Rancang Bangun *Object Detection* Pada Robot *Soccer* Menggunakan Metode *Single Shot Multibox Detector* (SSD MobileNetV2)

Cokorda Gde Wahyu Pramana[1], Duman Care Khrisne[2], Nyoman Putra Sastra[3]
[1,2,3] Program Studi Teknik Elektro, Fakultas Teknik, Universitas Udayana
Jl. Raya Kampus UNUD, Kampus Bukit Jimbaran, Jimbaran, Kabupaten Badung, Bali
cokwahyupramana@student.unud.ac.id[1], duman@unud.ac.id[2], putra.sastra@unud.ac.id[3]

ABSTRAK

Kecerdasan buatan atau *artificial intelligence* (AI) adalah teknologi yang menekankan pada kecerdasan mesin dalam memberikan respon layaknya manusia yang dikembangkan untuk membantu menunjang pekerjaan manusia. AI telah banyak diterapkan di berbagai bidang seperti industri, medis, pendidikan, bisnis, dan robotika. Pengembangan AI di bidang robotika menghasilkan robot *autonomous*, salah satu contohnya adalah robot *soccer* KRSBI-Beroda. Pada penelitian ini membahas mengenai penerapan AI dalam rancang bangun sistem pendeteksian objek menggunakan model *single shot multibox detector* (SSD) pada robot *soccer* KRSBI-Beroda. Penelitian ini bertujuan untuk menghasilkan model AI berupa jaringan saraf tiruan (JST) yang ditanamkan pada robot *soccer* agar dapat membedakan objek bola, gawang, robot, dan *obstacle*. Sistem pendeteksian objek ini dibangun dengan menggunakan metode *deep learning* yang dibantu dengan *framework* TensorFlow Object Detection API dengan menggunakan model SSD MobileNetV2 yang dijalankan menggunakan bahasa pemograman *python* pada *board* NVIDIA Jetson Nano yang terintegrasi pada kamera *webcam* C922 Pro. Model yang dibangun menggunakan *dataset* sebanyak 977 gambar yang terdiri dari 3064 objek di dalamnya yang dilatih sebanyak 200.000 *steps* pada Google Colaboratory. Hasil penelitian menunjukkan model dengan mAP rata-rata 0.80 dengan total *loss* rata-rata 1.5. Validasi terhadap model menghasilkan tingkat keberhasilan prediksi objek dengan akurasi rata-rata hingga 98.45 %.
Kata Kunci: *Artificial Intelligence* (AI), TensorFlow, Robot *Soccer*, SSD MobileNetV2

ABSTRACT

Artificial intelligence or AI is a technology that emphasizes machine intelligence in responding like humans, developed to help support human work. AI has widely applied in various fields such as industry, medical, education, business, and robotics. The development of AI in the field of robotics produces autonomous robots, one example is the KRSBI-Beroda soccer robot. This research discusses the application of AI in the design of object detection systems using the single-shot multibox detector (SSD) model on the KRSBI-Beroda soccer robot. This study aims to produce an AI model in the form of an artificial neural network (ANN) implanted in the soccer robot to distinguish between ball, goal, robot, and obstacle objects. This object detection system was built using a deep learning method assisted by the TensorFlow Object Detection API framework using the MobileNetV2 SSD model which is run using the python programming language on the NVIDIA Jetson Nano board which is integrated into the C922 Pro webcam camera. The model was built using a dataset of 977 images consisting of 3064 objects that were trained as many as 200,000 steps on Google Colaboratory. The results showed a model with an average mAP of 0.80 with an average total loss of 1.5. Validation of the model resulted in a success rate of object prediction with an average accuracy of up to 98.45%.
Key Words: *Artificial Intelligence* (AI), *TensorFlow, Soccer Robot, SSD MobileNetV2*

1. Pendahuluan.

KRSBI-Beroda (Kontes Robot Sepak Bola Indonesia Beroda) adalah salah satu divisi

yang dipertandingkan pada Kontes Robot Indonesia (KRI) yang kompetisi robot tingkat Nasional yang diselenggarakan Belmawa Kemenristekdikti. Pada kompetisi KRSBI, robot yang dipertandingkan adalah robot *soccer*, yaitu robot yang dapat bermain sepak bola layaknya manusia.

Robot *soccer* KRSBI-Beroda yang digunakan merupakan robot yang bergerak atau ber-manuver menggunakan roda dan melakukan permainan secara otomatis tanpa adanya campur tangan manusia (*autonomous*) dengan memanfaatkan sensor visual (kamera) sebagai alat visualisasi layaknya mata manusia, sehingga robot mampu melihat dan mendeteksi posisi objek yang berada di lapangan

Dengan memanfaatkan kamera sebagai sensor, robot dapat mengenal objek disekitarnya seperti bola, gawang, robot, dan halangan (*obstacles*). Untuk membantu robot *soccer* dalam melakukan pendeteksian terhadap objek di lapangan maka digunakan pendekatan *computer vision*, sehingga robot dapat memperoleh pemahaman dari setiap objek yang berhasil ditangkapnya. *Computer vision* menyediakan berbagai macam metode pendekatan untuk memecahkan masalah pendeteksian objek.

Salah satu penelitian tentang metode pendekatan *computer vision* yang telah dibuat oleh Handriko, Irwan dkk (2018) membahas tentang metode *color filtering* HSV yang digunakan untuk sistem pendeteksian objek pada *robot soccer*. Metode ini digunakan untuk memisahkan objek-objek berdasarkan warna dari objek yang ingin dideteksi. Dengan menggunakan *color filtering* proses pengolahan citra akan dapat lebih fokus dalam pendeteksian objek, sehingga metode ini banyak diterapkan pada perancangan *robot soccer*[1]. Namun metode ini memiliki kelemahan, yaitu bergantung pada kondisi lingkungan tertentu. Ketika kondisi lingkungannya berubah-ubah maka diperlukan lagi kalibrasi atau konfigurasi ulang untuk mengatur Hue, Saturation, Value atau HSV agar *robot soccer* dapat mendeteksi objek dengan benar, Untuk menutupi kelemahan metode *color filtering* HSV, maka diperlukan metode pendeteksian objek yang dapat beradaptasi dengan kondisi lingkungan yang berubah-ubah, yaitu dengan menerapkan *Artificial Intelligence* (AI) untuk mengatasi kelemahan pada penerapan *filter* konvensional yang sangat rentan dengan kondisi lingkungan yang berubah-ubah.

Berdasarkan permasalahan tersebut, maka dirancang sistem pendeteksian objek pada robot *soccer* dengan mengaplikasikan algoritma *Artificial Neural Network* (ANN) dengan model artsitektur *Convolutional Neural Network* (CNN) menggunakan metode pendekatan *Single Shot MultiBox Detector* (SSD MobileNetV2). Rancangan ini diimplementasikan menggunakan beberapa *tools*, yaitu *python* dan *library keras* serta *tensorflow*. Sedangkan, *platfom* yang digunakan adalah *board* NVIDIA Jetson Nano yang berfungsi sebagai *processor* untuk pemrograman *computer vision* dan kamera *webcam* untuk merekam objek. Hasil akhir yang diharapkan dalam penelitian ini adalah dengan pengaplikasian metode SSD MobileNetV2 pada sistem pendeteksian objek robot *soccer*, menghasilkan model deteksi dengan performa yang lebih baik dan lebih efektif dalam segi kecepatan dan keberhasilan dalam melakukan pendeteksian objek, sehingga robot mampu mendeteksi posisi bola, gawang, serta halangan yang ada disekitarnya dengan tepat.

2. Tinjauan Pustaka.

2.1. *Artificial Neural Network*

Artificial Neural Network atau ANN atau dalam Bahasa Indonesia disebut juga dengan jaringan saraf tiruan (JST) merupakan suatu teknik yang digunakan dalam melakukan pemecahan suatu masalah, seperti memprediksi, mencari, dan sesuatu dengan menerapkan cara kerja saraf biologi manusia.

JST dapat digunakan dalam hal untuk melakukan proses pencarian atau proses menemukan sesuatu tujuan yang diinginkan. Kinerja JST itu sendiri adalah melakukan suatu proses pembelajaran dari suatu model yang diinginkan berdasarkan data, kemudian JST yang akan melakukan proses untuk mencari atau menemukan target dengan melakukan pencocokan pola[2].

JST merupakan sistem adaptif yang dapat mengubah strukturnya untuk memecahkan masalah berdasarkan informasi eksternal maupun internal yang mengalir melalui jaringan tersebut. Keuntungan dari metode JST adalah dapat membangun fungsi non linier dan hanya memerlukan data masukan dan keluaran tanpa mengetahui dengan jelas proses dalam jaringan. JST dapat digunakan untuk memodelkan hubungan yang kompleks antara *input* dan *output* untuk menemukan pola-pola pada data. Hal ini cocok diterapkan pada data citra[3].

2.2. *Convolutional Neural Network*

Convolutional Neural Network atau CNN adalah salah satu metode *machine learning* dari pengembangan *Multi Layer Perceptron* (MLP) yang didesain untuk mengolah data dua dimensi. CNN termasuk dalam jenis *Deep Neural Network* (DNN) karena dalamnya tingkat jaringan dan banyak diimplementasikan dalam data citra. CNN memiliki dua metode; yakni klasifikasi menggunakan *feedforward* dan tahap pembelajaran menggunakan *backpropagation*. Cara kerja CNN memiliki kesamaan pada MLP, namun dalam CNN setiap *neuron* dipresentasikan dalam bentuk dua dimensi, tidak seperti MLP yang setiap neuron hanya berukuran satu dimensi[4].

2.3. *Single Shot MultiBox Detector*

Single Shot MultiBox Detector atau SSD adalah salah satu algoritma *computer vision* yang populer dalam deteksi objek. SSD adalah pendeteksi objek yang cukup cepat dan dapat digunakan pada *real-time* video. Detektor objek dapat menemukan lokasi beberapa jenis objek dalam suatu gambar. Deteksi dijelaskan oleh kotak pembatas atau *bounding box*, dan untuk setiap *bounding box* model juga memprediksi kelas.

SSD merupakan bagian dari pengembangan *deep learning* yang telah menjadi metode untuk tugas-tugas seperti pengenalan objek, yang jauh melampaui metode penglihatan komputer yang lebih tradisional yang digunakan dalam literatur. Dalam bidang visi komputer *Convolutional Neural Network* memiliki keunggulan dalam klasifikasi gambar atau pengelanan objek[5].

2.4. MobileNet

MobileNet adalah salah satu dari arsitektur *convolutional neural network* (CNN) yang dapat digunakan untuk mengatasi kebutuhan akan *computing resource* berlebih yang dapat digunakan pada *platform mobile*[6].

MobileNet merupakan pengembangan keluarga dari model *open source* computer visi *mobile* pertama untuk TensorFlow, yang dirancang untuk memaksimalkan akurasi secara efektif dengan tetap memperhatikan sumber daya yang terbatas untuk aplikasi yang terpasang disuatu perangkat.

Model MobileNet berukuran kecil dengan latensi rendah dan berdaya rendah untuk memenuhi kendala sumber daya dari berbagai kasus penggunaan. MobileNet dapat dibangun untuk klasifikasi, deteksi, *embeddings* dan segmentasi[7].

2.5. TensorFlow

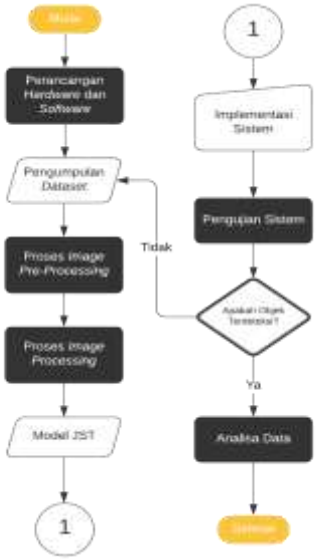
TensorFlow adalah *open source library* untuk komputasi numerik dan pembelajaran mesin skala besar. TensorFlow menggabungkan satu set model dan algoritma pembelajaran mesin dan pembelajaran dalam (alias jaringan saraf atau *neural network*).

TensorFlow dibangun dengan menggunakan bahasa pemrograman *python* yang menyediakan API *front-end* yang nyaman untuk membangun aplikasi dengan kerangka kerja, sambil mengeksekusi aplikasi tersebut dalam C ++/*python* kinerja tinggi.

TensorFlow dikembangkan dengan tujuan untuk melakukan pembelajaran mesin dan penelitian jaringan syaraf dalam. Tensorflow menggabungkan aljabar komputasi, teknik pengoptimalan kompilasi untuk mempermudah penghitungan ekspresi matematis, sehingga dapat mengurangi masalah waktu yang dibutuhkan untuk melakukan perhitungan[8].

3. Metode Penelitian

Rancangan metode penelitian sistem pendeteksian objek menggunakan model SSD MobileNetV2 pada robot *soccer* KRSBI-Beroda dapat dilihat pada Gambar 1 berikut.



Gambar 1. Diagram Alur Metode Penelitian

Proses perancangan sistem pendeteksian objek pada robot *soccer* dengan menggunakan metode *Single Shot Multibox Detector* (SSD MobileNetV2) secara umum secara umum dimulai dari proses perancangan *hardware*, yaitu dengan

perancangan rangkaian NVIDIA Jetson Nano dan kamera *webcam* pada Robot *Soccer* KRSBI-Beroda. Selanjutnya, dilakukan perancangan *software* yang dimulai dari proses pengumpulan *dataset*, *pre-processing images*, dan *processing images* untuk menghasilkan model JST SSD MobileNetV2.

Hasil perancangan *software* dan *hardware* direalisasi sehingga dapat dilakukan pengujian sistem. Proses selanjutnya, dilakukan pengujian dan analisa dari sistem yang telah dibuat. Jika salah satu sistem tidak bekerja dengan benar, maka akan dilakukan *troubleshooting* dan pengujian kembali.

3.1 Perancangan Perangkat Keras

Pada perancangan perangkat keras sistem pendeteksian objek dengan menggunakan metode SSD MobileNetV2 pada robot *soccer* terdiri dari NVIDIA Jetson Nano sebagai pengolah data yang ditangkap oleh kamera webcam C922 Pro. Citra objek yang ditangkap dari *webcam* akan diolah pada NVIDIA Jetson Nano dengan menggunakan model deteksi *neural network* SSD MobileNetV2, selanjutnya *output* yang dihasilkan berupa tampilan sistem pendeteksian berupa objek dengan kotak pembatas lengkap dengan kelas serta persentase akurasinya. Pada pemrosesan grafis dipilih NVIDIA Jetson NANO dengan prosesor Quad-core ARM A57 @ 1.43 GHz dan GPU 128-core Maxwell, karena memungkinkan untuk menjalankan jaringan saraf dengan AI secara simultan dan algoritma *computer vision* diterapkan untuk sistem pendeteksian objek. Skema perancangan hardware dapat dilihat pada Gambar 2.

Gambar 2. Skema Perancanagan *Hardware*

3.2 Perancangan Perangkat Lunak

Pada perancangan sistem pendeteksian objek yang diimplementasikan pada robot KRSBI-Beroda menggunakan bahasa pemrograman *python* dengan *library tensorflow* yang dikemas dengan *framework keras*.

Perancangan *software* ini dilakukan dengan mengunakan *compiler python IDLE* yang dibantu dengan *VSCode* sebagai *code editor*. Perancangan sistem *Detection Object* pada Robot *Soccer* menggunakan metode *Single Shot Multibox Detector* (SSD MobileNetV2) terdiri dari tiga sub tahapan, yaitu: 1) *Dataset Collection*, 2) *Traning Model* dan 3) *Implementation Model*. Proses tersebut dapat dilihat pada kerangka kerja pada Gambar 3 berikut ini.

Gambar 3. Kerangka Kerja Sistem

3.3 Gambaran Umum Sistem

Prinsip kerja dari sistem pendeteksian objek pada robot KRSBI-Beroda dengan menggunakan metode SSD MobileNetV2 ini terdiri dari beberapa proses, pertama dilakukan proses pengumpulan *dataset*, proses *pre-processing images* yang meliputi: tahap *labelling images*, konversi *dataset* XML ke CSV, lalu dari CSV ke TFRecord, dan tahap terakhir pembuatan label map. Selanjutnya proses *processing images* yang meliputi: tahap pemilihan model Pra-terlatih SSD MobileNetV2, konfigurasi *pipeline.config*, lalu dilakukan proses *training* model. Dengan menggunakan arsitektur CCN pada proses *training*, menghasilkan *classification, detection, prediction* model. Selanjutnya dilakukan proses NMS untuk menghasilkan model dengan IOU minimal 0.5. Selanjutnya model JST SSD MobileNetV2 dihasilkan.

Cara kerja sistem pendeteksian objek dimulai dari robot menangkap citra dari objek target yang diambil melaui kamera. Kemudian, citra tersebut menjadi data masukkan untuk sistem kemudian olah pada CPU dan GPU prosesor. Model terlatih SSD MobileNetV2 akan melakukan evaluasi dari setiap *frame* objek yang ditangkap kamera. Target yang ditentukan adalah objek dari bola, gawang, dan halangan serta robot. Sistem akan memberikan *output* berupada prediksi dan skor untuk setiap kelas.

Pada saat melakukan prediksi, jaringan menghasilkan skor untuk keberadaan setiap kategori objek di setiap kotak pembatas dan menghasilkan penyesuaian ke kotak pembatas untuk menyesuaikan dengan bentuk objek. Selain itu, jaringan menggabungkan prediksi dari beberapa fitur peta dengan resolusi

berbeda untuk secara alami menangani objek dari berbagai ukuran. Tahap terakhir adalah analisa dari hasil pendeteksian objek terhadap setiap kelas untuk melihat presentase akurasi dan kepresisian deteksi. Gambaran umum sistem pedenteksian objek dengan menggunakan metode SSD MobileNetV2 dapat dilihat pada Gambar 4 berikut ini.

Gambar 4. Gambaran Umum Sistem

4. Hasil dan Pembahasan

4.1. Analisa Model

Untuk mengetahui performa dari model yang telah dilatih, dilakukan analisa terhadap beberapa parameter yang dihasilkan dari proses *training* yang dapat dilihat melalui *TensorBoard* yang menyediakan layanan yang dapat digunakan untuk berbagi visualisasi pembelajaran mesin *TensorFlow*. *TensorBoard* menyediakan fitur *Visualizing Learning* yang memungkinkan pengguna untuk memvisualisasikan berbagai informasi tentang model yang dilatih, melacak data dari percobaan pembelajaran mesin, seperti grafik model; untuk kerugian, akurasi, atau metrik khusus; untuk menyematkan proyeksi, gambar, dan histogram bobot dan bias. Berikut ini merupakan parameter yang dihasilkan dari model terlatih, yaitu:

1. *Detection Precision* (mAP)

Precision atau ketepatan digunakan untuk mengukur seberapa akurat objek yang dapat diprediksi model, yakni persentase prediksi benar. Gambar 5 menunjukkan grafik kinerja kepresisian deteksi dari model yang telah dilatih sebanyak 200.000 langkah/*steps* yang mengacu pada COCO dalam melakukan perhitungan metrik *mean Average Precision* (mAP). Pada *DetectionBoxes_Precision* menghasilkan 6 grafik yang dapat dilihat pada *TensorBoard*, yaitu:

Gambar 5. Grafik *DetectionBoxes_Precision*

1. mAP

Grafik pertama menunjukkan grafik mAP atau AP *at* IoU = .50; .05; .95 (*primary challenge metric*) yang dihasilkan dari pelatihan model. Grafik tersebut menghasilkan nilai tertinggi, yaitu: 0.8351 dengan rata-rata ± 0.8 konvergen sampai *steps* ke 200.000 yang menunjukkan bahwa nilai rata-rata mAP mendekati 1.

2. mAP (*large*)

Grafik kedua menunjukkan grafik mAP *large* (*AP for large objects: area*>96^2) yang dihasilkan dari pelatihan model. Grafik tersebut menghasilkan nilai tertinggi, yaitu: 0.8430 dengan rata-rata ± 0.8 konvergen sampai *steps* ke 200.000 yang menunjukkan kinerja dari model dalam mendeteksi objek dengan ukuran yang besar.

3. mAP (*medium*)

Grafik ketiga menunjukkan grafik mAP *medium* (*AP for medium objects:* 32^2<*area*<96^2) yang dihasilkan dari pelatihan model. Grafik tersebut menghasilkan nilai tertinggi, yaitu: 0.8170 dengan rata-rata ± 0.8 konvergen sampai *steps* ke 200.000 yang menunjukkan kinerja dari model dalam mendeteksi objek dengan ukuran yang sedang.

4. mAP (*small*)

Grafik empat menunjukkan grafik mAP *small* (*AP for small objects: area*<32^2) yang dihasilkan dari pelatihan model. Grafik tersebut menghasilkan nilai dibawah ambang batas 0, yaitu: -1 dan tidak mengalami kenaikan ataupun penurunan walaupun dilatih selama 200.000 *steps* yang menunjukkan kinerja dari model dalam mendeteksi objek dengan ukuran yang kecil.

5. mAP@.50IOU

Grafik kelima menunjukkan grafik mAP@.50IOU (*AP at* IoU=.50 (*PASCAL VOC metric*)) yang dihasilkan dari pelatihan model. Grafik tersebut menghasilkan nilai tertinggi, yaitu: 0.9774 dengan rata-rata ± 0.9 konvergen sampai *steps* ke 200.000 yang menunjukkan kinerja dari model dalam menghasilkan *bounding box* dengan

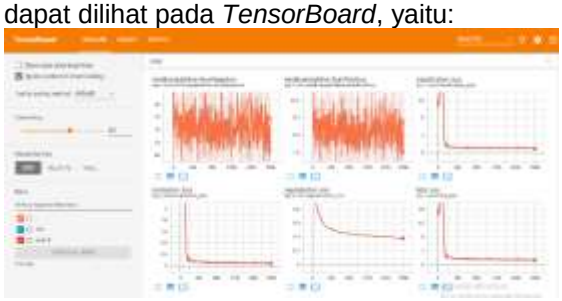
kebenaran dasar (*Ground Truth*) pada ambang batas/ *treshold* 0.50.

6. mAP@.75IOU

Grafik terakhir menunjukkan grafik mAP@.75IOU (*AP at* IoU=.75 (*strict metric*)) yang dihasilkan dari pelatihan model. Grafik tersebut menghasilkan nilai tertinggi, yaitu: 0.9772 dengan rata-rata ± 0.9 konvergen sampai *steps* ke 200.000 yang menunjukkan kinerja dari model dalam menghasilkan *bounding box* dengan kebenaran dasar (*Ground Truth*) pada ambang batas/ *treshold* 0.75.

2. *Loss*

Fungsi kerugian memberi tahu model tentang seberapa baik model tersebut dalam menyelesaikan tugas tertentu. Jika ingin membuat perubahan pada model sebelumnya dengan harapan (mencoba *hyperparameter* berbeda) untuk membuat model yang lebih baik, fungsi kerugian akan memberi informasi apakah model yang dihasilkan lebih baik daripada model yang dilatih sebelumnya. Jika prediksi model mati total, fungsi kerugian akan menghasilkan angka yang lebih tinggi. Jika prediksi cukup bagus, maka akan dihasilkan angka kerugian yang lebih rendah. Gambar 6 menunjukkan grafik kinerja kerugian (*loss*) dari model yang telah dilatih sebanyak 200.000 *steps*. Pada *loss* menghasilkan 6 grafik yang dapat dilihat pada *TensorBoard*, yaitu:



Gambar 6. Grafik *Loss*

1. *Loss HardExampleMiner/NumNegatives*

Grafik pertama merupakan grafik *loss HardExampleMiner/NumNegatives*. Grafik tersebut meghasilkan nilai terendah, yaitu: 25.31 dengan rata-rata ± 26.5 konvergen sampai *steps* ke 200.000 yang menunjukkan performa dari model dalam membedakan objek positif palsu dengan kebenaran dasar.

2. *Loss HardExampleMiner/NumPositives*

Grafik kedua menunjukkan *loss HardExampleMiner/NumPositives*. Grafik tersebut meghasilkan nilai terendah, yaitu: 8.375 dengan rata-rata ± 8.8 konvergen sampai *steps* ke 200.000 yang menunjukkan performa dari model dalam membedakan objek negatif palsu dengan kebenaran dasar.

3. *Classification Loss*

Grafik ketiga menunjukkan *classification_loss*. Grafik tersebut meghasilkan nilai terendah, yaitu: 0.808 dengan rata-rata ± 1.0 konvergen sampai *steps* ke 200.000 yang menunjukkan performa dari model dalam mengklasifikasikan objek.

4. *Localization Loss*

Grafik keempat menunjukkan *localization_loss*. Grafik tersebut meghasilkan nilai terendah, yaitu: 0.102 dengan rata-rata ± 0.2 konvergen sampai *steps* ke 200.000 yang menunjukkan bahwa performa dari model cukup baik untuk memprediksi *offset* kotak pembatas untuk setiap objek.

5. *Regularization Loss*

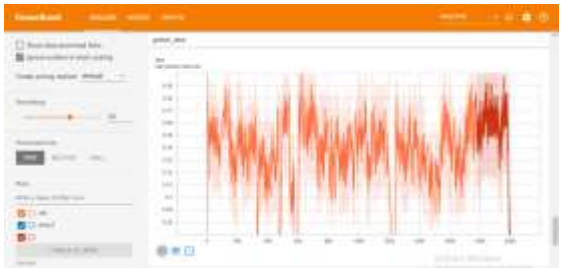
Grafik kelima menunjukkan *regularization_loss*. Grafik tersebut meghasilkan nilai terendah, yaitu: 0.387 dengan rata-rata ± 0.4 konvergen sampai *steps* ke 200.000 yang menunjukkan performa dari model dalam menggeneralisasi pola untuk setiap objek.

6. Total Loss

Grafik terakhir menunjukkan *total_loss* yang berfungsi untuk memberikan informasi dari total kerugian dari hasil pelatihan model selama 200.000 *steps*. Total loss merepresentasikan performa dari model dalam akurasinya dalam mendeteksi setiap objek. Grafik tersebut meghasilkan nilai terendah, yaitu: 0.977 dengan rata-rata ± 1.5 konvergen sampai *steps* ke 200.000 yang menunjukkan performa dari model dalam mendeteksi setiap objek.

3. *Global_Steps*

Global_steps didefinisikan sebagai jumlah *batch* data yang telah dilihat oleh grafik sepanjang proses *training* dilakukan. Setiap kali variabel *batch* berubah, bobot akan diperbarui sedemikian rupa sehingga meminimalkan kerugian. Grafik tersebut meghasilkan nilai terendah, yaitu: 0.07793 dengan rata-rata ± 0.300 konvergen sampai *steps* ke 200.000 yang menunjukkan bahwa performa dari model kurang stabil dalam memberikan keluaran yang tepat dalam sekumpulan masukan yang belum pernah dilihat sebelumnya. Gambar 7 menunjukkan grafik kinerja *global_steps* dari model yang telah dilatih sebanyak 200.000 *steps*.



Gambar 7. Grafik *Global_Steps*

4. *Learning Rates*

Learning_rates adalah *hyperparameter* yang mengontrol seberapa banyak model harus diubah sebagai respons terhadap estimasi error setiap kali bobot model diperbarui. *Learning_rates* digunakan untuk mengetahui efek kecepatan pembelajaran pada performa model, perilaku model serta mengetahui loss diminimalkan atau tidak. Kecepatan pembelajaran mengontrol seberapa cepat model beradaptasi dengan masalah. *Learning_rates* yang lebih kecil memerlukan lebih banyak periode pelatihan karena perubahan yang lebih kecil dilakukan pada bobot setiap pembaruan, sedangkan *learning_rates* yang lebih besar menghasilkan perubahan yang cepat dan memerlukan waktu pelatihan yang lebih sedikit. Kecepatan pembelajaran yang terlalu besar dapat menyebabkan model terlalu cepat menyatu ke solusi yang kurang optimal, sedangkan kecepatan pembelajaran yang terlalu kecil dapat menyebabkan proses training macet.Gambar 8 menunjukkan grafik kinerja *learning_rates* dari model yang telah dilatih sebanyak 200.000 *steps*. Dari grafik tersebut menghasilkan nilai konstan, yaitu: 0 konvergen sampai *steps* ke 200.000 yang menunjukkan terjadinya *underfitting* pada model, yaitu: model "kurang belajar" dari data pelatihan, mengakibatkan generalisasi yang rendah dan prediksi yang kurang dapat diandalkan.

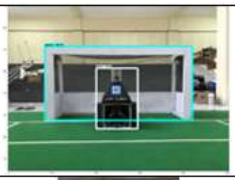
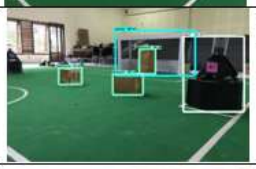
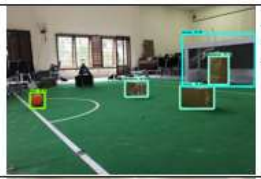
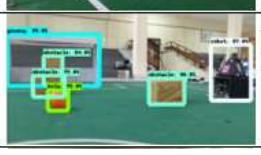
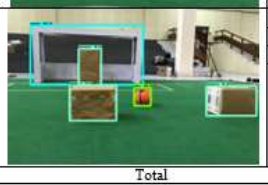
Gambar 8. Grafik *Learing_Rates*

4.2 Validasi Model

Untuk mengetahui akurasi dari model terlatih perlu dilakukan validasi terhadap model tersebut. Validasi dilakukan dengan cara membandingkan *ground truth* atau objek sebenarnya dengan prediksi model. Untuk mengetahui performa dari model SSD MobilNetV2 yang telah dilatih sebanyak 200.000 steps, dapat dilihat hasil uji coba model terhadap objek pada Tabel 1.

Tabel 1. Validasi Model

No	Hasil Deteksi	Kelas	Model	Manual	%
1.		Bola	1	1	99%
		Gawang	-	-	-
		Robot	-	-	-
		Obstacle	-	-	-
2.		Bola	1	1	100%
		Gawang	-	-	-
		Robot	-	-	-
		Obstacle	-	-	-
3.		Bola	1	1	100%
		Gawang	-	-	-
		Robot	-	-	-
		Obstacle	-	-	-
4.		Bola	1	1	100%
		Gawang	-	-	-
		Robot	1	1	98%
		Obstacle	-	-	-
5.		Bola	1	1	99%
		Gawang	1	1	100%
		Robot	-	-	-
		Obstacle	-	-	-
6.		Bola	-	-	-
		Gawang	-	-	-
		Robot	1	1	99%
		Obstacle	-	-	-
7.		Bola	-	-	-
		Gawang	-	-	-
		Robot	1	1	100%
		Obstacle	-	-	-
8.		Bola	-	-	-
		Gawang	-	-	-
		Robot	1	1	100%
		Obstacle	-	-	-
9.		Bola	-	-	-
		Gawang	1	1	100%
		Robot	1	1	87%
		Obstacle	-	-	-

10.		Bola	-	-	-
		Gawang	1	1	100%
		Robot	1	1	99%
		Obstacle	-	-	-
11.		Bola	-	-	-
		Gawang	-	-	-
		Robot	-	-	-
		Obstacle	1	1	100%
12.		Bola	-	-	-
		Gawang	1	1	98%
		Robot	-	-	-
		Obstacle	1	1	100%
13.		Bola	-	-	-
		Gawang	-	-	-
		Robot	1	1	99%
		Obstacle	2	2	99%; 99%
14.		Bola	1	1	80%
		Gawang	1	1	63%
		Robot	1	1	97%
		Obstacle	3	3	93%; 93%; 61%
15.		Bola	1	1	66%
		Gawang	1	1	59%
		Robot	1	1	84%
		Obstacle	2	3	99%; 92%; 0%
16.		Bola	1	1	70%
		Gawang	1	1	99%
		Robot	1	1	97%
		Obstacle	3	3	86%; 69%; 97%
17.		Bola	-	1	0%
		Gawang	-	-	-
		Robot	2	2	88%; 98%
		Obstacle	2	2	90%; 98%
18.		Bola	-	-	-
		Gawang	1	1	100%
		Robot	1	2	90%; 0%
		Obstacle	3	3	80%; 100%; 99%
19.		Bola	1	1	99%
		Gawang	1	1	81%
		Robot	-	1	0%
		Obstacle	3	3	96%; 99%; 97%
20.		Bola	1	1	75%
		Gawang	1	1	99%
		Robot	1	1	97%
		Obstacle	3	3	84%; 75%; 98%
21.		Bola	1	1	99%
		Gawang	1	1	100%
		Robot	-	-	-
		Obstacle	3	3	100%; 96%; 95%
22.		Bola	-	-	-
		Gawang	1	1	100%
		Robot	1	1	92%
		Obstacle	3	3	100%; 71%; 99%
23.		Bola	1	1	100%
		Gawang	1	1	100%
		Robot	-	-	-
		Obstacle	3	3	98%; 99%; 95%
24.		Bola	1	1	99%
		Gawang	-	1	0%
		Robot	1	1	95%
		Obstacle	3	3	93%; 98%; 97%
25.		Bola	1	1	100%
		Gawang	1	1	100%
		Robot	-	-	-
		Obstacle	3	3	98%; 99%; 99%
Total			82	87	

Pada Tabel 1 dilakukan validasi dengan menggunakan gambar acak sebanyak 25 gambar dengan total objek sebanyak 87 objek. Dari total 25 gambar, model SSD MobileNetV2 terlatih hanya dapat memprediksi 82 objek dari total objek sebenarnya sebanyak 87 objek. Untuk mengetahui presentase keberhasil model dalam memprediksi objek, maka dapat dirumuskan sebagai berikut:

$$\%\ Keberhasilan = \frac{Total\ Prediksi\ Model}{Total\ Objek\ Sebenarnya} \times 100\%$$

$$= \frac{82}{87} \times 100\%$$

$$\approx 94{,}85\%$$

Dari hasil perhitungan presentase keberhasilan, didapatkan angka presentase keberhasilan prediksi rata-rata adalah 94,85%.

5. Kesimpulan

Kesimpulan yang dapat diambil berdasarkan pembuatan, pengujian dan pembahasan penelitian ini adalah sebagai berikut:

1. Sistem pendeteksian objek pada robot *soccer* KRSBI-Beroda yang dibangun dengan menggunakan metode pendekatan *Single Shot MultiBox Detector* (SSD MobileNetV2) yang diimplementasikan pada *main controller* NVIDIA Jetson Nano dan kamera Logitech

C922 Pro sudah mampu melakukan pendeteksian objek dalam kondisi objek terdeteksi utuh atau terhalang objek lain dengan catatan kondisi objek yang terdeteksi ada dalam posisi dan jarak ideal yang dapat ditangkap kamera.

2. Pada sistem pendeteksian objek pada robot *soccer* KRSBI-Beroda yang dibangun dengan menggunakan metode pendekatan *Single Shot MultiBox Detector* (SSD MobileNetV2) menghasilkan presentase keberhasilan prediksi rata-rata 94,85% untuk memprediksi objek dengan model yang dilatih sebanyak 200.000 steps dengan data validasi sebanyak 25 gambar *random* yang berisi 87 objek didalamnya.

6. Daftar Pustaka

[1] Handriko, Irwan, and M. I. Nugraha, *PROCEEDING The 6th Indonesian Symposium on Robotic Systems and Control (ISRC)*, 6th ed. Yogyakarta: Lembaga Penelitian, Publikasi & Pengabdian Masyarakat Universitas Muhammadiyah Yogyakarta, 2018.

[2] M. Yanto, S. Defit, and G. W. Nurcahyo, "MEMPREDIKSI JUMLAH RESERVASI KAMAR HOTEL DENGAN METODE BACKPROPAGATION (Studi Kasus Hotel Grand Zuri Padang)," *J. KomTekInfo Fak. Ilmu Komput.*, vol. 2, no. 1, pp. 34–39, 2015, doi: 2356-0010.

[3] Yuliana, "Bab 8 Jaringan Saraf Tiruan (Neural Network)," in *Kecerdasan Buatan*, Yuliana, Ed. Surabaya: Yuliana, 2014, pp. 82–123.

[4] N. Sofia, "CONVOLUTIONAL NEURAL NETWORK," 2018. https://medium.com/@nadhifasofia/1-convolutional-neural-network-convolutional-neural-network-merupakan-salah-satu-metode-machine-28189e17335b#:~:text=Fungsi Softmax menghitung probabilitas dari,target untuk input yang diberikan.&text=Softm (accessed May 27, 2020).

[5] Honingds, "Single shot object detection SSD using MobileNet and OpenCV," 2019. https://honingds.com/blog/ssd-single-shot-object-detection-mobilenet-opencv/ (accessed May 27, 2020).

[6] R. O. Ekoputris, "MobileNet: Deteksi Objek pada Platform Mobile," 2018. https://medium.com/nodeflux/mobilenet-deteksi-objek-pada-platform-mobile-bbbf3806e4b3 (accessed May 27, 2020).

[7] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," 2017, [Online]. Available: http://arxiv.org/abs/1704.04861.

[8] S. R. Dewi, "DEEP LEARNING OBJECT DETECTION PADA VIDEO MENGGUNAKAN TENSORFLOW DAN CONVOLUTIONAL NEURAL NETWORK (Studi Kasus: Klasifikasi Gambar Meja dan Kursi Motif Ukiran Jepara)," UNIVERSITAS ISLAM INDONESIA, 2018.