Jurnal Inovasi Teknologi Terapan

Vol. 01, No. 1, (2023) e-ISSN : xxxx - xxxx

Perbandingan Kinerja Algoritma YOLO Dan RCNN Pada Deteksi Plat Nomor Kendaraan

Sesilia Amanda Putri[1*], Gilang Ramadhan[1], Zein Alwildan[1], Irwan[1], Riki Afriansyah[1]
[1]Politeknik Manufaktur Negeri Bangka Belitung, Sungailiat
*E-mail : amandaputrisesilia@gmail.com

Received 2 Januari 2023; Received in revised form 31 Januari 2023; Accepted 31 Januari 2023

Abstract

Vehicle license plate detection is a system used to automatically recognize and identify license plates on passing vehicles. This system is useful to assist in managing traffic, reducing crime, and increasing the efficiency of the transportation system. However, in its implementation, license plate detection often encounters difficulties due to variations in shape, color, and lighting conditions. In addition, vehicle license plates are also cut off or distorted in digital images, which can complicate the detection process. To solve these problems, an effective method is needed to identify objects from digital images. This study aims to evaluate and compare the performance of two popular algorithms used for vehicle license plate detection, namely YOLO (You only look once) and RCNN (Region Convolutional Neural network). The performance evaluation of the two algorithms uses the same dataset so that their accuracy can be compared. The results showed that the YOLOv4 algorithm has a higher level of detection accuracy than RCNN with an accuracy of 96% and 87.8%, respectively. Based on these results it can be concluded that YOLOv4 is more suitable for use in vehicle license plate detection applications with prioritized speeds.

Keywords*: Accuracy; RCNN; Vehicle license plate detection; You only look once V4.*

Abstrak

Deteksi plat nomor kendaraan merupakan sebuah sistem yang digunakan untuk secara otomatis mengenali dan mengidentifikasi plat nomor pada kendaraan yang melintas. Sistem ini berguna untuk membantu dalam mengelola lalu lintas, mengurangi kejahatan, dan meningkatkan efisiensi sistem transportasi. Namun pada penerapannya, pendeteksian plat nomor sering kali mengalami kesulitan karena adanya variasi dalam bentuk, warna, dan kondisi pencahayaan. Selain itu, plat nomor kendaraan juga sering terpotong atau terdistorsi pada citra digital, sehingga dapat mempersulit proses deteksi. Dalam upaya menyelesaikan masalah tersebut, maka diperlukan metode deteksi yang efektif dalam mengidentifikasi objek dari citra digital. Penelitian ini bertujuan untuk mengevaluasi dan membandingkan kinerja antara dua algoritma yang populer digunakan untuk deteksi objek plat nomor kendaraan, yaitu YOLO (*You Only Look Once*) dan RCNN (*Regional Neural network*). Evaluasi kinerja dari kedua algoritma menggunakan dataset yang sama agar dapat dibandingkan tingkat akurasinya. Hasil penelitian menunjukkan bahwa algortima YOLOv4 memiliki tingkat akurasi deteksi yang lebih tinggi dibandingkan RCNN dengan masing-masing sebesar 96% dan 87,8%. Berdasarkan hasil tersebut, dapat disimpulkan bahwa YOLOv4 lebih cocok digunakan untuk aplikasi deteksi plat nomor kendaraan dengan kecepatan yang diutamakan.

Kata kunci: Akurasi; RCNN; Deteksi plat nomor kendaraan; *You only look once V4.*

1. PENDAHULUAN

Dalam kegiatan sehari-hari, terdapat banyak kendaraan yang melintasi jalan-jalan di sekitar kita. Setiap kendaraan memiliki plat nomor unik, yang digunakan sebagai identifikasi dari setiap kendaraan. Plat nomor ini dapat digunakan sebagai bukti kepemilikan kendaraan tersebut sehingga dapat diterapkan untuk meningkatkan teknologi sistem pelayanan parker [1]. Namun, dalam beberapa kasus, terdapat kendaraan dengan plat nomor yang tidak terlihat jelas sehingga dapat menyulitkan dalam menangani masalah kepemilikan kendaraan tersebut. Untuk mengatasi

masalah ini, diperlukan sistem yang dapat mendeteksi plat nomor kendaraan dengan tepat. Sistem deteksi plat nomor kendaraan dapat dilakukan menggunakan teknik pengenalan pola menggunakan algoritma *object detection* yang bertujuan untuk mengidentifikasi objek-objek yang terdapat pada gambar atau video.

Pengambilan objek plat nomor kendaraan dapat menggunakan berbagai metode, terutama dalam proses pembelajaran mesin dan pemrosesan citra. Beberapa metode pembelajaran mesin yang dapat digunakan diantaranya adalah *neural network* dan *support vector machine*, sementara beberapa metode pemrosesan citra yang dapat digunakan adalah *template matching*, *edge detection*, dan *region-based approach*. Salah satu metode yang lebih efektif dalam menangani masalah pengolahan citra kompleks adalah metode *Deep learning*, yang merupakan teknik pembelajaran menggunakan jaringan syaraf tiruan berlapis yang mirip dengan otak manusia. Dalam *deep learning*, neuron-neuron bergabung membentuk jaringan syaraf yang sangat kompleks [2]. Salah satu teknik dari *deep learning* yang saat ini memiliki hasil terbaik dalam pengenalan citra yaitu *Convolutional Neural Network* (CNN) karena mampu mengekstraksi fitur penting dari setiap citra tanpa bantuan manusia [3]. Beberapa detektor yang menggunakan teknik CNN sebagai bagian dari proses deteksi objek, diantaranya RCNN, *Fast* RCNN, dan *Faster* RCNN. Selain itu, algoritma YOLO (*You only look once*) juga sering digunakan sebagai detektor. YOLO merupakan algoritma yang hanya membutuhkan satu langkah saja dalam mendeteksi objek menggunakan *neural network*. Sementara RCNN merupakan algoritma yang menggunakan teknik region proposal untuk menentukan daerah objek kemudian dilakukan deteksi. Kedua detektor ini sering digunakan dalam aplikasi pengenalan objek pada citra atau video [4].

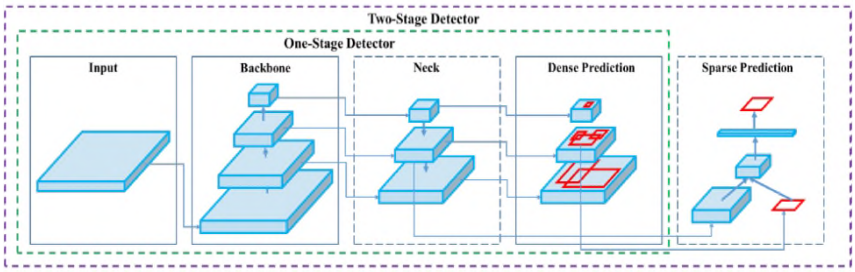
Beberapa penelitian sebelumnya yang telah dilakukan tentang pengenalan objek plat kendaraan dilakukan oleh: Penelitian Mohammad Ali dalam mengembangkan sistem deteksi nomor plat kendaraan menggunakan teknik *deep learning*. Algoritma CNN dipakai untuk melatih model deteksi dengan dataset citra yang mengandung nomor plat kendaraan. Hasil penelitian menunjukkan sistem deteksi plat kendaraan mencapai akurasi sebesar 99,2% [5]. Algoritma CNN juga digunakan oleh Muhammad Hassan untuk mendeteksi dan memahami plat nomor kendaraan secara *real-time*. Tingkat akurasi yang didapat dalam penelitiannya sebesar 95,42% dengan dataset lebih dari 40.000 gambar [6]. Selanjutnya pada penelitian K. R. Munjal menggunakan algortima YOLO untuk mendeteksi nomor plat pada citra yang diambil dari kamera mobil. Hasil penelitian menunjukkan bahwa metode YOLO mampu mendeteksi dengan akurasi tinggi pada citra yang diambil dari berbagai sudut dan kondisi cahaya [7]. Pada penelitian Muhammad Usman dalam melatih model deteksi menggunakan metode kombinasi antara *K-nearest neighbor* (k-NN) dan *Support vector machine* (SVM) dan mendapatkan hasil penelitian yang mencapai akurasi 97,5% [8]. Mochammad Zakiyamani membuat sistem yang dapat mengenali karakter pada plat nomor kendaraan Indonesia dengan metode CNN. Tingkat akurasi dari hasil pengujian plat kendaran mencapai 96% dengan tingkat kesalahan 11,78% [9]. Metode-metode yang digunakan dalam pengenalan citra pada penelitian terkait yang dibahas berhasil dilakukan pendeteksian dengan tingkat akurasi yang tinggi. Namun pada penerapannya masih terdapat beberapa kendala seperti ketergantungan pada kualitas citra, kondisi pencahayaan serta keterbatasan dalam menangani variasi seperti huruf, angka, dan simbol pada plat nomor. Oleh karena itu diperlukan penelitian lebih lanjut untuk mengembangkan metode deteksi plat nomor kendaraan yang lebih efektif dan akurat.

Pada penelitian ini penulis mengevaluasi dan membandingkan dua algoritma populer dalam *object detection*, yaitu YOLO dan RCNN. Pengolahan citra pada kedua algoritma tersebut memiliki kelebihan dan kekurangan masing-masing, sehingga akan dilakukan analisis terhadap kinerja dan hasil deteksi dari kedua algoritma tersebut. Pengujian akan dilakukan dengan menggunakan dataset yang sama untuk proses pelatihan. Tujuan pengujian dilakukan untuk melihat algoritma mana yang lebih baik dari segi kecepatan,

akurasi deteksi dan cara kerja. Hasil dari penelitian ini diharapkan dapat memberikan informasi bermanfaat bagi pengembang sistem deteksi plat nomor kendaraan dan menjadi solusi dalam mengetahui algoritma yang lebih efektif pada aplikasi tersebut.

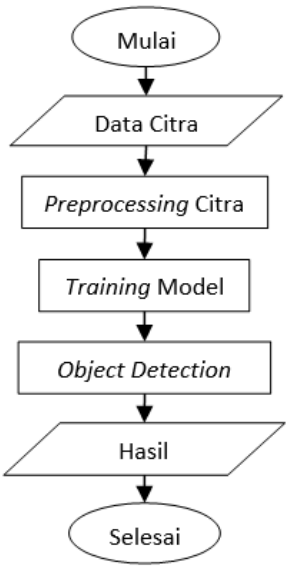
2. METODE PENELITIAN

Pada penelitian ini, metode deteksi objek menggunakan teknik pemrosesan citra digital yang terlibat dalam mengidentifikasi dan menemukan lokasi objek dalam gambar atau video.



Gambar 1. *Object Detector*

Dalam penerapannya, sistem deteksi objek terdiri dari jaringan inti (*backbone*), jaringan kepala (*head*) dan jaringan leher (*neck*). Jaringan inti dilatih pada dataset besar seperti *image*Net, sedangkan jaringan kepala (*head*) yang digunakan untuk memprediksi kelas dan *bounding boxes* objek dalam gambar yang diinput. Ada beberapa jenis jaringan inti yang digunakan dalam deteksi objek yaitu *ResNet, DenseNet, Darknet,* dan *MobileNet*. Pada bagian jaringan kepala, sistem dapat menggunakan detektor objek satu tahap atau dua tahap [10] . Contoh detektor objek satu tahap yaitu YOLO, SSD, RetinaNet, sementara contoh detektor objek tanpa *anchor* adalah *CornerNet* dan FCOS. Contoh detektor objek dua tahap termasuk seri RCNN misalnya, *fast R-CNN, faster-RCNN* dan R-FCN serta detektor tanpa *anchor* seperti *RepPoints*. Beberapa detektor objek juga memiliki jaringan *neck* yang disisipkan antara *backbone* dengan *head* dan digunakan untuk mengumpulkan peta fitur dari berbagai tahap jaringan inti. Contoh jaringan *neck* termasuk FPN, PAN, dan NAS-FPN [10]. Penggunaan *backbone*, *head*, dan *neck* ini bergantung pada kebutuhan dan tujuan penggunaan sistem deteksi objek. Pemilihan algoritma yang tepat akan mempengaruhi kecepatan, akurasi, dan kemampuan sistem dalam mendeteksi objek yang diinginkan. Alur dari proses pemrosesan citra plat nomor kendaraan dapat dilihat pada Gambar 2.



Gambar 2. Tahapan Eksperimen

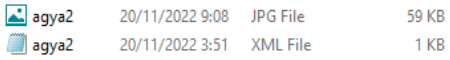
Dalam penelitian ini, peneliti akan menggunakan algoritma YOLO dan RCNN untuk dibandingkan hasil deteksinya. Proses pengolahan citra dimulai dengan mengumpulkan dataset citra plat nomor kendaraan untuk digunakan sebagai data pelatihan. Selanjutnya, menyiapkan model sesuai algoritma yang dipilih untuk proses deteksi. Setelah itu model akan dilatih menggunakan dataset yang disiapkan sebelumnya. Hasil model yang didapat akan evaluasi untuk mengetahui kualitas model sebelum diterapkan pada sistem deteksi.

2.1. Persiapan Dataset

Dalam penelitian ini, citra plat nomor kendaraan yang digunakan diperoleh dari dataset yang diunggah oleh Kadek Gunawan di Kaggle. Data yang diambil dari keseluruhan dataset tersebut terdiri dari 150 citra dengan format.jpg. Sebelum dapat digunakan dalam proses pelatihan, citra tersebut harus melalui tahap anotasi objek dengan melakukan proses *labelling*. *Labelling* merupakan proses pembuatan label pada gambar dengan cara memberikan *bounding boxes* beserta nama kelas pada objek yang akan dideteksi. Objek yang akan dideteksi diberi label dengan nama kelas "*platenumber*" dengan format file .txt untuk algoritma YOLOv4 dan file .xml untuk algoritma RCNN. File data tersebut berisi angka *annotation image*. Proses *labelling* ini dilakukan secara manual menggunakan *software MakesenseAI*. Dengan melakukan *labelling*, citra tersebut dapat digunakan dalam proses pelatihan algoritma YOLOv4 atau RCNN untuk mendeteksi objek plat nomor kendaraan di dalam citra.

Gambar 3. Proses Anotasi Dataset

Gambar 4 merupakan contoh hasil gambar yang sudah diberikan label:



Gambar 4. Dataset Hasil *Labelling*

2.2. Prinsip Kerja Algoritma YOLOv4 dan RCNN

2.2.1. Algoritma YOLOv4

Algoritma YOLO (*You only look once*) adalah salah satu metode *deep learning* yang dapat digunakan untuk mendeteksi objek di dalam citra digital. Algoritma ini beroperasi dengan cara mengambil sebuah citra dan melakukan prediksi objek yang terdapat di dalamnya secara sekaligus, sehingga dapat melakukan deteksi objek dengan kecepatan yang tinggi. YOLO mampu menemukan posisi objek dengan tingkat akurasi yang tinggi karena menggunakan jaringan saraf tiruan *convolutional* (CNN) untuk mengelompokkan citra menjadi beberapa bagian kecil (kotak atau *grid*) dan mengklasifikasikan setiap bagian tersebut sebagai kemungkinan keberadaan objek. Ada beberapa versi dari algoritma YOLO, di antaranya YOLOv1, YOLOv2, YOLOv3, dan YOLOv4. Dari keempat versi tersebut, YOLOv4 merupakan versi terbaru yang mengalami peningkatan dari versi sebelumnya, dengan meningkatkan kemampuan deteksi objek dan kecepatan proses [11]. Dalam suatu penelitian, algoritma YOLOv4 dapat digunakan sebagai metode untuk mendeteksi objek di dalam citra digital.

Arsitektur YOLOv4 terdiri dari beberapa lapisan konvolusi yang diikuti oleh lapisan pemrosesan regional (*regional processing layers*). Pada lapisan pertama terdapat input gambar atau citra yang akan diproses oleh algoritma YOLOv4. Arsitektur YOLOv4 menggunakan sebuah *backbone* yang berupa jaringan saraf konvolusi bernama Darknet53. Darknet53 adalah sebuah jaringan *Neural Network* (NN) yang digunakan untuk melakukan ekstraksi fitur, yang merupakan proses mengambil

informasi yang relevan dari sekumpulan data. Jaringan ini memiliki 53 lapisan, seperti yang ditunjukkan pada gambar 5. Lapisan-lapisan tersebut terdiri dari lapisan konvolusi yang masing-masing memiliki ukuran 3x3 dan 1x1, yang dipakai secara berturut-turut. Lapisan konvolusi merupakan lapisan NN yang digunakan untuk mengekstrak fitur dari data dengan mengaplikasikan *filter* ke data tersebut.

Setiap lapisan memiliki residual block yang dilakukan perhitungan berurutan dari 1 kali residual *block layer* 1x1 yang digunakan untuk mengurangi ukuran saluran dan *layer* berukuran 3x3. Seiring bertambahnya jumlah lapisan maka jumlah *filter* akan bertambah 2x lipat untuk mencari cakupan objek yang lebih luas. Setiap lapisan konvolusi ini diisi dengan *batch normalization layers* dan *leaky ReLu activation* [13].

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Gambar 5. Arsitektur YOLOv4

Selain itu, bagian *head* dari arsitektur YOLOv4 [12] terdiri dari beberapa lapisan konvolusi, lapisan *max pooling*, dan lapisan *fully connected* yang digunakan untuk mengklasifikasi objek yang terdapat dalam gambar [19]. Pada lapisan terakhir, algoritma YOLOv4 akan mengeluarkan hasil deteksi objek berupa "kotak pembatas" (bounding box) yang menandai lokasi, kelas, dan tingkat kepercayaan (*confidence*) dari objek yang terdeteksi. Kotak pembatas tersebut akan menandai batas-batas objek yang terdeteksi dan akan digunakan untuk mengidentifikasi objek tersebut dengan lebih spesifik.

Berikut ini merupakan beberapa parameter yang digunakan dalam algoritma YOLOv4:

1. *Batch Size*

Batch size adalah jumlah data yang akan diproses dalam satu iterasi pembelajaran. Pemilihan *batch size* dapat mempengaruhi kecepatan pelatihan dan akurasi model. *Batch size* yang lebih besar akan menyebabkan lebih banyak data yang diolah dalam satu iterasi, tetapi dapat menyebabkan terjadinya *overfitting*. Sebaliknya, *batch size* yang kecil akan membutuhkan waktu pelatihan yang lebih lama, tetapi dapat meningkatkan akurasi model [13].

2. *Filters*

Filter merupakan alat yang digunakan untuk menemukan fitur-fitur tertentu dalam citra yang sesuai dengan pola yang telah ditentukan di jaringan *convolutional*. Penggunaan *filters* yang lebih banyak akan menemukan lebih banyak fitur dalam citra, tetapi juga akan membutuhkan lebih banyak waktu proses dan data latih [14].

3. *Learning Rate*

Parameter yang digunakan untuk menentukan seberapa besar perubahan yang akan diterapkan pada model setiap kali iterasi pelatihan. *Learning rate* yang terlalu tinggi dapat menyebabkan model tidak stabil dan melebihi target, sementara *learning rate* yang terlalu rendah dapat memperlambat proses pelatihan dan menghasilkan model yang tidak optimal [14].

4. *Max batches*

Max batches adalah jumlah maksimum iterasi yang akan diproses selama proses pelatihan. Parameter ini sering digunakan untuk mengontrol panjang proses pelatihan dan memastikan bahwa model tidak overfit pada data pelatihan. Tingginya nilai iterasi membuat sistem mempelajari lebih banyak data latih. Nilai *max batches* tidak boleh kurang dari jumlah data latih dan harus disesuaikan dengan jumlah kelas objek yang akan dideteksi [13]. *Max batches* = *number class* × 2000.

5. *Confidence Threshold*

Confidence threshold merupakan nilai ambang batas yang digunakan untuk menentukan kepercayaan model terhadap keberadaan objek yang dideteksi. Nilai *confidence threshold* yang tinggi akan menghasilkan deteksi objek yang lebih akurat, tetapi juga akan mengurangi jumlah deteksi objek secara keseluruhan [13].

2.2.2. Algoritma RCNN

Algoritma RCNN (*Region Based Convolutional Neural Network*) adalah sebuah algoritma *deep learning* yang digunakan untuk pengolahan citra dan pengenalan objek [15]. Algoritma ini bekerja dengan menggunakan metode *sliding window*, yaitu dengan memindahkan daerah-daerah secara berurutan pada citra yang akan diolah, lalu melakukan deteksi objek pada setiap posisi daerah tersebut. Setiap daerah yang mengandung objek akan diberi label sebagai *Region of Interest* (ROI). Kemudian, RCNN akan mengekstrak fitur dari setiap ROI yang telah ditemukan, lalu mengklasifikasikan ROI tersebut ke dalam kelas objek yang sesuai menggunakan model klasifikasi yang telah dilatih sebelumnya. Algoritma RCNN merupakan salah satu metode objek deteksi yang cukup efektif, tetapi memiliki waktu proses yang relatif lama karena harus melakukan pemrosesan pada setiap daerah yang ditemukan. Setelah algoritma RCNN dikembangkan, terdapat beberapa varian dari RCNN yang juga dikembangkan, di antaranya adalah *Fast* R-CNN dan *Faster* RCNN [15]. Kedua algoritma tersebut dikembangkan untuk meningkatkan kecepatan proses algoritma RCNN tanpa menurunkan tingkat akurasi deteksi objek.

Arsitektur algoritma RCNN terdiri dari tiga bagian utama, yaitu *Extraction Layer*, *Classification Layer*, dan *Regression Layer*. *Extraction Layer* merupakan bagian pertama dari arsitektur RCNN yang menggunakan *Convolutional Neural Network* (CNN) untuk mengekstrak fitur-fitur yang relevan dari citra yang akan dikenali. Proses ekstraksi fitur ini dilakukan dengan cara mengalikan setiap pixel pada citra dengan *filter* yang telah ditentukan, kemudian hasilnya dijumlahkan dan direset ke dalam sebuah *feature map*. *Classification Layer* merupakan bagian kedua dari arsitektur RCNN yang bertugas untuk mengenali setiap objek di dalam citra berdasarkan fitur-fitur yang telah diekstrak oleh *Extraction Layer*. Setiap objek yang diidentifikasi akan diberikan label sesuai kelas yang telah ditentukan sebelumnya dengan menggunakan *Support vector Machine* (SVM). *Regression Layer* merupakan bagian terakhir dari arsitektur RCNN yang bertugas untuk menentukan lokasi objek di dalam citra dengan menggunakan informasi yang telah diperoleh dari *Classification Layer* dengan menggunakan regresi linear [16]. Regresi linear ini dilakukan dengan cara mencari nilai yang paling tepat untuk menjelaskan hubungan antara variabel terikat dan variabel bebas dalam suatu model linier sederhana.

Adapun parameter yang digunakan dalam algortima RCNN sebagai berikut:

1. *Epoch*

Epoch merupakan jumlah iterasi dari proses pelatihan model. Semakin banyak *epoch* yang dilakukan, maka semakin tinggi kemungkinan model mencapai konvergensi.

Namun, model akan membutuhkan waktu yang lebih lama dan juga dapat menyebabkan mengalami *overfitting* [15].

2. *Activation function*

Activation function adalah fungsi yang digunakan pada jaringan saraf tiruan untuk mengaktivasi neuron pada lapisan tersembunyi. Beberapa *activation function* yang umum digunakan antara lain *sigmoid*, *tanh*, dan ReLu [17].

3. *Loss function*

Loss function adalah fungsi yang digunakan untuk mengukur keberhasilan model dalam mengklasifikasikan objek. Beberapa *loss function* yang umum digunakan yaitu *cross-entropy loss dan mean squared error* (MSE). MSE mengukur seberapa jauh hasil prediksi model dari target yang sebenarnya sedangkan *cross-entropy-loss* mengukut seberapa jauh model terhadap kemungkinan keberadaan objek [15].

4. *Optimizer*

Optimizer merupakan algoritma yang digunakan untuk mengoptimalkan nilai parameter model pada RCNN. Beberapa *optimizer* yang umum digunakan antara lain SGD (*Stochastic Gradient Descent*), Adam, dan RMSprop (*Root Mean Square Propogation*) [18].

Parameter-parameter lain yang juga digunakan dalam algoritma RCNN seperti *batch size*, *learning rate*, *filters* dan lainnya tergantung pada konfigurasi model yang akan digunakan.

2.3. Eksperimen

Tahap eksperimen yang dilakukan setelah *preprocessing* data yaitu dilakukan pelatihan atau *training* pada data yang telah diolah. Proses pelatihan ini akan menghasilkan model yang nantinya akan digunakan untuk melakukan deteksi nomor plat kendaraan.

2.3.1.*Training* YOLOv4

Pelatihan model dilakukan secara *cloud* menggunakan *cloud notebook* yaitu *Google Colaboratory*. Proses ini menggunakan *library open-source Darknet* yang tersedia di *online repository* (*AlexeyAB*). Pada tahap ini menggunakan model yang telah dilatih sebelumnya (*pre-trained*) weights YOLOv4 dengan teknik *transfer learning*. Proses *transfer learning* pada Darknet menggunakan file custom .cfg, obj.data, obj.names, dan *pretrained weights*. Proses *training* menggunakan *Framework Neural Network Darknet* sebagai *load model* dan YOLOv4 sebagai *load weights* dengan konfigurasi seperti pada Tabel 1.

Tabel 1. Konfigurasi pada weights *YOLO*v4

Jenis Konfigurasi	Keterangan
Batch size	64
Subdivisions	16
Width x height	416 x 416
Max_batches	6000
Learning rate	0.001
Filters	18

Setelah proses pelatihan selesai, hasilnya akan tersimpan dalam bentuk *file weights*. Proses penyimpanan *file weights* dimulai dari 1000 iterasi dengan proses 2000 iterasi per 1 kelas objek. Proses akan berakhir sampai menghasilkan file best.weights. Peneliti membutuhkan waktu sekitar 2 jam untuk melakukan proses pelatihan ini. *File* best.weights inilah yang akan digunakan untuk proses pengujian pada deteksi gambar statis dan deteksi secara *real-time*.

2.3.2.*Training* RCNN

Training model pada RCNN menggunakan cara yang sama seperti saat melatih model YOLO yaitu dimulai dengan menggunakan data yang sudah dilakukan *preprocessing* sebelumnya. Model yang akan digunakan untuk melakukan konfigurasi dari *training* yaitu *Inception Resnet* V2, dimana model ini sudah disediakan oleh *tensorflow* sendiri. Dalam proses *training* model deteksi plat nomor menggunakan konfigurasi pada Tabel 2.

Tabel 2. Konfigurasi pada RCNN

Jenis Konfigurasi	Keterangan
Batch size	10
Epoch	50
Width x height	224 x 224
Learning rate	0.0001
Optimizer	Adam
Loss function	Mse
Activation function	ReLu, sigmoid

Proses *training* model dimulai dengan mengirimkan data ke model yang telah dikonfigurasi. Model kemudian akan melakukan proses belajar dengan menyesuaikan bobotnya dengan data yang diberikan. Pada saat melakukan proses *training* data penulis memerlukan waktu sekitar 4 jam untuk melakukan *training* sebanyak 50 steps tersebut. Model yang didapat dari hasil *training* disimpan dalam bentuk *file* model.h5. Model ini kemudian akan diuji dengan menggunakan data validasi untuk mengevaluasi kinerja model. Model yang telah melalui proses pelatihan dan validasi kemudian disimpan untuk digunakan pada proses deteksi plat nomor pada citra baru. Hasil dari tahap akhir *training* yaitu mendapatkan *bounding boxes* objek plat nomor kendaraan pada citra dengan mengklasifikasi objek yang terdeteksi menjadi kelas yang sesuai. Proses memprediksi probabilitas keberadaan objek pada setiap kelas yang ditentukan sebelumnya akan diberi label sesuai nama kelas beserta nilai *confidence*nya. Dapat dilihat pada Gambar 6 terdapat garis yang mengelilingi objek pada citra dengan label "*platenumber*: 1.00%" yang artinya pendeteksian objek plat pada citra berhasil dilakukan sesuai label objek dengan nilai pendeteksian 100%.

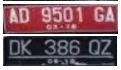
Gambar 6. *Bounding Boxes* Pada Citra

3. HASIL DAN PEMBAHASAN

Keseluruhan proses deteksi plat nomor kendaraan telah selesai dilakukan, selanjutnya dilakukan pengujian untuk mengetahui kinerja model pada deteksi citra baru. Berikut ini merupakan tabel uji coba pada 15 citra kendaraan dengan menggunakan algoritma YOLOv4 dan RCNN. Pada tabel tersebut terdapat kolom "Terdeteksi" dan "Nilai Pendeteksian" yang digunakan untuk menampilkan hasil deteksi objek pada citra. Kolom "Terdeteksi" digunakan untuk mengetahui apakah algoritma berhasil menemukan objek pada citra yang ditandai dengan *bounding boxes*. Sedangkan kolom "Nilai pendeteksian" menunjukkan tingkat akurasi dari deteksi objek pada *bounding boxes*. Hasil uji coba dapat dilihat pada Tabel 3. Berdasarkan Tabel 3 hasil pengujian tersebut, dapat dilihat bahwa algoritma YOLOv4 berhasil menemukan *bounding boxes* pada 15 citra pengujian, sementara algoritma RCNN mengalami 1 kesalahan dalam mendeteksi objek plat kendaraan. Rata-rata nilai pengujian deteksi dalam *bounding boxes* adalah diatas 80%. Perbedaan nilai

bounding boxes terjadi dikarenakan algoritma tidak sepenuhnya yakin terhadap deteksi objek yang dilakukan. Hal ini disebabkan oleh beberapa faktor seperti kemiringan objek, bentuk objek yang unik, kecerahan citra yang tidak sama, serta kualitas citra yang rendah. Penyebab lainnya mungkin terjadi karena adanya kesalahan pada algoritma yang digunakan sehingga dapat menyebabkan deteksi pada *bounding boxes* bervariasi.

Tabel 3. Hasil pengujian

No	Nama File	Plat	Algoritma *YOLO*v4		Algoritma *RCNN*	
			Terdeteksi	Nilai Pendeteksian	Terdeteksi	Nilai Pendeteksian
1	agya6.jpg		Ya	100%	Ya	96%
2	brio29.jpg		Ya	96%	Ya	100%
3	crv3.jpg		Ya	86%	Ya	88%
4	crv15.jpg		Ya	98%	Ya	88%
5	crv16.jpg		Ya	94%	Ya	92%
6	crv25.jpg		Ya	100%	Ya	98%
7	crv26.jpg		Ya	100%	Ya	100%
8	hitam24.jpg		Ya	88%	Tidak	-
9	hrv9.jpg		Ya	100%	Ya	92%
10	jazz9.jpg		Ya	100%	Ya	96%
11	jazz10.jpg		Ya	96%	Ya	92%
12	jazz13.jpg		Ya	98%	Ya	96%
13	r009.jpg		Ya	90%	Ya	86%
14	swift1.jpg		Ya	98%	Ya	96%
15	swift4.jpg		Ya	100%	Ya	98%

Berdasarkan nilai pendeteksian, dapat dihitung akurasi sebagai berikut:

$$Akurasi = \left(\frac{jumlah\ nilai\ deteksi}{total\ jumlah\ citra}\right) x100\%$$

Didapat akurasi hasil deteksi pada pengujian algoritma YOLOv4 dan algoritma RCNN dengan masing-masing akurasi sebesar 96% dan 87,8%. Dari data tersebut dapat disimpulkan bahwa algoritma YOLOv4 memiliki tingkat akurasi yang lebih tinggi daripada algoritma RCNN dalam melakukan deteksi objek pada gambar.

4. SIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa kedua algoritma YOLO dan RCNN memiliki kinerja yang baik dalam melakukan deteksi plat nomor kendaraan. Namun, setelah dibandingkan kinerja kedua algoritma tersebut dengan menggunakan dataset yang sama, ditemukan bahwa algoritma YOLOv4 lebih unggul dibandingkan dengan algoritma RCNN. Beberapa kesimpulan lebih detail sebagai berikut:

1. Algoritma YOLOv4 memiliki akurasi lebih besar dibandingkan algoritma RCNN dengan masing-masing akurasi sebesar 96% dan 87,8% pada percobaan deteksi 15 citra kendaraan.
2. Pada penelitian ini algoritma YOLOv4 memerlukan waktu lebih sedikit dalam melakukan proses *training* dibandingkan algoritma RCNN. Algoritma YOLOv4 membutuhkan waktu sekitar 2 jam, sedangkan RCNN membutuhkan waktu sekitar 4 jam.
3. Ukuran gambar dan parameter dapat mempengaruhi hasil dan waktu proses *training*. Seperti contoh, penggunaan *epoch* yang besar dapat meningkatkan akurasi namun juga dapat menyebabkan *overfitting*. Selain itu juga proses *training* data menjadi semakin lama.

Kesimpulan ini menunjukkan bahwa algoritma YOLO lebih cocok digunakan dalam aplikasi deteksi plat nomor kendaraan dibandingkan algoritma RCNN dari segi kecepatan dan akurasi deteksi.

DAFTAR PUSTAKA

[1] I. F. Adila, Y. Mandiri, "Simulasi Sistem Smart Parking", Politeknik Manufaktur Negeri Bangka Belitung", 2020.

[2] P. A. Nugroho, I. Fenriana, and R. Arijanto, "Implementasi Deep learning Menggunakan Convolutional Neural network (CNN) Pada Ekspresi Manusia", *ALGOR*, vol.2 no.1, pp.12-20, 2020.

[3] I. W. S. Eka Putra, "Klasifikasi Citra Menggunakan Convolutional Neural network (CNN) Pada Caltech 101", *Jurnal Teknologi ITS*, vol. 5, no. 1, pp. A65-A69, Mar. 2016.

[4] W. Chen, Z. Liu, Y. Ma, and L. Li, "A Comparison of YOLO and RCNN for Object Detection in Images and Videos", *IEEE Access*, vol. 7, pp. 132964-132974, 2019.

[5] M. Ali, "A License Plate Recognition *System Using Deep Learning* and Its Application in Vehicle Tracking", *International Journal of Emerging Technologies in Engineering Research*, vol.8, no.2, pp.138-143, 2020.

[6] M. Hassan, "Real-time License Plate Recognition using Convolutional Neural Network", *International Journal of Computer Applications,* vol.176, no.2, pp. 9-15, 2018.

[7] K. R. Munjal, "Automatic License Plate Recognition Using Deep Learning", *Journal of Engineering and Applied Sciences*, vol.13, no.9, pp.3223-3226, 2018.

[8] M. Usman, "A Licence Plate Recognition System Using Hybrid Method and Its Application in Traffic Management", *International Journal of Computer Applications* (IJCA), vol.165, no.5, 2021.

[9] M. Zakiyamani, T. I. Cahyani, D. Riana, S. Hardianti, "Deteksi Dan Pengenalan Plat Karakter Nomor Kendaraan Menggunakan OpenCV Dan *Deep learning* Berbasis Python", INTECOMS: *Journal of Information Technology and Computer Science*, vol. 5 no.1, pp. 56-64, 2022.

[10] A. Bochkovskiy, C. Y. Wang, H. Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of *Object detection*," *Institute of Information Science, Academia Sinica*, Taiwan. Apr. 2020.

[11] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, "You only look once: Unified, real-time object detection", *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 779-788, Des. 2016.

[12] J. Redmon, A. Farhadi, "YOLOv3: An Incremental Improvement", University of Washington, Apr. 2018.

[13] T. A. A. H. Kusuma, K. Usman, S. Saidah, "People Counting for Public Transportations Using You only look once Method", *Jurnal Teknik Informatika*, vol.2, no. 1, pp.57-66, Feb. 2021.

[14] X. Zhang, Q. Du, Y. Dan, Y. Hu, S. Gao, "A comprehensive survey of deep learning in remote sensing," ISPRS *Journal of Photogrammetry and Remote Sensing,* vol.158, pp. 3-32, 2019.

[15] R. Girshick, J. Donahue, T. Darrell, J. Malik, "Rich feature hierarchies for accurate *object detection* and semantic segmentation," *Proceedings of the IEEE Conference on computer vision and pattern recognition*, pp. 580-587, 2014.

[16] L. Wang, W. Ouyang, X. Wang, X. Chen, "Deep learning for object detection: A comprehensive review," *Pattern Recognition*, vol. 84, pp. 317-338, 2019.

[17] M. E. H. Chowdhury, S. A. H. Bhuiyan, S. M. A. Amin, M. A. Hannan, "A Review on Activation Functions: Comparisons and Applications", *Neural Computing and Applications,* vol. 31, no. 9, pp. 5185-5195, 2019.

[18] O. Ronneberger, P. Fischer, T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation", *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234-241, Springer, Cham, 2015.

[19] A. Rohim, R. M. Harmie, "Sistem Pendeteksi Buah Lada Berbasis Convolutional Neural network (CNN)", Politeknik Manufaktur Negeri Bangka Belitung, 2021.