IMPLEMENTASI *DEEP LEARNING* MENGGUNAKAN METODE *CONVOLUTIONAL NEURAL NETWORK* DAN ALGORITMA YOLO DALAM SISTEM PENDETEKSI UANG KERTAS RUPIAH BAGI PENYANDANG *LOW VISION*

Kevin Maulana Azhar[*)], Imam Santoso dan Yosua Alvin Adi Soetrisno

Program Studi Sarjana Departemen Teknik Elektro, Universitas Diponegoro
JL. Prof.Sudharto, SH, Kampus UNDIP Tembalang, Semarang 50275, Indonesia

[*)]*E-mail : kevinazhar54@gmail.com*

Abstrak

Penggunaan *deep learning* sudah diterapkan di berbagai sektor kehidupan karena memberikan dampak positif berupa kegiatan yang dikerjakan menjadi lebih efektif dan efisien. Implementasi *deep learning* dapat diterapkan untuk mendeteksi objek terntentu, salah satunya untuk mendeteksi objek uang kertas rupiah. Hal ini bertujuan untuk mempermudah penyandang *low vision* ketika akan menggunakan uang karena saat ini hanya menggunakan cara konvensional untuk membedakan nominal uang kertas rupiah seperti menyusun nominal ataupun membuat lipatan pada bagian uang. Pada tugas akhir ini, dirancang suatu sistem pendeteksian uang kertas rupiah pada citra bergerak menggunakan metode *Convolutional Neural Network* (CNN) untuk membantu penderita *low vision*. Algoritma untuk mendeteksi uang kertas rupiah menggunakan YOLO (*You Only Look Once*). Data latih yang digunakan berjumlah 1.260 gambar dengan 7.000 iterasi menghasilkan nilai mAP sebesar 97,65%. Hasil pengujian menggunakan data testing dengan jumlah data sebanyak 140 gambar bertujuan untuk mAP *cross validation* dan diperoleh nilai mAP sebesar 97,5%. Sementara itu, hasil pengujian dengan variasi pada berbagai kondisi uang menghasilkan tingkat mAP sebesar 88%.

Kata kunci : deep learning, deteksi, uang kertas rupiah, citra bergerak, CNN, YOLO.

Abstract

Use of deep learning has been applied in various sectors of life because it has a positive impact in the activities being carried out to be more effective and efficient. Implementation of deep learning can be applied to detect certain objects, one of them is to detect rupiah banknotes. This goals to make low vision easier to use money because currently they only use conventional methods to distinguish the nominal value, such as sorting the nominal or making folds on it. In this final project, a rupiah banknote detection system is designed on moving images using the Convolutional Neural Network (CNN) method to help people with low vision. The algorithm to detect rupiah banknotes uses YOLO (You Only Look Once). The training data used are 1,260 images with 7,000 iterations resulting in an mAP value of 97.65%. The results of the test using data testing with a total of 140 images aimed at mAP cross validation and obtained an mAP value of 97.5%. Meanwhile, the test results with variations in various money conditions resulted in an mAP level of 88%.
.

Kata kunci : deep learning, detection, rupiah banknotes, moving image,, CNN, YOLO.

1. Pendahuluan

Kecerdasan buatan atau sering disebut *Artificial Intelligence* (AI) sedang berkembang pesat di era modern saat ini. Kecerdasan buatan adalah bagian dari komputasi yang meneliti bagaimana mesin (komputer) dapat dibuat untuk berperilaku dan berpikir seperti manusia [1]. Salah satu pendekatan yang paling populer digunakan melalui *machine learning* (ML). Seperti namanya, pendekatan ini mencakup perancangan dan pembangunan algoritma untuk mengembangkan kemampuan komputer berdasarkan data yang telah diinputkan. Oleh karena itu machine learning membutuhkan data untuk digunakan dalam proses *training*. *Machine learning* memungkinkan mengolah banyak data dan mempelajari sebuah pola dalam data sehingga dapat dibuat prediksi di masa mendatang. Salah satu bagian machine learning yang paling banyak digunakan adalah *deep learning*.

Beberapa tahun terakhir *deep learning* telah menunjukan performa yang luar biasa. Hal ini sebagain besar dipengaruhi faktor komputasi yang lebih kuat, data set yang besar dan teknik untuk melatih jaringan yang lebih dalam [2]. Salah satu algoritma yang digunakan pada *deep learning* adalah *artificial neural network* (ANN) [3]. Jaringan ini memiliki sifat yang fleksibel dan dapat beradaptasi secara mandiri untuk menyelesaikan

permasalahan kompleks seperti pengenalan pola dan klasifikasi [4]. Kemampuan ANN dalam menyelesaikan permasalahan yang kompleks telah terbukti dari berbagai macam penelitian, seperti Analisis data, pengenalan pola, sistem kontrol, deteksi penomena kedokteran, dan sebagainya [5]. Selain algoritma ANN, *deep learning* juga memiliki algoritma lain seperti *convolutional neural networks* (CNN). Berbeda dengan ANN, pada algoritma CNN operasi linier menggunakan proses konvolusi dan bobot tidak dalam bentuk satu dimensi lagi, namun berbentuk minimal empat dimensi yang merupakan kernel konvolusi.

Perkembangan model algortima CNN yang banyak memberikan kemudahan dalam dunia *deep learning* menghasilkan beberapa model yang sudah terlatih dinamakan *pre trained model.* Pada *pre trained model* biasanya sudah dilakukan proses pelatihan pada dataset yang sangat besar seperti Imagenet, sehingga kualitas dari model tersebut sudah sangat baik. Beberapa contoh dari *pre trained model* yang populer digunakan dalam *deep learning* antara lain Alexnet, R-CNN, *Faster* R-CNN, *Single Shot Detector* (SSD), dan *You Only Look Once* (YOLO).

Beberapa bidang tertentu seperti olahraga, industri, dan medis telah banyak memanfaatkan teknologi *deep learning* dalam pengolahan citra digital, terutama pada bagian *object detection*. Terdapat beberapa penelitian pengolahan citra digital terkait pendeteksian objek yang telah dilakukan sebelumnya. Pada tahun 2016 terdapat penelitian mengenai *object tracking* untuk mendeteksi dan menghitung jumlah kendaraan secara otomatis menggunakan metode *kalman filter* dan *gaussian mixture* model [6]. Penelitian serupa juga dilakukan pada tahun 2019 namun menggunakan metode *convolutioinal neural network* [7]. Sementara itu, pada tahun 2017 telah dilakukan penelitian mengenai pendeteksian nominal uang yang diimplementasikan pada raspberry pi [8]. Penelitian di tahun yang sama dilakukan dalam pembuatan aplikasi pendeteksian nominal uang menggunakan metode *feature matching* [9]. Penelitian terhadap deteksi objek berupa manusia juga telah dilakukan, misal nya pada tahun 2019 dilakukan penerapan metode *convolutional neural network* dan *long short term memory* untuk pengenalan aktivitas manusia pada CCTV di area tambak udang [10]. Pada tahun yang sama juga dilakukan penelitian berupa monitoring ruangan untuk deteksi manusia berbasis CNN dengan fitur *push notification* [11].

Sejauh ini, para penderita *low vision* memakai cara konvensional seperti menyusun nominal uang kertas dan membuat lipatan uang untuk membedakan nominal uang tersebut. Namun, kedua cara tersebut masih mempunyai beberapa kelemahan, yaitu dari segi daya ingat, kondisi fisik uang dan tidak adanya faktor penentu kejujuran pada saat bertransaksi jual-beli barang dan jasa

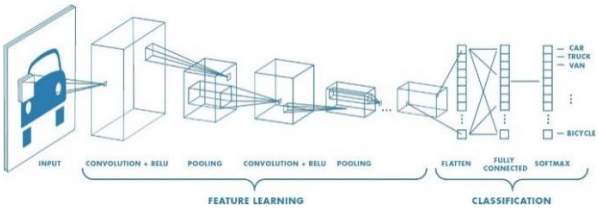
Penelitian ini bertujuan untuk mengimplementasikan *deep learning* pada aplikasi pendeteksi uang kertas rupiah untuk membantu penyandang low vision di Indonesia berbasis sistem operasi android. Algoritma yang digunakan untuk pengenalan uang adalah YOLOv4-tiny. Sistem *deep learning* ini akan dikolaborasikan kedalam aplikasi berbasis android sebagai pendukung utama untuk mengamati performa program dalam mendeteksi objek pada berbagai variasi kondisi.

2. Metode

2.1. Convolutional Neural Network

Convolutional Neural Network (CNN) adalah pengembangan dari *Multilayer Perceptron* (MLP) yang didesain untuk mengolah data dua dimensi [12]. CNN tergabung dalam jenis *Deep Neural Network* karena tingkat jaringan yang berlapis-lapis. CNN pertama kali dikembangkan oleh peneliti dari NHK *Broadcasting Science Research Laboratories*, Kunihiko Fukushima dan konsep tersebut dimatangkan oleh Yaan LeChun dalam penelitiannya [13].

Pada CNN dapat ditemukan sebuah *hidden layer*, dimana pada setiap *hidden layer* terdapat beberapa *neuron* yang saling terhubung satu sama lain. *Layer* akhir yang terhubung dengan hidden *layer* disebut *output layer* yang bertugas merepresentasikan hasil akhir dari proses konvolusi. Berdasarkan Gambar 1, diketahui secara umum lapisan CNN dibagi menjadi 2 bagian yaitu Feature Extraction *Layer* dan *Classification* Layer.



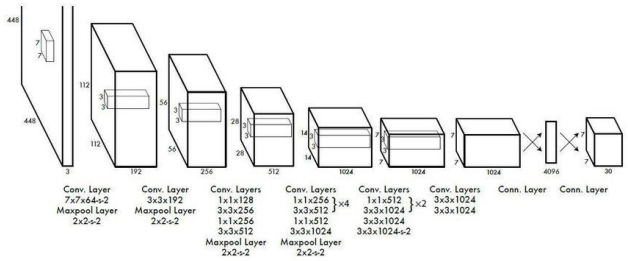
Gambar 1. Gambaran Umum Proses Konvolusi

2.2. YOLO

YOLO (*You Only Look Once*) merupakan susunan algoritma yang dimanfaatkan agar dapat melakukan pendeteksian sebuah objek secara *real-time*. Secara teknis, jaringan syaraf tiruan merupakan pendekatan yang digunakan oleh YOLO agar dapat mendeteksi sebuah objek. Proses YOLO diawali dengan membagi citra menjadi beberapa wilayah, kemudian memprediksi setiap *bounding box* dan probabilitas untuk setiap wilayah. Arsitektur YOLO terdiri dari 27 *layer* CNN, yaitu 24 *layer* yang terdiri dari *convolutional layer* dan *pooling layer* dengan ukuran 2x2 kernel, kemudian diikuti 2 *fully connected layer* dan sebuah *final decision layer* dengan ukuran kernel 1x1 seperti pada gambar 2 YOLO membagi citra masukan menjadi suatu *grid* berukuran S x S dimana

tiap sel dari *grid* tersebut akan memprediksi *bounding box* dan menghasilkan nilai untuk tiap kelas. Masing - masing *bounding box* terdiri dari lima nilai prediksi yaitu pusat koordinat x, pusat koordinat y, lebar sel (w), tinggi sel (h) dan nilai *confidence* [14].

Convolutional layer menggunakan filter sebagai parameter yang akan diupdate dalam proses pembelajaran. Filter diinisialisasi dengan nilai tertentu, kemudian dari kalkulasi *grid* pada citra dengan nilai filter dihasilkan nilai keluaran dalam bentuk *array*. Kemudian dengan menggeser (konvolusi) filter di setiap kemungkinan posisi pada citra untuk dihasilkan sebuah *activation map*.



Gambar 2. Arsitektur YOLO

Pada pembagian citra masukan menjadi ukuran *grid* S x S, bagian pusat yang menjadi titik tengah dari *grid cell* bertanggung jawab untuk proses pendeteksian objek. Masing-masing *grid* cell memprediksi *bounding box* dan nilai *confidence* untuk setiap *box*. Nilai *confidence* ini menyatakan berapa besar tingkat keyakinan sistem terhadap *bounding box* yang berisi inti informasi objek dan juga seberapa akurat *grid cell* dalam memprediksi *bounding box* tersebut. Secara matematis nilai *confidence* dapat dinyatakan sebagai berikut:

$$Confidence\ Score = P_r(object) * IOU_{pred}^{truth}$$

dimana Pr = probabilitas kelas objek, IOU = *Intersection Over Union*

IOU (*Intersection Over Union*) adalah salah satu metode yang digunakan untuk proses evaluasi terhadap akurasi deteksi objek dalam suatu data set. IOU memerlukan dua area untuk proses perpotongan (*intersection*) dan gabungan (*union*), yaitu area *ground-truth bounding box* dan *predicted bounding box* [15][16].

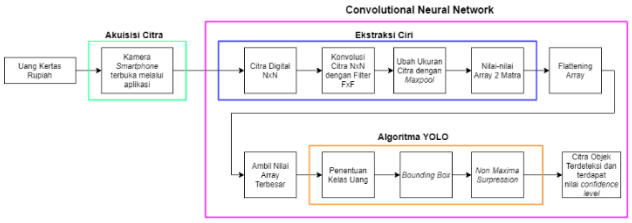
Persamaan IOU dapat dinyatakan sebagai berikut :

$$IOU = \frac{Irisan}{Gabungan}$$

2.3. Spesifikasi Sistem

Perancangan sistem pada penelitian tugas akhir ini terdiri dari tiga tahapan utama, yaitu penentuan spesifikasi komputer sebagai perangkat yang digunakan untuk proses *training* data, perancangan program pendeteksian uang yang ditulis menggunakan bahasa pemrograman *python* versi 3.7 menggunakan google colab, dan penentuan spesifikasi *smartphone*. Proses pendeteksian dibagikan menjadi empat belas kelas berbeda berdasarkan nominal uang kertas dalam rupiah. *Smartphone* merupakan perangkat akhir yang akan mejadi tempat pengujian dari hasil model *training* yang telah dilakukan pada perangkat komputer.

Informasi visual yang menjadi masukkan sistem berupa citra bergerak (video) yang diambil oleh kamera ponsel pintar *Samsung galaxy s7 edge* secara real-time. Penggunaan *library* Opencv digunakan untuk dapat meluncurkan kamera pada ponsel pintar. Dataset yang digunakan pada proses *training* didapatkan dengan melakukan pengambilan foto pada masing – masing nominal uang kertas rupiah menggunakan kamera ponsel pintar *Samsung galaxy s7 edge, Asus zenfone max pro m1* dan iPhone 8+ serta gambar dari *google images*.



Gambar 3. Diagram Blok Sistem Secara Keseluruhan

Citra uang yang ditangkap oleh kamera ponsel pintar akan diproses dalam sistem pengolahan citra berdasarkan model *training* yang telah diinputkan dalam program. Proses awal pada sistem setelah akuisisi citra adalah menjalankan CNN dengan melakukan konvolusi ukuran citra dengan filter dengan ukuran fxf sehingga dihasilkan keluaran berupa nilai *array* 2 matra. *Max pooling* digunakan pada program yang bertujuan untuk mereduksi input dari lapisan konvolusi. Kemudian dilakukan proses *flattening*, hasil dari proses *flattening* diolah pada *fully connected layer* dengan menjalankan algoritma YOLO untuk mendapatkan keluaran berupa citra objek yang telah terklasifikasi dengan *bounding box*. Perancangan sistem secara keseluruhan dapat digambarkan oleh diagram blok sistem pada Gambar 3.

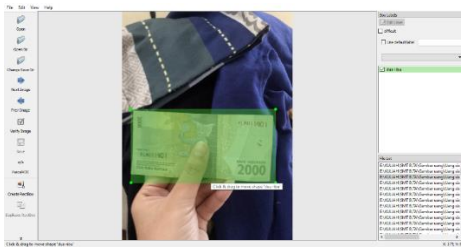
2.4. Pengumpulan Data Latih

Data yang digunakan pada penelitian ini berupa kumpulan gambar uang kertas rupiah emisi tahun 2016 yang terdiri dari 7 jenis nominal uang. Masing masing uang tersebut memiliki 2 kelas karena mempunyai sisi depan dan sisi belakang sehingga jumlah total keeluruhan adalah 14 kelas. Pengumpulan data untuk keempat belas kelas uang ini didapatkan dari data foto pribadi dan *google images*.

Pengumpulan data latih dilakukan dengan cara mengumpulkan citra sesuai dengan kelasnya. Masing – masing nominal uang memiliki data latih sebanyak 200 gambar terdiri dari 100 gambar sisi depan dan 100 gambar sisi belakang. Total data latih yang terkumpul dari 7 nominal uang kertas berjumlah 1.400 gambar. Data yang terkumpul selanjutnya dibagu menjadi 3 kategori yaitu Data *training*, *validation* dan *testing* dengan masing – masing bobot 70%, 20%, dan 10%.

2.5. Pelatihan Data

Pelatihan data latih dilakukan untuk proses ekstraksi ciri dari masing-masing kelas yang akan diklasifikasikan. Proses pelatihan dibagi menjadi dua bagian yaitu proses anotasi dan *training*. Gambar 4 merupakan contoh gambar dalam proses anotasi menggunakan aplikasi *labelimg*.



Gambar 4. Proses Anotasi Data Latih

Framework yang digunakan untuk proses *training* adalah *darknet* menggunakan *google colab*. Jumlah kelas yang digunakan untuk klasifikasi objek pada proses pelatihan tertera dalam file Obj.data yang telah ditentukan sebelumnya. Pada file ini juga terdapat persebaran jumlah data *training*, data validasi, dan data *testing*. Jumlah pada masing – masing persebaran data sebesar 70% untuk data *training* (980 gambar), 20% data validasi (280 gambar), dan 10% data *testing* (140 gambar). Sementara untuk mengidentifikasi jumlah kelas, program akan membaca file bertipe .labels yang berada pada data *training*. Jumlah *layer*, filter dan desain arsitektur CNN yang digunakan dirancang pada *file custom-yolov4-tiny-detector*.cfg yang kemudian dipanggil dari terminal untuk digunakan pada proses *training*. Banyaknya jumlah iterasi yang akan dijalankan yaitu sejumlah 7000 iterasi.

2.6. *Testing* Data

Proses *testing* dilakukan setelah proses *training* selesai dilakukan dengan tujuan untuk melakukan pengujian apakah program dapat mengenali sebuah objek diluar data *training* dan data validasi yang telah digunakan sebelumnya. Kumpulan gambar yang termasuk didalam data *testing* ditempatkan di direktori yang berbeda dengan data *training* dan data validasi. Jumlah data *testing* yang akan dipakai sebanyak 10% dari total 1.400 gambar.

Tahap pertama yang akan dilakukan yaitu memuat 38 *layer* yang sama digunakan pada proses *training*. Selanjutnya program akan melakukan proses deteksi menggunakan *file* bobot yang telah disimpan dan menghasilkan klasifikasi objek beserta tingkat *confidence level nya*.

3. Hasil dan Analisa

Setelah dilakukan proses *training* dan *testing* dengan total data sebanyak 1.400 gambar dan perancangan program, maka tahapan selanjutnya adalah melakukan pembahasan dan analisis hal mengenai pengujian yang akan dilakukan. Tujuan dilakukan pengujian ini adalah untuk mengetahui hasil keluaran dan tingkat keakuratan dari program yang telah diimplementasikan. Data masukan yang digunakan berupa citra bergerak yang berasal dari kamera ponsel Android. Pengujian dibagi menjadi 3 bagian, yaitu :

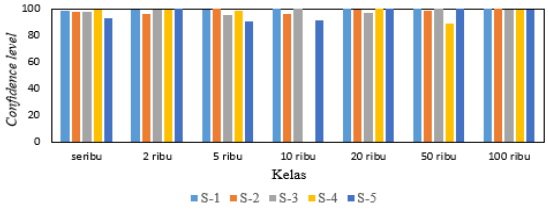
1. Pengujian performa pendeteksian program dengan variasi kondisi uang.
2. Pengujian performa pendeteksian program dengan kondisi cahaya redup.
3. Pengujian performa program terhadap waktu respon deteksi.

3.1. Pengujian Performa Pendeteksian Program Dengan Variasi Kondisi Uang

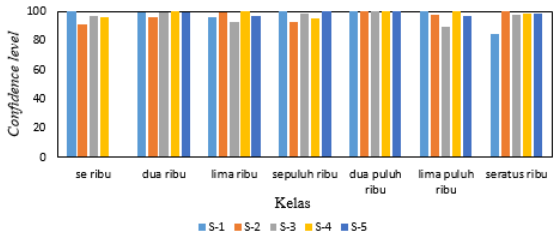
Pengujian performa program dengan variasi kondisi uang memiliki 3 variasi pecobaan yaitu kondisi uang seluruh bagian, kondisi uang bagian kiri, dan kondisi uang bagian kanan. Pemilihan variasi ini dibuat untuk menguji apakah program dapat mengenali objek ketika beberapa bagian uang saja yang masuk dalam frame kamera

3.1.1. Pengujian Performa Program Kondisi Uang Seluruh bagian

Pada pengujian performa program dengan kondisi uang seluruh bagian mengindikasikan ketika program akan mendeteksi objek uang, seluruh bagian uang masuk kedalam frame kamera. Pengujian ini akan dilakukan terhadap semua kelas sebanyak 14, dengan jumlah 5 gambar pada masing – masing kelas. Proses pendeteksian objek dilakukan dengan mengarahkan kamera *smartphone* ke objek uang yang akan dideteksi.



Gambar 5. Grafik Performa Program Kelas Seribu, 2 ribu, 5 ribu, 10 ribu, 20 ribu, 50 ribu, 100 ribu Seluruh Bagian

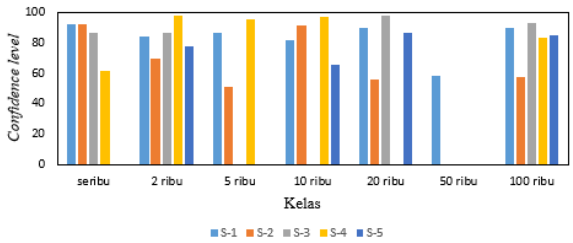


Gambar 6. Grafik Performa Program Kelas Se Ribu, Dua Ribu, Lima Ribu, Sepuluh Ribu, Dua Puluh Ribu, Lima Puluh Ribu, Seratus Ribu

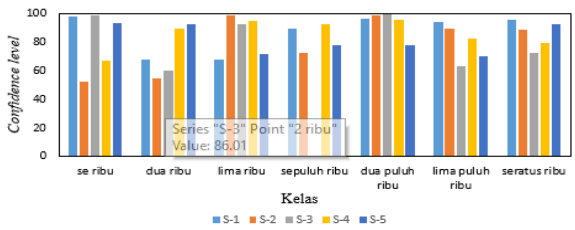
Berdasarkan Gambar 5 dan Gambar 6 diketahui bahwa nilai *confidence level* program dalam mendeteksi objek pada 14 kelas selalu berubah pada tiap sampelnya. Pada gambar terdapat parameter S-*n* yang memiliki arti sampel ke-n pada kelas tersebut, dimana *n* berada di rentang 1 sampai 5. Pada percobaan kali ini program tidak berhasil mendeteksi semua objek. Terdapat kesalahan prediksi yang dilakukan oleh program saat mendeteksi sampel keempat pada kelas "10 ribu" dan sampel kelima kelas "se ribu".

3.1.2. Pengujian Performa Program Kondisi Uang Bagian Kiri

Pada pengujian performa program dengan kondisi uang bagian kiri mengindikasikan ketika program akan mendeteksi objek uang, hanya bagian kiri uang masuk kedalam frame kamera. Pengujian ini akan dilakukan terhadap semua kelas sebanyak 14, dengan jumlah 5 gambar pada masing – masing kelas. Proses pendeteksian objek dilakukan dengan mengarahkan kamera *smartphone* ke objek uang yang akan dideteksi.



Gambar 7. Grafik Performa Program Kelas Seribu, 2 Ribu, 5 Ribu, 10 Ribu, 20 Ribu, 50 Ribu, 100 Ribu Bagian Kiri

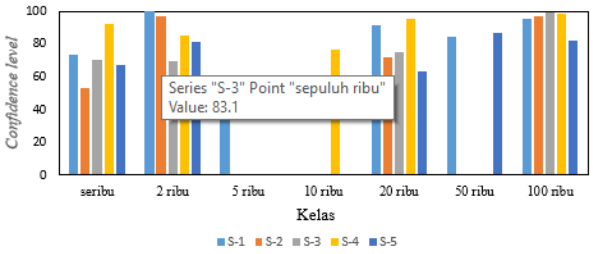


Gambar 8. Grafik Performa Program Kelas Seribu, 2 Ribu, 5 Ribu, 10 Ribu, 20 Ribu, 50 Ribu, 100 Ribu Bagian Kiri

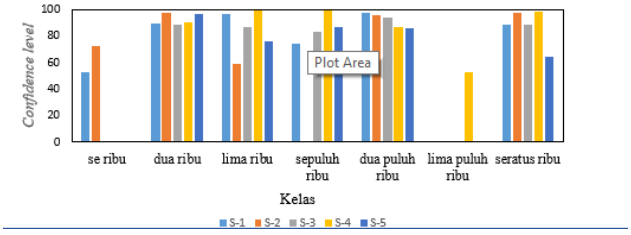
Berdasarkan Gambar 6 dan Gambar 7 diketahui bahwa nilai *confidence* program dalam mendeteksi objek pada 14 kelas selalu berubah pada tiap sampelnya. Pada gambar terdapat parameter S-*n* yang memiliki arti sampel ke-n pada kelas tersebut, dimana *n* berada di rentang 1 sampai 5. Pada percobaan kali ini program tidak berhasil mendeteksi semua objek. Program lebih kesulitan untuk mendeteksi bagian kiri dari sisi depan dibanding sisi belakang uang. Hal ini dapat kita lihat bahwa beberapa kelas yang merepresentasikan sisi depan uang memiliki *confidence level* yang tidak stabil dan bahkan terdapat 8 objek uang yang tidak dapat dikenali ataupun salah diprediksi oleh program. Sedangkan kelas – kelas yang merepresentasikan sisi belakang uang memiliki *confidence level* yang lebih merata daripada sisi depan dan hanya terdapat 1 objek uang yang tidak dapat dikenali oleh program. Hal ini dapat terjadi karena perbedaan data *training* pada masing – masing kelas yang digunakan sebelumnya sehingga mempengaruhi hasil deteksi uang jika bagian kiri saja yang masuk kedalam *frame* kamera.

3.1.3. Pengujian Performa Program Kondisi Uang Bagian Kanan

Pada pengujian performa program dengan kondisi uang bagian kanan mengindikasikan ketika program akan mendeteksi objek uang, hanya bagian kanan uang masuk kedalam frame kamera. Pengujian ini akan dilakukan terhadap semua kelas sebanyak 14, dengan jumlah 5 gambar pada masing – masing kelas. Proses pendeteksian objek dilakukan dengan mengarahkan kamera *smartphone* ke objek uang yang akan dideteksi.



Gambar 9. Grafik Performa Program Kelas Seribu, 2 Ribu, 5 Ribu, 10 Ribu, 20 Ribu, 50 Ribu, 100 Ribu Bagian Kanan



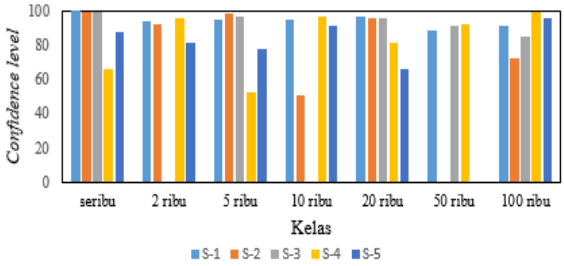
Gambar 10. Grafik Performa Program Kelas Seribu, 2 Ribu, 5 Ribu, 10 Ribu, 20 Ribu, 50 Ribu, 100 Ribu Bagian Kanan

Berdasarkan Gambar 9 dan Gambar 10 diketahui bahwa nilai *confidence level* program dalam mendeteksi objek pada 14 kelas selalu berubah pada tiap sampelnya. Pada gambar terdapat parameter S-*n* yang memiliki arti sampel ke-n pada kelas tersebut, dimana *n* berada di rentang 1 sampai 5. Pada percobaan kali ini program tidak berhasil mendeteksi semua objek. Program lebih kesulitan untuk mendeteksi bagian kanan dari sisi depan dibanding sisi belakang uang. Hal ini dapat kita lihat bahwa beberapa kelas yang merepresentasikan sisi depan uang memiliki *confidence level* yang tidak stabil dan bahkan terdapat 11 objek uang yang tidak dapat dikenali ataupun salah diprediksi oleh program. Sedangkan kelas – kelas yang merepresentasikan sisi belakang uang memiliki *confidence level* yang lebih merata daripada sisi depan dan terdapat 8 objek uang yang tidak dapat dikenali oleh program. Hal ini dapat terjadi karena perbedaan data *training* pada masing – masing kelas yang digunakan sebelumnya sehingga mempengaruhi hasil deteksi uang jika bagian kiri saja yang masuk kedalam *frame* kamera.

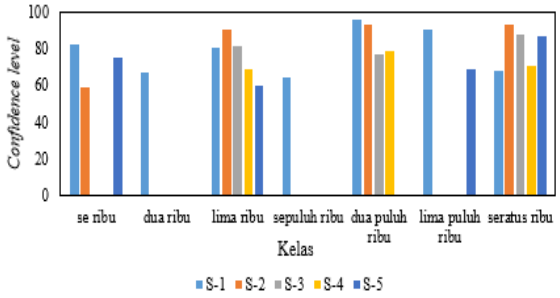
3.2. Pengujian Performa Pendeteksian Program Variasi Kondisi Cahaya Redup

Pada pengujian performa program dengan kondisi uang cahaya redup mengindikasikan ketika program akan mendeteksi objek uang, kondisi cahaya sekitar dalam keadaan redup. Pengujian ini akan dilakukan terhadap semua kelas sebanyak 14, dengan jumlah 5 gambar pada masing – masing kelas. Proses pendeteksian objek dilakukan dengan mengarahkan kamera *smartphone* ke objek uang yang akan dideteksi.

Berdasarkan Gambar 11 dan Gambar 12 diketahui bahwa nilai *confidence* program dalam mendeteksi objek pada 14 kelas selalu berubah pada tiap sampelnya. Pada gambar terdapat parameter S-n yang memiliki arti sampel ke-n pada kelas tersebut, dimana *n* berada di rentang 1 sampai 5. Pada percobaan kali ini program tidak berhasil mendeteksi semua objek. Program lebih kesulitan untuk mendeteksi sisi belakang dibanding sisi depan uang. Hal ini dapat kita lihat bahwa beberapa kelas yang merepresentasikan sisi belakang uang memiliki *confidence level* yang tidak stabil dan bahkan terdapat 14 objek uang yang tidak dapat dikenali ataupun salah diprediksi oleh program. Sedangkan kelas – kelas yang merepresentasikan sisi depan uang memiliki *confidence level* yang lebih merata daripada sisi belakang dan terdapat 4 objek uang yang tidak dapat dikenali oleh program. Hal ini dapat terjadi karena kondisi cahaya yang redup sehingga program sulit mengenali objek uang yang akan dideteksi.



Gambar 11. Grafik Performa Program Kelas Seribu, 2 Ribu, 5 Ribu, 10 Ribu, 20 Ribu, 50 Ribu, 100 Ribu Cahaya Redup



Gambar 12. Grafik Performa Program Kelas Seribu, 2 Ribu, 5 Ribu, 10 Ribu, 20 Ribu, 50 Ribu, 100 Ribu Bagian Kanan

3.3. Pengujian performa program terhadap waktu respon deteksi

Pada pengujian performa program terhadap waktu respon deteksi bertujuan untuk mengetahui kecepatan program untuk dapat mendeteksi objek uang. Perhitungan waktu respon dimulai ketika objek sudah masuk kedalam *frame* kamera dan berhenti saat program dapat mendeteksi nominal uang. Pengujian ini akan dilakukan terhadap semua kelas sebanyak 14, dengan jumlah 5 gambar pada masing – masing kelas. Data yang digunakan pada pengujian performa program terhadap waktu respon deteksi yaitu data pengujian performa program dengan kondisi uang seluruh bagian.

Setelah dilakukan pengujian dan analisis data dapat diketahui bahwa rata – rata waktu respon program untuk seluruh kelas berada diantara rentang 0,97 – 1,82 detik. Rata – rata waktu respon tercepat berada di kelas "lima puluh ribu" dengan nilai 0,99 detik, sedangkan waktu respon terlama berada di kelas "2 ribu" dengan nilai 1,82 detik. Kemudian dari data tersebut dapat dihitung rata – rata waktu respon secara keseluruhan dari program untuk mendeteksi objek uang dengan 14 kelas sebesar 1,28 detik.

3.4. Perhitungan Nilai mAP Keseluruhan Pengujian

Proses evaluasi performa program dapat dihitung menggunakan nilai yang didapatkan dari *precision* dan *recall* masing – masing kelas. Proses perhitungan *precision* dan *recall* menggunakan tiga parameter *confusion matrix* yaitu *true positive* (TP), *false positive* (FP) dan *false negative* (FN). Nilai *precision* diperoleh dengan cara perbandingan antara total parameter *true positive* (TP) dengan banyaknya data yang diprediksi positif. Sementara itu, nilai *recall* diperoleh dengan cara perbandingan antara total parameter *true positive* (TP) dengan banyaknya data yang sebenarnya positif. Tingkat akurasi dapat dihitung menggunakan metode *mean average precision* (mAP). Nilai mAP biasanya digunakan sebagai indikator performa akurasi dalam deteksi objek seperti pada algoritma *faster R-CNN*, SSD, dan termasuk YOLO.

Selanjutnya, untuk menghitung nilai *mean average precision* (mAP) digunakan teknik interpolasi 11 poin. Teknik ini akan menggunakan nilai tertinggi dari *precision* jika terdapat kesamaan nilai *Interpolated precision* adalah *precisoin* rata-rata yang diukur pada 11 tingkat *recall* dengan jarak yang sama yaitu dari 0,0, 0.1, 0,2, 0,3 hingga 1,0. Tabel 1 menunjukkan hubungan nilai *recall* dan *precision* setelah dilakukan teknik interpolasi 11 poin.

Tabel 1. Nilai *recall* dan *precision* setelah interpolasi 11 poin

Recall	*Interpolated Precision*
0,0	0,85
0,1	0,85
0,2	0,85
0,3	0,85
0,4	0,85
0,5	0,85
0,6	0,85
0,7	1,00
0,8	0,94
0,9	0,78
1,0	1,00

Berdasarkan Tabel 1 dapat diketahui hubungan masing – masing nilai *recall* dan *precision* setelah dilakukan teknik interpolasi 11 poin. Nilai *recall* pada rentang 0,0 hingga 0,6 memiliki nilai *precision* yang sama yaitu 0,85 karena pada pengujian keseluruhan nilai *recall* paling rendah berada di angka 0,6 sehingga rentang nilai *recall* yang berada dibawah angka 0,6 akan memiliki nilai *precision* yang sama. Selanjutnya proses perhitungan *mean average precision* (mAP) dapat dilakukan karena semua nilai *precision* dan *recall* sudah didapatkan. Perhitungan nilai *mean average precision* (mAP) ditunjukkan sebagai berikut.

$$mAP = \frac{1}{N} \sum_{recall(i)}^{N} precision\big(recall(i)\big)$$

$$= \frac{1}{11}[(7 * 0{,}85) + (1 * 1) + (1 * 0{,}94) + (1 * 0{,}78) + (1 * 1)]$$

$$= 0{,}88$$

Berdasarkan perhitungan nilai *mean average precision* (mAP) menggunakan rumus diatas, dapat diketahui bahwa nilai mAP program setelah melakukan pengujian variasi kondisi uang (seluruh bagian, bagian kanan, bagian kiri) dan variasi kondisi cahay redup menghasilkan nilai 88%.

4. Kesimpulan

Pada perhitungan nilai mAP sistem berdasarkan keseluruhan pengujian yang meliputi pengujian dengan berbagai variasi kondisi uang (seluruh bagian, bagian kiri, dan bagian kanan) serta penguhian pada variasi kondisi cahaya redup mencapai nilai mAP sebesar 88%. Rata – rata waktu respon program untuk dapat mendeteksi uang sebesar 1,28 detik.

Dari hasil penelitian ini, disarankan untuk menggunakan data latih dan iterasi pelatihan yang lebih banyak untuk mendapatkan tingkat akurasi.yang lebih tinggi. Disamping itu penggunaan kamera serta jarak dan sudut pengambilan citra serta proses anotasi citra yang optimal dapat meningkatkan akurasi.

Referensi

[1]. D. Muhammad, "Kecerdasan Buatan (Artificial Intelligence)," Jurnal Saintikom, vol. 5, no. 2, hal. 185-196, Agt. 2008.

[2]. G. Y. Bengio dan A. Courville, *Deep Learning (Adaptive Computation and Machine Learning Series)* USA : MIT Press, 2016.

[3]. A. Ahmad, "Mengenal Artificial Intelligence, Machine Learning, Neural Network, dan *Deep learning*," Jurnal Teknologi Indonesia, Jun. 2017.

[4]. L. Yifei, "Deep neural networks and fraud *detection*," U.U.D.M. Project Report 2017:38. 2017.

[5]. N. Yanti, "Penerapan Metode Neural Network Dengan Struktur Backpropagation Untuk Prediksi Stok Obat Di Apotek (Studi Kasus : Apotek ABC)," dipresentasikan di Seminar Nasional Aplikasi Teknologi Informasi, Yogyakarta, Indonesia, 2011.

[6]. H. V. A. Kautsar dan K. Adi, "Implementasi *Object* Tracking Untuk Mendeteksi Dan Menghitung Jumlah Kendaraan Secara Otomatis Menggunakan Metode Kalman Filter Dan Gaussian Mixture Model," Youngster Physics Journal, vol. 5, no. 1, hal 13-20, Jan. 2016.

[7]. M. A. Hudaya, "Implementasi *Object* Tracking Untuk Mendeteksi Dan Menghitung Jumlah Kendaraan Secara Otomatis Menggunakan Metode Kalman Filter Dan Gaussian Mixture Model," Laporan Tugas Akhir, Departemen Teknik Elektro, Universitas Diponegoro, Semarang. 2019

[8]. W. P. Candrawasih, "Rancang Bangun Alat Bantu Pendeteksi Nominal Uang Kertas Untuk Tuna Netra Menggunakan Kamera Berbasis Raspberry Pi," Laporan Tugas Akhir, Departemen Teknik Elektro, Institut Teknologi Malang, Malang. 2017.

[9]. G. Khoharja, "Aplikasi Deteksi Nilai Uang pada Mata Uang Indonesia dengan Metode Feature Matching," Laporan Tugas Akhir, Departemen Teknik Informatika, Universitas Kristen Petra, Surabaya. 2017.

[10]. M. A. Zulfikar, "Penerapan Metode Convolutional Neural Network (CNN) Dan Long Short Term Memory (LSTM) Untuk Pengenalan Aktivitas Manusia Pada Cctv Di Area Tambak Udang," Laporan Tugas Akhir, Departemen Teknik Elektro, Universitas Diponegoro, Semarang. 2020.

[11]. W. Swastika, dkk. "Monitoring Ruangan Untuk Deteksi Manusia Berbasis CNN Dengan Fitur Push Notification," Jurnal TEKNIKA, vol. 8, no. 2 hal 92-96, Nov. 2019.

[12]. S. R. Putra, "Implementasi Convolutional Neural Network Untuk Klasifikasi Obyek Pada Citra," Laporan Tugas Akhir, Jurusan Teknik Informatika, Institut Teknologi Sepuluh Nopember, Surabaya. 2015.

[13]. I. S. E. Putra, A. Y. Wijaya dan R. Soelaiman, "Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) Pada Caltech 101," Jurnal Teknik ITS, vol. 5 no.1, hal. A65-A69, 2016. ISSN: 2337-3539.

[14]. J. Redmon, S. Divvala, R. Girshick dan A. Farhadi, "You Only Look Once: Unified, Real-Time *Object Detection*," dipresentasikan di CVPR, Las Vegas, Amerika Serikat, Jun. 26 – Jul. 1, 2016.

[15]. Z. A. Fikriya, M. I. Irawan dan Soetrisno, "Implementasi Extreme Learning Machine untuk Pengenalan Objek Citra Digital," Jurnal Sains dan Seni ITS, vol. 6, no. 1, 2017, ISSN: 2337-3520.

[16]. S. Jupiyandi, F. Rizqullah, Y. Pratama, M. R. Dharmawan, dan I. Cholissodin,"Pengembangan Deteksi Citra Mobil Untuk Mengetahui Jumlah Tempat Parkir Menggunakan Cuda dan Modified Yolo," Jurnal Teknologi Informasi dan Ilmu Komputer, vol.6, no.4, hal. 413-419, 2018. ISSN: 2528-6579.