

Deteksi Objek Kereta Api menggunakan Metode *Faster R-CNN* dengan Arsitektur VGG 16

JASMAN PARDEDE, HENDRI HARDIANSAH

Program Studi Informatika, Institut Teknologi Nasional Bandung
Email : jasman@itenas.ac.id

Received 1 November 2021 | *Revised* 13 Februari 2022 | *Accepted* 3 April 2022

ABSTRAK

Kereta merupakan sebuah alat transportasi umum yang sering digunakan oleh masyarakat untuk berpergian dari kota asal ke kota tujuan. Mereka membutuhkan akan sarana transportasi umum untuk mempermudah aktifitas mereka. Namun kecelakaan di persimpangan jalan raya yang terlintasi oleh kereta api memiliki angka yang cukup besar akibat kelalaian dari petugas untuk menutup palang pintu kereta api. Maka dari itu penelitian ini dibuat agar mengetahui keberadaan kereta api berdasarkan jarak dan tingkat cahayanya dari siang sampai malam hari. Sistem dibangun menggunakan metode Faster RCNN dengan model arsitektur VGG16 untuk mengetahui keberadaan objek kereta api antara lokomotif dan gerbong berdasarkan tingkat cahaya dan jarak terhadap objek. Setelah dilakukan pengujian dengan jarak paling dekat ± 2 meter sampai ± 250 meter, diperoleh rata-rata akurasi untuk lokomotif sebesar 79,09%, dan akurasi untuk gerbong sebesar 97,05%. Sistem memperoleh keakurasian deteksi terhadap objek rata-rata akurasi deteksi objek lokomotif sebesar 86,40%, dan rata-rata akurasi deteksi objek gerbong sebesar 97,23%.

Kata kunci: Deteksi Objek, Faster RCNN, VGG, Kereta Api, Jarak, Lux

ABSTRACT

Railway is a public transportation that is often used by the public to travel from the home town to the destination city. They need public transportation to facilitate their activities. But accidents at the intersection of the highway crossed by the train has a considerable number due to the negligence of the officer to close the railway stopbars. Therefore, this study was made to know the existence of trains based on their distance and light level from day to night. The system was built using the Faster RCNN method with the VGG16 architectural model to determine the existence of railway objects between locomotives and carriages based on the level of light and distance to the object. After testing with the closest distance of ± 2 meters to ± 250 meters, obtained an average accuracy for locomotives of 79.09%, and accuracy for carriages of 97.05%. The system obtained accuracy of detection of objects with an average detection accuracy of locomotive objects of 86.40%, and an average detection accuracy of car objects of 97.23%.

Keywords: Object Detection, Faster RCNN, VGG, Railway, Distance, Lux

1. PENDAHULUAN

Kereta api adalah sebuah alat transportasi yang berjalan di atas rel yang terbuat dari serangkaian gerbong yang ditarik oleh lokomotif atau bagian paling depan dari rangkaian kereta api yang terdapat mesin penggerak untuk menarik gerbong **(KBBI, 2020)**. Sehingga mempermudah untuk manusia untuk melakukan perjalanan jarak jauh dengan lebih cepat dan lebih efektif.

Kereta api terkadang melewati suatu persimpangan jalan padat kendaraan yang memungkinkan kecelakaan terjadi ketika melewati persimpangan jalan tersebut. PT Kereta Api Indonesia (KAI) mencatat, selama tahun 2019 telah terjadi 260 kali kecelakaan yang mengakibatkan 76 nyawa melayang pada perlintasan kereta api (KA) sebidang. Lalu tahun sebelumnya, terjadi 395 kecelakaan yang menyebabkan 245 orang luka berat sampai meninggal dunia. Total selama dua tahun telah terjadi 655 kecelakaan **(Tempo, 2019)**.

Palang pintu perlintasan kereta api merupakan salah satu pengamanan utama dalam mencegah terjadinya kecelakaan kereta api saat melewati persimpangan jalan raya. Kecelakaan terbesar adalah telatnya ditutup palang pintu kereta api yang mengakibatkan mobil pengangkut BBM milik PT Pertamina menerobos masuk dan terjadi kecelakaan antara mobil pengangkut BBM dan kereta api yang menyebabkan penutupan jalan di Bintaro **(Liputan 6, 2013)**.

Deteksi objek diharapkan mampu membantu meminimalisir angka kecelakaan yang terjadi di persimpangan jalan akibat kelalaian dari petugas palang pintu, dimana deteksi objek memiliki suatu tugas yang sangat penting dalam sebuah pengolahan citra *digital*. Deteksi objek berhubungan dengan penentuan identitas sebuah objek yang sedang diamati dalam sebuah citra dari sekumpulan tanda yang sudah diketahui **(Khurana, dkk, 2013)**.

Faster R-CNN merupakan metode deteksi objek pengembangan dari CNN dan *Faster R-CNN* terdiri dari dua modul, modul pertama adalah *deep fully convolutional network* yang menjadi *region proposer*, dan modul kedua adalah detektor *Fast R-CNN* yang menggunakan *region proposed*. Seluruh sistem adalah satu jaringan terpadu untuk deteksi objek. Menggunakan terminologi jaringan neural yang baru-baru ini populer dengan mekanisme 'attention', modul RPN memberi tahu modul *Fast R-CNN* kemana harus mencari, kami memperkenalkan desain dan properti jaringan untuk *region proposer* **(K. Srijakkot, 2019)**.

Convolutional Neural Network (CNN) dirancang untuk memproses suatu data yang ada dalam bentuk banyak array, contohnya gambar warna yang terdiri dari 2D array yang mengandung piksel dalam tiga macam warna yaitu *Red*, *Green*, dan *Blue*. Ada berbagai macam bentuk CNN adalah 1D untuk sinyal dan urutan biasanya digunakan untuk bahasa, 2D untuk gambar atau suara; dan 3D untuk video atau gambar volumetric **(Y. Lecun, 2015)**.

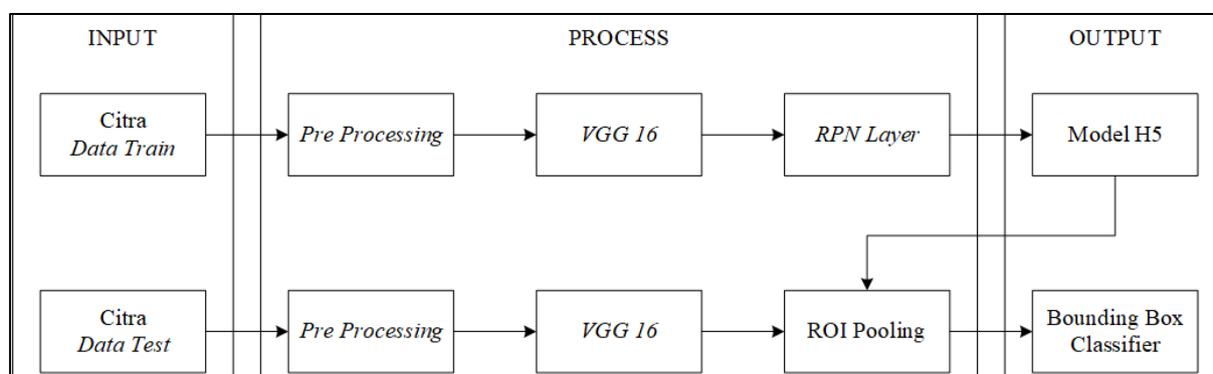
Perkembangan Arsitektur CNN melaju dipengaruhi kompetisi ILSVRC (*ImageNet Large Scale Visual Recognition Competition*) yang diselenggarakan setiap tahunnya oleh ImageNet yang menjadi standar dalam benchmark object Recognition. Pada kompetisi tersebut dilakukan evaluasi terhadap tingkat error dari suatu algoritma CNN dalam melakukan deteksi objek dan klasifikasi citra dalam skala yang besar. Pemenang kompetisi pada tahun 2012 dengan arsitektur CNN bernama AlexNet memiliki tingkat error sebesar 16,4%. Kemudian pemenang tahun 2013 dengan kode Clarifai dapat mengurangi tingkat error hingga sebesar 11,7%. Pada tahun 2014 terdapat 2 pemenang yang memiliki tingkat error dibawah 10% yaitu arsitektur VGG dengan jumlah penggunaan layer sebanyak 19 layer berhasil menurunkan nilai error hingga 7,3% dan arsitektur GoogLeNet dengan 22 layer menurunkan kembali nilai error hingga

6,7%. Pemenang kompetisi pada tahun 2015 dengan arsitektur ResNet dari Microsoft meningkatkan penggunaan layer hingga 152 layer yang berhasil menurunkan nilai error hingga 3,57%. Arsitektur DenseNet tingkat error sebesar 5,17%. Meskipun mendapatkan nilai error 3,57% akan tetapi layer yang digunakan pada arsitektur ResNet sebanyak 152 layer, maka operasi komputasi yang diperlukan meningkat lebih tinggi dari sebelumnya. Sehingga arsitektur GoogLeNet dapat menjadi arsitektur pilihan untuk melakukan pengenalan objek dengan akurasi tinggi akan tetapi tidak membutuhkan operasi komputasi yang terlalu tinggi seperti arsitektur ResNet **(Russakovsky, dkk. 2015)**.

Pada penelitian sebelumnya telah dilakukan penelitian menggunakan metode *Faster RCNN* dengan arsitektur VGG16 pada objek manusia dengan jarak 3 meter hingga 8 meter dengan rata rata akurasi sebesar 0.8613 untuk akurasi deteksi objek manusia **(K. Srijakkot, 2019)**. Metode *deep learning* yang digunakan pada penelitian yang lain sudah menunjukan performa yang baik dalam melakukan deteksi. Maka dari itu pada penelitian ini mencoba melakukan penelitian akan performa akurasi dari metode *Faster RCNN* terkait perbedaan tingkatan intensitas cahaya yang gunakan untuk pendeteksian dan juga jarak dari terdekat hingga terjauh akan deteksi objek itu sendiri.

2. METODE PENELITIAN

Pada penelitian kali ini, input citra diolah terlebih dahulu untuk mendapatkan file *.xml* yang didapat dari pelabelan menggunakan aplikasi *Imglab*. *Data train* terdiri dari citra itu sendiri dan juga file *.xml* yang nantinya akan digunakan untuk ke tahap pre-processing dan tahapan training untuk mendapatkan model *.h5* dengan arsitektur VGG16 untuk membandingkan hasil training dan testing, proses tersebut dapat dilihat pada Gambar 1.



Gambar 1. Blok Diagram Sistem

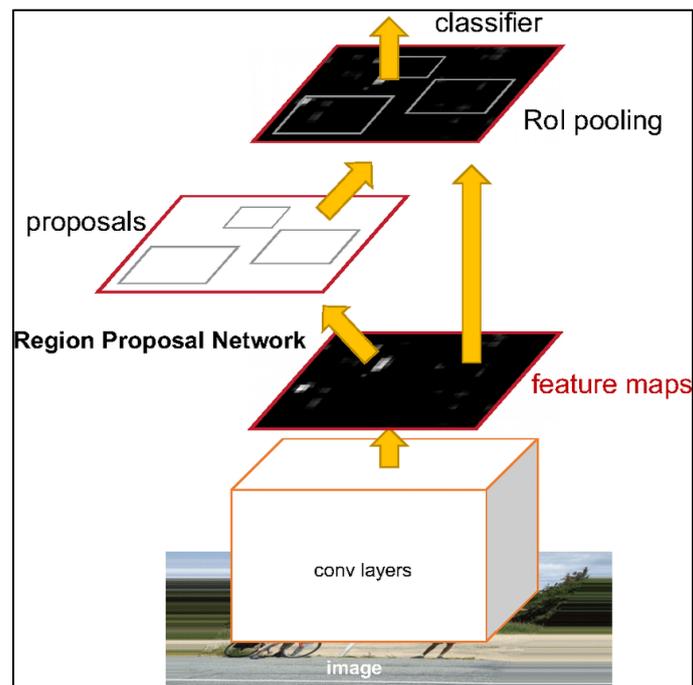
Pada proses deteksi citra lokomotif dan citra gerbong, citra input akan di deteksi dengan menggunakan *Pre-trained Model VGG16* yang sudah terlatih sebelumnya lalu *region proposal network* berperan sebagai *proposer* letak objek dengan menghitung antara irisan dan gabungan antara *anchor* dan *ground truth box* kemudian di *fully connected layer* di *RoI Pooling* untuk digunakan sebagai *external proposer* pada tahapan pengujian atau saat menggunakan data test untuk melakukan *bounding box* dan klasifikasi objek selanjutnya.

2.1. Deteksi Objek

Deteksi objek adalah teknik visi komputer untuk menemukan lokasi objek dalam gambar atau video. Algoritme deteksi objek biasanya memanfaatkan *machine learning* atau *deep learning* untuk menghasilkan hasil yang bermakna. Saat manusia melihat gambar atau video, kita dapat mengenali dan menemukan objek yang menarik dalam sekejap. Tujuan dari deteksi objek adalah untuk mereplikasi kecerdasan ini menggunakan komputer **(Mathwork, 2020)**.

2.2. Faster R CNN

Faster-RCNN adalah sistem deteksi objek terdiri dari dua modul. Modul pertama adalah *Regional Convolutional Neural Network*, dan modul kedua adalah detektor *Fast Region-based Convolutional Neural Network* yang menggunakan wilayah yang diusulkan. *Faster RCNN* ini terdiri dari 5 bagian utama: *deep convolutional network*, *Region Proposal Network*, *ROI Pooling Layer* dan *Fully Connected Layer*, *bounding box regressor*, dan *classifier*. Melalui *region proposal networks* dan *deep fully convolutional*, banyak objek yang diajukan untuk menjadi *proposer*, lalu dinormalisasi melalui *ROI Pooling Layer*. Kemudian, *Fully Connected layer* mengekstrak fitur yang baik untuk melakukan klasifikasi dan regresi tahapan tersebut dapat diilustrasikan seperti pada Gambar 2. Sebagian besar *proposer* tidak selalu sesuai dengan objek target dan regulator mencoba untuk menemukan yang paling sesuai. Namun, klasifikasi dilakukan pada proposal dan bukan pada yang paling sesuai. Dalam penggunaan praktis, ukuran objek bermacam-macam: misalnya mobil, manusia, dan plat nomor. Jika ukuran objek target bervariasi, banyak proposal yang berukuran lebih besar sehingga berisi objek kecil maupun objek besar. Dalam kasus ini, pengklasifikasi harus menangani tidak hanya variasi dari wilayah objek itu sendiri tetapi juga variasi dari wilayah luar objek. Ini tidak efisien untuk pengklasifikasi dan regressor dan menghasilkan positif palsu dan negatif palsu dalam pengujian (M. Roh, 2017).



Gambar 2. Metode Faster RCNN (Shaoqing Ren, 2016)

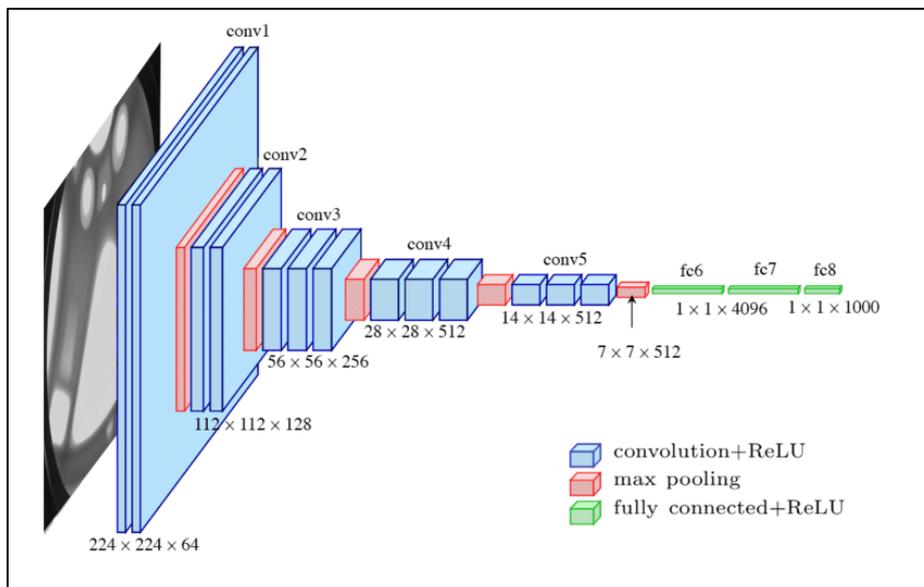
2.3. VGG16

VGGNet dikembangkan oleh Visual Geometry Group di Oxford University. Model ini menjuarai kontes ILSVRC sebagai 1st *Runner-up Image Classification* dan pemenang *Image Localization* pada tahun 2014. Model VGGNet dirancang mengurangi kernel ukuran besar pada AlexNet 1x1 dan 5x5 diganti dengan beberapa kernel 3x3 dengan 1 *stride* yang berguna untuk mengekstrak fitur kompleks (Simonyan & Zisserman, 2015). Kemudian menggunakan max pooling 3x3 dengan 2 strides. Model ini sudah terlatih pada dataset besar gambar ImageNet dengan 1000 kelas yang berbeda. Berikut konfigurasi arsitektur VGG dijelaskan pada Gambar 3.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 conv1-256	conv3-256 conv3-256 conv3-256	conv3-256 conv3-256 conv3-256 conv3-256
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Gambar 3. Konfigurasi Arsitektur VGGNet
Sumber : (Simonyan & Zisserman, 2015)

Pada penelitian ini digunakan arsitektur VGG16 yang di dalamnya terdapat proses 5 blok konvolusi yang terdiri dari operasi konvolusi 3x3 menggunakan 1 *stride* dengan jenis padding same/zero padding lalu di aktivasi menggunakan ReLU. Kemudian setiap selesai operasi blok konvolusional di reduksi menggunakan operasi maxpooling 2x2 dengan 2 *stride* dan diakhiri dengan 2 dense layer sebanyak 4096 node/neuron. Model VGG16 diilustrasikan pada Gambar 4.

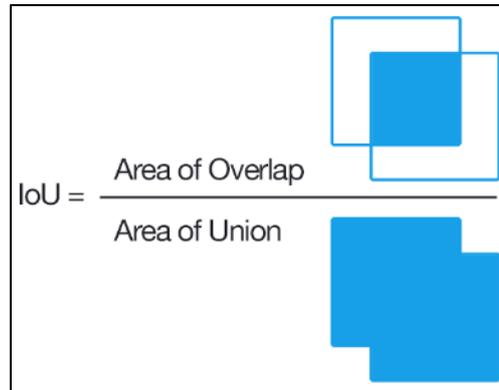


Gambar 4. Arsitektur VGG 16
(Sumber : researchgate.net)

2.4. Intersection over Union

Intersection over Union (IoU) merupakan metode evaluasi untuk mengukur akurasi deteksi objek terhadap suatu dataset. IoU membutuhkan 2 area yang pertama intersect adalah area

ground-truth bounding box yang merupakan *bounding box* aktual dan area yang dideteksi dari model yang dibangun (Rosebrock, A. 2016). Ilustrasi persamaan IoU dapat dilihat pada Gambar 4.



Gambar 4. Intersection over Union (Sumber : PYImageSearch)

2.5. Region Proposal Network

Region Proposal Network digunakan untuk menghasilkan proposal dari *input feature*. *Feature map* berdimensi 512 diekstraksi oleh jaringan konvolusional dasar (ConvNet) yang mengambil seluruh gambar sebagai input. *Feature map input* digunakan sebagai *input sliding window 3x3*, setiap *sliding window* dipetakan ke vektor fitur 512 dimensi. Vektor fitur ini dimasukkan ke dalam dua lapisan yang terhubung ke *fully-connected layers* dilanjutkan *box-regression layer* dan *box-classification*. Dalam, konsep baru *Anchor* diusulkan. Sebuah *anchor* dipusatkan pada jendela geser 3×3 , setiap *sliding window* yang memiliki sembilan *anchor* dengan kombinasi tiga skala [128 2, 256 2, 512 2] dan tiga rasio aspek [1: 1, 1: 2, 2: 1]. Lapisan klasifikasi kotak memeriksa apakah sebuah jangkar positif atau tidak dan lapisan kotak-regresi mengeluarkan koordinat dari kotak pembatas. Untuk melatih RPN, *loss function* didefinisikan sebagai berikut:

$$L = \frac{1}{N_{ds}} \sum L_{ds}(p, p^*) + \lambda \frac{1}{N_{reg}} \sum p^* L_{reg}(t, t^*) \quad (1)$$

Dimana indeks dari *Anchor* dalam *mini batch*,

p adalah kemungkinan proposal menjadi objek,

p^* adalah label sebenarnya dari proposal (ketika proposal adalah objek, $p^* = 1$, jika tidak $p^* = 0$),

t dan t^* adalah koordinat dari *bounding box regression* dan *ground-truth* masing-masing,

N_{cls} dan N_{reg} adalah dua parameter normalisasi,

L_{cls} adalah kerugian klasifikasi, itu adalah kerugian log selama dua kelas (objek dan bukan keberatan).

L_{reg} adalah kerugian regresi, istilah $p^* L_{reg}$ berarti kerugian regresi diaktifkan hanya untuk *anchor* positif

$$t_x = \frac{(x - x_a)}{w_a} \quad t_y = \frac{(y - y_a)}{h_a} \quad (2)$$

$$t_w = \log\left(\frac{w}{w_a}\right) \quad t_h = \log\left(\frac{h}{h_a}\right) \quad (3)$$

$$t_x^* = \frac{(x^* - x_a)}{w_a} \quad t_y^* = \frac{(y^* - y_a)}{h_a} \quad (4)$$

$$t_w^* = \log\left(\frac{w^*}{w_a}\right) \quad t_h^* = \log\left(\frac{h^*}{h_a}\right) \quad (5)$$

Untuk regresi kotak pembatas, parameter dari empat koordinat ditunjukkan pada Persamaan. (2). Dimana x dan y mewakili koordinat dari pusat kotak, w dan h mewakili berat dan tinggi dari kotak pembatas masing-masing, x , x_a , dan x^* untuk kotak yang diprioritaskan, kotak jangkar, dan *ground-truth box* masing-masing (demikian juga untuk y , w , h). Ini bisa berupa pemikiran *bounding box regression* dari kotak *anchor* ke kotak kebenaran dasar terdekat (**H. Huang, 2019**).

2.6. Region Of Interest Pooling Layer

Lapisan *pooling RoI* menggunakan *max pooling* untuk mengkonversi fitur di dalam *Region of Interest* ke dalam *feature maps* dengan tingkat spasial tetap $H \times W$ (misalnya, 7×7), di mana H dan W adalah hiper-parameter lapisan sistem dari RoI tertentu. RoI adalah jendela persegi sistem ke dalam *feature maps convolution*. Setiap RoI didefinisikan oleh empat tuple (r, c, h, w) yang menentukan sudut kiri atasnya (r, c) dan tinggi dan lebarnya (h, w). Lapisan *pooling RoI* menggunakan *max pooling* untuk mengonversi fitur di dalam wilayah yang valid yang menarik menjadi peta fitur kecil dengan tingkat spasial tetap $H \times W$ (misalnya, 7×7), di mana H dan W adalah hiper-parameter lapisan yang sistem dari RoI tertentu. Dalam makalah ini, RoI adalah jendela persegi sistem ke dalam peta fitur *convolution*. Setiap RoI didefinisikan oleh empat tuple (r, c, h, w) yang menentukan sudut kiri atasnya (r, c) dan tinggi dan lebarnya (h, w) (**Girshick, R. 2015**).

2.7. Bounding Box Regresor

Region Based CNN dalam langkah ini, setiap ROI diklasifikasikan dan kotaknya diregressi menggunakan metode R-CNN cepat yang diusulkan dalam. Pertama, fitur dari CNN dipotong oleh setiap ROI, dan hanya fitur yang dipotong yang dikumpulkan. Kemudian, fitur yang dikumpulkan melewati beberapa lapisan yang terhubung sepenuhnya dan tersembunyi. Kemudian, untuk setiap ROI, skor probabilitas kelas dan kotak pembatas regresi dihitung dari lapisan yang sepenuhnya terhubung. Akhirnya, kami mengumpulkan kotak pembatas dengan skor. Selain itu, untuk setiap kelas, kami menggabungkan kotak pembatas yang dikumpulkan menggunakan penekanan non-maksimum untuk menghindari deteksi duplikat (**Chungkeun Lee, 2016**).

2.8. Classifier

Classifier merupakan proses yang digunakan untuk mengklasifikasikan RoI yang sudah diidentifikasi pada RPN ke dalam kelas atau target yang sesuai. Teknik yang digunakan pada tahap ini adalah CNN.

2.3. Kereta Api

Menurut kamus besar Bahasa Indonesia arti kata kereta api adalah yang terdiri atas rangkaian gerbong (kereta) yang ditarik oleh lokomotif, dijalankan dengan tenaga uap (atau listrik), berjalan di atas rel (rentangan baja dan sebagainya), digunakan untuk kendaraan umum (**KBBI, 2020**).



Gambar 5. Kereta Api
(Sumber : jabarnews)

3. HASIL DAN PEMBAHASAN

3.1. Dataset

Pada dataset citra-citra lokomotif dan gerbong yang didapat dari pengambilan citra secara langsung yang berlokasi di JPO (Jembatan Penyebrangan Orang) di jalan Kebon Jukut dan di Jembatan jalan Pasir Kaliki Bandung menggunakan kamera dari iPhone 6s dengan kamera utama 12 MP dan bukaan lensa $f/2.2$, menggunakan mode *burst shot* dan *square mode*. Hasil dari pengambilan citra dapat dilihat di Gambar 6.



Gambar 6. Hasil Pengambilan Citra Dataset

Dataset kemudian di bagi menjadi data *training* dan *testing* sebanyak 70% citra *training*, dan 30% untuk citra *testing*. Setelah dataset dibagi menjadi citra training dan citra testing, citra training diberi label menggunakan labelImg, menghasilkan file *.xml untuk pelabelan citra training citra untuk digunakan pada proses training yang berguna sebagai untuk *ground truth box* untuk menghasilkan *anchor* pada proses training dan di konfolusi. *Training* sitem dalam mendeteksi lokomotif dan gerbong yang terdapat pada citra dengan menggunakan metode convolution neural network dengan arsitektur VGG16. Hasil pelabelan menggunakan labelImg dapat dilihat pada Gambar 7.

```

<?xml version="1.0"?>
- <annotation>
  <folder>train</folder>
  <filename>00014.jpg</filename>
  <path>C:\Users\Hendri Hardiansah\Downloads\dataset\train\00014.jpg</path>
  - <source>
    <database>Unknown</database>
  </source>
  - <size>
    <width>2048</width>
    <height>2048</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  - <object>
    <name>Lokomotif</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    - <bndbox>
      <xmin>1009</xmin>
      <ymin>795</ymin>
      <xmax>1187</xmax>
      <ymax>1120</ymax>
    </bndbox>
  </object>
</annotation>

```

Gambar 7. Hasil Pelabelan Citra menggunakan LabelImg

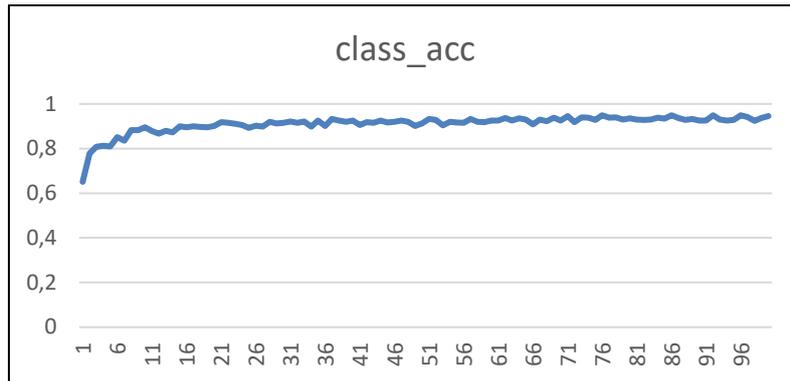
Implementasi ini dilakukan dengan dua mesin. Penelitian ini melakukan proses training menggunakan google collaboratory dengan spesifikasi GPU Nvidia Tesla P100 16GB, RAM 27GB, dan storage 68GB. Proses testing menggunakan processor Intel Core i5-1135G7 @2.4GHz, RAM 8GB, storage 512GB. Tensorflow dan keras sebagai penunjang dari penelitian ini.

Tabel 2. Dataset Deteksi Objek Kereta Api

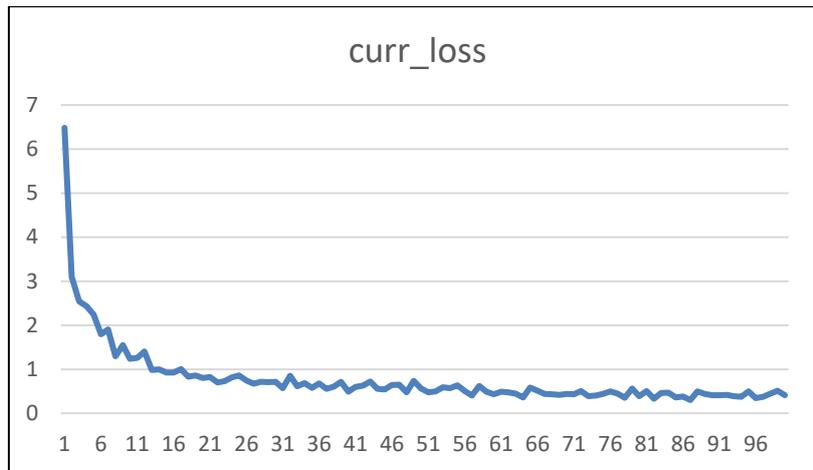
	Training		Testing	
	Lokomotif	Gerbong	Lokomotif	Gerbong
Siang	223	214	111	80
Sore	138	315	52	123
Malam	49	309	20	158
Total	1248		544	

3.2. Training

Penelitian ini pada proses training dibangun menggunakan google collaboratory dengan 100 epoch, *optimizer* Adam, *learning rates* 0,0005 dan 256 *batch size* dalam training dengan menggunakan proses arsitektur VGG16 yang mendapatkan model "*.h5". Memanggil citra training kemudian citra tersebut dibagi dengan 256 dari *batch size* sehingga 1 epoch melewati proses 256 citra kemudian diselesaikan sampai 100 epoch dimana dengan arsitektur VGG16 mendapatkan nilai terendah *total loss* yang terbagi menjadi *loss* pada training 0.0001 dan untuk nilai terendah *total loss function* 0,3. Untuk nilai tertinggi dari training accuracy 1 dan untuk nilai tertinggi dari 0.9455. Training dilakukan selama 4 jam. Model yang sudah didapatkan diintegrasikan dengan *website* menggunakan framework flask untuk dilakukan pengujian terhadap citra lokomotif dan gerbong untuk mengetahui tingkat akurasi yang didapatkan dari model tersebut. Pada Gambar 8 untuk hasil akurasi dan Gambar 9 hasil loss seperti hasil dari training dari arsitektur VGG16.



Gambar 8. Hasil Nilai Akurasi Klasifikasi



Gambar 9. Hasil Nilai Loss Klasifikasi

3.3. Pengujian

Tabel 3. Tabel Pengujian Sistem

No. Pengujian	Skenario Pengujian	Hasil
1	Melakukan pengujian deteksi objek pada citra lokomotif dan gerbong pada objek kereta api	Mendapatkan nilai <i>Accuracy</i> , <i>Precession</i> , <i>Recall</i> , dan <i>F1 Score</i>
2	Mengetahui <i>accuracy</i> klasifikasi tingkatan cahaya yang berbeda dan jarak yang fluktuatif pada citra lokomotif dan gerbong pada objek kereta api	Mendapatkan nilai <i>Accuracy</i> klasifikasi pada objek kereta api

Pengujian dibagi menjadi dua pengujian yaitu pengujian kinerja sistem dan pengujian deteksi objek antara lokomotif dan gerbong. Jarak yang didapat berdasarkan koordinat tempat berdiri pengambilan citra kepada koordinat objek kereta api menggunakan google maps dan pengurangan jarak dilakukan dengan mencari kecepatan rata rata kereta api dengan cara seperti berikut ini :

$$Kecepatan\ Rata\ Rata = \frac{Jarak\ Citra\ Awal}{Citra\ Terdekat[n]} \quad (6)$$

$$Jarak\ Objek = Jarak\ Citra\ Sebelumnya - Kecepatan\ Rata\ Rata \quad (7)$$



Gambar 10. Penentuan Jarak Citra Awal Menggunakan *Google Maps*

3.3.1 Pengujian Deteksi Kereta Api menggunakan Faster RCNN

Untuk mengukur kerja sistem dilakukan pengujian dengan menghitung nilai akurasi, presisi, recall dan F1 score. Persamaan nilai akurasi, presisi, recall dan F1 score sebagai berikut :

$$Akurasi = \frac{TP_{Lokomotif} + TP_{Gerbong}}{\sum TP + TN + FP + FN} \quad (8)$$

$$Presisi = \frac{TP_{Lokomotif} + TP_{Gerbong}}{\sum TP + FP} \quad (9)$$

$$Recall = \frac{TP_{Lokomotif} + TP_{Gerbong}}{\sum TP + FN} \quad (10)$$

$$F1\ Score = 2 * \left(\frac{Presisi * Recall}{Presisi + Recall} \right) \quad (11)$$

Keterangan :

- $TP_{Lokomotif}$ = True Positive jika input citra terdapat lokomotif dan dibaca terdapat lokomotif oleh sistem,
- $TP_{Gerbong}$ = True Positive jika input citra terdapat gerbong dan dibaca terdapat gerbong oleh sistem,
- $\sum TP$ = Jumlah keseluruhan dari gerbong dan lokomotif yang terdeteksi benar oleh sistem,
- TN = True Negative jika input citra tidak terdapat lokomotif atau gerbong dan dibaca tidak terdapat lokomotif atau gerbong oleh sistem,
- FP = False Positive Jika input citra tidak terdapat lokomotif atau gerbong tetapi dibaca terdapat lokomotif atau gerbong oleh sistem,
- FN = False Negative jika input citra terdapat lokomotif atau gerbong tetapi dibaca tidak terdapat lokomotif atau gerbong oleh sistem.

Data citra yang digunakan untuk penelitian ini menggunakan hasil foto langsung sebanyak 544 citra gambar, yang dimana terbagi menjadi citra lokomotif dan citra gerbong dengan pembagian waktu siang, sore, dan malam hari. Berikut merupakan hasil pengujian deteksi kereta api :

Tabel 4. Tabel Pengujian Deteksi Kereta Siang

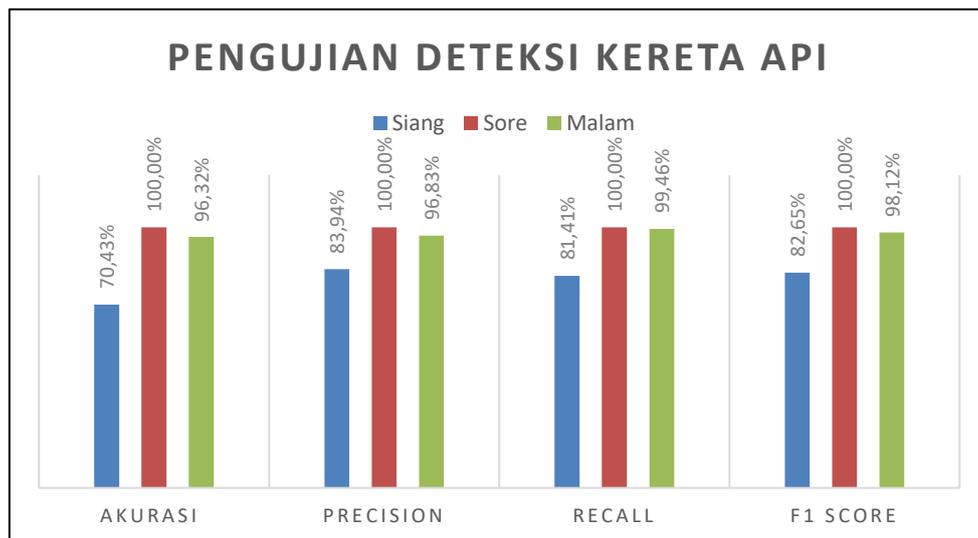
		Lokomotif	Gerbong	Tidak terdeteksi	Total	Akurasi	Precision	Recall	F1 Score
Siang	Lokomotif	77	7	37	121	70,43%	83,94%	81,41 %	82,65 %
	Gerbong	24	85	0	109				
	Tidak Ada	0	0	0	0				
	Total	101	92	37	230				

Tabel 5. Tabel Pengujian Deteksi Kereta Sore

		Lokomotif	Gerbong	Tidak terdeteksi	Total	Akurasi	Precision	Recall	F1 Score
Sore	Lokomotif	53	0	0	53	100%	100%	100%	100%
	Gerbong	0	137	0	137				
	Tidak Ada	0	0	0	0				
	Total	53	137	0	230				

Tabel 6. Tabel Pengujian Deteksi Kereta Malam

		Lokomotif	Gerbong	Tidak terdeteksi	Total	Akurasi	Precision	Recall	F1 Score
Malam	Lokomotif	22	5	0	27	96,32%	96,83%	99,46 %	98,12 %
	Gerbong	1	161	1	163				
	Tidak Ada	0	0	0	0				
	Total	23	166	1	190				



Gambar 11. Grafik Pengujian Deteksi Kereta Api

3.3.2 Pengujian Tingkat Akurasi Klasifikasi

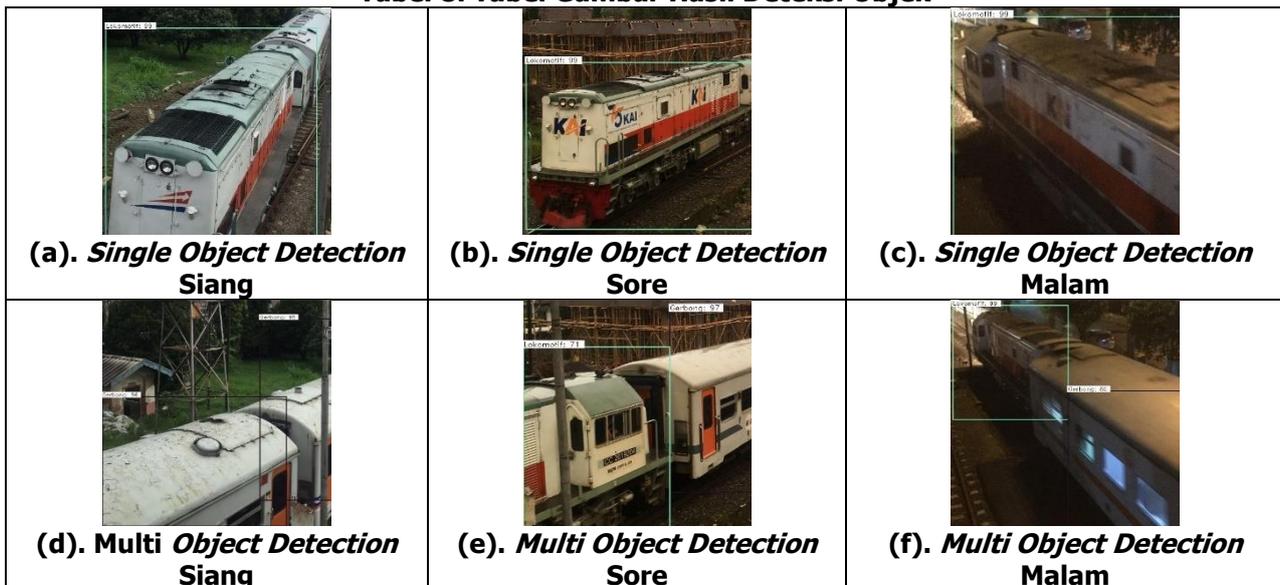
Pada pengujian ini pencarian tingkat akurasi klasifikasi untuk deteksi objek lokomotif dan gerbong pada penelitian deteksi objek kereta api menggunakan metode *Faster RCNN* dengan arsitektur VGG 16.

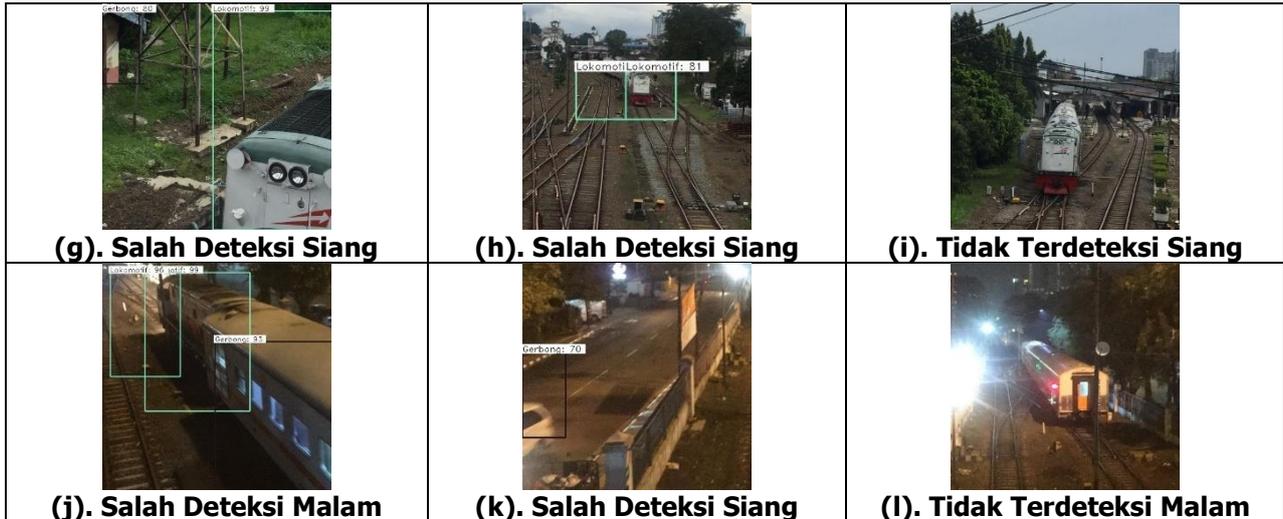
Tabel 7. Hasil Pengujian Akurasi Deteksi Kereta Api

Kondisi Cahaya	Jarak kepada objek	Objek		Rata-Rata Deteksi Objek		
		Lokomotif	Gerbong			
Siang	Jarak 0-20 M	98,46	97,04	97,75	69,48	87,83
	Jarak 20-40 M	43,09	97,38	70,23		
	Jarak 40-60 M	-	97,60	97,60		
	Jarak 150-200 M	0	-	0,00		
	Jarak 200-250 M	81,80	-	81,80		
Sore	Jarak 0-20 M	97,58	97,14	97,36	96,93	
	Jarak 20-40 M	96,50	-	96,50		
Malam	Jarak 0-20 M	97,64	97,76	97,70	97,08	
	Jarak 20-40 M	-	99,00	99,00		
	Jarak 40-60 M	-	94,55	94,55		

Dari hasil pengujian diatas terdapat 535 objek yang terdeteksi benar, mengalami salah deteksi sebanyak 37 objek, dan citra yang tidak terdeteksi sebanyak 38 citra. Sehingga dapat ditarik kesimpulan rata rata deteksi objek untuk deteksi objek kereta api dalam 3 kondisi cahaya dan jarak yang *tentative* sebesar 87,83% dan terjadi kesalahan deteksi sebesar 12,17% untuk deteksi. Berikut adalah sistem mengeluarkan pelabelan untuk objek yang terdeteksi sebagai lokomotif dan gerbong untuk citra yang terdeteksi lokomotif atau gerbong dan menampilkan tingkat akurasi deteksi seperti pada Tabel 8.

Tabel 8. Tabel Gambar Hasil Deteksi Objek





Hasil deteksi objek berdasarkan deteksi kereta api memiliki rata rata akurasi deteksi objek sebesar 0.915 dari jarak terdekat hingga sejauh 60 meter. Akurasi deteksi objek menunjukkan hasil yang lebih baik dibanding terhadap objek manusia yang sebelumnya sudah dilakukan terhadap objek deteksi manusia yang memiliki tingkat akurasi 0,861.

4. KESIMPULAN

Berdasarkan pengujian yang telah dilakukan, Metode *Faster RCNN* dengan model VGG 16 mampu melakukan klasifikasi, deteksi, dan bounding box dalam mendeteksi objek kereta api yang terbagi antara lokomotif dengan rata rata *accuracy* sebesar 0.86 dan gerbong sebesar 0.97, dengan evaluasi pengujian kinerja system *accuracy*, *precision*, *recall*, dan *F-Measure* masing masing sebesar 0.88, 0.93, 0.93, dan 0.93. Dengan hasil tersebut metode deteksi objek *Faster RCNN* dengan model VGG16, memiliki kinerja baik dalam deteksi objek dengan berbagai macam kondisi cahaya, dengan jarak terdekat hingga sejauh 60 meter.

DAFTAR RUJUKAN

- Benuwa, B., Zhan, Y., Ghansah, B. (2016). A Review of Deep Machine Learning. *International Journal of Engineering Research in Africa*. 10.4028/www.scientific.net/JERA.24.124
- Girshick, R. (2015). ICCV paper the Open Access Version by Computer Vision Foundation. *Fast R-CNN*.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-December*, 770–778. <https://doi.org/10.1109/CVPR.2016.90>
- Huang, H., Zhou, H., Yang, X., Zhang, L., Qi, L., & Zang, A. Y. (2019). Faster R-CNN for marine organisms detection and recognition using data augmentation. *Neurocomputing*, 337(xxxx), 372–384. <https://doi.org/10.1016/j.neucom.2019.01.084>
- KBBI. 2020. "Kereta Api" : <https://kbbi.kemdikbud.go.id/entri/Kereta%20Api>.

- Khan, A., Sohail, A., Zahoora, U., & Qureshi, A. S. (2020). A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, 53(8), 5455–5516. <https://doi.org/10.1007/s10462-020-09825-6>
- Khurana, K., & Awasthi, R. (2013). Techniques for Object Recognition in Images and Multi-Object Detection. *International Journal of Advanced Research in Computer Engineering & Technology*, 2(4), 1383–1388.
- Lee, C., Kim, H. J., & Oh, K. W. (2016). Comparison of faster R-CNN models for object detection. *International Conference on Control, Automation and Systems, (Iccas)*, (pp. 107–110). <https://doi.org/10.1109/ICCAS.2016.7832305>
- Mathworks. 2020. "What Is Object Detection?" <https://www.mathworks.com/discovery/object-detection.htm>.
- Mathworks. 2018. "What Is Deep Learning?" <https://www.mathworks.com/discovery/deep-learning.html>.
- Nagataries, D., Hardiristanto, S., Purnomo, M. H., & Klasik, A. A. G. (2012). Deteksi Objek pada Citra Digital Menggunakan Algoritma Genetika untuk Studi Kasus Sel Sabit. *Journal of Electrical Engineering*.
- Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6), 1137–1149. <https://doi.org/10.1109/TPAMI.2016.2577031>
- Roh, M. C., & Lee, J. Y. (2017). Refining faster-RCNN for accurate object detection. *Proceedings of the 15th IAPR International Conference on Machine Vision Applications, MVA 2017*, (pp. 514–517). <https://doi.org/10.23919/MVA.2017.7986913>
- Rosebrock, A. (2016). *Intersection over Union (IoU) for object detection. Diambil kembali dari PYImageSearch: https://www.pyimagesearch.com/2016/11/07/intersectionover-union-iou-for-object-detect*
- Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3), 211–252. <https://doi.org/10.1007/s11263-015-0816-y>
- Srijakkot, K., Kanjanasurat, I., Wiriyakrieng, N., & Benjangkaprasert, C. (2020). The comparison of Faster R-CNN and Atrous Faster R-CNN in different distance and light condition. *Journal of Physics: Conference Series*, 1457(1), 0–6. <https://doi.org/10.1088/1742-6596/1457/1/012015>

Tempo, 2019. "KAI Catat 655 Kecelakaan di Perlintasan Kereta 2 Tahun Terakhir"
<https://bisnis.tempo.co/read/1244503/kai-catat-655-kecelakaan-di-perlintasan-kereta-2-tahun-terakhir>.

Y. Lecun, Y. Bengio, dan G. Hinton. (2015). Deep learning. *Nature*, 521(7553), 436–444.