Terakreditasi SINTA Peringkat 2
Surat Keputusan Dirjen Penguatan Riset dan Pengembangan Ristek Dikti No. 10/E/KPT/2019
masa berlaku mulai Vol. 1 No. 1 tahun 2017 s.d Vol. 5 No. 3 tahun 2021

Terbit online pada laman web jurnal: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 4 No. 3 (2020) 476 - 481 ISSN Media Elektronik: 2580-0760

Deteksi Buah untuk Klasifikasi Berdasarkan Jenis dengan Algoritma CNN Berbasis YOLOv3

Hr.Wibi Bagas N[1], Evang Mailoa[2], Hindriyanto Dwi Purnomo[3]
[1),2),3)]Teknik Informatika, Teknologi Informasi, Universitas Kristen Satya Wacana
[1]672013220@student.uksw.edu, [2]Evang.mailoa@uksw.edu, [3]Hidriyanto.fti@gmail.com

Abstract

The fruit is part of the flowers in plants that are produced from pollination of the pistils and stamens. The shape and color of many fruits with a variety, with the type of single fruit, double fruit and compound fruit. This study asks for the development of 10 pieces detection applications to help the sensor agriculture sector for 10 pieces detection. The data in this study used the image of 10 fruits namely Mangosteen, Delicious, Star Fruit, Water Guava, Kiwi, Pear, Pineapple, Salak, Dragon Fruit, and Strawberry. Training and testing using CNN algorithms and YOLOv3 machine learning methods with the support of the work of the Darknet53 neural network. The analysis was conducted using 2,333 images of data from 10 classes. The training process is carried out up to 5,000 iterations stored in checkpoints. The implementation of the detection of 10 pieces was carried out on Google Collaboratory through imagery with two tests. Accuracy in the detection of 10 pieces can reach more than 90% in the first test of each fruit and an average of 70% in the second test for images outside the test data.

Keywords: Detection,YOLOv3(You Only Lock Once), CNN(Convolutional Neural Network), Darknet, Google Colaboratory.

Abstrak

Buah merupakan bagian dari bunga pada tumbuhan yang dihasilkan dari penyerbukan putik dan benang sari. Bentuk dan warna buah banyak serta beragam, dengan tipe buah tunggal, buah ganda dan buah majemuk. Penelitian ini diajukan untuk pengembangan aplikasi deteksi 10 buah untuk membantu dalam bidang pertanian sebagai sensor untuk deteksi 10 buah. Data dalam penelitian ini menggunakan citra 10 buah yaitu Manggis, Apel delicious, Belimbing, Jambu air, Kiwi, Pir, Nanas, Salak, Buah naga, dan Stroberi. Pelatihan dan pengujian menggunakan algoritma *CNN* dan metode pembelajaran mesin *YOLOv3* dengan kerangka kerja jaringan saraf Darknet53 . Analisa yang dilakukan menggunakan data 2.333 gambar dari 10 kelas. Proses latihan dilakukan hingga 5.000 iterasi yang disimpan dalam checkpoint. Penerapan deteksi 10 buah ini dilakukan pada google colaboratory melalui citra dengan dua kali pengujian. Akurasi dalam deteksi 10 buah ini bisa mencapai lebih dari 90% pada pengujian pertama dari setiap buah dan rata-rata 70% pada pengujian kedua untuk citra diluar data test.

Kata kunci: Deteksi,YOLOv3(You Only Lock Once), CNN(Convolutional Neural Network), Darknet, Google Colaboratory.

1. Pendahuluan

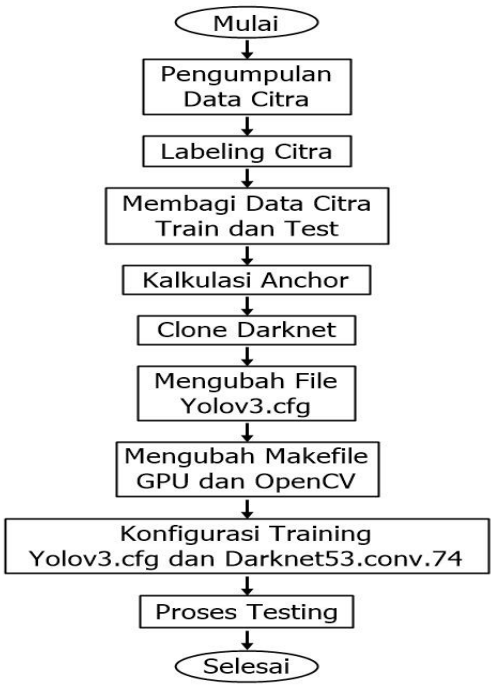
Deteksi buah merupakan salah satu teknologi yang sekarang banyak dikembangkan teknologi komputer. Pengenalan buah digunakan untuk membandingkan suatu citra buah dengan database buah dan menghasilkan akurasi tinggi yang paling cocok dengan citra buah tersebut. Data citra buah pada penelitian ini memiliki 10 kelas yang dibagi menjadi train dan test. Beberapa metode sudah digunakan untuk mengembangkan teknologi deteksi buah yang *modern*, seperti metode deteksi objek dengan metode pembelajaran mesin SVM(Support Vector Machine)[1] dan Sistem deteksi buah dan efektor akhir untuk pemanenan robotik dari apel fuji[2]. Dengan digunakannya teknologi deteksi buah ini memungkinkan dari bidang pertanian mengidentifikasi buah menggunakan bantuan komputer dan bantuan perantara kamera sebagai pengelola citra. Tujuan dari penelitian ini adalah untuk mengembangkan suatu metode deteksi buah melalui gambar yang akurat dan handal terhadap berbagai citra buah secara otomatis. Tahap pertama melakukan klasifikasi biner yang digunakan pelatihan menggunakan data yang berupa citra-citra buah dari 10 buah. Pemilihan untuk citra diambil dari buah yang tidak mentah dan tidak terlalu masak, 10 buah yang menjadi bahan citra antara lain buah Manggis, Apel delicious, Belimbing, Jambu air, Kiwi, Pir, Nanas, Salak, Buah naga, dan Stroberi yang

Diterima Redaksi : 06-04-2020 | Selesai Revisi : 12-05-2020 | Diterbitkan Online : 20-06-2020

nantinya akan diberi label. Tahap pemberian label pada citra dilakukan menggunakan LabelImg. Penerapan pembelajaran mesin deteksi objek berbeda dengan pembelajaran mesin lain seperti SVM(Support Vector Machine, SSD(Singel Multibox Shot Detector), Faster R- CNN(Faster Regional Convolutional Neural Network) dll. Pembelajaran mesin pada penelitian ini menggunakan YOLOv3 yang merupakan model pembelajaran mesin terbaru saat ini dan 3x lebih cepat yang beroperasi dalam 22 m/s pada 28,2 mAP(mean Average Precision) dan fps pada yolo dasar 45 frame per detik [3]. Deteksi buah diterapkan dengan bahasa python pada google colaboratory yang mendukung GPU(Graphics Processing Unit) tesla K80 hingga 12GB, dengan adanya GPU bisa menghemat waktu konfigurasi training dengan jumlah data lebih dari 2.000 untuk mendapat akurasi tinggi.

2. Metode Penelitian

Pengumpulan data citra untuk deteksi 10 buah diambil dari buah yang siap di konsumsi tidak terlalu mentah dan tidak terlalu masak. Secara garis besar alur pembuatan deteksi 10 buah terlihat seperti gambar 1.



Gambar 1. Alur deteksi 10 buah

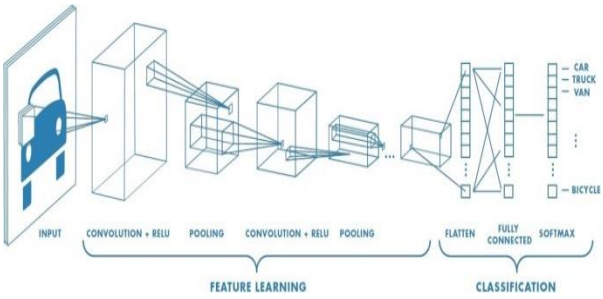
Citra data yang dikumpulkan sebanyak 2.333 citra dengan tipe file JPG dan ukuran 768x1024 akan di beri label dari LabelImg menggunakan anaconda prompt, citra yang sudah di beri label di bagi menjadi *train* dan *test*. Tidak ada aturan pasti untuk pembagian, tetapi dalam penelitian ini dibagi dengan presentase 91.5% untuk train = 2133 citra dan 8.5% test = 200 citra sebagai pengujian. Dalam menunjang deteksi 10 buah, peneliti juga menggunakan metode anchor dimana metode ini menggunakan 4 titik yang menjadi kotak jangkar pada citra. Citra data yang dibagi dua digunakan untuk model pelatihan dan pengujian. Proses pelatihan diharapkan menghasilkan jumlah *loss* kecil sehingga menghasilkan akurasi tinggi untuk pengujian, metode yang digunakan ialah CNN(Convolutional Neural Network) berbasis YOLOv3(You Only Lock Once) dengan perhitungan *loss* menggunakan MSE(Mean Squared Error)[4]. Kerangka kerja menggunakan jaringan saraf Darknet serta menggunakan bahasa python pada Google Colaboratory.

2.1. Anaconda

Anaconda adalah salah satu dari banyak platform open source yang memfasilitasi penggunaan bahasa pemrograman open source (R, Python) untuk pemrosesan data skala besar, analitik prediktif, dan komputasi ilmiah. Komunitas riset lingkungan dapat memilih untuk mengadaptasi penggunaan salah satu dari bahasa pemrograman R atau Python untuk mengalisis masalah sains data pada platform Anaconda[5].

2.2. CNN(Convolutional Neural Network)

CNN adalah pengembangan dari Multilayer Perceptron (MLP) yang didesain untuk mengolah data dua dimensi. CNN termasuk dalam jenis Deep Neural Network karena kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra khususnya pada kasus klasifikasi citra[6].



Gambar 2. Lapisan CNN[7]

a. Feature Learning

Lapisan yang terdapat dalam *Feature Learning* berguna untuk mentranslasikan suatu *input* menjadi *feature* berdasarkan ciri *input* yang berbentuk angka dalam vektor.

Convolutional Layer : Menghitung *output* dari *neuron* yang terhubung ke daerah lokal dalam *input*, masing-masing menghitung produk titik antara bobot mereka dan wilayah kecil yang terhubung ke dalam volume *input*.

Rectified Linear Unit (ReLU) : Menghilangkan *vanishing gradient* dengan cara menerapkan fungsi aktivasi element sebagai f(x)=max(0,x) alias aktivasi elemen akan dilakukan saat berada di ambang batas 0.

Pooling layer : Adalah lapisan yang mengurangi dimensi dari *feature map* atau lebih dikenal dengan langkan untuk *downsampling*, sehingga mempercepat komputasi karena parameter yang harus diupdate semakin sedikit dan mengatasi *overfitting*.

b. Classification

Lapisan ini berguna untuk mengklasifikasikan tiap *neuron* yang telah diekstraksi fitur pada sebelumnya, terdiri dari :

Flatten : Membentuk ulang fitur (*reshape feature map*) menjadi sebuah vector agar bisa digunakan sebagai *input* dari *fully-connected layer*.

Fully-connected : Lapisan FC (yaitu terhubung sepenuhnya) akan menghitung skor kelas. Seperti Jaringan Saraf biasa dan seperti namanya, setiap *neuron* dalam lapisan ini akan terhubung ke semua angka dalam volume.

Softmax : Fungsi *Softmax* menghitung probabilitas dari setiap kelas target atas semua kelas target yang memungkinkan dan akan membantu untuk menentukan kelas target untuk *input* yang diberikan[7].

2.3. YOLOv3(You Only Lock Once)

YOLOv3 sejauh ini menjadi salah satu algoritma pendeteksi terbaik, memiliki penerapan berdasarkan YOLOv2 dalam deteksi muti-skala, klasifikasi multi-tabel,dll. YOLOv3 menskala gambar asli ke ukuran 416 x 416 menggunakan skala struktur piramida yang mirip dengan jaringan FPN, dan membagi gambar asli menjadi sel S × S sesuai dengan ukuran peta fitur. Jaringan saraf convolutional memprediksi 4 nilai untuk setiap kotak pembatas pada setiap sel, yaitu koordinat (x, y) dan lebar w dan tinggi h dari target, yang masing-masing dicatat sebagai xt, yt, wt, ht. [8].

YOLOv3 sangat cepat dan lebih akurat Dalam MAP diukur pada 0,5 IOU YOLOv3 setara dengan *Focal Loss* tetapi sekitar 4x lebih cepat. Selain itu dapat dengan mudah menukar antara kecepatan dan akurasi hanya dengan mengubah ukuran model, tanpa perlu pelatihan ulang. Jaringan ini membagi gambar menjadi daerah dan memprediksi kotak pembatas dan probabilitas untuk setiap wilayah. Kotak pembatas ini diberi bobot oleh probabilitas yang diprediksi dengan menerapkan jaringan saraf tunggal ke gambar.

2.4. Darknet

Darknet merupakan kerangka kerja jaringan neural open *source* yang ditulis dalam C dan CUDA yang cepat dan mudah dipasang, mendukung komputasi CPU dan juga GPU yang membuat lebih efisien serta mengevaluasi dengan lebih cepat[9]. Selain C dan CUDA darknet juga mendukung kompilasi pada OpenCV dengan mengubah Makefile dalam Darknet. Darknet yang digunakan dalam penelitian ini adalah Darknet53.conv.74.

2.5. Google Colaboratory

Google Colaboratory (alias Colab) adalah proyek yang memiliki tujuan untuk menyebarluaskan pendidikan dan penelitian pembelajaran mesin. *Colaboratory* menyediakan *runtime Python* 2 dan 3 yang telah dikonfigurasikan sebelumnya dengan pembelajaran mesin yang penting dan perpustakaan kecerdasan buatan, seperti *Tensorflow*, *Matplotlib*, dan *Keras*.
Layanan *Google* ini menyediakan *runtime* yang dipercepat GPU, juga sepenuhnya dikonfigurasikan dengan perangkat lunak yang sebelumnya diuraikan. Infrastruktur Kolaborasi *Google* di-host di *platform Google Cloud*[10].

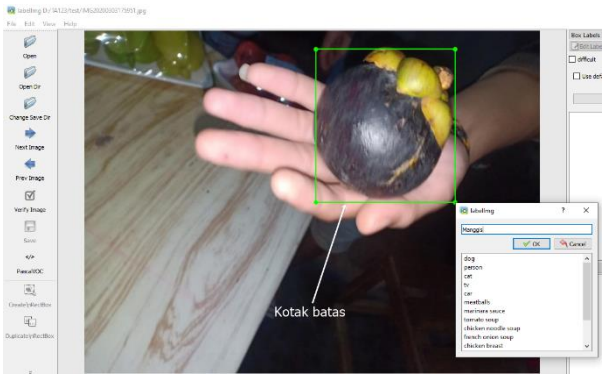
3. Hasil dan Pembahasan

3.1. Pengumpulan data

Pengumpulan data dalam membuat sistem deteksi 10 buah dilakukan dengan pengumpulan data sebagai tahap awal. Pengumpulan data dilakukan dengan bantuan kamera untuk mengelola citra dari 10 buah yaitu buah Manggis = 233 citra, Apel delicious = 228 citra, Belimbing = 232 citra, Jambu air = 230 citra, Kiwi = 235 citra, Pir = 239 citra, Nanas = 238 citra, Salak = 233 citra, Buah naga = 234 citra, Stroberi = 231 citra. Total data yang dikumpulkan mencapai 2.333 dengan ukuran 768x1024 untuk setiap citra.

3.2. Pemberian label

Proses pelatihan dan pengujian deteksi 10 buah dilakukan dengan tahap memberi label pada setiap data anotasi citra train dan citra test melalui labelImg yang dibuka menggunakan Anaconda prompt. Memberikan label pada citra berguna untuk pengenalan citra deteksi objek pada tahap konfigurasi *training* maupun tahap pengujian.



Gambar 3. Proses Label

Proses labeling pada citra seperti pada gambar 3, dilakukan dengan memberi kotak pada objek serta memberi nama pada kotak tersebut. Pemberian kotak dengan tipe YOLO akan menghasilkan format .txt yang berisi titik sudut sehingga membetuk kotak jangkar.

3.3. Kalkulasi Anchor

Memberikan kotak jangkar pada objek citra dari labelimg akan menghasilkan output dengan format .txt yang diperlukan untuk menghitung anchor. Perhitungan anchor dilakukan untuk menghitung rata-rata dari keseluruhan kotak jangkar pada setiap citra dengan proses clustering untuk membagi titik data menjadi kelompok yang akan dihubungkan dengan centroid shape untuk membentuk titik potong yang seimbang. Kalkulasi anchor dari kotak jangkar proses labeling digunakan untuk melengkapi source code anchor pada YOLOv3-fruits.cfg.

3.4. Makefile

Deteksi 10 buah dilanjutkan dengan membuka *library* openCV dan GPU pada google colaboratory dari makefile yang terdapat di darknet. OpenCV dan GPU akan di aktifkan dengan source code seperti terlihat pada gambar 4 untuk masuk pada pelatihan dan pengujian.

```
!sed -i 's/OPENCV=0/OPENCV=1/g' Makefile
!sed -i 's/GPU=0/GPU=1/g' Makefile

!make
```

Gambar 4. Membuka OpenCV dan GPU pada Makefile

Pelatihan dan pengujian menggunakan *library* OpenCV untuk menampilkan gambar dan GPU agar proses training berjalan lebih cepat dengan jumlah citra train sebanyak 2.133. GPU yang diaktifkan akan secara *default* menjalankan jaringan pada kartu grafis ke-0 pada sistem.

3.5. Proses Training

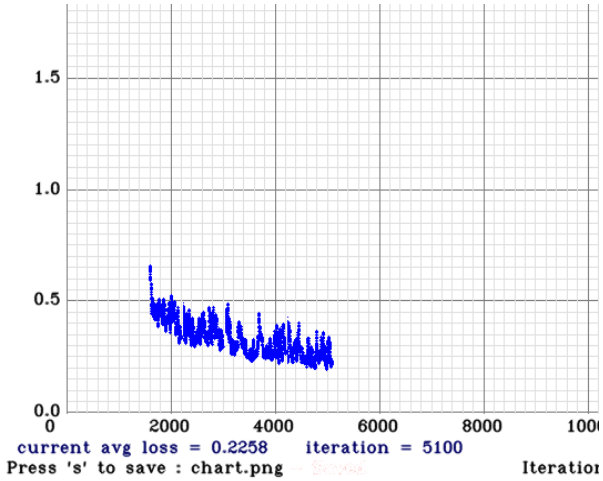
Proses awal training untuk data pelatihan sebanyak 2.133 citra dengan menjalankan source code yang ditunjukan gambar dibawah ini.

```
!./darknet detector train
data_ten2/fruits.data
cfg/yolov3-fruits.cfg
darknet53.conv.74 -dont_show
```

Gambar 5. Awal Konfigurasi Training

Pada proses *training* dilakukan dengan memanggil fruits.data yang berisi classes =10, train = data_ten2/trainten2.txt, valid = data_ten2/trainten2.txt, names = data_ten/buah.names dan backup/ untuk folder penyimpanan konfigurasi training. Selain fruits.data untuk konfigurasi *Training* juga memasukkan yolov3-fruits.cfg sebagai source code konfigurasi dengan batch=64 yang berarti data yang digunakan dalam sekali iterasi dan subdivisions=32 sebagai 32 citra yang akan di kirim ke GPU. Kerangka kerja jaringan saraf yang digunakan pada proses training menggunakan darknet.conv.74.

Konfigurasi *training* pada penelitian ini bisa dilakukan sampai iterasi 20.000, tetapi proses konfigurasi *training* dihentikan pada iterasi 5.100 karena jumlah loss sudah di bawah 0,5 . Proses konfigurasi *training* akan disimpan secara bertahap pada iterasi per-100 sebagai _last.weights dan per-1000 disimpan dengan _1000.weights, proses ini akan di simpan pada folder backup/.



Gambar 6. Grafik Loss pada iterasi 5100

Hasil pada iterasi 5.100 divisualisasikan pada gambar 6 merupakan grafik lanjutan chekpoint dari iterasi 1800. Iterasi tersebut menghasilkan jumlah loss = 0.2258 merupakan hasil *training* model dengan jumlah *loss* terkecil. Covolution layer pada proses training untuk penelitian ini dilakukan dengan filter 32,64,128,256,512 dan 1024, reize input citra 416x416. –dont_show sebagai *flag* yang digunakan untuk tidak menampilkan hasil augmentasi data agar menghindari kesalah selama pelatihan untuk iterasi tinggi.

3.6. Pengujian

Pengujian yang dilakukan dari iterasi 5.100 dengan *loss* = 0.2258 di terapkan pada citra *test* sebanyak 200 citra dari total semua kelas untuk menuju tahap pengujian.

```
!./darknet detector test
data_ten2/fruits.data
cfg/yolov3-fruits.cfg
backup/yolov3-fruits_last.weights
IMGyolo/testyolo/IMG20200303175951.jpg -
i 0 -thresh 0.5
```

Gambar 7. Source code Proses Test

Proses pengujian dari hasil konfigurasi *training* pada iterasi 5.100 di uji dengan melakukan *detector test* seperti di perlihatkan gambar 7, proses *test* ini dilakukan dengan menjalankan *source code detector test* dan

memanggil fruits.data, yolov3-fruits.cfg sebagai *source code* konfigurasi, yolov3-fruits_last.weights sebagai konfigurasi training terakhir yang disimpan secara *checkpoint* di folder backup dan juga memasukkan IMG20200303175951.jpg sebagai input gambar yang akan di deteksi. Output yang dihasilkan dari detector test menampilkan proses convolution layer dan presentase prediksi objek dari citra

```
def imShow(path):
import cv2
import matplotlib.pyplot as plt
%matplotlib inline
  image = cv2.imread(path)
  height, width = image.shape[:2]
resized_image = cv2.resize(image,(3*width
, 3*height), interpolation = cv2.INTER_CU
BIC)
  fig = plt.gcf()
  fig.set_size_inches(18, 10)
  plt.axis("off")
  plt.imshow(cv2.cvtColor(resized_image,
  cv2.COLOR_BGR2RGB))
  plt.show()
```

Gambar 8. Source code Menampilkan Hasil Pengujian

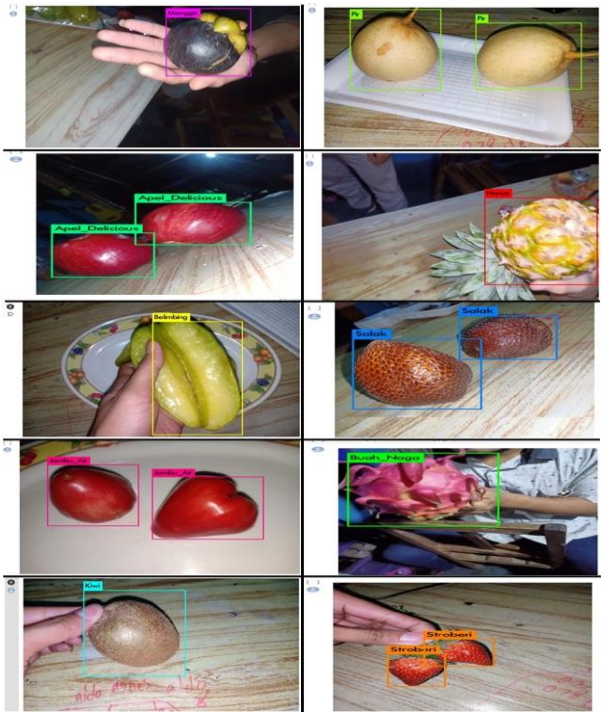
```
def upload():
  from google.colab import files
  uploaded = files.upload()
  for name, data in uploaded.items():
    with open(name, 'wb') as f:
      f.write(data)
      print ('saved file', name)
def download(path):
  from google.colab import files
  files.download(path)

imShow('predictions.jpg')
```

Gambar 9. Source code menampilkan Hasil pengujian

Hasil pengujian akan di tampilkan melalui source code yang ada pada gambar 8 dan gambar 9. Menampilkan gambar dengan *library* pyhton sehingga tampilan akan secara default untuk tipe gambar dan mengoptimalkan sumbu dan objek gambar untuk di tampilkan sesuai dengan *input* citra yang di deteksi.

3.7. Pengujian Pertama

Tahap pengujian pertama dilakukan dengan Citra 10 buah pada data test dari setiap kelas. Citra yang dideteksi akan memunculkan kotak jangkar dengan nama label pada objek di citra yang masukkan pada detector test untuk masing-masing kelas diuraikan di bawah ini :



Gambar 10. Hasil Pengujian Pertama

Hasil dari proses pelatihan dan pengujian pada 10 kelas berhasil di deteksi pada pengujian pertama seperti yang terlihat pada gambar 10. Pengujian pertama ini dilakukan dengan memasukkan citra 10 buah dari data *test* dengan menampilkan kotak jangkar dan nama label.

Tabel 1. Akurasi Hasil Pengujian Pertama

Nama Buah	Warna Label	Akurasi
Manggis	Ungu	Manggis: 96%
Apel delicious	Hijau	Apel_Delicious: 98%
Belimbing	Kuning	Belimbing: 93%
Jambu air	Merah muda	Jambu_Air: 100%
Kiwi	Biru	Kiwi: 100%
Pir	Hijau muda	Pir: 100%
Nanas	Merah	Nanas: 90%
Salak	Biru tua	Salak: 99%
Buah naga	Hijau	Buah_Naga: 98%
Stroberi	Orange	Stroberi : 99%
Rata-rata		97,3%

Presentase dari hasil pengujian citra 10 buah dari setiap kelas dengan rata-rata lebih dari 97,3% yang terlihat pada tabel 1. Akurasi deteksi buah dengan citra rata-rata 90% juga sudah banyak dilakukan dengan metode berbeda seperti metode Maks-RCNN pada deteksi buah untuk robot pemanen stroberi di lingkungan non-struktural dengan akurasi 95,78% dari 100 citra yang diuji[11].

3.8. Pengujian Kedua

Pengujian kedua juga di lakukan dengan memasukkan gambar diluar data *test* yang tidak melewati proses

pemberian label untuk melihat apakah sistem deteksi 10 buah juga bekerja dengan baik atau tidak.

Gambar 11. Hasil Pengujian Kedua

Proses pengujian kedua dilakukan dengan memasukkan citra yang di ambil secara acak di google *imgaes* dengan posisi objek yang terlihat jelas. Pada gambar 11 pengujian kedua dilakukan dengan memasukkan citra pada *source code detector test* , hasil dari pengujian kedua juga masih bekerja dengan baik. Pengujian kedua juga menampilkan deteksi pada objek dari 10 kelas.

Tabel 2. Hasil Akurasi Pengujian Kedua

Nama Buah	Warna Label	Akurasi
Manggis	Ungu	Manggis: 94%
Apel delicious	Hijau	Apel_Delicious: 75%
Belimbing	Kuning	Belimbing: 77%
Jambu air	Merah muda	Jambu_Air: 87%
Kiwi	Biru	Kiwi: 86%
Pir	Hijau muda	Pir: 76%
Nanas	Merah	Nanas: 89%
Salak	Biru tua	Salak: 99%
Buah naga	Hijau	Buah_Naga: 72%
Stroberi	Orange	Stroberi: 86%
Rata-rata		84,1%

Hasil dari pengujian kedua ditunjukkan pada tabel 2 dengan akurasi rata-rata 84,1% untuk semua kelas. Citra yang diambil dari google images secara acak dengan kategori minimal ukuran citra 300x300 serta dengan posisi objek yang terlihat jelas.

4. Kesimpulan

Setelah dilakukan pengumpulan data, pelatihan dan pengujian terhadap perancangan sistem, penerapan algoritma CNN mempermudah pembelajaran mesin dan berhasil mendeteksi objek dari dua tahap pengujian dengan akurasi rata-rata 70-90% dari setiap citra. Penggunaan *google colaboratory* yang menyediakan GPU hingga 12GB mempermudah dan mempercepat perancangan sistem deteksi 10 buah. Keakuratan untuk mendeteksi citra dipengaruhi pada proses awal training dengan citra yang masuk pada data test seperti pengujian pertama dan ukuran citra dengan minimal 300x300 seperti pengujian kedua. Walaupun sudah melakukan praktik terbaik untuk penulisan ini, masih harus mengembangkan jumlah data dari segi posisi objek pada citra dan proses training untuk iterasi yang lebih besar sehingga tidak membuat akurasi deteksi tumpul.

Daftar Rujukan

[1] Y Sahertian, J., & Sanjaya, A. S. A. (2017). Deteksi Buah Pada Pohon Menggunakan Metode Svm Dan Fitur Tekstur. SEMNASTEKNOMEDIA ONLINE, 5(1), 4-3.

[2] Bulanon, D. M., & Kataoka, T. (2010). Fruit detection system and an end effector for robotic harvesting of Fuji apples. Agricultural Engineering International: CIGR Journal, 12(1).

[3] Ammar, A., Koubaa, A., Ahmed, M., & Saad, A. (2019). Aerial Images Processing for Car Detection using Convolutional Neural Networks: Comparison between Faster R-CNN and YoloV3. *arXiv preprint arXiv:1910.07234*.

[4] Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., & Savarese, S. (2019). Generalized intersection over union: A metric and a loss for bounding box regression. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 658-666).

[5] Kadiyala, A., & Kumar, A. (2017, October 31). Apllications of Python to evaluate environmental data science. Environmetal Progress & Sustainable Energy, 36, 1580–1586.

[6] I Wayan Suartika E. P, Arya Yudhi Wijaya, dan Rully Soelaiman, Klasifikasi Citra Menggunakan Convolutional Neural Network (Cnn) pada Caltech 101, JURNAL TEKNIK ITS Vol. 5, No. 1, (2016) ISSN: 2337-3539 .

[7] N. Sofia,2018. Convolutional Neural Network, [Online] (Update 9 Juni 2018) Tersedia di : https://medium.com/@nadhifasofia/1-convolutional-neural-network-convolutional-neural-network-merupakan-salah-satu-metode-machine-28189e17335b/. [Accessed 6 Maret 2020].

[8] Weicong, D., Longxu, J., Guoning, L., & Zhiqiang, Z. (2018). Real-time airplane detection algorithm in remote-sensing images based on improved YOLOv3. Opto-Electronic Engineering, 45(12), 180350.

[9] J. Redmon,2013-2018. Darknet : YOLO:Real-Time Object Detection, [Online] (Update 2018) https://pjreddie.com/darknet/yolo/. [Accessed 7 maret 2020].

[10] T. Carneiro, R. V. Medeiros Da NóBrega, T. Nepomuceno, G. Bian, V. H. C. De Albuquerque and P. P. R. Filho, "Performance Analysis of Google Colaboratory as a Tool for Accelerating Deep Learning Applications," in IEEE Access, vol. 6, pp. 61677-61685, 2018.

[11] Yu, Y., Zhang, K., Yang, L., & Zhang, D. (2019). Fruit detection for strawberry harvesting robot in non-structural environment based on Mask-RCNN. Computers and Electronics in Agriculture, 163, 104846.