JURNAL ILMIAH FIFO **P-ISSN 2085-4315 / E-ISSN 2502-8332**
DOI: http://dx.doi.org/10.22441/fifo.2020.v12i2.002

Investigasi pengaruh *Step Training* pada Skema *Same-Padding* untuk Metode Faster R-CNN dalam Teknologi *Augmented Reality*

Anky Aditya P[1], Suryo Adhi Wibowo[2], Rissa Rahmania[3]

[1,2,3]*Fakultas Teknik Elektro, Universitas Telkom*
Jl. Telekomunikasi, Terusan Buah Batu, Dayeuhkolot, Bandung, 40257
[1]ankyaditya17@gmail.com, [2]suryoadhiwibowo@telkomuniversity.ac.id,
[3]saniarahmani@telkomuniversity.ac.id

Abstract

Augmented Reality (AR) is a technology with the concept of combining real-world dimensions with virtual world dimensions that are displayed in realtime. In the AR environment, interaction techniques used can vary. Marker-based AR is one type of AR that allows virtual objects to be displayed in the real world by using markers as pointers. In the use of marker-based AR required object detection method used for tracking markers. In this study, a system that can detect objects in the form of fingertips will be designed. In designing the system the Faster Region-based Convolutional Neural Network (Faster R-CNN) method is used. R-CNN Faster is an object detection method which is a combination of the Fast R-CNN method and the Region Proposal Network (RPN). The results of the detection parameters will be used for tracking, namely the coordinates x, y, width, and length. This research uses the Faster R-CNN method because it has a faster computing speed compared to the previous method, namely Particle Filter. The Faster R-CNN method uses ResNet architecture as the core of CNN. The system configuration to be tested is the 25K, 50K and 75K step training with the same-padding scheme. The testing process is taken from a video consisting of 10800 training data and 3600 test data. The best system configuration based on parameter priority for AR technology is obtained in the 50K step training.
***Keyword**: augmented reality, convolutional neural network, faster region-based convolutional neural network, region proposal network, ResNet.*

Abstrak

Augmented Reality (AR) adalah teknologi dengan konsep menggabungkan dimensi dunia nyata dengan dimensi dunia virtual yang ditampilkan secara *real-time*. Dalam lingkungan AR, teknik interaksi yang digunakan dapat bermacam – macam. *Marker-based* AR merupakan salah satu jenis AR yang memungkinkan objek virtual ditampilkan ke dalam dunia nyata dengan digunakannya *marker* sebagai *pointer*-nya. Dalam penggunaan AR berbasis *marker* diperlukan metode deteksi objek yang digunakan untuk *tracking marker*. Dalam penelitian ini akan dirancang sebuah sistem yang dapat mendeteksi objek berupa ujung jari. Dalam perancangan sistem tersebut digunakan metode *Faster Region-Based Convolutional Nueral Network* (Faster R-CNN). *Faster* R-CNN merupakan salah satu metode deteksi objek yang merupakan gabungan dari metode *Fast* R-CNN dan *Region Proposal Network* (RPN). Hasil dari parameter deteksi akan digunakan untuk *tracking*, yaitu koordinat x, y, *width*, dan *length*. Penelitian ini menggunakan metode *Faster* R-CNN karena memiliki kecepatan komputasi yang lebih cepat dibandingkan dengan metode sebelumnya yaitu *Particle Filter*. Metode *Faster* R-CNN mengunakan arsitektur ResNet sebagai inti dari CNN. Konfigurasi sistem yang akan diuji adalah *step training* 25K, 50K dan 75K dengan skema *same-padding*. Proses pengujian diambil dari video yang terdiri dari 10800 data latih dan 3600 data uji. Konfigurasi sistem terbaik berdasarkan prioritas parameter untuk teknologi AR didapatkan pada *step training* 50K.
Keyword: *augmented reality, convolutional neural network, faster region-based convolutional neural network, region proposal network, ResNet.*

I. Pendahuluan

Seiring dengan perkembangan teknologi khususnya di bidang *computer vision* yang semakin pesat. Hal ini membuat interaksi antara manusia dan komputer semakin beragam dan menarik. Salah satu pengembangan interaksi ini ialah *Augmented Reality* (AR), dimana AR adalah sebuah teknologi yang meningkatkan efek visual pengguna dalam lingkungan dunia nyata dengan menambahkan objek virtual atau digital seperti teks, gambar 2D, maupun 3D [1]. AR terbagi menjadi 2 jenis, yaitu *marker-based* AR dan *markerless* AR. *Marker-based* AR merupakan salah satu jenis AR yang memungkinkan objek virtual ditampilkan ke dalam dunia nyata dengan digunakannya *marker* yang bekerja sebagai *pointer* sedangkan *markerless* AR tidak menggunakan *marker* sebagai *pointer* untuk berinterkasi. Umumnya pada *marker-based* AR objek yang digunakan sebagai *pointer* untuk berinteraksi adalah tangan atau kertas berwarna [2].

Deteksi objek merupakan bagian dari AR yang sangat penting. Karena AR sangat sensitif dengan apa yang menjadi *marker* tersebut. Jika satu bagian *marker* saja yang hilang atau tidak terlihat di kamera maka AR tidak akan bekerja dengan semestinya. Penelitian ini mengusulkan untuk menggunakan metode yang dinamakan *Faster Region-Based Convolutional Neural Network* (*Faster* R-CNN) untuk deteksi objek sebagai pengganti deteksi objek yang digunakan untuk *tracking marker* pada penelitian sebelumnya [3].

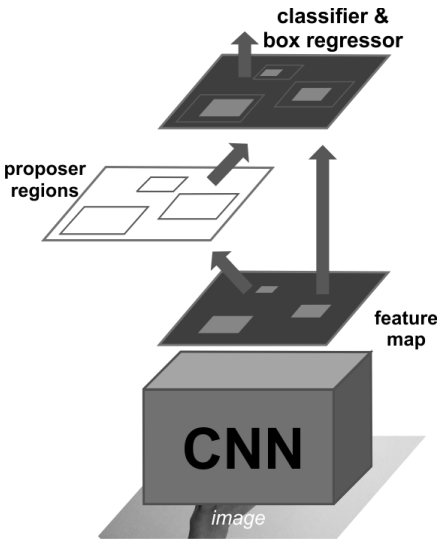
Metode ini akan diaplikasikan dalam pengembangan *marker-based* AR. *Faster* R-CNN merupakan salah satu metode deteksi objek berbasis *Convolutional Neural Network* (CNN). *Faster* R-CNN memiliki kecepatan komputasi yang lebih cepat dibandingkan R-CNN dikarenakan pada *Faster* R-CNN tidak memerlukan proses CNN pada setiap *region proposal*-nya. Penerapan metode tersebut bertujuan untuk meningkatkan performa kerja *augmented reality* karena kecepatan komputasinya yang lebih tinggi.

Faster R-CNN telah diterapkan dalam beberapa penelitian. Dalam penelitian Shaoqing diketahui bahwa dengan diterapkannya *Region Proposal Network* (RPN) dan *Shared* CNN pada *Faster* R-CNN dapat meningkatkan *mean average precision* (mAP) sebesar 1,2% [4]. Selain itu, kecepatan komputasinya hanya membutuhkan 0,2 detik per citra yang dilatih [4].

Penelitian ini menerapkan *marker-based* AR berbasis objek deteksi dengan menggunakan metode *Faster* R-CNN untuk mendapatkan interaksi yang lebih akurat dan mudah untuk digunakan. Studi kasus yang diambil adalah penggunaan *notebook* dalam sehari – hari agar menjadi lebih interaktif. Dengan ini diharapkan penelitian ini dapat digunakan sebagai referensi di penelitian selanjutnya. Dalam pengaplikasiannya pengguna dapat menggerakkan *pointer* dengan ujung jari tanpa harus berinteraksi langsung dengan *notebook*-nya.

II. Metodologi Penelitian

Menurut [4] metode *Faster* R-CNN merupakan pengembangan dari algoritma *Fast* R-CNN yang menghasilkan kecepatan dalam pelatihan data dan kecepatan dalam pengenalan objek [5]. Arsitektur dari *Faster* R-CNN terdiri dari beberapa bagian yaitu CNN dan RPN sebagai *region proposer*, dan *Fast* R-CNN sebagai *detector*. Secara umum arsitektur dari Faster R-CNN diilustrasikan seperti pada Gambar 1.

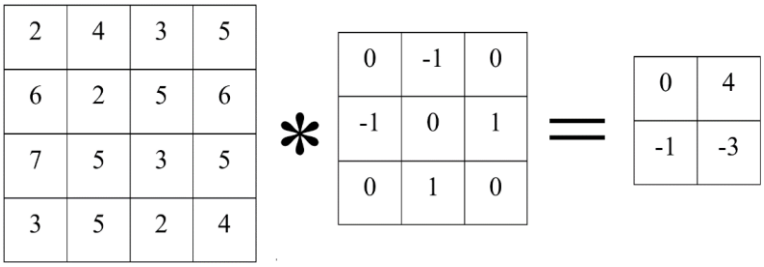


Gambar 1. Arsitektur *Faster* R-CNN.

Pada CNN sendiri terdapat 3 komponen utama, yaitu *convolutional layer*, *pooling layer*, dan *fully-connected layer* [6]. Beberapa *layer* tersebut memiliki peran yang berbeda-beda. *Convolutional layer* merupakan *layer* utama pada CNN. *Layer* ini melakukan operasi konvolusi dengan matriks *kernel* atau matriks *filter* untuk mengekstraksi fitur dari citra *input*, salah contoh nilai fitur dari sebuah citra tersebut yaitu *edge detection*, *corner detction*, dan sebagainya. Jumlah dan ukuran *filter* dapat berubah – ubah. Nilai matriks *filter* dapat dibuat acak agar hasil konvolusi dapat bervariasi. CNN juga dapat meningkatkan performa dari proses *tracking* objek [10]. Secara umum operasi konvolusi dapat dihitung dengan persamaan

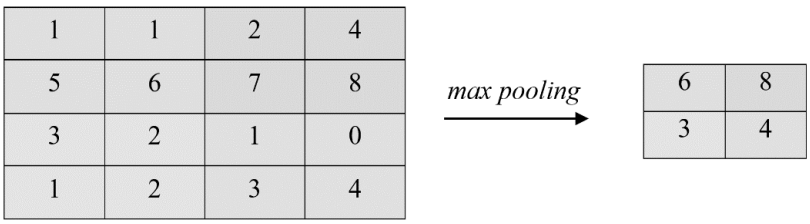
$$h(x) = f(x) * g(x), \tag{1}$$

dengan f(x) dan g(x) merepresentasikan citra *input* dan *filter* atau *kernel windows*. Hasil konvolusi pada citra dapat berbeda – beda tergantung dengan *padding* dan *stride* yang digunakan. Fitur dari CNN juga dapat meningkatkan visual *tracking* [11] [12]. Operasi konvolusi dapat dilihat pada Gambar 2.



Gambar 2. Operasi konvolusi.

Pooling layer pada CNN untuk melakukan *subsampling* atau pengurangan ukuran matriks dengan menggunakan operasi *pooling*. *Pooling layer* biasanya dilakukan setelah *convolutional layer*. Pada penelitian ini digunakan operasi *max pooling*, dimana nilai yang diambil analah nilai maksimal atau mengambil fitur terkuat dari nilai fitur citra. Contoh penerapan *max pooling* dapat dilihat pada Gambar 3.



Gambar 3. *Max pooling.*

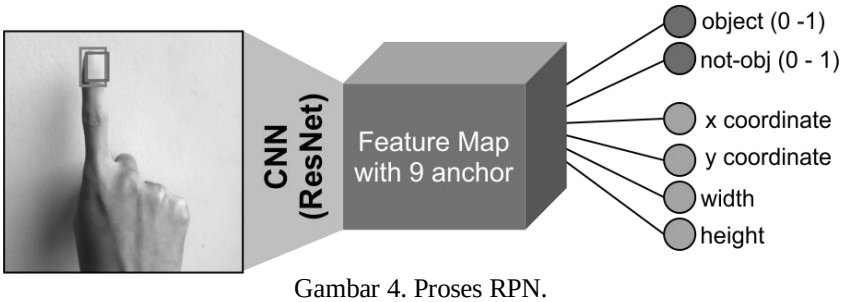
Fully-Connected Layer (FC-*layer*) adalah lapisan dimana semua *neuron* aktivasi dari *layer* sebelumnya terhubung semua dengan neuron di *layer* selanjutnya seperti halnya jaringan saraf otak yang terdapat pada manusia. FC-*layer* biasanya dilakukan setelah *convolutional layer* dan *pooling layer*. *Input* untuk FC-*layer* berupa *feature map* yang dihasilkan dari *convolutional layer* dan *pooling layer*. Nilai *output* dari FC-*layer* bersifat sementara dan memiliki perilaku yang stokastik / tidak dapat ditentukan. Secara umum, keluaran pada FC-*layer* dapat ditentukan dengan persamaan

$$h(x) = g\left(b + \sum_i w_i x_i\right), \tag{2}$$

dengan w_i, x_i, b, dan g didefinisikan sebagai nilai masukan / nilai fitur, nilai bobot koneksi *neuron*, *bias*, dan fungsi aktivasi. Fungsi dari FC-*layer* pada CNN adalah sebagai *classifier* untuk melakukan serangkaian proses yang akhirnya memberikan *probabilitas* untuk *multiclass classification output*.

Metode terpenting dari *Faster* R-CNN sendiri adalah RPN, dimana RPN adalah sebuah *neural network* yang menggantikan peran *selective search* pada metode sebelumnya [7], yang berguna untuk mengajukan *region*. RPN menghasilkan beberapa *bounding box*, setiap *box* memiliki 2 nilai probabilitas apakah pada lokasi tersebut terdapat objek atau tidak. Untuk mendapatkan nilai probabilitas tersebut RPN menggunakan metode *anchor* [4]. *Anchor* adalah sebuah area yang berfungsi untuk menilai apakah pada daerah tersebut ada objek atau tidak dengan cara menghitung nilai *Intersect of Union* (IoU). Proses RPN dapat diilustrasikan seperti pada Gambar 4.

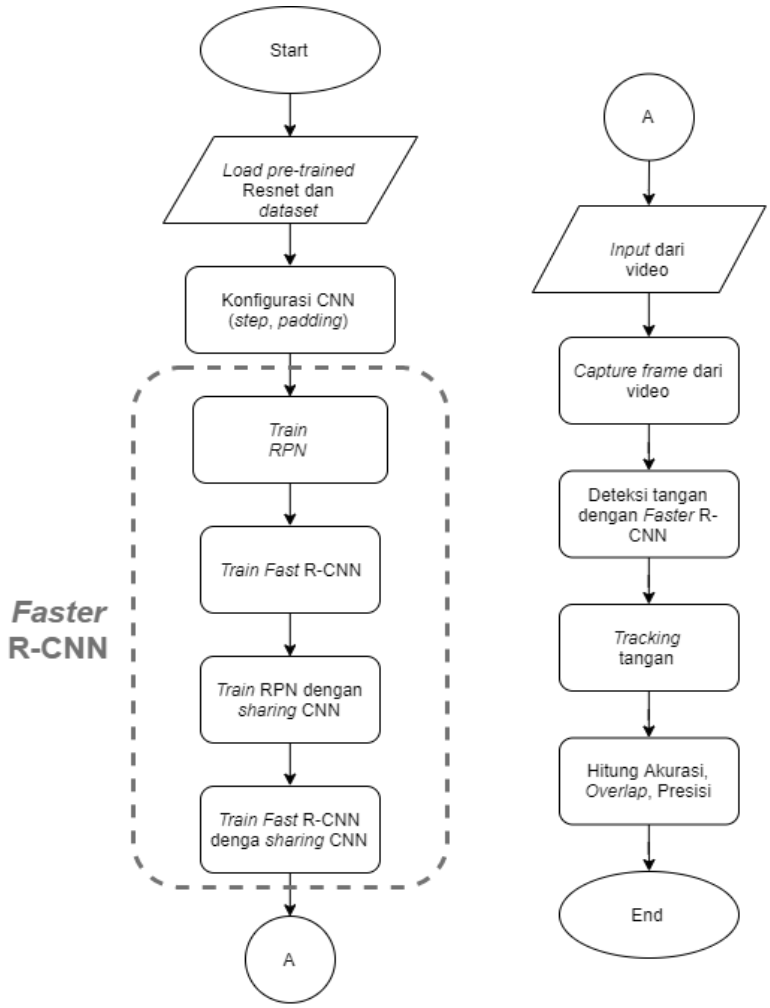
DOI: http://dx.doi.org/10.22441/fifo.2020.v12i2.002



Gambar 4. Proses RPN.

RPN dapat mendeteksi beberapa ukuran objek yang muncul pada sebuah citra dengan beberapa ukuran rasio dan skala. Skala 128, 256, dan 512 menggunakan rasio 1:1, 1:2, dan 2:1. Jadi pada RPN terdapat 9 tipe *anchor* secara keseluruhan. Dengan 9 tipe *anchor* akan menghasilkan banyak sekali kemungkinan untuk menentukan pada bagian tertentu terdapat objek atau tidak, maka dari itu RPN akan mengabaikan *cross-boundary anchor* dan menerapakan *Non-Max Suppression* (NMS) untuk mendapatkan hanya satu *anchor* yang memiliki nilai IoU tertinggi pada salah satu objek.

Dalam penelitian ini dibuat sistem deteksi objek bergerak dengan data masukan berupa video. Skema dari penelitian menggunakan arsitektur ResNet 101. ResNet sendiri merupakan model yang menggunakan *deep residual learning framework*. Setiap *layer* pada *framework* ini, memiliki referensi ke *layer* sebelumnya. Hal ini menjadikan proses optimasi menjadi lebih mudah daripada *layer network* lainnya yang tidak memiliki keterhubungan. Dengan menggunakan ResNet dapat meningkatkan akurasi dibandingkan dengan *neural network* yang tidak menggunakan ResNet [8]. Alur kerja sistem dapat dilihat dapat pada Gambar 5.



Gambar 5. Diagram alir sistem.

Disini penulis menggunakan arsitektur CNN ResNet yang sudah dilatih dengan *dataset* COCO yang memiliki 80 *class*. Maka dari itu penulis menggunakan metode *Transfer Learning*, dimana metode tersebut memanfaatkan pengetahuan yang diperoleh dari proses *learning network* sebelumnya, yang dimaksud dengan pengetahuan sebelumnya adalah berupa *weight* pada setiap *node neuron* pada *network* saat klasifikasi [9].

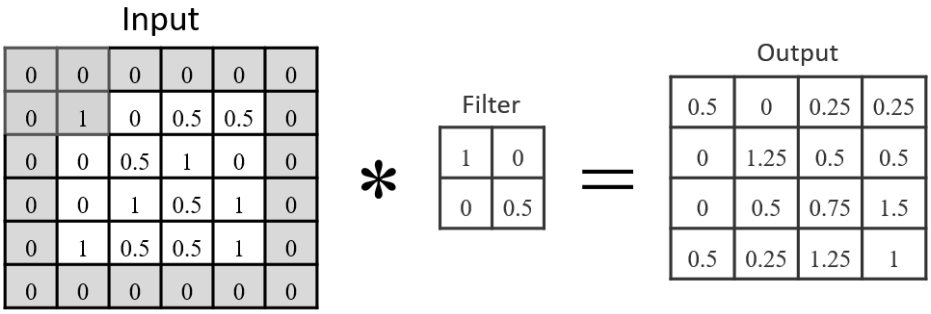
Sedangkan *dataset* yang digunakan pada penelitian ini hanya menggunakan 1 kriteria kelas. Proses *Load Pre-trained* ResNet dan *dataset* hanya dilakukan pada saat mulainya sistem. *Dataset* yang digunakan pada sistem berupa citra ujung jari. *Dataset* dibagi menjadi tiga, yaitu data latih, data validasi, dan data tes. Data validasi merupakan 50% data dari data latih. Dengan rincian seperti Tabel 1.

Tabel 1. Contoh Penulisan Nomer dan Judul Tabel

Jarak	Data Latih	Data Validasi	Data Uji	Lokasi	Kondisi Cahaya
30 cm	1800	900	600	Indoor	Terang
40 cm	1800	900	600	Indoor	Terang
50 cm	1800	900	600	Indoor	Terang
30 cm	1800	900	600	Outdoor	Terang
40 cm	1800	900	600	Outdoor	Terang
50 cm	1800	900	600	Outdoor	Terang

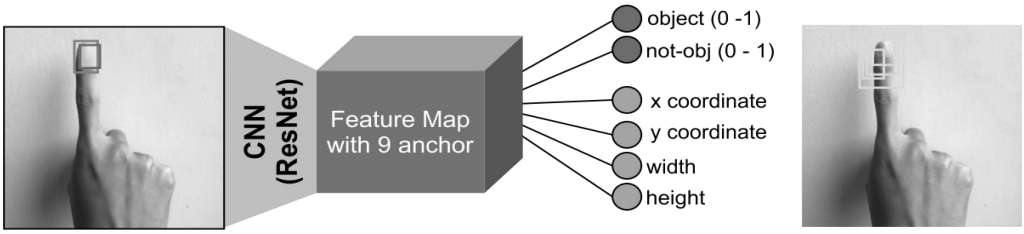
Penelitian ini menggunakan konfigurasi *filter* pada *convolutional layer* dengan 5 varian *depth filter* berukuran 3×3 dan 7×7 sesuai dengan model yang sudah ada [8]. *Convolutional layer* digunakan sebagai ekstraksi ciri / fitur pada citra masukan sedangkan untuk klasifikasinya menggunakan *fully-connected layer*. Untuk konfigurasi *padding*, digunakan *same-padding* dikarenakan pada [13] skema *same-padding* mendapatkan performansi terbaik, skema *same-padding* yang digunakan menggunakan jenis *zero-padding* dengan *step training* 25K, 50K, dan 75K.

Skema *same-padding* mengambil fitur sebanyak besar dimensi masukan. Pada umumnya jika kita memiliki masukan berukuran 4×4 maka jika menggunakan skema *same-padding* dimensi keluarannya akan sama dengan dimensi masukan yaitu 4×4. *Same-padding* diilustrasikan seperti pada Gambar 6.



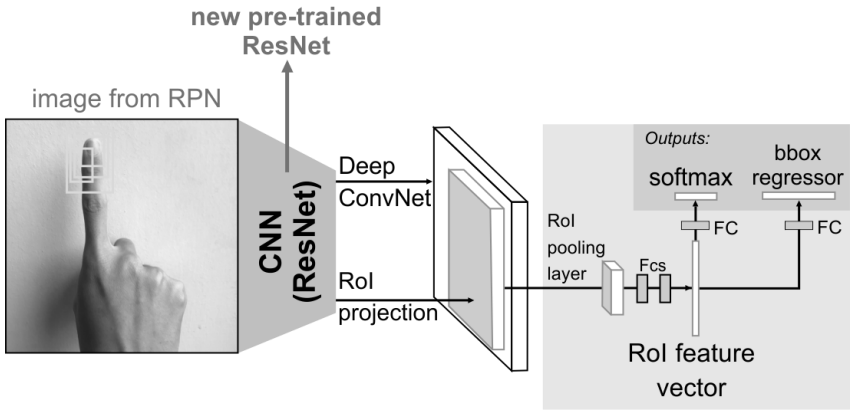
Gambar 6. *Same-padding*.

Proses *train* RPN pada Gambar 5 adalah tahap awal dalam *Faster* R-CNN. Dalam proses ini akan dilakukan *training proposer* dimana *proposer* yang digunakan yaitu RPN. Proses *train* RPN berfungsi sebagai subsistem yang berguna untuk menghasilkan *region proposer* pada setiap citra yang masuk ke dalam sistem. *Train* RPN pada sistem ini menggunakan parameter – parameter dari *pre-trained* ResNet. Keluaran pada proses ini terdapat 6 bagian yaitu probabilitas objek, probabilitas bukan objek, x koordinat, y koordinat, lebar, dan tinggi. Bagian – bagian tersebut yang akan membentuk *bounding box*. Proses *train* RPN dapat dilihat seperti Gambar 7.



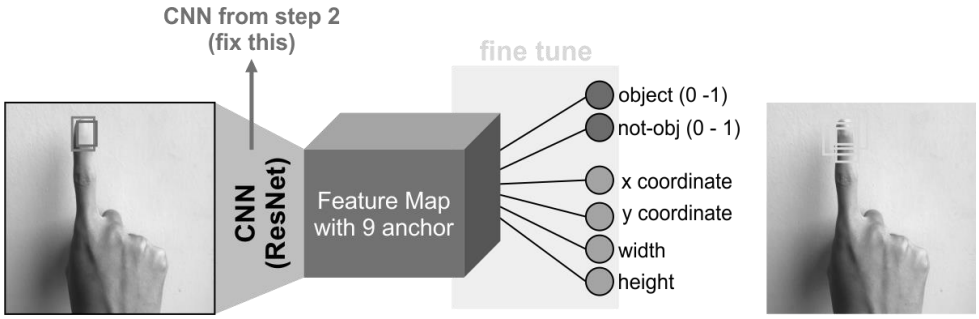
Gambar 7. *Train* RPN.

Train detector merupakan proses *train* pada *Fast* R-CNN. Citra *input training* pada proses ini menggunakan citra *output* dari proses *train* RPN sebelumnya. Akan tetapi untuk *pre-trained network* pada proses ini tidak menggunakan *network* dari proses sebelumnya, akan tetapi meggunakan *pre-trained* ResNet yang baru atau bisa disebut independen. Keluaran pada proses ini sistem dapat mendeteksi objek yang diinginkan. Proses *train detector* diilustrasikan pada Gambar 8.



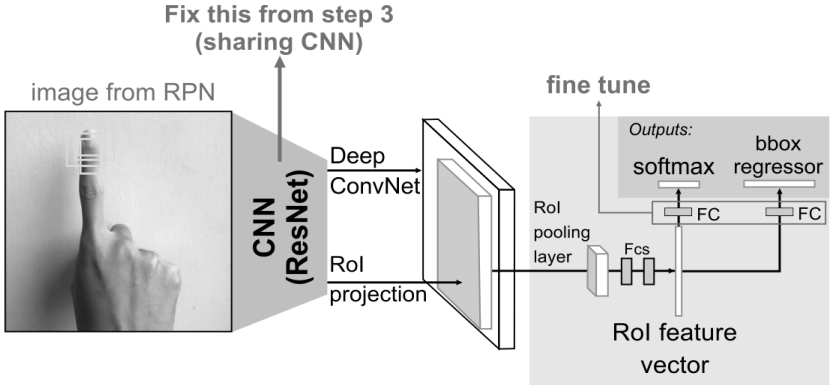
Gambar 8. *Train detector*.

Setelah melakukan *train detector* maka selanjutnya adalah *train proposer*, *train proposer* pada tahap ini sama dengan *train proposer* sebelumnya, akan tetapi CNN yang dipakai merupakan CNN dari tahap kedua yaitu CNN dari *train detector*. Dan menetapkan CNN sehingga tidak ada perubahan pada parameternya. Karena menggunakan CNN dari *train detector* maka akan dilakukan *fine-tune* di FC-*layer*. Maka nanti akan didapatkan *proposer* yang berbeda dari tahap awal. Proses *train proposer* dengan *sharing* CNN dapat dilihat pada Gambar 9.



Gambar 9. *Train proposer* dengan *sharing* CNN.

Selanjutnya adalah proses *train detector*, proses ini mirip seperti proses *train detector* sebelumnya, akan tetapi CNN yang dipakai merupakan CNN dari tahap ketiga yaitu CNN dari *train proposer*. Sama dengan tahap ketiga pada proses ini akan dilakukan penetapan CNN agar parameternya tidak berubah dan melakukan *fine-tune* di FC-*layer*. Proses *train detector* dengan *sharing* CNN diilustrasikan pada Gambar 10.



Gambar 10. *Train detector* dengan *sharing* CNN.

Setelah melakukan proses *training* pada *network* yang digunakan. Maka akan dilakukan proses deteksi pada video data uji sesuai dengan rincian pada Tabel 1. Dari video data uji yang digunakan maka akan dilakukan proses pengambilan *sampling* dari video. Proses pengambilan *frame* pada video berguna untuk mengubah *input* yang sebelumnya berupa video menjadi citra. Proses deteksi tangan merupakan bagian terpenting dalam sistem ini. Keluaran dari deteksi tangan berupa *bounding box*. Parameter dari *bounding box* nanti akan digunakan untuk proses *tracking*. Parameter yang digunakan dalam proses *tracking* adalah koordinat x dan y.

Hasil dari deteksi yang telah dilakukan akan digunakan parameter koordinat x dan y dari *bounding box*. Parameter koordinat x dan y berperan sebagai titik *centroid*. Titik *centroid* kemudian di plot dan dijadikan sebagai *track*. Setelah berhasil melakukan deteksi dan *tracking* maka selanjutnya akan dilakukan perhitungan parameter performansi. Pada penelitian ini parameter performansi yang akan diuji adalah akurasi, *Intersection of Union* (IoU) dan presisi.

Akurasi digunakan untuk mengukur kebenaran sistem dalam mendeteksi ujung jari. Akurasi dapat dihitung dengan persamaan

$$A = \frac{T}{N} \times 100\ \%, \tag{1}$$

dengan T dan N didefinisikan sebagai jumlah data yang dideteksi dengan benar dan total keseluruhan data.

IoU digunakan untuk menghitung akurasi ketepatan prediksi *bounding box* dengan *ground-truth box* dari objek. Persamaan yang digunakan adalah

$$IoU = \frac{B_{GT} \cap B_{ac}}{B_{GT} \cup B_{ac}}, \tag{2}$$

dengan B_{GT}, dan B_{ac} didefinisikan sebagai *ground-truth box* dan *boundary box actual*.

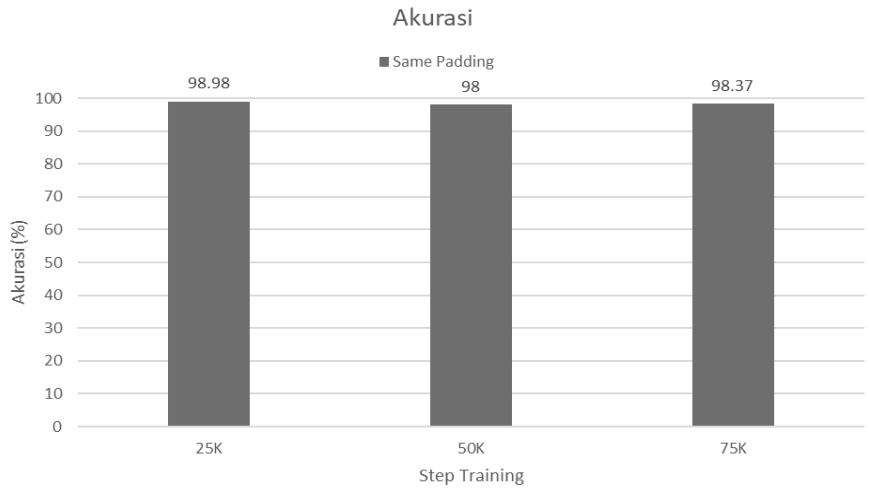
Presisi digunakan untuk menghitung jarak *centroid* / titik tengah *ground-truth box* dan *bounding box actual* dengan menerapkan rumus jarak *euclidean* dengan persamaan

$$J = \sqrt{\left(C_{x_{GT}} - C_{x_{ac}}\right)^2 + \left(C_{y_{GT}} - C_{y_{ac}}\right)^2}, \tag{3}$$

dengan C_{GT} dan C_{ac} merupakan letak koordinat x dan y untuk *centroid ground-truth box* dan *centroid bounding box actual*.

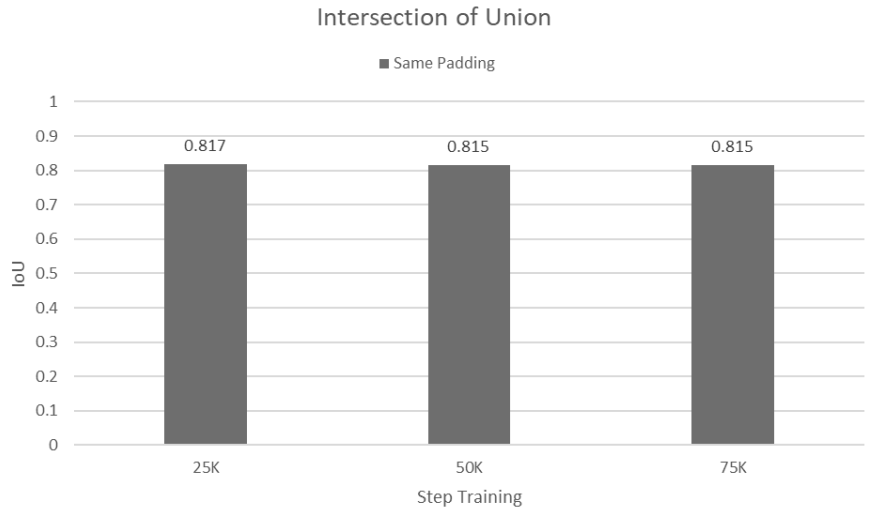
III. Hasil dan Pembahasan

Pada penelitian ini telah dilakukan pengujian terhadap model sistem dengan menggunakan video yang berdimensi 1280×720 dengan *Frame Rate* sebanyak 30fps. Dengan pembagian *indoor* dan *outdoor* untuk setiap jarak yang dipakai seperti pada Tabel 1. Model sistem yang dievaluasi merupakan percobaan dari skema *same-padding*.



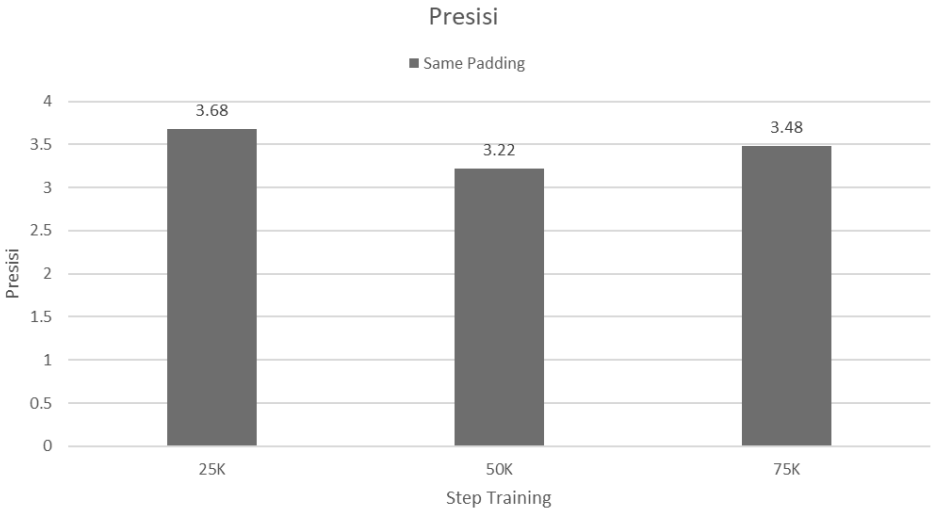
Gambar 11. Perbandingan performansi akurasi pada sampel *step* untuk *same-padding*.

Pada Gambar 11 menunjukkan bahwa semakin banyak *step training*, hasil performansi yang didapat cenderung stabil. Ketentuan parameter akurasi ditentukan dari tingkat IoU, dimana jika tingkat IoU dari setiap objek yang dideteksi ≥ 0,5 maka dapat diartikan bahwa objek tersebut terdeteksi. Dari Gambar 11 dapat diasumsikan bahwa jika tidak terjadi reduksi fitur/informasi penting yang hilang maka semakin banyak *step* performansi akurasi yang didapat cenderung stabil. Pada *step training* 25K menunjukkan performansi akurasi tertinggi yaitu 98,98%. Peningkatan dan penurunan yang terjadi pada performansi akurasi pada setiap *step training* tidak dapat ditentukan secara mutlak, hal tersebut dikarenakan *step training* memiliki sifat yang stokastik. Hasil pada *step training* 50K dapat diindikasikan sebagai *over-fitting*, akan tetapi untuk membuktikan hal tersebut diperlukan *sample step trainig* yang lebih banyak lagi. *Over-fitting* terjadi karena sistem terlalu spesifik dalam mengenali suatu objek pada citra.



Gambar 12. Perbandingan performansi IoU pada sampel *step* untuk *same-padding*.

Pada Gambar 12 ditampilkan perbandingan IoU dari masing-masing *step training* untuk skema *same-padding*. Seperti pada performansi akurasi pada Gambar 11, untuk skema *same-padding* memiliki pola yang berbeda yaitu semakin banyak *step training* hasil performansi yang didapat cenderung menurun. Hal tersebut dapat diindikasikan terjadinya *over-fitting* pada sistem, dikarenakan pada *same-padding* tidak terjadi reduksi dimensi/fitur sehingga sistem memproses fitur yang sangat banyak dan terlalu detail dalam mengklasifikasikan suatu objek pada citra. Untuk memastikan *over-fitting* diperlukan jumlah *sample step training* yang lebih banyak lagi agar hasil yang didapat lebih valid. *Step training* dengan nilai IoU terbaik tidak dapat disimpulkan bahwa skema tersebut memiliki performansi yang terbaik juga pada parameter lainnya, hal ini dikarenakan hasil IoU dan presisi pada setiap objek yang terdeteksi memiliki nilai yang berbeda-beda.



Gambar 13. Perbandingan performansi presisi pada sampel *step* untuk *same-padding*.

Gambar 13 menunjukkan grafik dari performansi presisi, dari grafik tersebut dapat dilihat bahwa skema *same-padding* memiliki presisi terbaik pada *step training* 50K yaitu 3,22. Semakin kecil presisi yang didapat maka pergerakan *pointer* pada sistem AR akan semakin mulus/lebih interaktif. Pada performansi presisi terjadi peningkatan dan penurunan pada setiap *step training*-nya, hal tersebut dikarenakan *step training* memiliki sifat yang stokastik. Hasil pada *step training* 25K dapat diasumsikan *loss* pada sistem masih cenderung besar, sedangkan pada *step training* 75K dapat diindikasikan terjadinya *over-fitting*. Untuk mebuktikan hal tersebut diperlukan *sample step training* yang lebih banyak lagi.

Gambar 14. Hasil deteksi ujung jari.

Pada Gambar 14 menunjukkan hasil dari deteksi ujung jari menggunakan model sistem dengan skema *padding* dan *step training* 25K. Hasil *confidence* yang didapat juga mencapai 99% pada ujung jari yang berhasil dideteksi. Hasil performansi akurasi pada metode *Faster* R-CNN memiliki hasil yang lebih baik dibandingkan dengan objek deteksi yang menggunakan R-CNN pada [14] dengan nilai akurasi sebesar 90%. Validasi data hasil pengujian dilakukan dengan cara membandingkan dengan hasil performansi [13], pada *paper* [13] dinyatakan bahwa pada skema *same-padding* semakin banyak *step-training* hasil yang didapatkan cenderung lebih bagus dan stabil.

IV. Kesimpulan

Setelah melakukan pengujian pada model sistem berdasarkan parameter yang telah ditentukan, maka dapat disimpulkan bahwa semakin banyak *step training* performansi yang didapat cenderung stabil seperti pada percobaan skema *same-padding*. Dari hasil dan pembahasan dapat disimpulkan bahwa model dengan skema *same-padding* pada *step training* 25K mendapatkan performansi terbaik berdasarkan prioritasi parameter untuk teknologi AR. Model dengan *step training* 50K mendapatkan nilai performansi akurasi sebesar 98%, IoU sebesar 0,815, dan presisi sebesar 3,22. Sedangkan model dengan *step training* 25K mendapatkan hasil dengan performansi cenderung buruk untuk teknologi AR diantara skema lainnya, dengan performansi akurasi 98,98%, IoU sebesar 0,817, dan presisi sebesar 3,68. Hasil yang didapat cenderung stabil dikarenakan pada skema *same-padding* tidak terjadi reduksi dimensi/informasi yang hilang dan *step training* itu sendiri memiliki sifat yang stokastik sehingga tidak dapat ditentukan secara mutlak. Model dengan *step training* 50K akan memiliki gerakan *pointer* yang lebih mulus dibandingkan dengan skema lainnya. Dari hasil yang didapat akan diterapkan pada penggunaan *notebook* sehari – hari agar menjadi lebih interaktif. Dalam pengaplikasiannya pengguna dapat menggerakkan *pointer* pada *notebook* dengan ujung jari tanpa harus berinteraksi langsung dengan *notebook*-nya.

V. Daftar Pustaka

[1] S. Malik, C. McDonald, and G. Roth. 2002. *Hand tracking for interactive pattern-based augmented reality*. in Proceedings of the 1st International Symposiumon Mixed and Augmented Reality. IEEE Computer Society.
[2] T. Lee and T. Hollerer. 2007. *Handy ar: Markerless inspection of augmented reality objects using fingertip tracking*. in 2007 11th IEEE International Symposiumon Wearable Computers. IEEE.
[3] S. S. Rao. 2010. *Sixth sense technology*. in 2010 International Conference on Communication and Computational Intelligence (INCOCCI). IEEE.
[4] S. Ren, K. He, R. Girshick, and J. Sun. 2015. *Faster r-cnn: Towards real-time object detection with region proposal networks*. In Advances in neural information processing systems.
[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik. 2014. *Rich feature hierarchiesfor accurate object detection and semantic segmentation*. in Proceedings of the IEEE conference on computer vision and pattern recognition.
[6] F-f. Li, J. Johnson, and S. Yeung. 2018. *Lecture 5 Convolutional Neural Networks*. pp. 1–78. [Online]. Available: http://cs231n.stanford.edu/slides/2018/cs231n2018lecture05.pdf
[7] R. Girshick. 2015. *Fast r-cnn*. in Proceedings of the IEEE international conferenceon computer vision.
[8] K. He, X. Zhang, S. Ren, and J. Sun. 2016. *Deep residual learning for image recog-nition.* in Proceedings of the IEEE conference on computer vision and patternrecognition.
[9] S. J. Pan and Q. Yang. 2009. *A survey on transfer learning*. IEEE Transactions on knowledge and data engineering.
[10] S. A. Wibowo, H. Lee, E. K. Kim, and S. Kim. 2017. *Convolutional shallow features for performance improvement of histogram of oriented gradients in visual object tracking*. Mathematical Problems in Engineering.
[11] S. A. Wibowo, H. Lee, E. K. Kim, and S. Kim. 2018. *Collaborative learning based on convolutional features and correlation filter for visual tracking*. International Journal of Control, Automation and Systems.
[12] S. A. Wibowo, H. Lee, E. K. Kim, and S. Kim. 2017. *Visual tracking based on complementary learners with distractor handling*. Mathematical Problems in Engineering.
[13] A. Wiranata, S. A. Wibowo, R. Patmasari, R. Rahmania, and R. Mayasari. 2018. *Investigation of padding schemes for faster r-cnn on vehi-cle detection*. International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC). IEEE.
[14] A. Y. Nasirudin, S .A. Wibowo, and R. Purnamasari. 2018. *Performance analysis on fine-tuned region-based cnn for object recognition*. Symposium of Future Telecommunication and Technologies (SOFTT).