *deep max-pooling convolutional neural network*, sebuah

deep neural network dengan lapisan konvolusional dan *max-pooling* [34]. Model ini digunakan untuk deteksi, klasifikasi, dan prediksi pada gambar.

Sebuah penelitian menjelaskan bahwa *deep learning* adalah model komputasi yang terdiri dari beberapa lapisan pemrosesan untuk mempelajari suatu representasi data dengan tingkat abstraksi yang berbeda [26]. Metode secara umum digunakan untuk pengenalan suara, pengenalan objek secara visual, deteksi obyek, dan banyak domain lainnya. *Deep learning* layaknya sistem sensori, di mana aliran informasi yang memiliki koneksi secara internal dengan neuron dan setiap neuron membantu memproses informasi berikutnya [35].

Metode pada *deep learning* adalah metode *representation-learning* yang memungkinkan mesin untuk diisi dengan data mentah secara otomatis untuk menemukan representasi yang diperlukan untuk deteksi dan klasifikasi. Hal ini diperoleh dari pembelajaran representasi dengan beberapa level representasi dan penyusunan modul nonlinear yang dimulai dari satu level (input mentah) menjadi representasi pada level yang lebih tinggi dengan komposisi transformasi yang memadai sehingga fungsi yang sangat kompleks dapat dipelajari, misalnya sebuah gambar yang hadir dalam bentuk piksel. Fitur pada lapisan pertama reprensentasi akan mewakili ada atau tidaknya tepi pada orientasi dan lokasi gambar tersebut. Kemudian, lapisan kedua akan mendeteksi motif dengan melihat susunan tepi tertentu, lapisan ketiga merangkai motif menjadi kombinasi yang lebih besar sesuai dengan bagian obyek yang sudah dikenal, dan lapisan berikutnya akan mendeteksi objek sebagai kombinasi dari bagian tersebut.

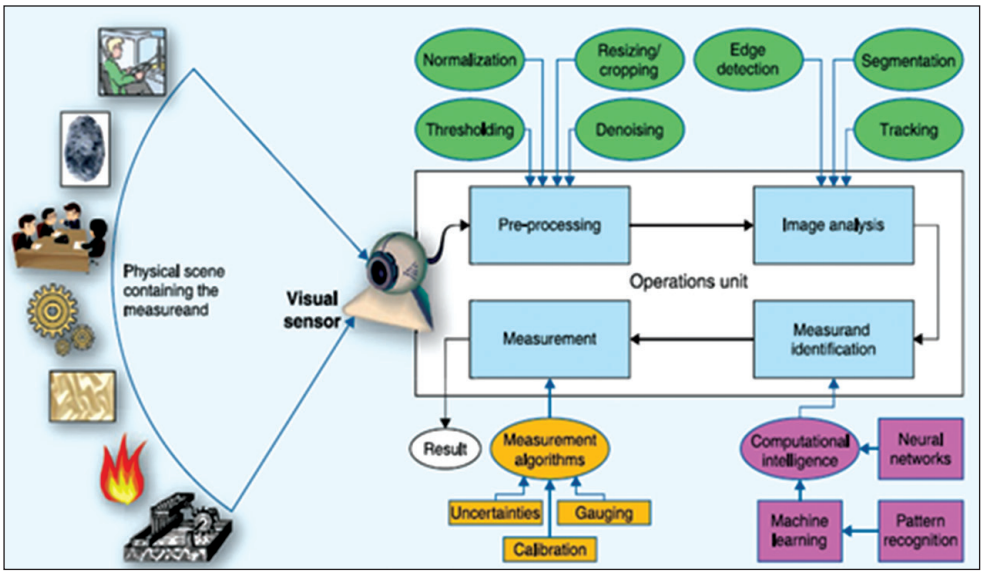
4. Pengukuran Berbasis Visi Komputer

Pembelajaran mendalam merevolusi banyak bidang visi komputer (*computer vision*). Sejak ledakan eksplosif dalam tantangan ImageNet pada 2012 [36], kinerja model telah meningkat dengan kecepatan yang taktertandingi. Menurut Shirmohammadi dan Ferrero, kemajuan yang berkelanjutan dan cepat dari teknologi perangkat keras atau lunak dalam kamera dan sistem komputasi, pengguna kini memiliki akses ke kamera dan unit komputasi yang lebih kecil, murah, dan berkualitas [37].

Hal ini membawa angin segar pada metode berbasis visi yang terdiri dari pemrosesan gambar dan kecerdasan komputasi yang dapat diimplementasikan dengan lebih mudah dan lebih murah daripada sebelumnya. Terlihat pada Gambar 2 bahwa sebuah gambar diperoleh oleh sensor visual dan diumpankan ke berbagai unit operasi untuk melakukan pra-pemrosesan gambar (warna hijau), inteligen komputasional (warna violet), dan operasi pengukuran (warna kuning). Empat tahapan dalam melakukan pengukuran berbasis visi komputer adalah sebagai berikut.

a. Pra-Pemrosesan

Tujuan tahap ini adalah mempersiapkan gambar mentah untuk tahap operasi selanjutnya. Gambar yang diperoleh dari sensor visual dapat memiliki kekurangan, seperti silau, *noise*, kabur. Selain itu, gambar mungkin tidak dalam format yang diperlukan



Gambar 2. Arsitektur Pengukuran Berbasis Visi Komputer [36]

pada operasi berikutnya, misalnya gambar sidik jari yang biasanya diperoleh dalam skala abu-abu, tetapi perlu dikonversi menjadi hitam putih murni tanpa latar belakang apapun untuk tahap pemrosesan. Tahapan ini juga melakukan operasi, seperti normalisasi yang memodifikasi intensitas piksel dan kontras bagian-bagian gambar, ambang batas yang mengubah gambar menjadi gambar hitam dan putih biner, *denoising* yang *meridge* gambar dari aditif *white Gaussian noise* atau jenis kebisingan lainnya, pengubahan ukuran, dan pemangkasan. Operasi ini adalah pemrosesan sinyal, khususnya pemrosesan gambar, dengan banyak metode dan algoritma yang tersedia untuk implementasinya.

b. Analisis Gambar

Tujuan tahap ini adalah untuk menganalisis gambar dan mengekstrak informasi yang diperlukan untuk menemukan pengukuran dan melakukan pengukuran nanti. Tahap ini juga menggunakan operasi pemrosesan gambar, seperti segmentasi yang membagi gambar menjadi beberapa segmen yang masing-masing mewakili sesuatu dalam adegan, mendeteksi tepi yang menemukan tepi objek dalam adegan, dan membantu pengguna mengidentifikasi objek yang menarik, serta melacak objek setelah terdeteksi dan ketika mereka bergerak melalui adegan. Pada akhir tahap analisis gambar, luaran (*output*) dapat berupa pengukuran dan informasi yang dapat mengarah pada identifikasi pengukuran. Dalam beberapa kasus, pengguna dapat melewati tahap mengidentifikasi dan langsung menuju ke tahap pengukuran.

Sebagai contoh, untuk menghitung jumlah orang dalam ruangan dengan menghitung jumlah wajah, setelah wajah terdeteksi pada tahap analisis gambar, pengguna dapat langsung menghitungnya tanpa operasi lebih lanjut. Namun, dalam beberapa aplikasi, lebih banyak operasi diperlukan untuk mengidentifikasi apa yang menjadi target pengukuran.

c. Identifikasi target

Tujuan tahap ini adalah untuk mengidentifikasi pengukuran spesifik dan dalam gambar, jika belum diidentifikasi pada tahap analisis gambar sebelumnya. Teknik yang digunakan di sini sebagian besar didasarkan pada kecerdasan komputasi, terutama pembelajaran mesin (*machine learning*), dan khususnya pengenalan pola dan pencocokan pola, di mana yang pertama memberikan pencocokan yang paling masuk akal dari input yang diberikan ke *output* dan memperkenalkan beberapa ketidakpastian, sedangkan yang terakhir terlihat untuk melaporkan kecocokan tepat dari input yang diberikan ke pola apriori. Pada tahap ini, pengguna dapat menemukan, mencocokkan, dan mengidentifikasi pola, bentuk, dan kelas objek tertentu untuk mengidentifikasi pengukuran. Pengenalan karakter optik dan jaringan saraf (*neural network*) juga dilakukan pada tahap ini jika diperlukan. Sebagai contoh, dengan memasukkan gambar ke dalam *support vector machine* (SVM) yang sebelumnya telah dilatih dengan gambar makanan serupa dalam hal warna, tekstur, bentuk, dan ukuran, pengguna dapat mengidentifikasi bahan apa yang ada dalam makanan dengan tingkat akurasi tertentu.

d. Pengukuran

Pada tahap ini pengguna memiliki pengukuran dan dapat melakukan operasi pengukuran yang diperlukan, seperti pengukuran yang memberi dimensi pengukuran dan kelilingnya, luas, dan volume. Pengukuran secara temporal saat melacak mengukur dan mengecek kondisinya dari waktu ke waktu. Kalibrasi adalah persyaratan lain pada tahap ini. Referensi diperlukan untuk mengetahui dimensi bahan makanan, dalam hal ini adalah ibu jari pengguna yang telah diukur sebelumnya dan dapat digunakan untuk kalibrasi di sini. Contoh lain pengukuran temporal adalah mempertimbangkan aplikasi pemantauan keberadaan pengemudi untuk mendeteksi kondisi menguap. Langkah pertama adalah pengguna harus dapat mendeteksi dan melacak mulut manusia yang tertutup, kemudian mendeteksi apakah mulut manusia yang sama terbuka dan tertutup sesuai dengan pola tertentu selama waktu tertentu. Hubungan temporal antara berbagai keadaan mulut manusia adalah yang paling penting. Jika tidak, akan ada positif palsu (*false positive*) karena menyanyi atau berbicara akan disalahartikan sebagai menguap.

5. *Convolutional Neural Network* (CNN)

CNN merupakan salah satu jaringan saraf yang paling populer karena memiliki banyak lapisan, seperti *convolutional layer*, *non-linearity layer*, *pooling layer*, dan *fully connected layer*. CNN digunakan untuk menangani data gambar, seperti kumpulan data klasifikasi gambar (Image Net), visi komputer, dan pemrosesan bahasa (NLP) [38]. CNN menjadi efisien dalam mempelajari data dikarenakan dapat mengekstrak *low-level features*, mendeteksi, dan mengklasifikasikan obyek [26].

Menurut Karphaty, arsitektur dari CNN dibagi menjadi dua bagian besar, yaitu *feature extraction layer* dan *fully-connected layer* [39]. *Feature extraction layer* adalah proses yang melakukan *encoding* dari sebuah gambar menjadi *features* berupa angka yang merepesentasikan gambar tersebut. Proses ini terdiri dari dua bagian, yaitu *convolutional layer* dan *pooling layer*.

a. *Convolutional Layer*

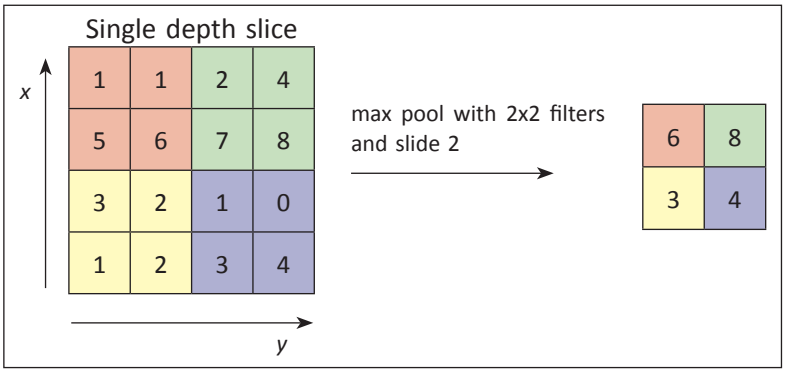
Lapisan ini terdiri dari neuron yang tersusun sedemikian rupa sehingga membentuk sebuah filter dengan panjang dan tinggi (piksel). Lapisan konvolusi merupakan lapisan pertama tempat berbagai fitur hasil ekstrak dari *input layer*, misalnya lapis pertama pada *extraction feature* adalah lapis konvulsi berukuran 5 x 5 x 3, yang berarti panjang 5 piksel, tinggi 5 piksel dengan ketebalan 3 yang merupakan *channel* dari gambar.

Setelah proses konvolusi, fungsi aktivasi merupakan tahapan sebelum memasuki *pooling layer* dan sering digunakan pada *convolutional network*, yaitu *tanh()* atau *reLU*. Beberapa peneliti menggunakan funsi aktivasi *reLU* karena sifatnya lebih berfungsi dengan baik dan memiliki fungsi nilai *output* dari *neuron* dapat dikatakan sebagai nol jika nilai input adalah negatif. Jika nilai input dari fungsi aktivasi adalah positif, *output* dari neuron adalah nilai input aktvasi itu sendiri.

b. *Pooling Layer*

Lapisan ini terdiri dari filter dengan ukuran dan *stride* tertentu yang bergeser pada seluruh area *feature map*. *Pooling* yang biasa digunakan adalah *max pooling* dan *average pooling*. Tujuan dari *pooling layer* adalah untuk mengurangi dimensi pada *feature map*. Metode ini dikenal sebagai *down-sampling* untuk mempertahankan informasi penting dari hasil *output* sehingga mempercepat komputasi karena parameter yang harus dimutakhirkan semakin sedikit dan mampu mengatasi *overfitting*. Jenis metode *pooling* adalah *max pooling*, *average pooling*, *mixed pooling*, *stride pooling*, *fractional pooling*, dan *stochastic pooling*. Menurut Singh dkk., metode *max pooling* dan *average pooling* adalah metode yang umum digunakan dalam praktik karena kesederhanaannya, kemudahan dalam implementasi, dan memiliki hasil yang sebanding dengan metode penyatuan lainnya [40].

Hal terpenting dalam pembuatan model CNN adalah memilih banyak jenis lapisan *pooling* [41]. Lapisan *pooling* bekerja pada setiap tumpukan di *feature map* dan mengurangi ukurannya. Pada umumnya bentuk lapisan *pooling* menggunakan filter berukuran 2 x 2 yang diaplikasikan dengan *stride* sebanyak 2 dan kemudian beroperasi pada setiap irisan dari *input*. Model demikian mampu mengurangi *feature map* hingga 75% dari ukuran aslinya. Gambar 3 memperlihatkan operasi *max pooling*.



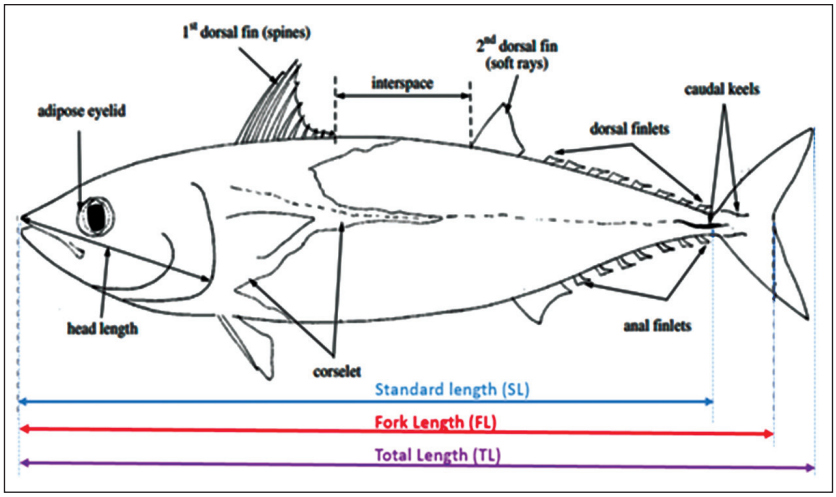
Gambar 3. Max-Pooling [41]

Max pooling ada operasi *pooling* dengan menghitung nilai maksimum atau nilai terbesar di setiap *patch* dari setiap peta fitur. Penyatuan maksimum adalah operasi penyatuan yang menghitung nilai maksimum, atau terbesar, di setiap *patch* dari setiap peta fitur. Hasilnya adalah peta fitur yang dikumpulkan yang menyoroti fitur yang paling ada di *patch*, bukan keberadaan rata-rata fitur. Hal ini ditemukan bekerja lebih baik ketimbang *average pooling*, terutama untuk tugas klasifikasi gambar.

6. Pengukuran Panjang Ikan

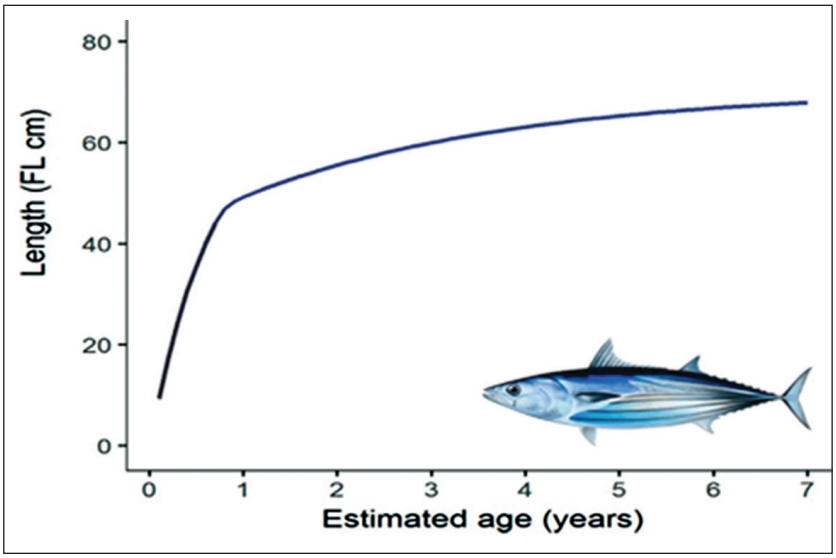
Gambar 4 di bawah ini menunjukkan cara mengukur beragam jenis ikan dan kerang-kerangan dengan panduan sebagai berikut [42].

a. Panjang total (*total length*, TL) diukur mulai dari bagian terdepan moncong atau bibir (*premaxillae*) hingga ujung ekor.
b. Panjang cagak atau panjang garpu (*fork length*, FL) diukur mulai dari bagian terdepan moncong atau bibir (*premaxillae*) hingga ekor.
c. Panjang standar (*standard length*, SL) diukur mulai dari bagian terdepan moncong atau bibir (*premaxillae*) hingga pertengan pangkal sirip ekor (pangkal sirip ekor bukan berarti sisik terakhir karena sisik-sisik tersebut biasanya memanjang sampai ke sirip ekor).



Gambar 4. Ukuran Panjang Ikan [42]

Ikan cakalang menjadi dewasa ketika mulai berusia dua tahun, panjang 41–43 cm, dan rata-rata umur maksimumnya mencapai tujuh tahun. Bobot rata-rata ikan cakalang yang ditangkap di Samudra Hindia berkisar dari 3 kg untuk penangkapan dengan *purse sein*, 2,8 kg untuk kapal umpan Maladewa, dan 4–5 kg untuk *gillnet* [43][44].



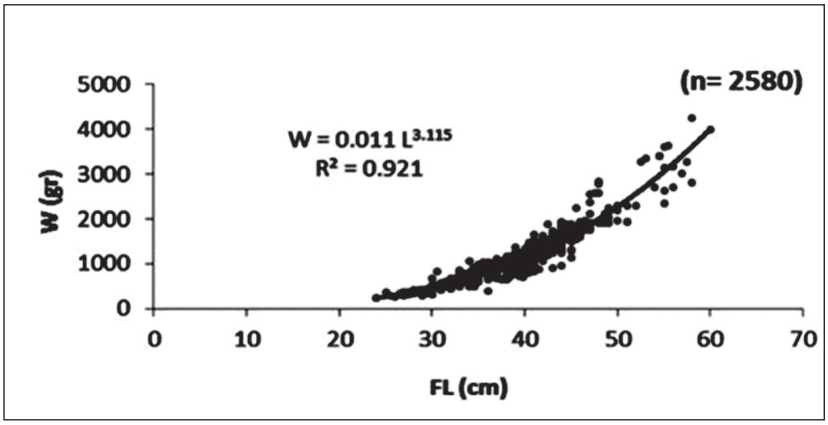
Gambar 5. Hubungan Panjang dengan Umur Ikan Cakalang [45]

Model pertumbuhan ikan cakalang dicirikan pertumbuhan yang cepat pada tahap pertama, diikuti oleh pertumbuhan yang lebih lambat dalam tahap kedua. Karena tinggi awal tingkat pertumbuhan, cakalang dapat mencapai 45cm FL pada tahun pertama kehidupan, 50–65 cm FL di tahun kedua, di mana tingkat pertumbuhan melambat, dan umur tua berkisar hingga 70 cm. Ikan cakalang dianggap memasuki umur produktif setelah mencapai dua tahun dengan angka harapan hidupnya sampai tujuh tahun dan usia produktifnya mencapai 50–70 cm. Berat ikan tergantung dari lingkungan hidupnya sehingga memungkinkan memiliki panjang yang sama dengan berat berbeda [16][46].

Hubungan antara panjang dan berat badan ikan dirumuskan dengan persamaan sebagai berikut [27][47].

$$W = a\ L^b$$

Hubungan berat-panjang (HBP) dihitung menggunakan persamaan di atas, di mana *a* dan *b* adalah koefisien, sedangkan *L* adalah panjang cagak (cm) dan *W* adalah berat (g).

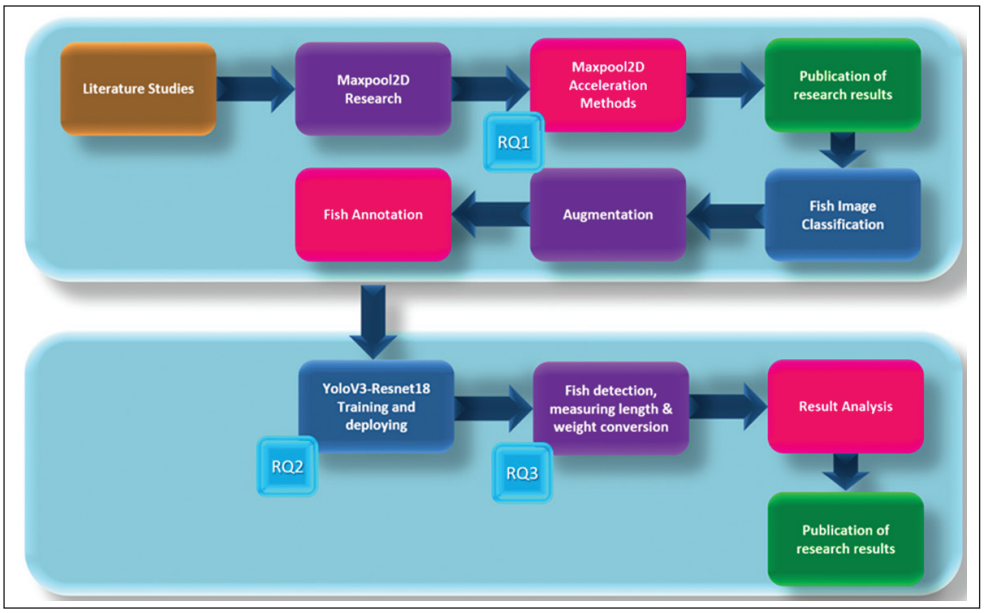


Gambar 6. Hubungan Panjang Cagak dan Bobot Ikan Cakalang [27]

7. Metode Penelitian

Seperti yang terlihat pada Gambar 7, kerangka penelitian dimulai dengan melakukan studi literatur dan fokus pada algoritma *maxpool* untuk menemukan cara mempercepat algoritma *maxpool* dan untuk mengumpulkan gambar ikan dengan berbagai latar belakang (klasifikasi, augmentasi, dan anotasi), *training YOLOv3-ResNet18*, dan melakukan *deploying* untuk mendeteksi ikan dari sebuah gambar untuk ditemukan jenis ikan, *bounding box* agar dapat menghitung panjang ikan dan konversi ke berat ikan.

Sumber daya komputasi yang digunakan saat training pada Huawei Cloud adalah *compute-intensive 1 instance* (GPU), yaitu *random access memory* (RAM) 64 GB, GPU NVIDIA-V100 (32 GB), 8 Cores. Implementasi menggunakan dua vCPUs dengan



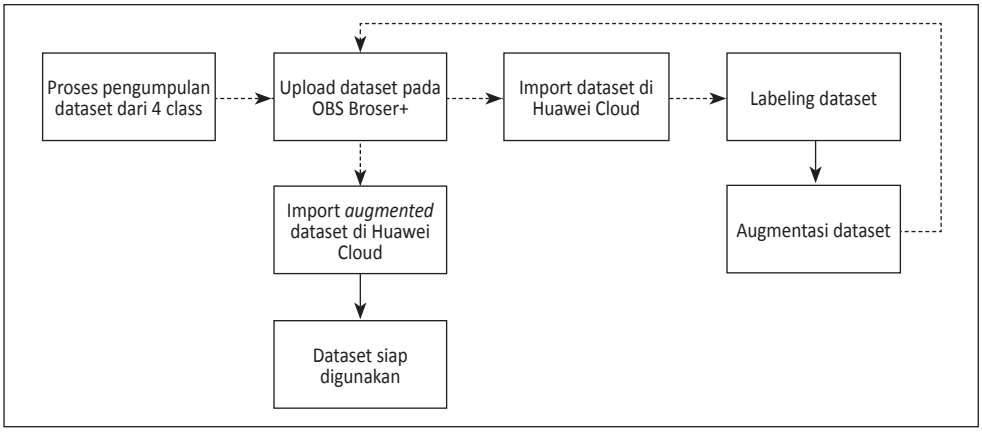
Gambar 7. Kerangka Penelitian [48]

RAM sebesar 8 GB dan implementasi modifikasi algoritma *maxpool* adalah dengan sebuah *notebook* dengan Processor Intel™ Co™TM) i7-8750H CPU @ 2.20GHz, 2208 Mhz, 6 Cores, RAM sebesar 16 GB. Tahapan evaluasi hasil penelitian melakukan evaluasi dari hasil program.

Metode penelitian disertasi ini menggunakan pendekatan eksperimental kuantitatif. Karakteristik utama dari pendekatan ini adalah adanya kontrol yang lebih besar terhadap lingkungan penelitian dan manipulasi beberapa viariabel untuk mengamati pengaruhnya terhadap viariabel lain [49].

Data sekunder ikan diperoleh dari situs web Flickr [50]. Gambar tersebut memiliki *creative commons attribution license* yang memungkinkan untuk pengguna saling berbagi dan menyesuaikan materi. Gambar 8 memperlihatkan pengolahan *dataset* yang diawali dari pengumpulan data berupa gambar dari empat *class* objek yang akan dilakukan deteksi, yaitu cakalang, tongkol, ikan lemadang, dan cumi-cumi, sebanyak seratus gambar untuk setiap *class*. Total *dataset* sebelum dilakukan augmentasi adalah empat ratus gambar.

Proses dilanjutkan dengan pengunggahan (*upload*) *dataset* pada *object storage* pada Huawei Cloud dengan OBS Browser+ dan proses impor data dari *object storage* untuk diolah pada proses *labelling*. Pelabelan (*labelling*) dilakukan dengan cara memberikan *bounding box* secara manual kepada objek yang ingin dilakukan deteksi. Selanjutnya, tahap augmentasi adalah menerapkan sembilan jenis filter pada *dataset* gambar yang telah diberikan label. Filter yang digunakan adalah *flip*, *rotate*, *scale*, *blur*, *crop*, *histogram equal*, *light contrast*, *light arithmetic*, dan *saturation*. Setelah



Gambar 8. Pengolahan *Dataset* [48]

dilakukan proses augmentasi, jumlah *dataset* yang ada menjadi total 4.000 gambar. Kemudian melakukan proses publikasi *dataset* untuk *training* dan data validasi dengan rasio 80:20. Gambar yang sudah siap diberi *bounding box* dan label *Katsuwonus pelamis, Euthynnus affinis, Coryphaena hippurus*, *Loligo chinenis*, dan *Person*. Label *Loligo chinensis* berjumlah lebih banyak karena satu gambar dapat berisi lebih dari satu cumi-cumi. Pada Huawei Cloud à ModelArts à Algorithm Management à Subscription memilih YOLOv3-Resnet sebagai algoritmanya.

C. HASIL DAN PEMBAHASAN

1. Modifikasi Algoritma Maxpool

Modifikasi algoritma Maxpool diilustrasikan seperti seseorang yang diberi tugas untuk memindahkan 64 penumpang dari satu lokasi ke lokasi lain dengan sebuah mobil kecil atau sebuah minibus. Jika ia menggunakan sebuah mobil kecil, ia perlu melakukan 16 kali perjalanan bolak-balik dengan sekali angkut 4 penumpang. Sementara itu, jika ia menggunakan sebuah minibus, ia cukup melakukan 4 kali perjalanan bolak-balik dengan sekali angkut 16 penumpang sekaligus. Dapat dibayangkan jika ternyata jarak tempuh perjalanan kedua lokasi tersebut berjauhan dan terjebak kemacetan. Dengan menggunakan mobil kecil, pemindahan 64 penumpang akan memakan waktu lebih lama dan bahan bakar lebih banyak dibandingkan menggunakan sebuah minibus.

Peneliti memperbanyak jumlah *maxpool* dari *maxpool* tensor 2-*axis*: [4,4] menjadi 16 *maxpool* dengan ukuran filter 2 x 2. Ukuran dari *maxpool* tensor 2-axis:[16,16] artinya memiliki panjang 16 dan lebar 16 (*TileBlock* berukuran 16x16).

Perhitungan *maxpool* dapat dilihat pada Gambar 9, dimulai ketika *row* = 0 dan *col* = 0, 4 baris dan 4 kolom pertama merupakan sebuah *maxpool* dengan filter 2 x 2 yang merupakan sebuah *maxpool* dengan 2-*axis* tensor (diberi garis kotak warna hitam). Dalam satu proses *maxpool* dengan filter 2 x 2 akan mendapat 4 buah nilai yang akan dicari nilai maksimal atau nilai terbesarnya. Proses *maxpool* tradisional akan

1	2	5	6	1	2	5	6	1	2	5	6	1	2	5	6
3	4	7	8	3	4	7	8	3	4	7	8	3	4	7	8
9	10	13	14	9	10	13	14	9	10	13	14	9	10	13	14
11	12	15	16	11	12	15	16	11	12	15	16	11	12	15	16
1	2	5	6	1	2	5	6	1	2	5	6	1	2	5	6
3	4	7	8	3	4	7	8	3	4	7	8	3	4	7	8
9	10	13	14	9	10	13	14	9	10	13	14	9	10	13	14
11	12	15	16	11	12	15	16	11	12	15	16	11	12	15	16
1	2	5	6	1	2	5	6	1	2	5	6	1	2	5	6
3	4	7	8	3	4	7	8	3	4	7	8	3	4	7	8
9	10	13	14	9	10	13	14	9	10	13	14	9	10	13	14
11	12	15	16	11	12	15	16	11	12	15	16	11	12	15	16
1	2	5	6	1	2	5	6	1	2	5	6	1	2	5	6
3	4	7	8	3	4	7	8	3	4	7	8	3	4	7	8
9	10	13	14	9	10	13	14	9	10	13	14	9	10	13	14
11	12	15	16	11	12	15	16	11	12	15	16	11	12	15	16

Gambar 9. 16 Buah Maxpool Tensor 2D

menghasilkan 4 buah nilai, yaitu 131, 130, 127, dan 127. Kemudian *col* bergeser 4 ke kanan, proses berulang untuk mencari 4 nilai terbesar dari *maxpool* berikutnya.

Pada algoritma *maxpool* yang dimodifikasi, disebut *Maxpool2DT4D*, ketika *row* = 0 dan *col* = 0, maka dicari ke-16 nilai untuk sel berwarna cokelat terlebih dahulu (lihat Gambar 10). Kemudian *col* naik 2 dan dimulai pencarian ke-16 nilai untuk sel berwarna kuning. Selanjutnya, *row* = 2 dan *col* = 0 untuk mencari ke-16 nilai sel berwarna hijau, kemudian *col* naik 2 untuk mencari ke-16 nilai sel berwarna biru. Nilai *output* dari *maxpool* sejumlah 64 sudah terhitung ketika proses ini selesai.

16	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	127	131	130	130	130	132	132	132	133	131	131	131	132	134	134	134
1	128	128	128	129	130	129	130	130	130	129	129	129	130	131	132	131
2	127	126	126	127	128	127	128	128	127	127	127	127	128	128	129	129
3	125	126	126	126	126	126	126	126	125	126	126	126	126	126	127	126
4	124	125	125	125	124	125	126	125	124	125	126	126	125	127	125	124
5	126	125	126	125	125	126	127	125	125	125	125	125	125	125	124	123
6	126	125	125	127	127	128	127	125	125	125	125	125	125	125	124	124
7	125	126	126	128	127	128	128	126	126	126	125	126	126	126	125	125
8	125	126	127	127	127	128	128	128	127	126	126	126	126	126	126	127
9	125	126	127	127	128	128	128	128	127	126	127	126	126	126	127	128
10	126	126	127	127	128	128	127	127	128	128	128	126	126	128	128	128
11	128	128	128	128	129	129	129	129	129	129	128	126	127	127	128	128
12	128	128	128	128	128	129	128	130	128	129	128	127	128	128	128	128
13	128	128	129	128	128	129	129	129	128	129	129	128	129	129	128	128
14	128	129	129	129	129	128	129	130	129	129	129	129	129	130	128	129
15	130	130	129	130	129	129	130	130	130	130	129	129	128	130	129	129

16	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	131				132				133				134			
1																
2																
3																
4	126				126				125				127			
5																
6																
7																
8	126				128				127				126			
9																
10																
11																
12	128				129				129				129			
13																
14																
15																

Gambar 10. Perhitungan Maxpool2DT4D untuk Sel Berwarna Cokelat

Maka *row* = 0 dan *col* bisa langsung melompat ke-16 dan proses berulang untuk mengolah *TileBlock* berikutnya. Berdasarkan cara kerja di atas, terlihat *Maxpool2DT4D* tidak memerlukan sumber daya tambahan dari segi perangkat keras (*hardware*), melainkan mengubah susunan pengulangannya.

Hal yang membedakan antara algoritma *maxpool* tradisional dan *Maxpool2DT4D* hasil modifikasi adalah dari banyaknya jumlah iterasi. Sebagai contoh, ada sebuah *TileBlock* sebesar 16 x 16 dengan ukuran filter sebesar 2 x 2. Metode *maxpool* tradisional mencakup 16 kali iterasi untuk mendapatkan 4 nilai yang menghasilkan 64 dari *TileBlock* tersebut. Sementara *Maxpool2DT4D* melakukan hanya 4 kali iterasi untuk mendapatkan 16 nilai yang menghasilkan 64.

Algoritma *Maxpool2D* dan *Maxpool2DT4D* diimplementasikan menggunakan bahasa pemograman Python 3.75. Library Package, yaitu OpenCV dan *numpy*.

```
def pool2dt4d_16x16(img, tileMaxpool=16):
    '''
    :img = 426 x 426 (Yolvov3 input resolution 416x416)
    :out2 = 213 x 213
    :param img: input image
    :param tileMaxpool: window size
    :return:
    '''
    H, W = img.shape
    filterMaxpool = int(math.sqrt(tileMaxpool)) # 16 --> 4
    cellMaxpool = int(math.sqrt(filterMaxpool)) # 4 --> 2
    Nh = H // cellMaxpool   # Nh = NewHeight
    Nw = W // cellMaxpool   # Nw = NewWidth

    # img   size = 345 x 518 divide by 16 --> 21 tileBlockRow
    #              plus 9 pixel x 32 tileBlockCol plus 6 pixel
    # image size = 348 x 520 (multiple of filterMaxpool)
    if (H % tileMaxpool) == 0:
        maxpoolMaxRow = math.floor(H // tileMaxpool)
    else:
        maxpoolMaxRow = math.floor(H // tileMaxpool) + 1
    if (W % tileMaxpool) == 0:
        maxpoolMaxCol = math.floor(W // tileMaxpool)
    else:
        maxpoolMaxCol = math.floor(W // tileMaxpool) + 1

    totalRow = maxpoolMaxRow * tileMaxpool
    totalCol = maxpoolMaxCol * tileMaxpool
    image = np.zeros((totalRow, totalCol))
    image[0:H, 0:W] = img
    temp = np.zeros((totalRow//2+filterMaxpool, totalCol//2+filterMaxpool))
    m = 0
```

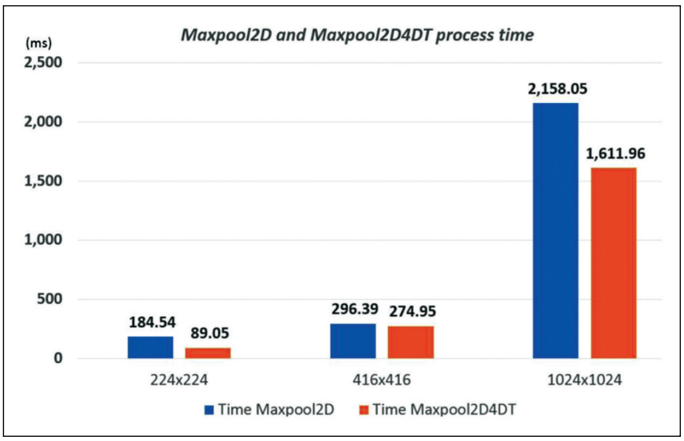
Gambar 11. *Maxpool2D*

```
    for maxpoolRow in range(0, totalRow, tileMaxpool):
        for maxpoolCol in range(0, totalCol, tileMaxpool):
            # Get a tileBlock consists 16x16
            lastPart = image[maxpoolRow:maxpoolRow + tileMaxpool, maxpoolCol:maxpoolCol + tileMaxpool]
            blr, blc = np.shape(lastPart)  # get the shape of the lastPart
            if (blr != tileMaxpool) or (blc != tileMaxpool):  # Check if we are the last block
                aBlock = np.zeros((tileMaxpool,tileMaxpool))
                aBlock[0:blr, 0:blc] = lastPart    # copy the lastPart into 16x16
            else:
                aBlock = lastPart
            # We process each cell. 16 values are calculated at a time
            for cellRow in range(cellMaxpool):
                for cellCol in range(cellMaxpool):
                    if wantDebug:
                        print("\nmaxpoolRow {}, maxpoolCol {}, cellRow {}, cellCol {}"
                              .format(maxpoolRow, maxpoolCol, cellRow, cellCol))
                    for row in range(filterMaxpool):
                        for col in range(filterMaxpool):
                            # Take oneCell, find itsMax
                            sourceRow  = (row * filterMaxpool) + (cellMaxpool * cellRow)
                            sourceRow2 = (row * filterMaxpool) + (cellMaxpool * (cellRow+1))
                            sourceCol  = (col * filterMaxpool) + (cellMaxpool * cellCol)  # 0, 4, 8, 12
                            sourceCol2 = (col * filterMaxpool) + (cellMaxpool * (cellCol+1))  # 0, 4, 8, 12
                            oneCell = aBlock[sourceRow:sourceRow2, sourceCol:sourceCol2]

                            targetRow = maxpoolRow//2 + (row * cellMaxpool + cellRow)
                            targetCol = maxpoolCol//2 + (col * cellMaxpool + cellCol)
                            itsMax = np.max(aBlock[sourceRow:sourceRow2, sourceCol:sourceCol2])
                            if wantDebug:
                                print("maxPool #{}, oneCell[{},{}] = \n {} \ntemp[{},{}] = {}"
                                      .format(m, sourceRow, sourceCol, oneCell, targetRow, targetCol, itsMax))
                            temp[targetRow, targetCol] = itsMax
                            m = m + 1
    out2 = temp[0:Nh, 0:Nw]
    print("total maxPool2dt4d = ", m)
    print(out2.shape)
    return out2
```

Gambar 12. *Maxpool2DT4D*

Untuk membuktikan bahwa algoritma *Maxpool2DT4D* hasil modifikasi lebih cepat dibandingkan *maxpool* tradisional, maka dilakukan percobaan untuk memproses sebuah foto dengan resolusi berbeda-beda, yaitu 224x224, 416x416 dan 1024x1024 piksel (Gambar 13).



Gambar 13. Grafik Perbandingan Maxpool dan Maxpool2DT4D

Sesuai dengan analisis algoritma, program Maxpool2D terlihat menggunakan struktur pengulangan dua lapis, sedangkan Maxpool2DT4D menggunakan struktur pengulangan enam lapisan (*nested loops*) dan memiliki notasi Big-O(n^2). Tabel 2 dan Tabel 3 memperlihatkan analisis jumlah *loop* untuk setiap resolusi *maxpool2D* dan *maxpool2DT4D* dan Gambar 14 memperlihatkan grafik jumlah loopnya, terjadi penurunan jumlah *loop* sebanyak 25%.

Tabel 2 Jumlah *Loop* Maxpool2d

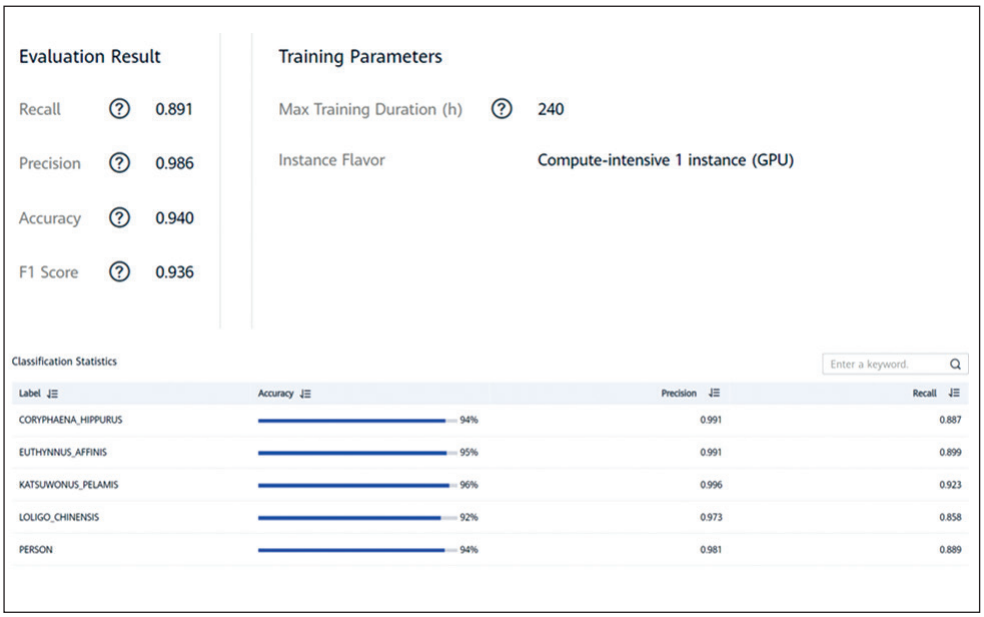
Lebar x Tinggi	Resolusi (n)	y	x	Jumlah Loop maxpool2D
224x224	224	224	224	50,176
512x512	512	512	512	262,144
1024x1024	1,024	1,024	1,024	1,048,576
5120x5120	5,120	5,120	5,120	26,214,400
10240x10240	10,240	10,240	10,240	104,857,600

Tabel 3 Jumlah *Loop* Maxpool2d4td

Lebar x Tinggi	Resolusi (n)	maxpolRow	maxpoolCol	cellRow	cellCol	row	col	Jumlah Loop maxpool2DTD
224x224	224	14	14	4	4	2	2	12,544
512x512	512	32	32	4	4	2	2	65,536
1024x1024	1,024	64	64	4	4	2	2	262,144
5120x5120	5,120	320	320	4	4	2	2	6,553,600
10240x10240	10,240	640	640	4	4	2	2	26,214,400

Gambar 14. Grafik Jumlah *Loop* Maxpool2D dan Maxpool2DT4D

2. Pengenalan Objek

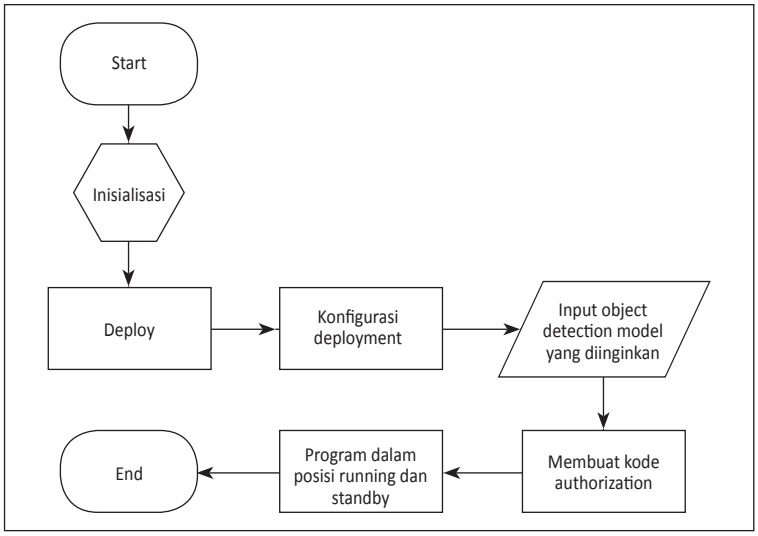


Gambar 15. Hasil *Training* Pengenalan Objek

Gambar 15 menunjukkan proses *training* menggunakan 1 buah GPU dengan hasil *recall* sebesar 0,891, *precision* sebesar 0,896, *accuracy* sebesar 0,940, dan *F1 score* 0,936.

Spesifikasi *object detection model* diperoleh setelah proses *training* selesai. Spesifikasi ini didapatkan dari hasil pengujian atau evaluasi yang sudah dilakukan secara otomatis ketika proses *training* berlangsung. *Recall* merupakan indikator untuk mengetahui rasio antara data *true positive* yang sudah diberi label oleh *object detection model* terhadap data yang sebenarnya pada realita atau seberapa banyak prediksi yang benar dari semua ikan sesungguhnya. *Precision* merupakan rasio antara prediksi *true positive* oleh *object detection model* terhadap data *true positive* yang diberi label oleh pengguna (*user*) atau seberapa banyak ikan cakalang yang sebenarnya diberi label oleh pengguna. *Accuracy* merupakan rasio antara pemberian *label* yang benar pada objek atau target deteksi terhadap seluruh objek yang ingin dilakukan deteksi atau seberapa banyak objek yang diberikan label dengan benar terhadap objek yang ingin dideteksi. Kemudian, *F1 score* merupakan *harmonic average* antara *precision* dan *recall* atau dapat disebut juga keseimbangan antara *precision* dan *recall*. *Object detection model* ini sudah siap digunakan untuk melakukan deteksi objek yang diinginkan.

Gambar 16 menunjukkan diagram alir proses *deploy* yang dilakukan dalam Huawei Cloud pada layanan *real-time services*. Setelah memilih menu *deploy*, pengguna akan diarahkan untuk langsung melakukan konfigurasi, seperti memilih jenis *object detection model* yang ingin digunakan, *timer*, aksesibilitas, dan jenis perangkat keras yang ingin digunakan. Dalam melakukan implementasi, peneliti menggunakan



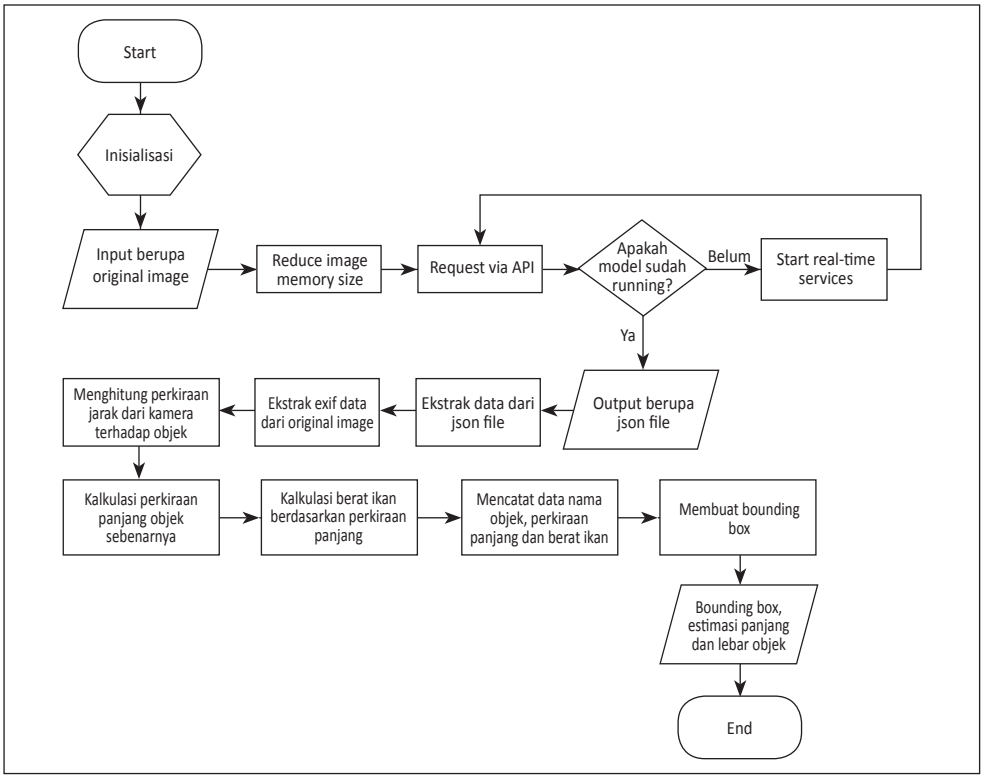
Gambar 16. Diagram Alir Model *Deployment*

object detection model yang sebelumnya sudah selesai dalam proses *training* serta memilih *public resource pools* agar dapat diakses oleh banyak pengguna dalam waktu bersamaan dan perangkat keras 2 vCPUs dengan memori RAM sebesar 8 GB. Setelah konfigurasi selesai, langkah selanjutnya adalah menambah *code authorization* yang berfungsi sebagai kode akses *object detection model* yang sedang berjalan pada layanan *real-time services* Huawei Cloud.

3. Estimasi Panjang dan Berat

Seperti yang terlihat pada Gambar 17, pada saat mulai, pengguna memberikan *input image* yang dapat diambil dari berbagai macam sumber berbeda. Namun, jika ingin melakukan kalkulasi perkiraan jarak dari kamera terhadap objek, perkiraan panjang dan berat objek diperlukan *input image* yang memiliki data *exchangeable image file* (EXIF). Biasanya, data EXIF tertanam bersama *image file* yang diambil dari kamera digital. Pengguna juga dapat melakukan *image input* yang tidak memiliki data EXIF, tetapi *output* hanya akan berupa *bounding box* dari objek yang terdeteksi dan tidak dapat melakukan estimasi jarak dari kamera terhadap objek serta perkiraan panjang dan berat objek

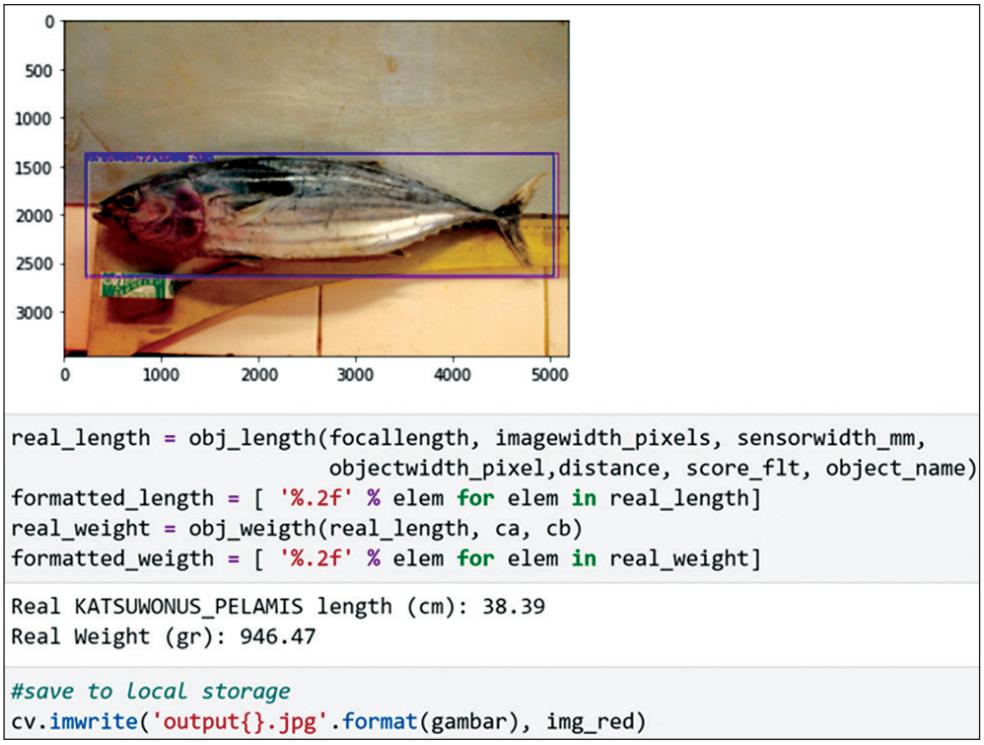
Uji coba kali ini melibatkan sumber *input image* yang memiliki data EXIF dan tidak. Input *image* yang memiliki data EXIF dari kamera digital akan memiliki ukuran lebih dari 1 MB sehingga diperlukan *compress image size* agar data gambar yang dikirim kepada Huawei Cloud tidak mengalami *time out* karena membutuhkan waktu di atas 30 detik. Setelah berhasil diperkecil, gambar akan dikirim pada *Huawei Cloud* melalui API dan mendapatkan *authorization code*, kemudian dideteksi dan akan mengirimkan kembali *output* kepada pengguna berupa format *.json* yang berisikan *class name*, *bounding box coordinate*, dan *detection score*. Namun, layanan *real-time*



Gambar 17. Diagram Alir Uji Coba

services harus diaktifkan terlebih dahulu untu mendapatkan *output* dengan format tersebut. Data dari .json *file* ini berfungsi untuk menggambar *bounding box* pada *local environment*. Data koordinat *bounding box* juga dapat digunakan untuk menemukan perkiraan panjang objek dengan satuan piksel. Setelah data .json *file* berhasil diolah, program sudah dapat membuat *bounding box* dalam *local environment.*

Selanjutnya adalah tahap pengukuran estimasi jarak antara kamera dan objek. Data estimasi jarak antara kamera dan objek ini diperoleh dari data EXIF pada sebuah gambar yang ditangkap menggunakan kamera digital (penelitian menggunakan Canon Kiss X7 dan Panasonic DMC-FH1). Data EXIF akan mengambil perkiraan jarak terjauh (*distance upper*) dan perkiraan jarak terdekat (*distance lower*), kemudian didapatkan nilai rata-rata yang akan digunakan untuk estimasi jarak dari kamera terhadap objek. Setelah itu, diperlukan juga *focal length* dan *image width pixel* dari data EXIF tersebut. Terakhir, diperlukan juga *sensor width* yang didapatkan dari data spesifikasi kamera yang akan digunakan untuk mencari perkiraan panjang objek. Setelah mendapatkan perkiraan panjang dan berat objek, data akan disimpan dalam bentuk .csv *file* beserta dengan *class name* pada *local storage* untuk mempermudah melakukan pencatatan data dari objek yang sudah diketahui, dalam kasus ini berarti mencatat juga seberapa banyak hasil tangkapan ikan yang sudah diperoleh dan data juga dapat diolah lagi setelah itu.




```
real_length = obj_length(focallength, imagewidth_pixels, sensorwidth_mm,
                         objectwidth_pixel,distance, score_flt, object_name)
formatted_length = [ '%.2f' % elem for elem in real_length]
real_weight = obj_weigth(real_length, ca, cb)
formatted_weigth = [ '%.2f' % elem for elem in real_weight]
```

```
Real KATSUWONUS_PELAMIS length (cm): 38.39
Real Weight (gr): 946.47
```

```
#save to local storage
cv.imwrite('output{}.jpg'.format(gambar), img_red)
```

Gambar 18. Perhitungan Panjang Ikan [48]

Uji coba dengan gambar yang memiliki data EXIF akan menggunakan sepuluh gambar yang telah ditangkap menggunakan kamera Canon Kiss X7. Data yang diambil adalah ikan cakalang. Gambar dikirim pada *real-time services* Huawei Cloud untuk mendapatkan hasil prediksi, data *class name*, *coordinate*, dan *prediction score*. Gambar yang dikirim akan dilakukan kompresi terhadap *file size* terlebih dahulu.

Kalkulasi penjang ikan diperoleh dari *bounding box* yang terdeteksi kemudian diambil selisih koordinat x-nya, mengambil jarak informasi EXIF sehingga diketahui jaraknya, serta menampilkan gambar ikan dalam piksel, kelas beserta skornya, panjang dan beratnya. Sebagai contoh, ikan tongkol dengan *ground truth* panjang 46 cm dan bobot 1.410 gram dengan jarak antara kamera dan ikan sejauh 50 cm. Hasil percobaan mencatat estimasi panjang dan berat ikan tongkol sebanyak sepuluh kali pengambilan foto dengan perhitungan persentase eror estimasi panjang diperoleh 2,59%.

%Error Estimasi panjang = $-X100$

Error Estimasi panjang = 2,59%

Sedangkan persentase *error* estimasi bobot diperoleh 15.67%.

$$\%\text{Error Estimasi bobot} = \frac{1630{,}94 - 1410}{-410} \times 100$$

Error Estimasi bobot = 15,67%

Data dari hasil uji coba yang diperoleh menunjukkan bahwa *object detection model* mampu mengenali perbedaan setiap *class* dengan baik. Terlihat rerata tingkat persentase ketepatan mendeteksi sebesar 89,55% sebagai bukti keberhasilan deteksi yang sangat baik (lihat Tabel 4). Hasil penelitian yang dilakukan lebih baik dibandingkan beberapa penelitian yang menggunakan CCN dengan hasil 81,4% [27], 83,2% [28], dan 88,12% [9].

Tabel 4. Ketepatan Prediksi Tiap Kelas

No	Kelas	Akurasi (%)
1.	Katsuwonus Pelamis	82,30
2.	Euthynuss Affinis	91,40
3.	Coryphaena Hippurus	89,40
4.	Loligo Chinensis	95,10
	Rerata	**89,55**

Algoritma YOLOV3-Resnet18 sudah banyak digunakan dan beberapa di antaranya sudah menjadi produk komersial untuk mendeteksi objek dan mengenalinya, tetapi *dataset* yang digunakan pada umumnya menggunakan ImageNet. YOLOV3-Resnet18 yang dilatih dengan *dataset* sendiri sebanyak 4.000 gambar ikan dengan masing-masing 1.000 gambar tiap kelasnya dapat mengenali cakalang, tongkol, ikan lemadang, dan cumi-cumi.

D. KESIMPULAN DAN SARAN

Percepatan proses pada *maxpool layer* adalah dengan mengurangi jumlah *loop* dan memperbanyak perhitungan hasil dalam setiap iterasinya. Hasil percobaan memperoleh percepatan Maxpool2DT4D dengan rata-rata 221.01 ms (27,99%) dan kompleksitas *maxpool* dapat diturunkan menjadi 25% dari jumlah *loop* semula. Cara mendeteksi objek ikan dari sebuah gambar dengan latar belakang berbeda-berbeda dan jarak yang statis adalah dengan melakukan *training* gambar cakalang, tongkol, ikan lemadang, dan cumi-cumi menggunakan YOLOV3-Resnet18 yang dapat mendeteksi objek dengan tingkat akurasi 89,55%. Perhitungan estimasi panjang objek yang ditangkap oleh kamera digital ditentukan oleh besarnya ukuran APS-C *sensor size* dalam penelitian ini adalah 22,3 x 14,9 mm sehingga jika menggunakan kamera lain dengan ukuran sensor berbeda dapat menyesuaikan dalam program perhitungannya. Penempatan kamera terhadap objek secara perpendikular mengestimasi panjang objek yang tertangkap pada sensor lensa kamera dalam piksel dengan mendeteksi *bounding box*-nya, jarak kamera dengan objek yang diketahui, dan karakteristik lensa kamera yang digunakan sehingga panjang ikan yang sesungguhnya dapat dihitung. Dari hasil percobaan diperoleh eror estimasi panjang sebesar 2,59% dan eror estimasi berat sebesar 15,67%.

Saran pengembangan yang dapat diberikan adalah melanjutkan penelitian dengan menerima input berupa video dan melakukan uji coba secara langsung dengan menempatkan kamera pada kapal nelayan berjarak 50 cm perpendikular dari atas dek kapal dan kamera dipasang pada saat akan ikan masuk ke lubang palka. Proses kalibrasi dilakukan setelah menempatkan kamera dengan mengukur seekor ikan dengan panjang diketahui dan menyesuaikan faktor kalibrasi sehingga sistem dapat mendeteksi ikan dengan panjang yang sebenarnya. Sistem dikembangkan lebih lanjut untuk dapat diterapkan pada kondisi lapangan yang sesungguhnya dengan memperhatikan juga faktor kabut, hujan, dan suasana terang atau mendung. Sistem mengambil titik GPS kapal nelayan ketika melakukan penangkapan. Cara kerja berbasis Maxpool2DT4D dapat diadaptasikan untuk melakukan modifikasi algoritma *convolve2D*. Perlu penelitian lebih lanjut untuk mengalamati permasalahan *resources* dan *computational time*.

DAFTAR PUSTAKA

[1] Hariono. "Nilai dan Volume Ekspor Tuna, Cakalang, Tongkol Periode Januari-Maret (Triwulan I) Tahun 2019 Mengalami Kenaikan." KKP.go.id. Diakses pada 16 Oktober 2019 [Daring.] https://kkp.go.id/djpdspkp/bbp2hp/artikel/11444-nilai-dan-volume-ekspor-tuna-cakalang-tongkol-periode-januari-maret-triwulan-i-tahun-2019-mengalami-kenaikan

[2] M. Ambari. "Menelusuri Keberadaan Tuna yang Terancam Punah di Indonesia." Mongabay.co.id. Diakses pada 17 April 2018 [Daring.] https://www.mongabay.co.id/2017/02/20/menelusuri-keberadaan-tuna-yang-terancam-punah-di-indonesia/

[3] Undang-Undang Republik Indonesia Nomor 16 Tahun 1997 Tentang Statistik (Indonesia, 1997). Diakses pada 16 Oktober 2019. [Daring.] https://peraturan.bpk.go.id/Home/Details/45944

[4] S. Triharyuni dan B. I. Prisantoso, "Species and size composition of tuna longline catches in the Southes of Java, Indian ocean," *Jurnal Saintek Perikanan*, vol. 8, no. 1, pp. 52–58, 2012. https://doi.org/10.14710/ijfst.8.1.52-58

[5] M. R. M. Shafry, A. Rehman, R. Kumoi, N. Abdullah, dan T. Saba, "A new approach in measuring fish length using fish length from digital images (FiLeDI) framework," *International Journal of the Physical Sciences*, vol. 7, no. 4, pp. 607–618, 2012. https://doi.org/10.5897/ijps11.1581

[6] C. I. Anumudu dan T. O. Mojekwu, "Advanced techniques for morphometric analysis in fish," *Journal of Aquaculture Research & Development*, vol. 06, no. 08, pp. 6–11, 2015. https://doi.org/10.4172/2155-9546.1000354

[7] M. H. Jamaluddin dkk., "The effectiveness of fish length measurement system using non-contact measuring approach," *Jurnal Teknologi*, vol. 77, no. 20, pp. 67–74, 2015. https://doi.org/10.11113/jt.v77.6554

[8] M. Man, N. Abdullah, M. S. M. Rahim, dan I. M. Amin, "Fish length measurement: The results from different types of digital camera," *Journal of Advanced Agricultural Technologies*, vol. 3, no. 1, pp. 67–71, 2016. https://doi.org/10.18178/joaat.3.1.67-71

[9] M. Hao, H. Yu, dan D. Li, "The measurement of fish size by machine vision: A review," dalam *9th International Conference on Computer and Computing Technologies in Agriculture (CCTA)*, 2017.

[10] G. Sanchez-Torres, A. Ceballos-Arroyo, dan S. Robles-Serrano, "Automatic measurement of fish weight and size by processing underwater hatchery images," *Engineering Letters*, vol. 26, no. 4, pp. 461–472, 2018.

[11] R. Islamadina, N. Pramita, F. Arnia, K. Munadi, dan T. M. Iqbal, "Pengukuran badan ikan berupa estimasi panjang, lebar, dan tinggi berdasarkan visual capture," *Jurnal Nasional Teknik Elektro dan Teknologi Informasi (JNTETI)*, vol. 7, no. 1, pp. 57–63, 2018. https://doi.org/10.22146/jnteti.v7i1.401

[12] G. Habib dan S. Qureshi, "Optimization and acceleration of convolutional neural networks: A survey," *Journal of King Saud University: Computer and Information Sciences*, vol. 34, no. 7, pp. 4244–4268, 2020. https://doi.org/10.1016/j.jksuci.2020.10.004

[13] NVIDIA. "Developer Resources for Deep Learning and AI." NVIDIA. Diakses pada 30 November 2018. [Daring.] https://www.nvidia.com/en-us/deep-learning-ai/developer/

[14] A. Baldominos, Y. Saez, dan P. Isasi, "A survey of handwritten character recognition with MNIST and EMNIST," *Applied Sciences (Switzerland)*, vol. 9, no. 15, 2019. https://doi.org/10.3390/app9153169

[15] S. Liawatimena dkk., "A fish classification on images using transfer learning and Matlab," dalam *1st 2018 Indonesian Association for Pattern Recognition International Conference, INAPR 2018 - Proceedings*, 2019, pp. 108–112. https://doi.org/10.1109/INAPR.2018.8627007

[16] E. Nurdin dan A. S. Panggabean, "Musim penangkapan dan struktur ukuran cakalang (Katsuwonus pelamis Linnaeus, 1758) di sekitar rumpon di Perairan Palabuhanratu," *Jurnal Penelitian Perikanan Indonesia*, vol. 23, no. 4, pp. 299, 2017. https://doi.org/10.15578/jppi.23.4.2017.299-308

[17] Z. Zou, Z. Shi, Y. Guo, dan J. Ye, "Object detection in 20 years: A survey," arXiv:1905.05055v2 [cs.CV], pp. 1–39, 2019. doi: http://arxiv.org/abs/1905.05055

[18] B. Hariharan, P. Arbeláez, R. Girshick, dan J. Malik, "Simultaneous detection and segmentation," dalam *Computer Vision-ECCV 2014: Part VII*, D. Fleet, T. Padjla, B. Schiele, dan T. Tuytelaars, Eds., Zurich, Swiss, 2014, pp. 297–312. https://doi.org/10.1007/978-3-319-10584-0_20

[19] B. Hariharan, P. Arbeláez, R. Girshick, dan J. Malik, "Hypercolumns for object segmentation and fine-grained localization," dalam *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, pp. 447–456. https://doi.org/10.1109/CVPR.2015.7298642

[20] J. Dai, Y. Li, K. He, dan J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," dalam *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, dan R. Garnett, Eds., 2016, pp. 379–387.

[21] K. He, G. Gkioxari, P. Dollar, dan R. Girshick, "Mask R-CNN," dalam *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2980–2988. https://doi.org/10.1109/ICCV.2017.322

[22] A. Karpathy dan L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," dalam *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3128–3137.

[23] K. Xu dkk., "Show, attend and tell: Neural image caption generation with visual attention," dalam *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37, 2015 pp. 2048–2057.

[24] Q. Wu, C. Shen, P. Wang, A. Dick, dan A. Van Den Hengel, "Image captioning and visual question answering based on attributes and external knowledge," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 6, pp. 1367–1381, 2018. https://doi.org/10.1109/TPAMI.2017.2708709

[25] K. Kang dkk. "T-CNN: Tubelets with convolutional neural networks for object detection from videos," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 2896–2907, 2018. https://doi.org/10.1109/TCSVT.2017.2736553

[26] Y. Lecun, Y. Bengio, dan G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. https://doi.org/10.1038/nature14539

[27] S. Jin, X. Yan, H. Zhang, dan W. Fan, "Weight-length relationships and Fulton's condition factors of skipjack tuna (Katsuwonus pelamis) in the western and central Pacific Ocean," *PeerJ*, vol. 2015, no. 2, 2015 https://doi.org/10.7717/peerj.758

[28] X. Li, M. Shang, H. Qin, dan L. Chen, "Fast accurate fish detection and recognition of underwater images with fast R-CNN," dalam *OCEANS 2015 – MTS/IEEE Washington*, Washington DC, AS, 2015, pp. 1–5. doi: 10.23919/OCEANS.2015.7404464

[29] S. Cui, Y. Zhou, Y. Wang, L. Zhai, "Fish detection using deep learning," *Applied Computational Intelligence and Soft Computing*, vol. 2020, Art. no. 3738108, 2020. doi: https://doi.org/10.1155/2020/3738108

[30] E. M. Ditria, S. Lopez-Marcano, M. Sievers, E. L. Jinks, C. J. Brown, dan R. M. Connolly, "Automating the analysis of fish abundance using object detection: Optimizing animal ecology with deep learning," *Frontiers in Marine Science*, vol. 7, no. June, pp. 1–9, 2020. https://doi.org/10.3389/fmars.2020.00429

[31] M. Dhankar dan N. Walia, "An introduction to artificial intelligence," dalam *Emerging Trends in Big Data, IoT and Cyber Security*, M. Kumar, R. Choudary, dan S. K. Pandhey, Eds. 2020, pp. 105–111.

[32] A. Entwistle, "What is artificial intelligence?" *Engineering Materials and Design*, vol. 32, no. 3), pp. 1–14, 1988. https://doi.org/10.1201/9781003080626-1

[33] S. Shalev-Shwartz dan S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge, Inggris: Cambridge University Press, 2014.

[34] A. Giusti, D. C. Cireşan, J. Masci, L. M. Gambardella, dan J. Schmidhuber, "Fast image scanning with deep max-pooling convolutional neural networks," dalam *2013 IEEE International Conference on Image Processing, ICIP 2013 - Proceedings*, 2013, pp. 4034–4038. https://doi.org/10.1109/ICIP.2013.6738831

[35] K. Raza dan S. Hong, "Fast and accurate fish detection design with improved yolo-v3 model and transfer learning," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 2, pp. 7–16, 2020. https://doi.org/10.14569/ijacsa.2020.0110202

[36] O. Russakovsky dkk., "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015. https://doi.org/10.1007/s11263-015-0816-y

[37] S. Shirmohammadi dan A. Ferrero, "Camera as the instrument: The rising trend of vision based measurement," *IEEE Instrumentation and Measurement Magazine*, vol. 17, no. 3, pp. 41–47, 2014. https://doi.org/10.1109/MIM.2014.6825388

[38] S. Albawi, T. A. Mohammed, dan S. Alzawi, "Understanding of a convolutional neural network," dalam *2017 International Conference on Engineering and Technology (ICET)*, Antalya, Turki, 2017. doi: 10.1109/ICEngTechnol.2017.8308186

[39] Karpathy, A. (2018). *CS231n Convolutional Neural Networks for Visual Recognition*. Diakses pada 10 Desember 2019. [Daring.] https://cs231n.github.io/

[40] P. Singh, P. Raj, dan V. P. Namboodiri, "EDS pooling layer," *Image and Vision Computing*, vol. 98, Art. no. 103923, 2020. https://doi.org/10.1016/j.imavis.2020.103923

[41] C. Y. Lee, P. W. Gallagher, dan Z. Tu, "Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree," dalam *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016*, 2016, pp. 464–472.

[42] M. J. Holden dan D. F. S. Raitt. *Manual Of Fisheries Science Part 2 - Methods of Resource Investigation and their Application*, 1974. [Daring.] http://www.fao.org/3/F0752E/F0752E03.htm

[43] Anonymous. (2016). *Skipjack tuna Updated : December 2016 Skipjack tuna Updated : December 2016. December*, 1–15.

[44] W . Dimmlich. *Smart fishing initiative (SFI): Species overview skipjack tuna (Katsuwonus pelamis)*, 2015, pp. 1–5. Diakses pada 10 Desember 2018. [Daring.] https://wwfint.awsassets.panda.org/downloads/fact_sheet_wwf_sfi_skipjack_tuna_april_2015_final.pdf

[45] I. Artetxe-Arrate dkk., "A review of the fisheries, life history and stock structure of tropical tuna (skipjack Katsuwonus pelamis, yellowfin Thunnus albacares and bigeye Thunnus obesus) in the Indian Ocean. *Advances in Marine Biology*, vol. 8, pp. 89–99, 2020. doi: https://doi.org/10.1016/bs.amb.2020.09.002

[46] L. H. Baidhowie, S. Redjeki, H. Endrawati, S. Rejeki, dan H. Endrawati, "Morfometri dan komposisi isi lambung Katsuwonus pelamis yang didaratkan di Pantai Puger Jember Jawa Timur," *Journal of Marine Research*, vol. 8, no. 1, pp. 69–74, 2019. https://ejournal3.undip.ac.id/index.php/jmr/article/view/24331

[47] J. S. Huxley, *Problems of Relative Growth*. New York: The Dial Press, 1932.

[48] C. R. Kothari dan G. Garg, *Research Methodology: Methods And Techniques* (edisi keempat). New Delhi, India: New Age International, 2019.

[49] S. Liawatimena, "Akselerasi maxpool pada pengenalan obyek dan pengukuran panjang Katsuwonus Pelamis, Euthynnus Affinis, Coryphaena Hippurus dan Loligo Chinensis menggunakan Convolutional Neural Network," Disertasi Doktoral, Universitas Bina Nusantara, 2022.

[50] Anonymous. (n.d.) Cakalang. Diakses pada 7 Mei 2018. [Daring.] https://www.flickr.com/photos/tags/cakalang/

[51] S. Liawatimena. (2022). Cakalang. Diakses pada 19 Mei 2022. [Daring.] https://github.com/sliawatimena/cakalang