SISTEM IDENTIFIKASI NOMINAL UANG LOGAM MENGGUNAKAN *TENSORFLOW* DAN *CONVOLUTIONAL NEURAL NETWORK* BERBASIS RASPBERRY PI

Rima Apriani Doga[1], Hendro F. J. Lami[2] dan Stephanie I. Pella[3]

[1]Program Studi Teknik Elektro, Universitas Nusa Cendana, Jl. Adisucipto Penfui Kupang
Email: rimaaprianidoga@gmail.com
[2] Studi Teknik Elektro, Universitas Nusa Cendana, Jl. Adisucipto Penfui Kupang
Email: h.lami@staf.undana.ac.id
[3] Studi Teknik Elektro, Universitas Nusa Cendana, Jl. Adisucipto Penfui Kupang
Email: s.i.pella@staf.undana.ac.id

ABSTRAK

Penelitian ini bertujuan mendesain sebuah sistem untuk mendeteksi nominal uang logam dan menampilkan jumlah dari uang yang teridentifikasi menggunakan *TensorFlow* dan algoritma *Convolutional Neural Network*. Penelitian terbagi atas dua bagian, yaitu proses pelatihan (*training*) *model* dan proses pengujian *model*. Proses pelatihan *model* dimulai dari tahapan mengumpulkan *dataset* berupa 511 gambar dimana 80% dari *dataset* merupakan citra latih dan 20% dari *dataset* adalah citra uji. Kemudian tahapan *preprocessing* yaitu melakukan pelabelan objek pada setiap gambar menjadi 4 kelas yaitu Rp.100, Rp.200, Rp.500 dan Rp.1000. Tahapan terakhir adalah *training dataset* dimana *feeding data TFRecord* didapat dari file CSV hasil konversi berkas XML pelabelan gambar. *Training dataset* dilakukan hingga langkah ke 100.000 dengan menghasilkan *loss* sebesar 2.5 dan *model* hasil pelatihan ini diuji pada sistem yaitu Raspberry Pi yang terintegrasi dengan kamera. Pengujian sistem dilakukan dengan menggunakan jumlah objek, jarak kamera, kemiringan kamera, dan kondisi pencahayaan yang bervariasi. Setiap *set* pengujian dilakukan sebanyak 50 kali untuk mendapatkan hasil yang akurat. Hasil pengujian menunjukan tingkat keberhasilam 100% untuk identifikas 1-3 objek, kemudian menurun secara linear dengan pertambahan jumlah objek. Pengujian jarak dan kemiringan kamera menunjukan jarak ideal antara kamera dan objek adalah 12-16 cm dan posisi kemiringan kamera yang baik adalah 0-20^o terhadap bidang mendatar dimana rata-rata 87% dari objek dikenali dengan benar. Hasil pengujian pengaruh kondisi pencahayaan menunjukan sistem berkerja optimal pada kondisi pencahaayaan 120-200 lux.

Kata Kunci: *Deep Learning*, Deteksi Uang Logam, CNN, *TensorFlow*, Raspberry Pi
Author : Rima Apriani Doga, Hendro F. J. Lami dan Stephanie I. Pella

1. PENDAHULUAN

Latar Belakang

Uang memiliki peranan strategis dalam perekonomian karena fungsi utamanya sebagai alat pembayaran. Penggunaan uang logam dalam kehidupan sehari-hari masih dibutuhkan karena memberikan kemudahan dalam transaksi, baik transaksi ekonomi jual beli di pusat perbelanjaan maupun dengan mesin penjual minuman ringan dan makanan ringan otomatis. Mesin penjual minuman menggunakan teknologi pengolahan citra digital untuk mendeteksi uang logam berdasarkan bentuk citra dari koin yang dimasukkan. Pada penelitian sebelumnya dilakukan pengenalan uang logam berbasis pengolahan citra digital menggunakan fitur jarak antar piksel pada citra untuk membandingkan jarak suatu citra dan citra lainnya bertujuan untuk mencari kemiripan antara citra (Wibawa, 2012). Penelitian pendeteksian uang logam dengan metode yang sama dilakukan yaitu berdasarkan warna dan ukuran citra tiap-tiap objek koin uang logam (Banurea, 2017).

Di era digital sekarang ini kehidupan manusia berhubungan dengan teknologi komputasi, manusia terus mengembangkan teknologi untuk meringankan pekerjaannya, bidang yang sedang berkembang adalah kecerdasan buatan atau yang dikenal dengan *Artificial Intelligence* (AI). Penerapan *Artificial Intelligence* khususnya *Deep Learning* yang merupakan bagian dari *Machine Learning* saat ini memiliki hasil signifikan dalam pengenalan citra dan banyak digunakan untuk menyelesaikan permasalahan yang berkaitan dengan *object detection* dan *image classification*. Penelitian terdahulu yang dilakukan oleh Taufiq (2018) menggunakan metode *Convolutional Neural Network* (CNN) untuk mengidentifikasi nomor kendaraan menghasilkan akurasi sebesar 99%. Metode yang sama digunakan dalam penelitian untuk klasisfikasi gambar meja dan kursi ukiran jepara menghasilkan tingkat akurasi hingga 99% oleh Dewi (2018). Metode *Convolutional Neural Network* (CNN) berusaha meniru sistem pengenalan citra pada *visual cortex* manusia tetapi memiliki kelemahan yang sama dengan metode *Deep Learning* lainnya yaitu proses pelatihan yang

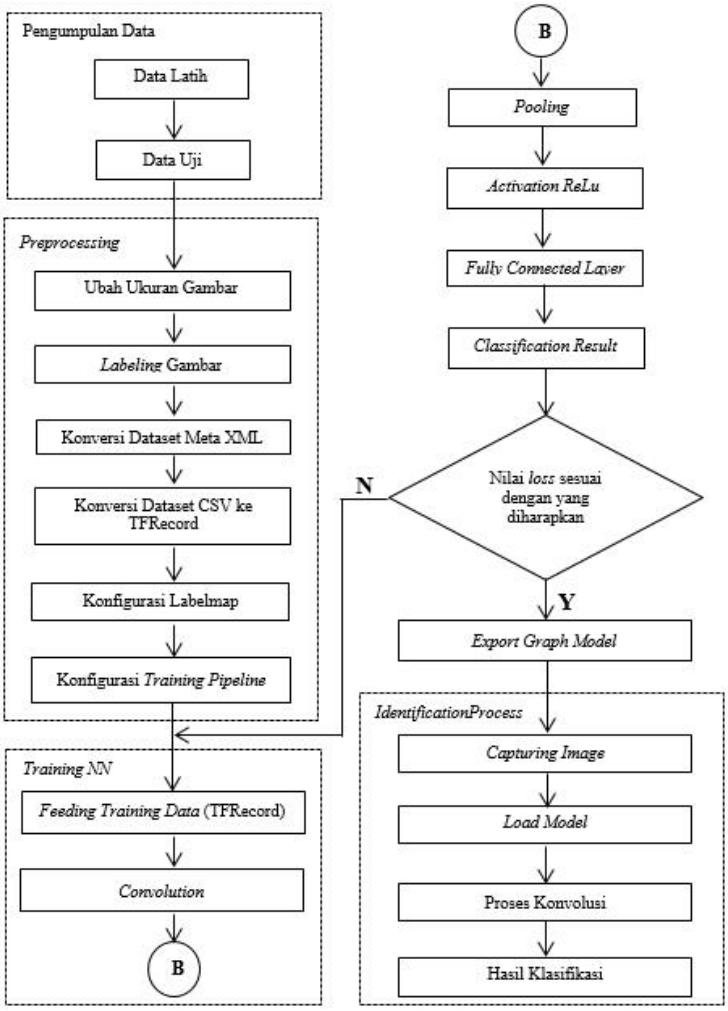
lama. Hal tersebut dapat diatasi dengan perkembangan perangkat keras menggunakan teknologi *General Purpose Graphical Processing Unit* (GPGPU) dimana *tools* yang mendukung komputasi secara paralel dan terdokumentasi dengan baik adalah *Tensor Flow Object Detection API* yang dirilis oleh Google. Penelitian terdahulu berkaitan dengan metode *Deep Learning* menggunakan *single board computer* untuk efisiensi biaya berhasil mendeteksi dan mengenali objek menggunakan Raspberry Pi (Lami, 2019).
Penelitian ini dilakukan dengan menerapkan metode *Convolutional Neural Network* untuk mendapatkan tingkat akurasi yang baik dan hasil yang signifikan dalam pengenalan citra yaitu sebuah sistem untuk mengidentifikasi nominal uang logam menggunakan *TensorFlow* berbasis Raspberry Pi.
Jurnal ini ditulis dengan sistematika pada bagian pendahuluan berisikan latar belakang, pada bagian perancangan system dijabarkan mengenai jenis dan sumber data, tahapan penelitian serta arsitektur jaringan. Pada hasil dan pembahasan dibahas implementasi sistem dan pengujian model hasil training dan bagian terakhir adalah kesimpulan.

2. PERANCANGAN SISTEM

Jenis dan Sumber Data
Sumber data terbagi menjadi dua yaitu data primer dan data sekunder. Data primer yang digunakan merupakan data yang diperoleh dari hasil pengambilan gambar uang logam yang akan digunakan untuk *training model*, sementara data sekunder adalah data yang diperoleh dari peneliti dari sumber yang sudah ada, yaitu pengunduhan gambar uang logam dari internet. Jumlah data yang digunakan berjumlah 511 gambar yang terdiri dari:

a. Data *Training*, data gambar yang digunakan untuk proses *training* berjumlah 414 data gambar dan 414 data label tiap gambar.
b. Data *Testing*, data gambar yang digunakan untuk proses *testing* berjumlah 97 data gambar dan 97 data label tiap gambar.

Tahapan Penelitian

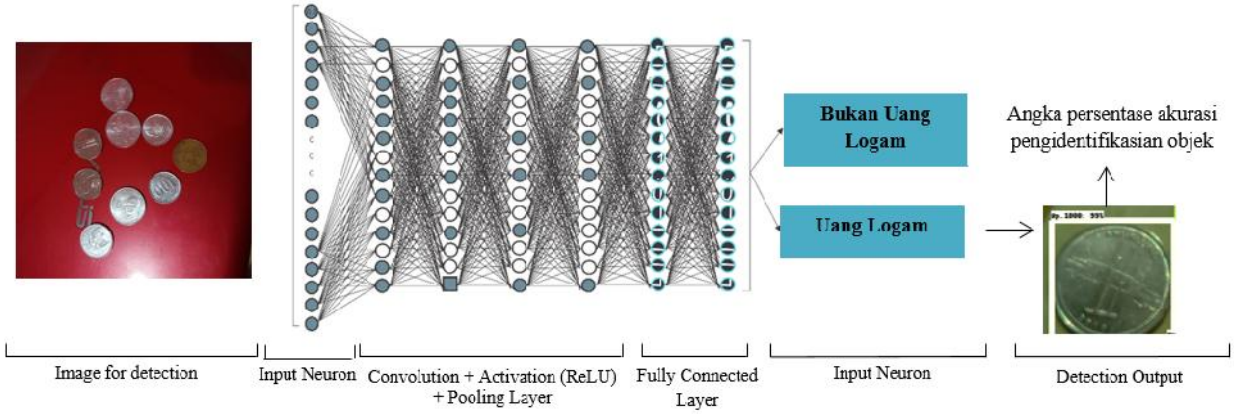


Gambar 1. Tahapan Penelitian

Gambar 1 adalah gambaran tahapan penelitian secara detail yang diawali dengan mengumpulkan data berupa gambar dan dipisahkan sebagai data latih dan data uji. Data gambar ini diubah ukurannya menjadi 774x774 piksel, dilakukan pelabelan objek pada gambar sebagai Rp.100, Rp.200, Rp.500 dan Rp.1000 dimana data inilah yang dikonversi menjadi CSV sebagai *TFRecord*. Kemudian melakukan konfigurasi *labelmap* dan *training pipeline* untuk dilakukan *training neural network*. *Export graph model* dilakukan saat *training* mencapai nilai *loss* yang diharapkan. *Model* hasil *training* ini kemudian digunakan untuk mengidentifikasi objek pada Raspberry Pi.

Arsitektur Jaringan

Algoritma *Convolutional Neural Network* mempunyai arsitektur jaringan dalam proses pelatihan. Dimana arsitektur jaringan ini terdiri dari beberapa bagian jaringan yaitu *Image for detection*, *Input Neuron*, *Convolution + Activation* (ReLU) + *Pooling layer*, *Fully connected layer*, *Classification* dan *Detection output*. Berikut ini adalah visualisasi arsitektur jaringan dari *Convolutional Neural Network*:

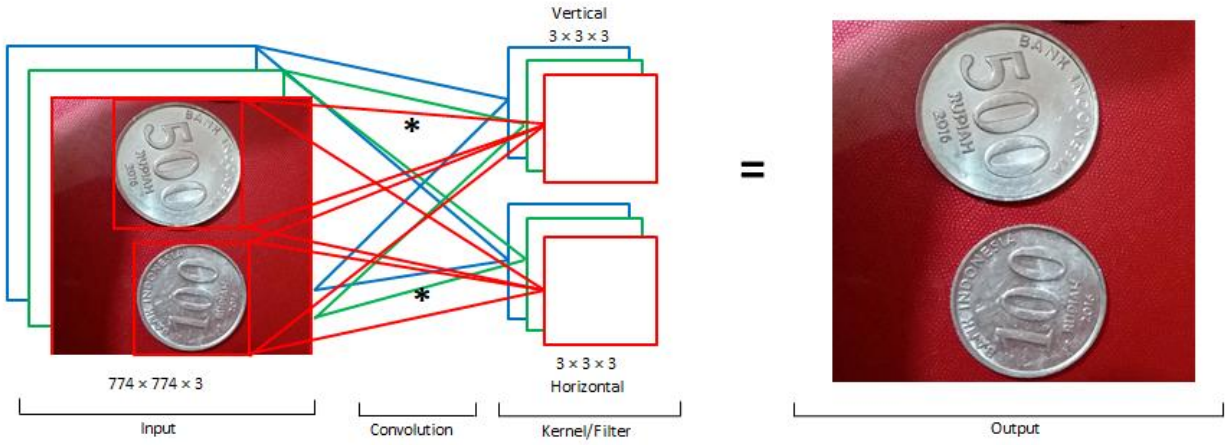


Gambar 2. Arsitektur Jaringan

Gambar 2 adalah arsitektur jaringan dengan gambar 774x774 piksel dengan warna RGB (*Red*, *Green*, *Blue*) yaitu sebanyak 3 *channel* sehingga yang masuk kedalam *layer* pertama atau bagian *Input neuron* sebanyak 1.797.228 *neuron* yaitu 774×774×3, tiap *neuron* memiliki nilai parameter tersendiri yaitu antara 0 sampai 225. Dimana ukuran gambar yang digunakan untuk pelatihan adalah 150kB hingga 300kB.

1. *Convolution Layer*

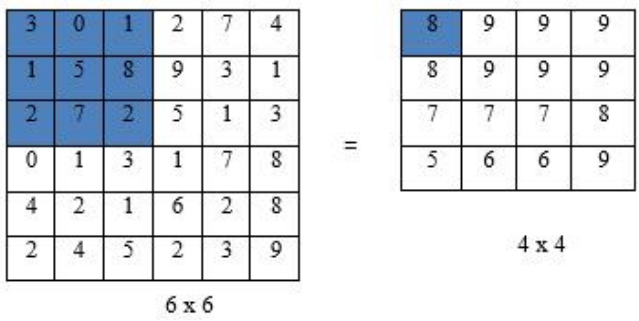
Bagian *input layer* adalah gambar dengan tinggi 774 dan lebar 774 piksel dimana input gambar memiliki 3 parameter warna dasar yaitu warna merah, hijau dan biru. Lapisan konvolusi terbentuk oleh hasil konvolusi dari kernel atau *filter* dengan piksel-piksel gambar ini. *Filter* digunakan untuk menentukan pola apa yang akan dideteksi yang selanjutnya dikonvolusi atau dikalikan dengan nilai pada matriks input, nilai pada masing-masing kolom dan baris pada matriks sangat bergantung pada jenis pola yang akan dideteksi. Berikut adalah visualisasi *convolution layer* pada penelitian ini:



Gambar 3. *Convolution Layer*

2. *Pooling Layer*

Pooling layer menjaga ukuran data ketika *convolution*, yaitu dengan melakukan *downsampling* (pereduksian sampel). Dengan *pooling*, data dapat direpresentasikan menjadi lebih kecil, mudah dikelola dan mudah mengontrol *overfitting pooling layer* yaitu lapisan setelah setelah *convolutional layer*. Pada prinsipnya *pooling layer* terdiri dari sebuah *filter* dengan ukuran dan *stride* tertentu yang akan bergeser pada seluruh area *feature map* seperti pada gambar:



Gambar 5. *Pooling Layer*

Pooling layer pada Gambar 5 merupakan *pooling layer* dengan menggunakan metode *Max Pooling*. Terdapat lapisan dengan ukuran 6×6, jika menggunakan *filter* 3x3 dengan *stride* 1 maka diperoleh hasil *Max pooling* dengan ukuran 4x4.

3. *Fully Connected Layer*

Pada Lapisan yang terhubung secara penuh (*fully connected layer*), setiap *neuron* memiliki konkesi penuh ke semua aktivasi dalam lapisan sebelumnya. Pada akhir *convolution layer* dan *pooling layer*, jaringan umumnya menggunakan lapisan yang terhubung sepenuhnya di mana setiap piksel dianggap sebagai *neuron* terpisah seperti jaringan saraf biasa yang disebut sebagai *Fully Connected Layer*. *Fully Connected Layer* merupakan lapisan terakhir yang terhubung sepenuhnya yang mengandung banyak *neuron* sebagai jumlah kelas yang harus diprediksi.

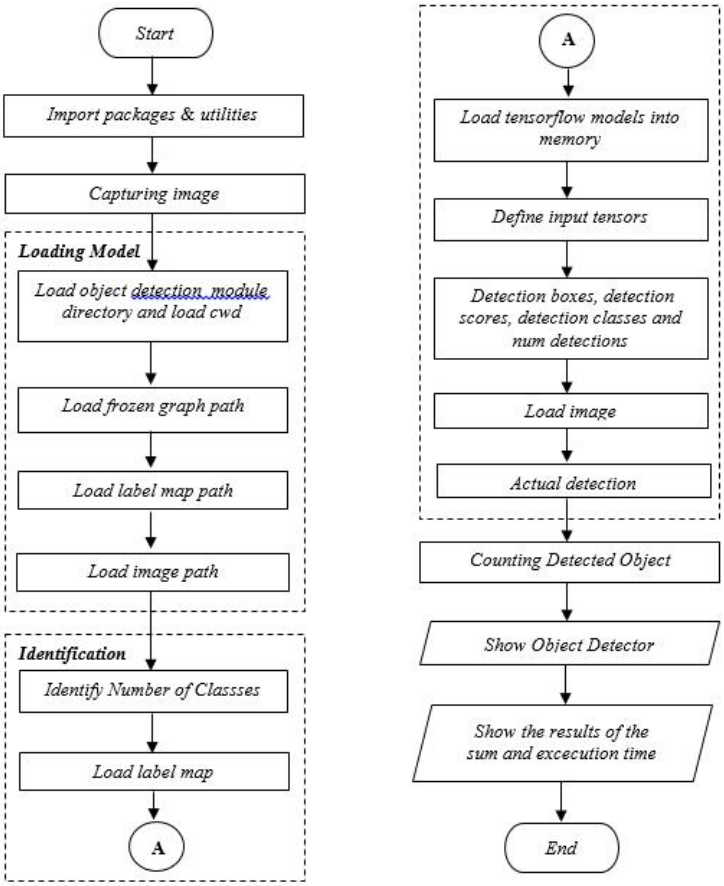
Gambar 6. *Fully Connected Layer*

Gambar 6 diatas merupakan visualisasi dari lapisan *fully connected layer* dimana pada proses ini dilakukan penyatuan setiap piksel yang dianggap sebagai uang logam dengan nominal tertentu.

3. HASIL DAN PEMBAHASAN

Implementasi Sistem

Implementasi merupakan penerapan cara kerja sistem berdasarkan hasil analisa dan juga perancangan yang telah dibuat sebelumnya ke dalam suatu bahasa pemrograman tertentu. Pada penelitian ini *model* hasil pelatihan diimplementasikan ke Raspberry Pi untuk mengidentifikasi objek. Berikut adalah *flowchart* kerja sistem:



Gambar 7. *Flowchart* Kerja Sistem

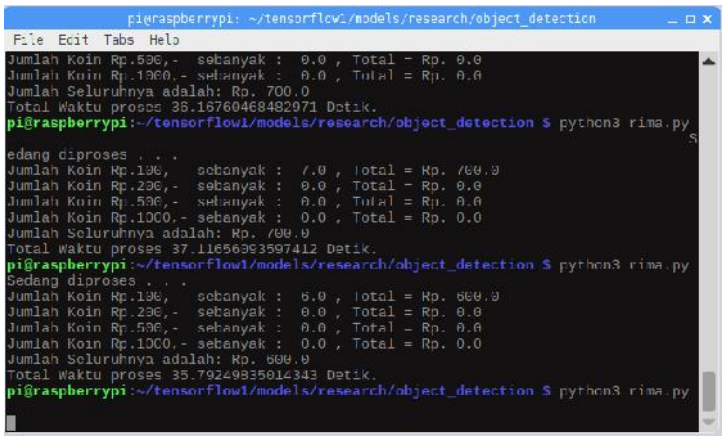
Gambar 7 adalah *flowchart* kerja sistem yaitu pada Raspberry Pi dan kamera untuk menguji *model* hasil *training*. Adapun program diawali dengan meng-*import* semua *packages* dan *utilities*, mengambil gambar oleh kamera sebagai inputan yang akan disimpan pada direktori dan akan terganti tiap kali program dieksekusi, me-*loading model* hasil *training*, mengidentifikasi objek pada gambar, menghitung jumlah dari seluruh nominal objek, menampilkan gambar hasil identifikasi, serta menampilkan hasil jumlah objek terdeteksi dan waktu yang dibutuhkan untuk mengeksekusi program.

1. Tampilan *Object Detector*

Gambar 8. Tampilan *Object Detector*

Gambar 8 adalah *object detector* yaitu menunjukan tampilan sistem yaitu hasil identifikasi objek yaitu *box* objek terdeteksi, kelas objek dan persentase *confidence* dimana gambar input yang sudah diproses dan hasilnya identifikasinya di tampilkan.

2. Tampilan Hasil Penjumlahan dan Waktu Eksekusi Program

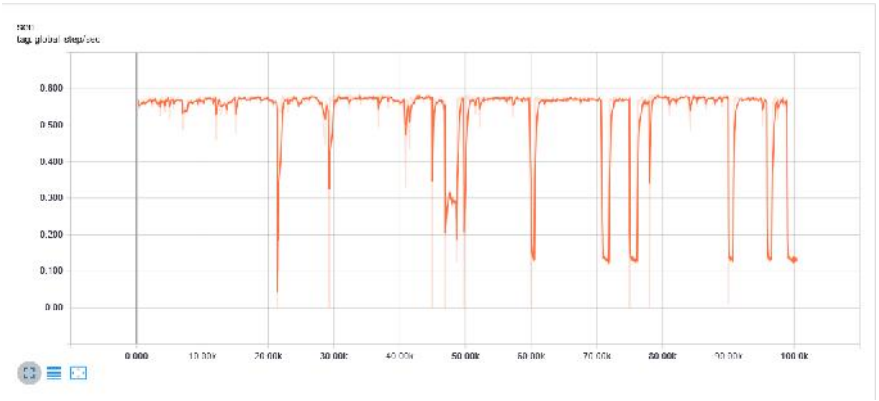


Gambar 9. Tampilan hasil penjumlahan objek dan waktu eksekusi program

Gambar 9 menunjukan hasil penjumlahannya serta waktu yang dibutuhkan untuk mengeksekusi program. Dimana waktu untuk mengeksekusi program ini mulai dihitung dari meng-*import packages* sampai kepada menampilkan hasil jumlah objek terdeteksi.

Model Hasil *Training*

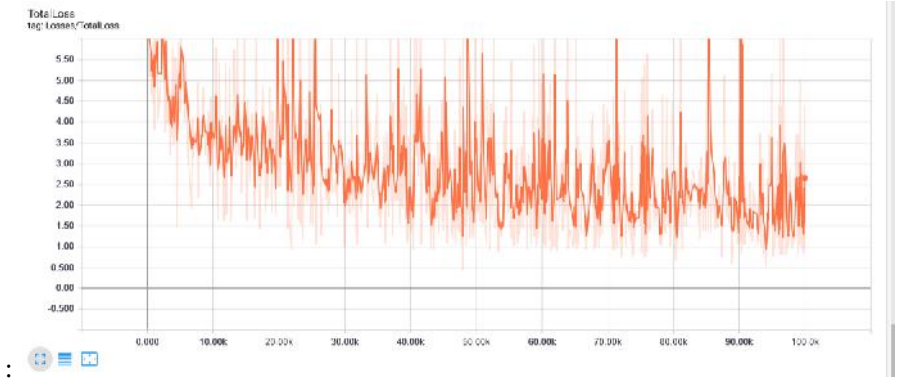
1. *Training Process*



Gambar 10. Grafik Global *Training* Step

Gambar 10 menunjukan bahwa *dataset* gambar pada awal pelatihan sampai pada akhir pelatihan memiliki tingkat akurasi hingga 0.5 dan terjadi penurunan di beberapa *step*, tetapi secara keseluruhan dapat dikatakan langkah pelatihan ini maksimal.

2. *Total Loss*

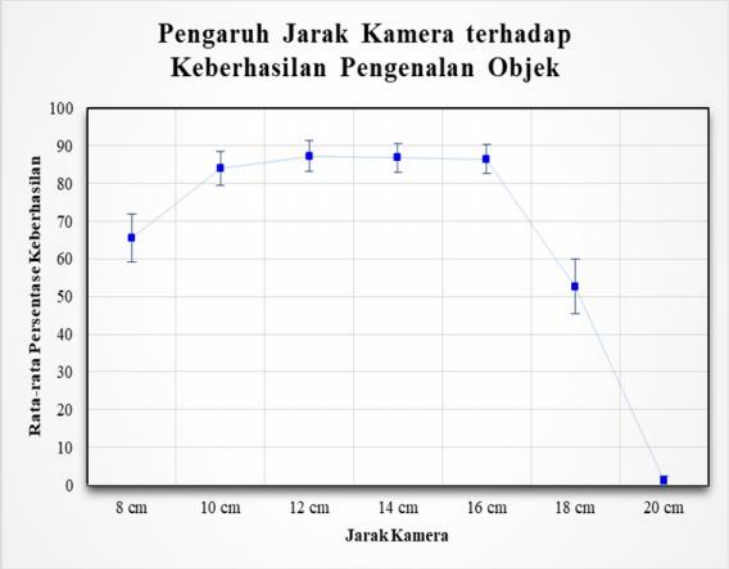


Gambar 11. Grafik *Total Loss*

Gambar 11 adalah grafik *total loss* dimana nilai *total loss* ini adalah penjumlahan nilai *classification loss* dan *localization loss*. Dimana *loss* yang dihasilkan pada saat proses pelatihan sampai dengan selesai yaitu dari *step* ke 1 hingga *step* ke 100.000, dari grafik dapat disimpulkan bahwa dalam proses *training* nilai *loss* yang dihasilkan mengalami fluktuatif yaitu dengan *range* nilai *loss* dari 0 hinga lebih dari 5.5. Nilai *loss* ini kemudian stabil atau tidak terjadi fluktuatif yang berarti setelah step ke 90.000 dengan rata-rata nilai *loss* 2.5 sesuai dengan ketentuan *ideal loss* pada *model ssdlite mobilenet v2 coco* dimana nilai *ideal loss*-nya adalah 2.

Pengujian *Model* Hasil *Training*

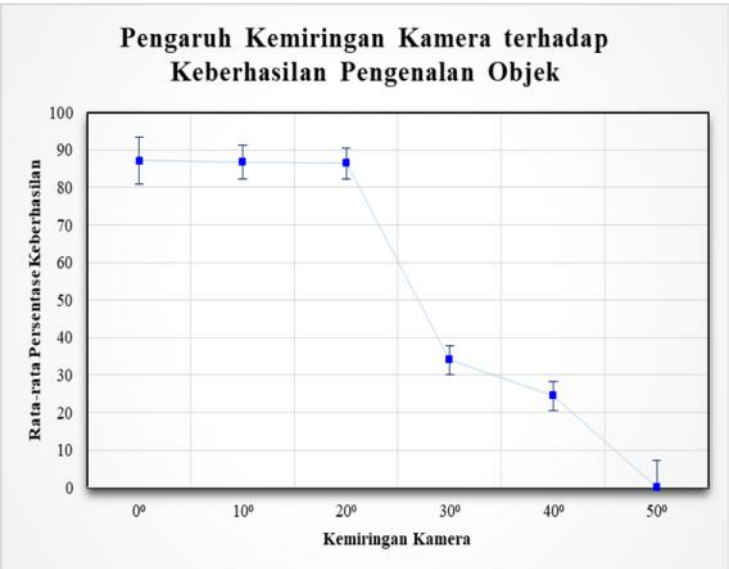
1. Pengujian Jarak Kamera



Gambar 12. Grafik Pengaruh Jarak Kamera terhadap Keberhasilan Pengenalan Objek

Gambar 12 adalah Grafik yang menunjukan pengaruh jarak kamera terhadap keberhasilan pengenalan atau identifikasi objek. Pengujian pengaruh jarak kamera terhadap objek dilakukan untuk mengetahui jarak ideal kamera dari objek untuk mendapatkan hasil identifikasi yang optimal. Dalam pengujian ini pencahayaan yang digunakan adalah 120 lux hingga 200lx dan kemiringan 0^{o} dengan dan mengubah jarak kamera dari objek. Grafik menunjukan rata-rata persentase keberhasilan dengan nilai maksimum adalah di jarak 12 cm yaitu 87,2% dan diikuti jarak 14 cm sebesar 86.8%, 16 cm 86.4% dan 10 cm yaitu 84%.

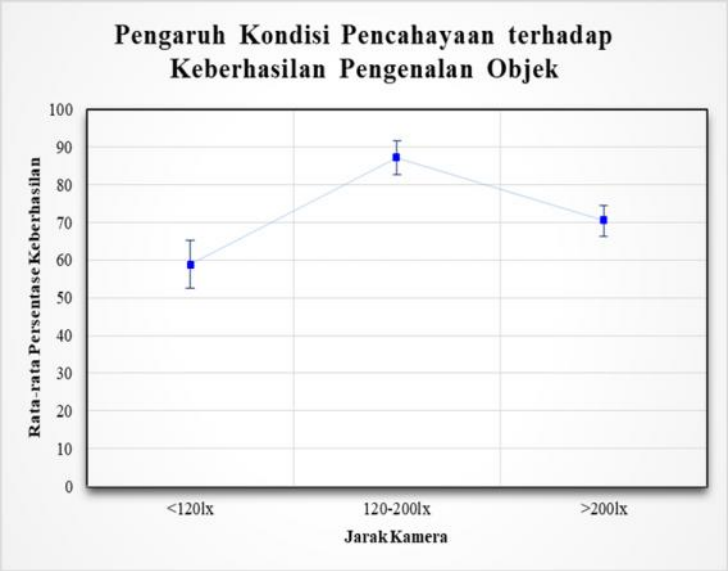
2. Pengujian Kemiringan Kamera



Gambar 13. Grafik Pengaruh Kemiringan Kamera terhadap Keberhasilan Pengenalan Objek

Gambar 13 adalah Grafik yang menunjukan pengaruh kemiringan kamera terhadap keberhasilan pengenalan atau identifikasi objek. Pengujian pengaruh kemiringan kamera terhadap objek dilakukan untuk mengetahui kemiringan ideal kamera dari objek untuk mendapatkan hasil identifikasi yang optimal. Dalam pengujian ini pencahayaan yang digunakan adalah 120 lux hingga 200lx dan jarak kamera 12 cm dengan mengubah kemiringan kamera. Grafik menunjukan rata-rata persentase keberhasilan dengan nilai maksimum adalah pada kemiringan 0^{o} yaitu 87,2% dan diikuti 10^{o} yaitu 86,8% dan 20^{o} sebesar 86.4%. Grafik menunjukan rata-rata persentase keberhasilan dengan nilai maksimum adalah pada kemiringan 0^{o} yaitu 87,2% dan diikuti 10^{o} yaitu 86,8% dan 20^{o} sebesar 86.4%.

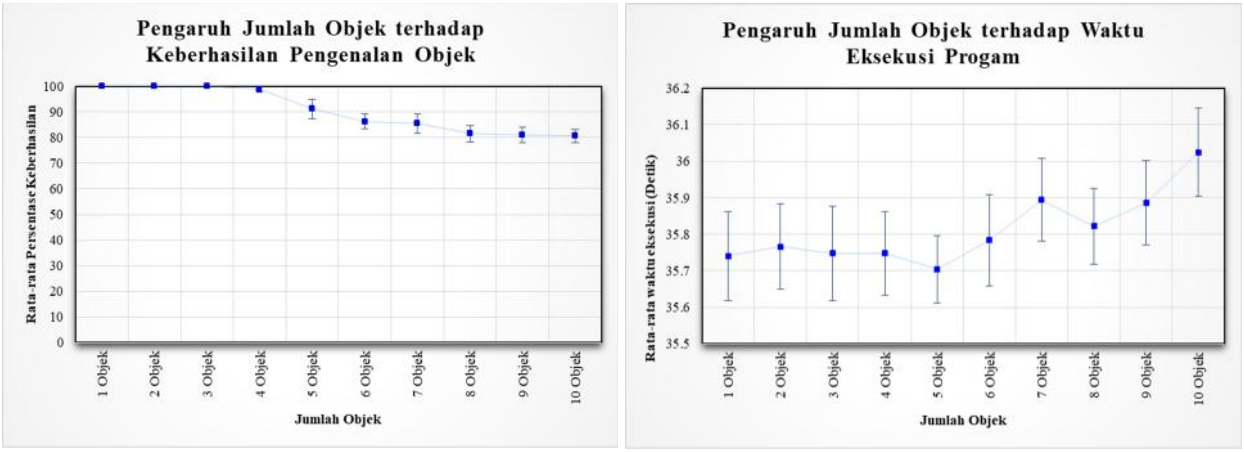
3. Pengujian Kondisi Pencahayaan



Gambar 14. Grafik Pengaruh Kondisi Pencahayaan terhadap Keberhasilan Pengenalan Objek

Gambar 14 adalah Grafik yang menunjukan pengaruh kondisi pencahayaan terhadap keberhasilan pengenalan atau identifikasi objek. Dalam pengujian ini kemiringan kamera adalah 0^{o} dan jarak kamera 12 cm dengan mengubah kondisi pencahaan. Grafik menunjukan rata-rata persentase keberhasilan dengan nilai maksimum adalah pada kondisi pencahayaan sebesar 120-200 lux yaitu 87,2% dan diikuti 200 lux sebesar 70,4%, sedangkan untuk kondisi pencahayaan >200 lux sebesar 58,8%.

4. Pengujian Jumlah Objek



Gambar 15. Grafik Pengaruh Jumlah Objek terhadap Keberhasilan Pengenalan Objek dan Waktu Eksekusi

Gambar 15 adalah Grafik yang menunjukan pengaruh jumlah objek terhadap keberhasilan pengenalan objek atau identifikasi objek. Pengujian pengaruh jumlah objek terhadap akurasi keberhasilan pengidentifikasian dilakukan untuk mengetahui kemampuan sistem dalam mengidentifikasi objek dalam jumlah tertentu dan mengetahui waktu yang dibutuhkan untuk mengeksekusi program. Dalam pengujian ini kemiringan kamera adalah 0^{o} dan jarak kamera

12 cm dengan kondisi pencahayaan adalah 120-200 lux dengan mengubah jumlah objek. Proses pengidentifikasian dieksekusi oleh sistem dengan rata-rata waktu yang dibutuhkan adalah 35-36 detik dan rata-rata 100% dari objek teridentifikasi benar untuk jumlah 1-3 objek kemudian menurun secara teratur untuk 4-10 objek dengan persentase keberhasilan 80.6-99%.

4. KESIMPULAN

1. Perancangan sistem terdiri dari 2 bagian yaitu mempersiapkan *model* sebagai data dan membuat sistem. Mempersiapkan *model* meliputi tahapan mengumpulkan *dataset* citra, *preprocessing* citra dan *training model*. Membuat sistem yaitu Raspberry Pi dan kamera untuk menguji *model* dengan mempertimbangkan posisi kamera, kondisi pencahayaan dan jumlah objek pada saat pengujian.
2. Tahap pelatihan *dataset* menggunakan 511 citra meliputi 414 citra latih dan 97 citra uji dan dilakukan hingga langkah ke 100.000 menghasilkan nilai *loss* sebesar 2.5.
3. Pengujian pengaruh jarak dan kemiringan kamera menunjukan bahwa jarak ideal antara kamera dan objek adalah 12-16 cm dan posisi kemiringan kamera yang baik adalah 0-20^{o} dimana rata-rata 87% dari objek dikenali dengan benar. Hasil pengujian pengaruh kondisi pencahayaan menunjukan sistem dapat mengidentifikasi objek dengan benar hingga 87.2% yaitu pada kondisi pencahaayaan 120-200 lux. Proses pengidentifikasian dieksekusi oleh sistem dengan rata-rata waktu yang dibutuhkan adalah 35-36 detik dan rata-rata 100% dari objek teridentifikasi benar untuk jumlah 1-3 objek kemudian menurun secara teratur untuk 4-10 objek dengan persentase keberhasilan 80.6-99%.

DAFTAR PUSTAKA

Banurea, Sodipta. (2017). *Pengenalan Nilai nominal Uang Logam Rupiah*. Universitas Sanata Dharma, Yogyakarta

Dewi, Syarifah. (2018). *Deep Learning Object Detection pada Video Menggunakan TensorFlow dan Convolutional Neural Network*. Universitas Islam Indonesia, Yogyakarta

Erhan, D, Szegedy, C and Toshev, A. (2014). "Scale Object Detection Using Deep Neural Network". *The CVF*, Vol.2147-2154

Kokoulin, A, Tur, A and Yuzhakov A. (2018). "Convolutional Neural Networks Application in Plastic Waste Recognition and Sorting". *ElConRus*, Vol. 1094-1098

Krizhevsky, A, Sutskever, I and Hinton, G. (2012). "Imagenet Classification With Deep Concolutional Neural Networks". *NIPS*, Vol.1097-1105

Lami, H and Pella, S. (2019). "Implementasi Deteksi dan Pengenalan Wajah pada Sistem Ujian Online Menggunakan Metode Deep Learning Berbasis Raspberry Pi". *Media Elektro*, Vol. 8, 1-4.

Maggiori, E, Tarabalka, Y and Charpiat, G. (2016). "Convolutional Neural Networks for Large-Scale Remote-Sensing Image Classification. *INRIA*, Vol. 645-657

Suyanto. (2018). *Machine Learning Tingkat Dasar dan Lanjut*. Informatika, Bandung

Taufiq, Imam. (2018). *Deep Learning untuk Deteksi Tanda Nomor Kendaraan Bermotor Menggunakan Algortitma Convolutional Neural Network dengan Python dan TensorFlow*. Sekolah Tinggi Manajemen Informatika dan Komputer AKAKOM, Yogyakarta

Wibawa, W, Sari, J and Ananda. (2012). "Perancangan dan Pembuatan Aplikasi untuk Mendeteksi Uang Logam dengan Metode Euclidean". *Jurnal Teknik Informatika*, Vol. 1, 1-7