BAB IV
HASIL DAN PEMBAHASAN

4.1 Pendahuluan

Pada pendahuluan bab ini, akan dibahas hasil pengolahan berbagai citra dari dataset retina menggunakan metode yang telah dijelaskan pada Bab III. Dataset yang digunakan dalam penelitian ini adalah dataset DRIVE dan STARE. Tahapan yang dilakukan di Bab IV meliputi augmentasi dataset, *preprocessing* (peningkatan citra dengan metode *Grayscale*, teknik penghilangan *noise* menggunakan *Gaussian blur* dan CLAHE), segmentasi pembuluh darah dengan arsitektur ResVNet, serta evaluasi dan hasil pengukuran kinerja matriks.

4.2 Augmentasi Data

Sebelum memasuki tahap *preprocessing* dengan menggunakan metode *Grayscale, Gaussian Blur*, dan CLAHE, dataset harus melalui tahap augmentasi data terlebih dahulu. Augmentasi data adalah teknik yang digunakan untuk memodifikasi citra tanpa menghilangkan inti atau esensi dari citra tersebut. Augmentasi yang digunakan dalam penelitian ini meliputi *flip* horizontal, *flip* vertical, dan *grid distortion*. pada dataset DRIVE bagian *training* dilakukan proses augmentasi *flip* horizontal, *flip vertikal*, dan *grid distortion* sehingga setalah dilakukan proses augmentasi tersebut jumlah dataset *training* DRIVE menjadi 80 dan dataset *testing* tetap 20% karena tidak dilakukan proses augmentasi. Pada dataset STARE, dilakukan proses augmentasi menggunakan *flip* horizontal, *flip vertikal*, dan *grid distortion* sehingga dataset yang digunakan menjadi 16 testing dan 64 training. Setelah dilakukan augmentasi data menggunakan *flip* horizontal, *flip* vertical, dan *grid distortion* pada kedua dataset (DRIVE dan STARE), jumlah dataset yang digunakan meningkat. Persentase penggunaan dataset

menjadi 80% untuk *training* dan 20% untuk test. Detail jumlah dataset yang digunakan setelah proses augmentasi akan ditampilkan pada tabel 4.1.

Tabel 4.1. Jumlah Dataset Setelah Proses Augmentasi Data

No.	Dataset	Jumlah dataset sebelum augmentasi		Jumlah dataset setelah augmentasi	
		Test	Train	Test	Train
1.	DRIVE	20	20	20	80
2.	STARE	4	16	16	64

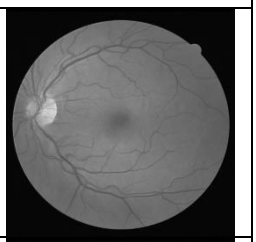
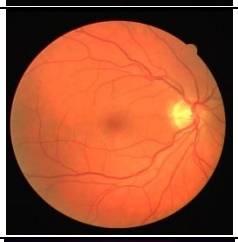
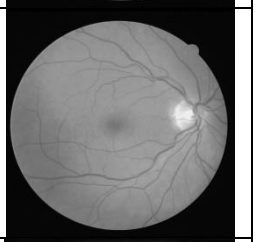
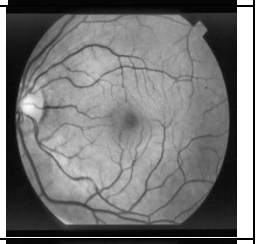
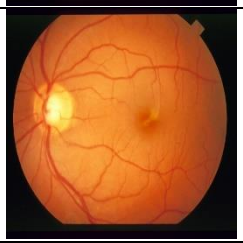
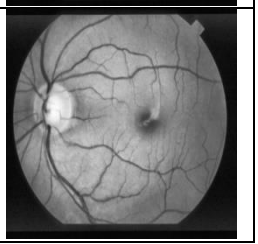
Pada tabel 4.1 terdapat jumlah dataset sebelum dan sesudah dilakukan augmentasi. Pada dataset DRIVE terdapat data *test* dan *train*, pada data *test* tidak dilakukan augmentasi karena keterbatasan penyimpanan sedangkan data train dilakukan proses augmentasi menggunakan *flip horizontal*, *flip vertical*, dan *Grid Distortion.* Dataset Stare terdapat data test dan data train, semua data *test* dan *train* pada stare dilakukan proses augmentasi menggunakan *flip horizontal, flip vertical*, dan *Grid Distortion*

4.3 Prepocessing

4.3.1 Grayscale

Setelah konversi ke citra *grayscale*, langkah selanjutnya adalah melakukan enhancement citra dengan menggunakan *Gaussian blur*. *Gaussian blur* digunakan untuk menghilangkan *noise* dan detail kecil dalam citra dengan menghaluskan perubahan intensitas di seluruh citra. Proses *Gaussian blur* dilakukan dengan menerapkan kernel Gaussian pada setiap piksel dalam citra. Untuk setiap piksel, nilai baru dihitung sebagai rata-rata tertimbang dari nilai-nilai piksel dalam jendela 3x3 dengan bobot yang diberikan oleh kernel Gaussian. Contoh hasil *preprocessing* menggunakan metode *Gaussian blur* yang diterapkan pada dataset DRIVE dan STARE dengan ukuran kernel 3x3 akan ditampilkan pada tabel 4.2.

Tabel 4.2. Tabel Preprocessing Grayscale

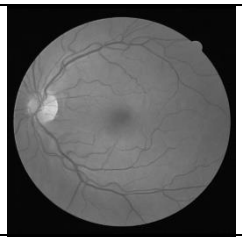
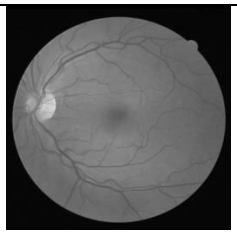
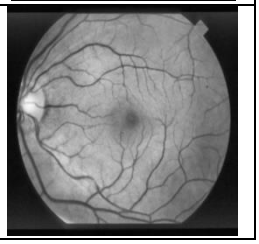
No.	Dataset	Nama File	Citra Asli	Citra Hasil Grayscale
1.	DRIVE	01_test		
		02_test		
2.	STARE	im0077		
		im0235		

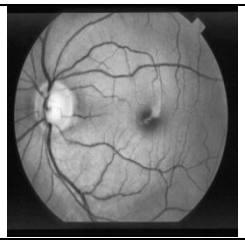
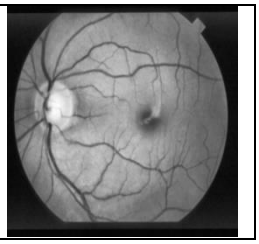
Pada tabel 4.2, terdapat perbandingan antara citra asli dan citra setelah dikonversi menjadi *grayscale*. Citra asli menunjukkan variasi warna yang kaya dan kontras yang mencolok, memberikan tampilan yang menarik dan informasi tambahan mengenai objek. Ketika citra asli diubah menjadi *grayscale*, informasi warna akan hilang, mengubah gambar menjadi skala abu-abu dan mengurangi detail warna. Namun, citra *grayscale* memungkinkan fokus yang lebih baik pada struktur dan bentuk objek, memudahkan analisis tekstur dan deteksi tepi dengan meningkatkan kontras antara berbagai tingkat intensitas cahaya.

4.3.2 Gaussian Blur

Setelah dilakukan konversi kecitra *grayscale*, selanjutnya akan dilakukan proses *enhancement* citra dengan menggunakn *gaussian blur*. *Gaussian blur* digunakan untuk menghilangkan *noise* dan detail kecil yang terdapat dalam citra tersebut dengan cara menghaluskan perubahan I tensitas diseluruh citra. Proses *gaussian blur* dilakukan dengan menerapkan kernel gaussian pada setiap pixel dalam citra. untuk setiap pixel, nilai baru dihitung sebagai rata-rata tertimbang dari nilai-nilai pixel dalam jendela 3x3 dengan bobot yang diberikan oleh kernel gaussian. Berikut ini contoh hasil *preprocessing* menggunakan metode *gaussian blur* yang dilakukan pada dataset DRIVE dan STARE dengan kernel size yang digunakan ialah 3x3 akan ditampilkan pada tabel 4.3.

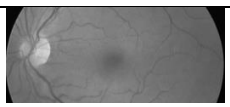
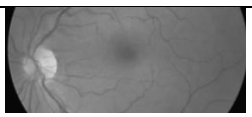
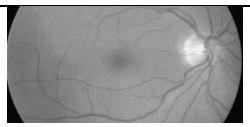
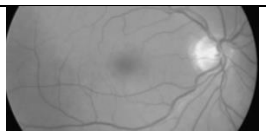
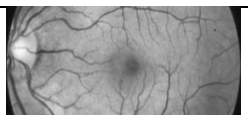
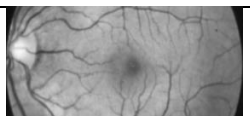
Tabel 4.3. Tabel Gaussian Blur

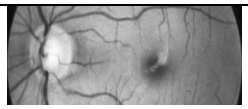
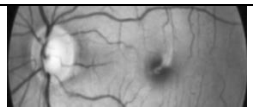
No.	Dataset	Nama File	Citra Hasil Grayscale	Citra Hasil Gaussian Blur
1.	DRIVE	01_test		
		02_test		
2.	STARE	im0077		

P a d a		im0235			

tabel 4.3 terdapat citra setelah dilakukan proses *grayscale* dan citra *grayscale* setelah dilakukan *Gaussian Blur*. Pada citra *grayscale* sebelum dilakukan *Gaussian Blur*, masih terdapat *noise* dan detail kecil yang sangat jelas. Setelah dilakukan *Gaussian Blur*, detail kecil dalam citra menjadi lebih halus dengan kontur objek dan tepi yang lebih lembut, serta *noise* dalam citra berkurang secara signifikan. Detail gambar sebelum dan sesudah dilakukan proses enhancement menggunakan *gaussian blur* akan ditampilkan pada tabel 4.4.

Tabel 4.4. Detail Gambar Gaussian Blur

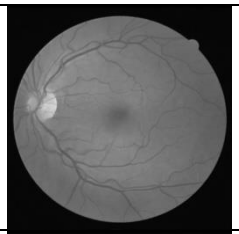
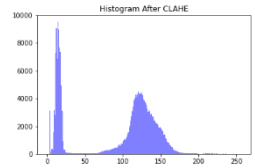
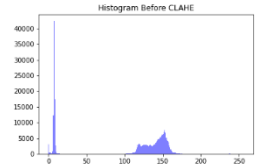
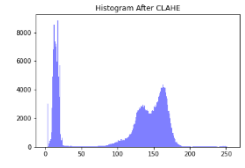
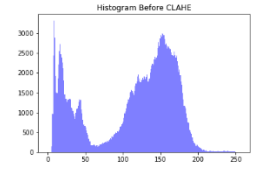
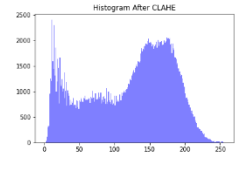
No.	Dataset	Nama File	Citra Hasil Grayscale	Detail Gambar Hasil Grayscale	Citra Hasil Gaussian Blur	Detail Gambar Hasil Gaussian Blur
1.	DRIVE	01_test		Mean Pixel Value: 98.66472625732422 Standard Deviation: 63.979541572067134		Mean Pixel Value: 98.69657516479492 Standard Deviation: 63.8231056145968
		02_test		Mean Pixel Value: 81.95447540283203 Standard Deviation: 52.701358396531546		Mean Pixel Value: 81.98560333251953 Standard Deviation: 52.536845825790486
2.	STARE	im0077		Mean Pixel Value: 111.47238159179688 Standard Deviation: 58.16996532215282		Mean Pixel Value: 111.50434112548828 Standard Deviation: 57.96128016658749

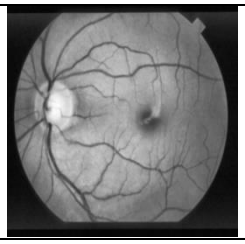
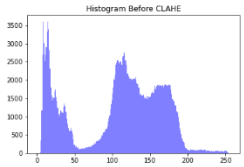
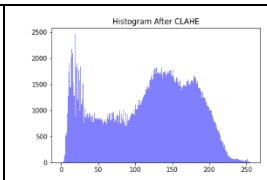
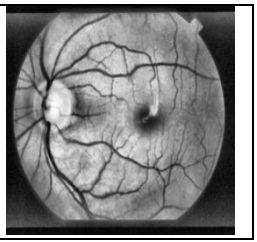
		im0235		Mean Pixel Value: 106.32274627685547 Standard Deviation: 58.72298637333581		Mean Pixel Value: 106.35441589355469 Standard Deviation: 58.55567480119016

4.3.3 CLAHE

Setelah dilakukan *gaussian blur*, tahap terakhir dalam proses *preprocessing* penelitian ini ialah menggunakan metode CLAHE. Metode CLAHE digunakan untuk memperbaiki kualitas citra yang rendah serta menghilangkan *noise* yang terdapat dalam citra. Pada tabel 4.5 akan menampilkan contoh histogram dari citra sebelum menggunakan CLAHE dan sesudah menggunakan CLAHE, serta hasil citra dari menggunakan metode CLAHE.

Tabel 4.5. Contoh Hasil Histogram Dan Citra dari Metode CLAHE

No.	Dataset	Nama File	Citra Hasil Gaussian Blur	Histogram *gaussian blur*	Histogram *clahe*	Citra hasil *clahe*
1.	DRIVE	01_test				
		02_test				
2.	STARE	im0077				

		im0235				

Pada tabel 4.5, terlihat perbedaan antara histogram sebelum dan sesudah penerapan metode CLAHE (*Contrast Limited Adaptive Histogram Equalization*). Histogram setelah penerapan CLAHE menunjukkan distribusi kontras yang lebih merata, dengan peningkatan kontras di berbagai tingkat intensitas. Puncak-puncak pada histogram menjadi lebih beragam, mencerminkan peningkatan kontras lokal di bagian gambar. Metode CLAHE juga meningkatkan detail gambar dengan meningkatkan frekuensi pada berbagai tingkat intensitas dalam histogram. Setelah menerapkan CLAHE pada gambar, terlihat peningkatan kontras lokal dan penegasan detail, menghasilkan gambar yang lebih jelas dengan detail.

Untuk membandingkan kualitas citra sebelum dan sesudah dilakukan proses preprocessing, akan dilakukan pengukuran menggunakan PSNR (Peak Signal-to-Noise Ratio) dan SSIM (Structural Similarity Index Measure). PSNR adalah ukuran yang digunakan untuk mengevaluasi kualitas citra hasil preprocessing dibandingkan dengan citra asli. PSNR dinyatakan dalam satuan desibel (dB). Semakin tinggi nilai PSNR, semakin baik kualitas citra hasil preprocessing, yang menunjukkan bahwa perubahan pada citra lebih kecil dan tetap mendekati citra asli. SSIM adalah metode yang digunakan untuk mengukur kesamaan struktur antara dua citra. SSIM memiliki rentang nilai dari 0 hingga 1, di mana nilai 1 menunjukkan kesamaan sempurna. Semakin tinggi nilai SSIM, semakin mirip struktur dan kualitas citra hasil preprocessing dengan citra asli. Hasil pengukuran kualitas citra sebelum dan sesudah dilakukan preprocessing pada dataset DRIVE akan disajikan dalam **Tabel 4.6**.

Tabel 4.6 Hasil Pengukuran Kualitas Citra Sebelum Dan Sesudah Dilakukan Preprocessing Pada Dataset DRIVE

No.	Nama File	PSNR (dB)	SSIM
1.	01_test	26.05	0.8534
2.	02_test	27.21	0.8595
3.	03_test	24.30	0.8605
4.	04_test	26.09	0.8618
5.	05_test	26.42	0.8640
6.	06_test	25.89	0.8649
7.	07_test	25.99	0.8614
8.	08_test	27.00	0.8731
9.	09_test	28.10	0.8673
10.	10_test	25.39	0.8641
11.	11_test	26.45	0.8649
12.	12_test	26.72	0.8607
13.	13_test	26.75	0.8636
14.	14_test	26.61	0.8543
15.	15_test	25.01	0.8585
16.	16_test	27.02	0.8613
17.	17_test	27.87	0.8642
18.	18_test	27.88	0.8644
19.	19_test	23.69	0.8559
20.	20_test	26.67	0.8687

Tabel 4.6 menunjukkan hasil pengukuran PSNR dan SSIM pada dataset DRIVE setelah dilakukan proses enhancement menggunakan metode grayscale, Gaussian blur, dan CLAHE. Nilai PSNR pada dataset DRIVE berkisar antara 23.69 dB hingga 28.10 dB, dengan rata-rata nilai di atas 20 dB, yang menunjukkan bahwa citra hasil preprocessing memiliki kualitas yang cukup baik, dan perubahan yang terjadi setelah proses enhancement masih dalam batas yang wajar. Semakin tinggi nilai PSNR, semakin kecil tingkat distorsi atau perbedaan antara citra asli dan citra

hasil preprocessing. Selain itu, nilai SSIM pada dataset DRIVE menunjukkan angka di atas 80% (0.8), berkisar antara 0.8534 hingga 0.8731, yang mencerminkan bahwa struktur citra hasil preprocessing masih sangat mirip dengan citra aslinya, sehingga perubahan yang terjadi selama proses enhancement tidak merusak detail penting dalam citra. Secara keseluruhan, hasil ini menunjukkan bahwa metode preprocessing yang digunakan efektif dalam meningkatkan kualitas citra tanpa menghilangkan informasi penting. Nilai PSNR dan SSIM yang konsisten tinggi pada dataset ini mengindikasikan bahwa citra hasil preprocessing dapat digunakan untuk analisis lebih lanjut dengan tingkat akurasi yang baik.

Selain hasil pengukuran kualitas citra sebelum dan sesudah dilakukan preprocessing pada dataset DRIVE, terdapat juga hasil pengukuran kualitas citra sebelum dan sesudah dilakukan preprocessing pada dataset STARE yang akan disajikan dalam **Tabel 4.7**.

Tabel 4.7 Hasil Pengukuran Kualitas Citra Sebelum Dan Sesudah Dilakukan Preprocessing Pada Dataset STARE

No.	Nama File	PSNR (dB)	SSIM
1.	im00040	24.67	0.9191
2.	im00041	24.58	0.9189
3.	im00042	24.68	0.9191
4.	im00043	24.79	0.9158
5.	im00770	24.82	0.8796
6.	im00771	24.82	0.8797
7.	im00772	24.80	0.8795
8.	im00773	24.75	0.8821
9.	im01390	22.50	0.8700
10.	im01391	22.48	0.8701
11.	im01392	22.49	0.8699
12.	im01393	22.46	0.8728
13.	im02350	24.81	0.8726

14.	im02351	24.78	0.8727
15.	im02352	24.81	0.8725
16.	im02353	24.76	0.8707

Tabel 4.7 menampilkan hasil pengukuran PSNR dan SSIM pada dataset STARE setelah dilakukan proses enhancement menggunakan metode grayscale, Gaussian blur, dan CLAHE. Nilai PSNR berkisar antara 22.46 dB hingga 24.82 dB, dengan nilai tertinggi pada citra im00770 dan im00771 (24.82 dB) serta nilai terendah pada citra im01393 (22.46 dB). Nilai PSNR yang konsisten di atas 20 dB menunjukkan bahwa citra hasil preprocessing memiliki kualitas yang baik, di mana perubahan yang terjadi masih dalam batas wajar dan tidak menimbulkan distorsi yang signifikan. Sementara itu, nilai SSIM berada pada rentang 0.8699 hingga 0.9191, dengan nilai tertinggi pada citra im00040 dan im00042 (0.9191). SSIM yang konsisten di atas 0.8 mencerminkan bahwa struktur dan detail penting pada citra tetap terjaga, meskipun telah dilakukan proses enhancement. Secara keseluruhan, hasil ini mengindikasikan bahwa metode preprocessing yang digunakan efektif dalam meningkatkan kualitas visual citra tanpa menghilangkan informasi penting, sehingga citra hasil preprocessing layak digunakan untuk analisis lebih lanjut.

4.4 Segmentasi Pembuluh Darah menggunakan Arsitektur ResVNet

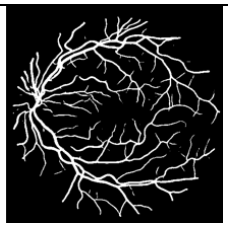
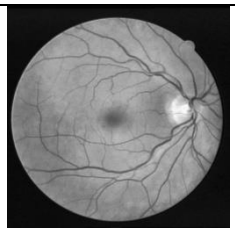
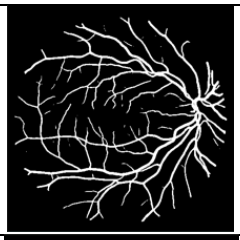
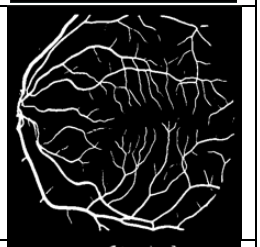
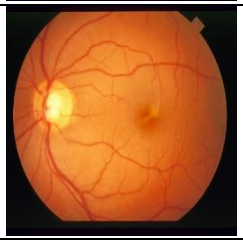
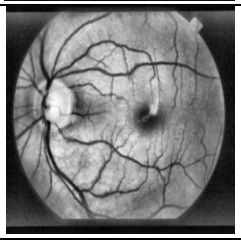
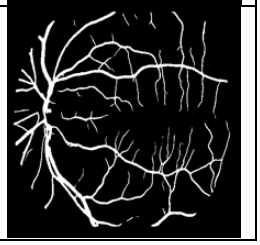
Setelah melalui tahap augmentasi dataset yang menghasilkan distribusi citra sebesar 80% untuk train dan 20% untuk test, dilanjutkan dengan enhancement citra pada tahap *preprocessing* menggunakan metode *grayscale, Gaussian blur,* dan CLAHE. Tahap berikutnya adalah segmentasi pembuluh darah menggunakan arsitektur ResVNet. Arsitektur ResVNet adalah modifikasi dari arsitektur U-Net, yang biasanya terdiri dari tiga bagian: *encoder*, *bridge*, dan *decoder*. Pada ResVNet, bagian *bridge* dihilangkan, sehingga hanya tersisa *encoder* dan *decoder*, yang kemudian disebut arsitektur V-Net. Selanjutnya, arsitektur V-Net dimodifikasi dengan mengganti *encoder* dengan arsitektur ResNet,

sehingga dinamakan arsitektur ResVNet. Gambar arsitektur ResVNet yang digunakan dapat dilihat pada Gambar 3.8

Pada proses *training* arsitektur ResVNet, bagian *encoder* menggunakan arsitektur ResNet, di mana setiap proses terdiri dari dua tahap *konvolusi*. Proses *konvolusi* pertama mencakup *konvolusi*, batch normalization, dan fungsi aktivasi ReLU, sedangkan proses *konvolusi* kedua mencakup *konvolusi* dan batch normalization. Selanjutnya, *shortcut connection* diterapkan, yang kemudian ditambahkan ke hasil *konvolusi* kedua, dan diterapkan fungsi aktivasi ReLU. Jumlah saluran pada layer *encoder* pertama berubah dari 1 menjadi 64, dari 64 menjadi 128 pada layer *encoder* kedua, dan dari 128 menjadi 256 pada layer *encoder* ketiga. Pada bagian *decoder*, arsitektur U-Net digunakan, di mana setiap prosesnya terdiri dari dua lapisan *konvolusi*onal, batch *normalization*, dan satu lapisan *dropout*. Jumlah saluran adalah 256 pada layer *decoder* pertama, kemudian berubah dari 256 menjadi 128 pada layer *decoder* kedua, dan dari 128 menjadi 64 pada layer *decoder* ketiga. Pada layer *decoder* terdapat proses *upsampling* untuk mengembalikan citra ke ukuran aslinya. Fungsi aktivasi yang digunakan adalah ReLU, dan dropout diterapkan pada setiap layer untuk menghindari *overfitting* dengan nilai 0.2 (20%). Ketika jumlah saluran adalah 64 pada layer *decoder* ketiga, jumlah saluran berubah menjadi 2 sebagai *output* segmentasi map dengan fungsi aktivasi *sigmoid.*

Hyperparameter yang digunakan pada proses *training* dengan metode ResVNet meliputi batch size sebesar 2, learning rate sebesar 0.0001 (1e-4), *epoch* sebanyak 30, dan loss function DiceBCELoss. Setelah melalui proses *training*, hasil pengujian citra untuk segmentasi pembuluh darah pada dataset DRIVE dan STARE diperoleh. Contoh hasil segmentasi pembuluh darah menggunakan metode ResVNet dengan terlebih dahulu melakukan tahapan preprocessing akan ditampilkan pada Tabel 4.8.

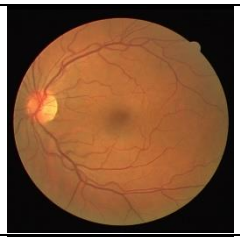
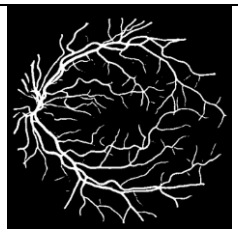
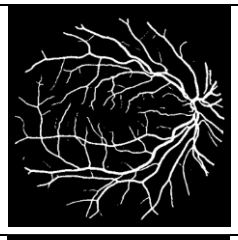
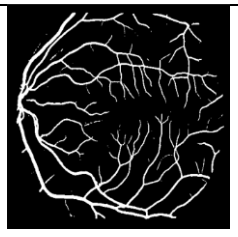
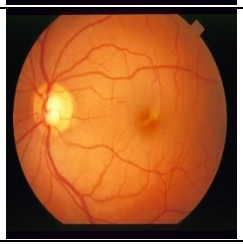
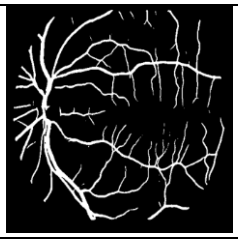
Tabel 4.8. Contoh Hasil Pengujian Segmentasi Pembuluh Darah Menggunakan Arsitektur ResVNet Dengan Menggunakan Tahapan Preprocessing Terlebih Dahulu

No.	Dataset	Nama File	Citra Asli	Citra Hasil Preprocessing	Segmentasi Menggunakan Arsitektur ResVNet
1.	DRIVE	01_test			
		02_test			
2.	STARE	im0077			
		im0235			

Selain hasil segmentasi pembuluh darah menggunakan metode ResVNet pada citra retina dengan melewati tahapan preprocessing (*grayscale*, *gaussian blur*, dan *clahe*), terdapat juga hasil segmentasi

pembuluh darah menggunakan metode ResVNet tanpa melewari tahapan preprocessing akan ditampilkan pada tabel 4.9.

Tabel 4.9. Contoh Hasil Pengujian Segmentasi Pembuluh Darah Menggunakan Arsitektur ResVNet Tanpa Menggunakan Tahapan Preprocessing

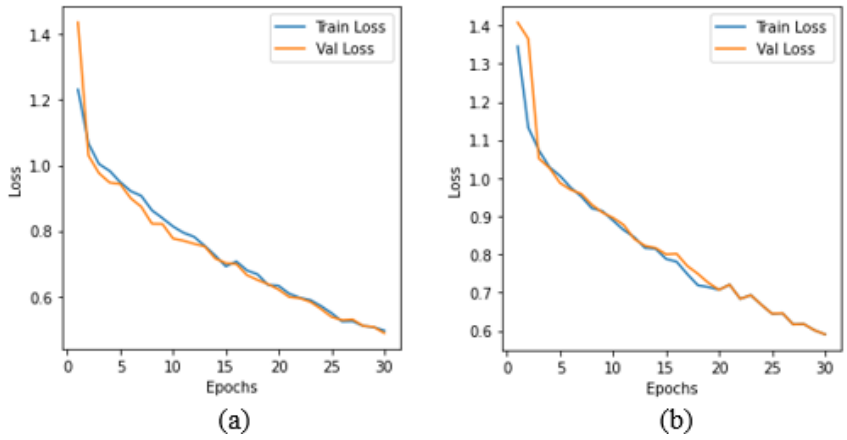
No.	Dataset	Nama File	Citra Asli	Segmentasi Menggunakan Arsitektur ResVNet
1.	DRIVE	01_test		
		02_test		
2.	STARE	im0077		
		im0235		

4.5 Evaluasi dan Hasil Pengukuran Kinerja Matriks

4.5.1 Training

Pada tahap pelatihan, dilakukan untuk menghasilkan bobot yang akan digunakan pada tahap pengujian. Penelitian ini menggunakan 30 *epoch* dan *batch size* sebesar 2. Proses segmentasi dilakukan dengan

menggunakan arsitektur ResVNet. Pada arsitektur ResVNet yang digunakan, langkah pertama adalah menginisialisasi nilai bobot untuk setiap *epoch*. Untuk setiap *epoch*, nilai loss (error) harus dihitung. Bobot harus disimpan jika nilai error pada data validasi lebih kecil dari *epoch* sebelumnya, dan sebaliknya. Bobot juga harus diperbarui untuk *epoch* berikutnya guna menghasilkan bobot terbaik. Grafik *training* loss dan *validation* loss dapat dilihat pada gambar 3.9.



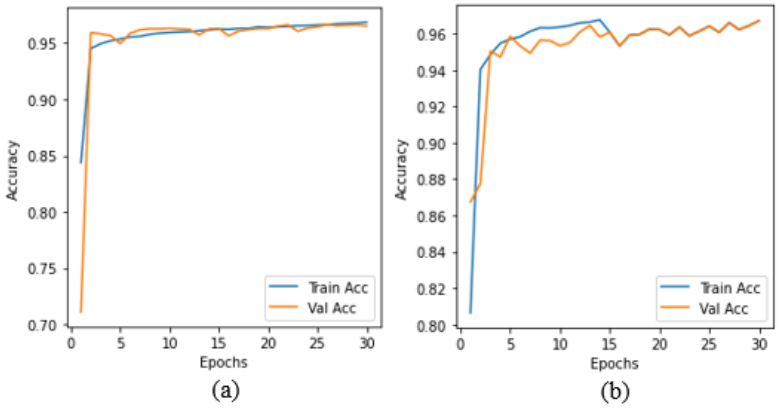
Gambar 3.9. Grafik Training Loss dan Validation Loss (a) DRIVE dan (b) STARE

Pada gambar 3.9. (a), terdapat grafik *training* loss dan *validation loss* pada arsitektur ResVNet pada dataset DRIVE, dapat dilihat grafik *loss* proses *training* mengalami penurunan pada setiap *epoch*. Nilai *train_loss* yang diperoleh pada *epoch* pertama sebesar 1.2, lalu pada *epoch* selanjutnya nilai *train_loss* terus menurun hingga mencapai 0.4 pada *epoch* ke-30. Sedangkan, nilai *val_loss* (loss validasi) juga menunjukkan tren penurunan, dengan nilai awal sebesar 1.4 pada *epoch* pertama, yang kemudian menurun secara bertahap hingga mencapai 0.4 pada *epoch* ke-30. Meskipun ada beberapa fluktuasi kecil, nilai *val_loss* tetap menunjukkan penurunan yang konsisten seiring dengan bertambahnya jumlah *epoch*. Dari Grafik *training* loss dan *validation loss* pada dataset DRIVE ini menunjukkan bahwa arsitektur ResVNet yang dilatih berhasil mempelajari pola data dengan baik, yang ditunjukkan oleh penurunan nilai

train_loss dan *val_loss*. Hasil ini mengindikasikan bahwa arsitektur ResVNet tidak mengalami masalah *overfitting*, karena penurunan *val_loss* sejalan dengan penurunan *train_loss*.

Grafik *training* loss dan *validation* loss untuk dataset STARE, yang ditampilkan pada gambar 3.9. (b), dapat dilihat grafik *loss* proses *training* mengalami penurunan pada setiap *epoch*. Nilai *train_loss* yang diperoleh pada *epoch* pertama sebesar 1.3, lalu pada *epoch* selanjutnya nilai *train_loss* terus menurun hingga mencapai 0.5 pada *epoch* ke-30. Sedangkan, nilai *val_loss* (loss validasi) juga menunjukkan tren penurunan, dengan nilai awal sebesar 1.4 pada *epoch* pertama, yang kemudian menurun secara bertahap hingga mencapai 0.5 pada *epoch* ke-30. Menariknya, pada grafik ini, nilai *train_loss* dan *val_loss* cenderung sangat dekat atau bahkan sama pada beberapa *epoch* terakhir, seperti pada *epoch* ke-20, ke-21, dan seterusnya. Grafik ini menunjukkan bahwa arsitektur ResVNet berhasil melakukan generalisasi dengan baik terhadap data validasi, yang ditunjukkan oleh konsistensi penurunan nilai *train_loss* dan *val_loss*. Hasil ini mengindikasikan bahwa arsitektur ResVNet mampu belajar secara efektif tanpa *overfitting*, karena nilai *val_loss* bergerak sejalan dengan *train_loss* tanpa ada tanda-tanda kenaikan yang signifikan.

Selain grafik *training* loss dan *validation* loss, terdapat juga grafik *training accuracy* dan *validation accuracy* yang menggambarkan performa model selama proses pelatihan. *Training accuracy* mengukur kinerja model pada data pelatihan, sedangkan *validation accuracy* menunjukkan kemampuan model dalam menggeneralisasi data baru. Hasil dari grafik *training accuracy* dan *validation accuracy* ini akan ditampilkan pada gambar 4.0.

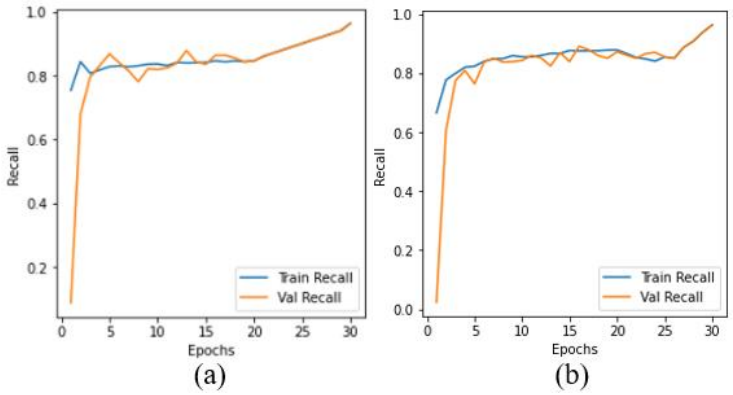


Gambar 4.1. Grafik Training Accuracy dan Validation Accuracy (a) DRIVE dan (b) STARE

Grafik *training accuracy* dan *validation accuracy* untuk dataset DRIVE dan STARE dapat dilihat pada gambar 4.1. Pada gambar 4.1, terlihat bahwa *accuracy* meningkat seiring bertambahnya jumlah *epoch* hingga mencapai *epoch* ke-30. Pada gambar 4.1(a), grafik menunjukkan *training accuracy* dan *validation accuracy* untuk dataset DRIVE, dapat dilihat grafik akurasi proses *training* mengalami peningkatan pada setiap *epoch*. Nilai akurasi yang diperoleh pada *epoch* pertama sebesar 0.84, lalu pada *epoch* selanjutnya nilai akurasi terus meningkat, mencapai 0.96 pada *epoch* terakhir. Kemudian, nilai akurasi untuk data validasi (*val_acc*) juga menunjukkan tren peningkatan yang signifikan. Pada *epoch* pertama, nilai *val_acc* sebesar 0.71, namun pada *epoch* selanjutnya nilai ini meningkat dengan cukup stabil hingga mencapai 0.96 pada *epoch* ke-29. Secara umum, nilai *val_acc* berkisar di antara 0.95 hingga 0.96, menunjukkan stabilitas performa arsitektur ResVNet pada data validasi. Grafik ini menunjukkan bahwa arsitektur ResVNet memiliki performa yang baik dengan kemampuan generalisasi yang cukup kuat, sebagaimana ditunjukkan oleh tren peningkatan akurasi yang konsisten baik pada data *training* maupun data validasi. Nilai akurasi yang stabil dan tinggi pada kedua metrik tersebut menunjukkan bahwa arsitektur ResVNet tidak mengalami *overfitting* dan berhasil belajar dengan baik dari data yang ada.

Demikian pula, pada grafik *training accuracy* dan *validation accuracy* untuk dataset STARE yang ditampilkan pada gambar 4.1 (b), dapat dilihat grafik akurasi proses *training* mengalami peningkatan pada setiap *epoch*. Nilai akurasi yang diperoleh pada *epoch* pertama sebesar 0.80, lalu pada *epoch* selanjutnya nilai akurasi terus meningkat, mencapai 0.96 pada *epoch* terakhir. Sedangkan, nilai akurasi untuk data validasi (*val_acc*) juga menunjukkan tren peningkatan yang konsisten, dimulai dari 0.86 pada *epoch* pertama dan mencapai 0.96 pada *epoch* terakhir. Grafik ini menunjukkan bahwa nilai akurasi untuk data validasi relatif stabil dan mengikuti tren peningkatan dari nilai akurasi pada data *training*. Secara keseluruhan, grafik ini menunjukkan bahwa arsitektur ResVNet berhasil mempelajari pola dengan baik, dengan kedua metrik akurasi (*train_acc* dan *val_acc*) mendekati nilai maksimal dan tidak menunjukkan tanda-tanda *overfitting*. Nilai akurasi untuk data *training* dan validasi yang stabil serta mendekati satu sama lain menunjukkan bahwa arsitektur ResVNet memiliki performa yang baik dan generalisasi yang kuat.

Selain grafik *training accuracy* dan *validation accuracy* , juga terdapat grafik *training recall* dan *validation recall*. Grafik ini menggambarkan perubahan nilai *recall* dari model yang digunakan selama proses pelatihan dan evaluasi pada dataset validasi. Grafik *training recall* dan *validation recall* dapat dilihat pada gambar 4.2.



Gambar 4.2. Grafik Training Recall dan Validation Recall (a) DRIVE dan (b) STARE

Pada gambar 4.2. (a), grafik *training recall* dan *validation recall* untuk dataset DRIVE menunjukkan tren yang lebih stabil dibandingkan grafik sebelumnya. Nilai recall yang diperoleh pada epoch pertama adalah 0.7531, dan menunjukkan peningkatan yang signifikan pada epoch-epoch selanjutnya, mencapai 0.8431 pada epoch kedua. Selanjutnya, meskipun terdapat beberapa fluktuasi kecil, nilai train recall secara keseluruhan mengalami peningkatan yang konsisten, mencapai 0.9627 pada epoch terakhir. Ini menunjukkan bahwa model berhasil belajar dan meningkatkan kemampuannya dalam mengenali kelas positif selama proses training. Di sisi lain, validation recall (val_recall) menunjukkan tren yang relatif lebih stabil namun tidak konsisten di awal. Pada epoch pertama, nilai val_recall hanya 0.0876, yang sangat rendah, namun secara dramatis meningkat menjadi 0.6785 pada epoch kedua. Nilai val_recall kemudian meningkat secara bertahap, mencapai 0.8629 pada epoch ke-16. Setelah itu, grafik menunjukkan stabilitas yang baik, dengan nilai val_recall berfluktuasi di sekitar 0.86 hingga 0.91 sebelum akhirnya mencapai 0.9627 pada epoch terakhir. Secara keseluruhan, grafik ini menunjukkan bahwa arsitektur ResVNet tidak hanya mampu meningkatkan training recall tetapi juga berhasil mempertahankan nilai validation recall yang solid di akhir proses. Hal ini mengindikasikan bahwa arsitektur ResVNet mampu melakukan generalisasi yang baik terhadap data yang tidak terlihat dan mampu menangkap kelas minoritas secara efektif. Namun, meskipun terdapat fluktuasi di awal pada validation recall, kenaikan nilai yang stabil di akhir menunjukkan bahwa arsitektur ResVNet mulai mengatasi tantangan tersebut, sehingga performanya semakin baik dalam mendeteksi kelas positif.

Sementara itu, pada grafik *training recall* dan *validation recall* untuk dataset STARE yang ditampilkan pada gambar 4.1. (b), menunjukkan bahwa proses training mengalami peningkatan yang signifikan sepanjang setiap epoch. Nilai recall yang diperoleh pada epoch pertama adalah 0.6656, dan secara bertahap meningkat pada epoch-epoch berikutnya. Pada epoch ke-15, nilai recall mencapai 0.8763, menandakan bahwa model

semakin baik dalam mengenali semua kelas selama proses training. Meskipun terdapat beberapa fluktuasi minor, nilai train recall tetap stabil dan terus menunjukkan tren peningkatan yang solid, mencapai 0.9628 pada epoch terakhir. Di sisi lain, validation recall (val_recall) pada epoch pertama sangat rendah, yaitu 0.0241, tetapi mengalami peningkatan yang cepat pada epoch selanjutnya, mencapai 0.8906 pada epoch ke-15. Namun, setelah itu, nilai val_recall mengalami fluktuasi dengan penurunan dan kenaikan yang bervariasi. Pada epoch terakhir, nilai recall untuk data validasi mencapai 0.9628, sama dengan nilai train recall. Secara keseluruhan, grafik ini menunjukkan bahwa arsitektur ResVNet tidak hanya mampu meningkatkan training recall tetapi juga berhasil mencapai nilai validation recall yang solid di akhir proses. Hal ini mengindikasikan bahwa arsitektur ResVNet dapat melakukan generalisasi dengan baik terhadap data yang tidak terlihat dan mampu menangkap pola dengan efektif. Meskipun ada beberapa fluktuasi dalam nilai validation recall, pencapaian yang konsisten di akhir menunjukkan bahwa arsitektur ResVNet memiliki kemampuan yang baik dalam mendeteksi kelas positif, sehingga model ini tampaknya tidak mengalami masalah overfitting yang signifikan.

4.5.2 Testing

Setelah melakukan pengujian pada segmentasi pembuluh darah menggunakan arsitektur ResVNet dengan melalui tahapan preprocessing terlebih dahulu, diperoleh nilai *confusion* matriks yang akan disajikan dalam tabel 4.10.

Tabel 4.10. Hasil Confusion Matriks Dari Arsitektur ResVNet Dengan Menggunakan Tahapan Preprocessing

No.	**Dataset**	***Confusion Matriks***			
		TP	**FP**	**FN**	**TN**
1.	DRIVE	4687423	97232	82451	375774
2.	STARE	3788543	71905	66149	267707

Pada tabel 4.10, terdapat nilai TP, FP, FN, dan TN yang dihasilkan dari pengujian segmentasi pembuluh darah menggunakan arsitektur

ResVNet dengan melewati tahapan preprocessing terlebih dahulu, yang ditampilkan pada tabel *confusion matriks*, nilai tersebut dimasukkan kerumus pada persamaan (2.19) untuk menghitung *accuracy* , lalu pada persamaan (2.20) untuk menghitung *sensitivity* , kemudian pada persamaan (2.21) untuk menghitung *Accuracy*, dan terakhir persamaan (2.22) untuk menghitung *Jaccard score*. Adapun perhitungannya akan ditampilkan dibawah ini:

$$accuracy\ (DRIVE) = \frac{TP + TN}{TP + FP + FN + TN} x\ 100\%$$

$$= \frac{4687423 + 375774}{4687423 + 97232 + 82451 + 375774} x\ 100\%$$

$$= \frac{5063197}{5242880} x100\%$$

$$= 96{,}57\%$$

$$accuracy\ (STARE) = \frac{TP + TN}{TP + FP + FN + TN} x\ 100\%$$

$$= \frac{3788543 + 267707}{3788543 + 71905 + 66149 + 267707} x\ 100\%$$

$$= \frac{4056250}{4194304} x100\%$$

$$= 96{,}71\%$$

$$Sensitivity\ (DRIVE) = \frac{TP}{(TP + FN)} x\ 100\%$$

$$= \frac{4687423}{(4687423 + 82451)} x\ 100\%$$

$$= \frac{4687423}{4769874} x100\%$$

$$= 98{,}27\%$$

$$Sensitivity\ (STARE) = \frac{TP}{(TP + FN)} x\ 100\%$$

$$= \frac{3788543}{(3788543 + 66149)} x\ 100\%$$

$$= \frac{3788543}{3854692} x100\%$$

$$= 98{,}28\%$$

$$Precision\ (DRIVE) = \frac{TP}{(TP + FP)} x\ 100\%$$

$$= \frac{4687423}{(4687423 + 97232)} x\ 100\%$$

$$= \frac{4687423}{4784655} x100\%$$

$$= 97{,}97\%$$

$$Precision\ (STARE) = \frac{TP}{(TP + FP)} x\ 100\%$$

$$= \frac{3788543}{(3788543 + 71905)} x\ 100\%$$

$$= \frac{3788543}{3860448} x100\%$$

$$= 98{,}14\%$$

$$Jaccard\ Score\ (DRIVE) = \frac{TP}{TP + FP + FN} x\ 100\%$$

$$= \frac{4687423}{4687423 + 97232 + 82451} x\ 100\%$$

$$= \frac{4687423}{4867106} x100\%$$

$$= 96{,}31\%$$

$$Jaccard\ Score\ (STARE) = \frac{TP}{TP + FP + FN} x\ 100\%$$

$$= \frac{3788543}{3788543 + 71905 + 66149} x\ 100\%$$

$$= \frac{3788543}{3926597} x 100\%$$

$$= 96{,}48\%$$

Hasil dari perhitungan tersebut digunakan untuk mengevaluasi kinerja arsitektur ResVNet dalam melakukan proses segmentsi pembuluh darah pada retina dengan melewati tahapan preprocessing telebih dahulu, adapun hasil pengukuran kinerja matriks yang didapatkan akan ditampilkan pada tabel 4.11.

Tabel 4.11. Hasil Pengukuran Kinerja Matriks ResVNet Dengan Menggunakan Tahapan Preprocessing

No.	Dataset	*Accuracy* (%)	*Sensitivity* (%)	Precision (%)	Jaccard Score (%)
1.	DRIVE	96,57	96,27	97,97	96,31
2.	STARE	**96,71**	**96,28**	**98,14**	**96,48**

Berdasarkan pengukuran kinerja matriks pada tabel 4.11, untuk segmentasi pembuluh darah pada dataset DRIVE dengan menggunakan arsitektur ResVNet menghasilkan nilai *accuracy* sebesar 96,57%, yang mana berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6, nilai *accuracy* yang didapatkan termasuk kedalam kategori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan telah mampu melakukan segmentasi dengan sangat baik. Selanjutnya nilai *sensitivity* yang diperoleh sebesar 96,27% dimana berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6 hasil *sensitivity* yang diperoleh termasuk kedalam ketegori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan telah mampu dengan sangat baik mendeteksi fitur yang ingin disegmentasi dari citra. *Precision* yang dihasilkan sebesar

97,97% berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6 hasil *Accuracy* yang dihasilkan termasuk kedalam kategori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan memiliki kemampuan sangat baik dalam mengidentifikasi piksel-piksel positif dari keseluruhan prediksi prositif. Adapun nilai *jaccard score* yang dihasilkan sebesar 96,31% berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6 hasil *jaccard score* yang dihasilkan termasuk kedalam kategori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan memiliki kemampuan yang sangat baik dalam mendeteksi antara *ground truth* dan prediksi.

Selanjutnya untuk segmentasi pembuluh darah pada dataset STARE dengan menggunakan arsitektur ResVNet dengan melalui tahapan preprocessing terlebih dahulu berdasarkan tabel 4.11, menghasilkan nilai *accuracy* sebesar 96,71%, yang mana berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6, nilai *accuracy* yang didapatkan termasuk kedalam kategori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan telah mampu melakukan segmentasi dengan sangat baik. Selanjutnya nilai *sensitivity* yang diperoleh sebesar 96,28% dimana berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6 hasil *sensitivity* yang diperoleh termasuk kedalam ketegori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan sangat baik untuk mendeteksi fitur yang ingin disegmentasi dari citra. *Precision* yang dihasilkan sebesar 98,14% berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6 hasil *Precision* yang dihasilkan termasuk kedalam kategori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan memiliki kemampuan sangat baik dalam mengidentifikasi piksel-piksel positif dari keseluruhan prediksi positif. Adapun nilai *jaccard score* yang dihasilkan sebesar 96,48% berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6 hasil *jaccard score* yang dihasilkan termasuk kedalam kategori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan memiliki kemampuan yang sangat baik dalam mendeteksi antara ground truth dan prediksi.

Segmentasi pembuluh darah pada citra retina juga dilakukan tanpa menggunakan tahapan preprocessing dengan menggunakan metode ResVNet, adapun hasil confusion matrix nya akan ditampilka pada tabel 4.12.

Tabel 4.12. Hasil Confusion Matriks Dari Arsitektur ResVNet Dengan Menggunakan Tahapan Preprocessing

No.	**Dataset**	***Confusion Matriks***			
		TP	**FP**	**FN**	**TN**
1.	DRIVE	4702696	81959	96107	362118
2.	STARE	362118	111048	54509	3749400

Pada tabel 4.12, terdapat nilai TP, FP, FN, dan TN yang dihasilkan dari pengujian segmentasi pembuluh darah menggunakan arsitektur ResVNet tanpa menggunakan tahapan preprocessing, yang ditampilkan pada tabel *confusion matriks*, nilai tersebut dimasukkan kerumus pada persamaan (2.19) untuk menghitung *accuracy* , lalu pada persamaan (2.20) untuk menghitung *sensitivity* , kemudian pada persamaan (2.21) untuk menghitung *Accuracy*, dan terakhir persamaan (2.22) untuk menghitung *Jaccard score*. adapun hasil pengukuran kinerja matriks yang didapatkan akan ditampilkan pada tabel 4.13.

Tabel 4.13. Hasil Pengukuran Kinerja Matriks ResVNet Dengan Menggunakan Tahapan Preprocessing

No.	**Dataset**	***Accuracy* (%)**	***Sensitivity* (%)**	**Precision (%)**	**Jaccard Score (%)**
1.	DRIVE	96,60	97,99	98,29	96,35
2.	STARE	96.09	**83.67**	71.55	62.78

Berdasarkan pengukuran kinerja matriks pada tabel 4.13, untuk

segmentasi pembuluh darah pada dataset DRIVE dengan menggunakan arsitektur ResVNet tanpa menggunakan tahapan preprocessing menghasilkan nilai *accuracy* sebesar 96,60%, yang mana berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6, nilai *accuracy* yang didapatkan termasuk kedalam kategori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan telah mampu melakukan segmentasi dengan sangat baik. Selanjutnya nilai *sensitivity* yang diperoleh sebesar 97,99% dimana berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6 hasil *sensitivity* yang diperoleh termasuk kedalam ketegori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan telah mampu dengan sangat baik mendeteksi fitur yang ingin disegmentasi dari citra. *Precision* yang dihasilkan sebesar 98,29% berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6 hasil *Accuracy* yang dihasilkan termasuk kedalam kategori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan memiliki kemampuan sangat baik dalam mengidentifikasi piksel-piksel positif dari keseluruhan prediksi prositif. Adapun nilai *jaccard score* yang dihasilkan sebesar 96,35% berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6 hasil *jaccard score* yang dihasilkan termasuk kedalam kategori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan memiliki kemampuan yang sangat baik dalam mendeteksi antara *ground truth* dan prediksi.

Selanjutnya untuk segmentasi pembuluh darah pada dataset STARE dengan menggunakan arsitektur ResVNet tanpa menggunakan tahapan preprocessing berdasarkan tabel 4.13, menghasilkan nilai *accuracy* sebesar 96,71%, yang mana berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6, nilai *accuracy* yang didapatkan termasuk kedalam kategori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan telah mampu melakukan segmentasi dengan sangat baik. Selanjutnya nilai *sensitivity* yang diperoleh sebesar 96,28% dimana berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6 hasil *sensitivity* yang diperoleh termasuk kedalam ketegori sangat baik, hal ini

menunjukkan bahwa arsitektur ResVNet yang digunakan sangat baik untuk mendeteksi fitur yang ingin disegmentasi dari citra. *Precision* yang dihasilkan sebesar 98,14% berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6 hasil *Precision* yang dihasilkan termasuk kedalam kategori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan memiliki kemampuan sangat baik dalam mengidentifikasi piksel-piksel positif dari keseluruhan prediksi positif. Adapun nilai *jaccard score* yang dihasilkan sebesar 96,48% berdasarkan kategori evaluasi kinerja arsitektur pada tabel 2.6 hasil *jaccard score* yang dihasilkan termasuk kedalam kategori sangat baik, hal ini menunjukkan bahwa arsitektur ResVNet yang digunakan memiliki kemampuan yang sangat baik dalam mendeteksi antara ground truth dan prediksi

4.6. Analisis hasil

Dalam penelitian ini, arsitektur ResVNet digunakan untuk melakukan segmentasi pembuluh darah pada citra retina. *Confusion matrix* digunakan untuk menghitung kinerja arsitektur berdasarkan *accuracy* , *sensitivity* , *precision*, dan *Jaccard score*. Perbandingan hasil segmentasi pembuluh darah pada citra retina dari penelitian ini dengan penelitian lain pada dataset DRIVE akan ditampilkan dalam tabel 4.14.

Tabel 4.14. Perbandingan Hasil Arsitektur ResVNet yang diusulkan untuk Segmentasi Pembuluh Darah Citra Retina Pada Dataset DRIVE Dengan Penelitian Lain

No.	**Peneliti**	**Metode Segmentasi**	***Accuracy*** **(%)**	***Sensitivity*** **(%)**	***Precision*** **(%)**	***Jaccard score*** **(%)**

1.	J. Ouyang dkk [22]	*LEA* U-Net	95,46	-	-	-
2.	Y. Zhan dkk [23]	*Bridge*- Net	95,65	78,53	-	-
3.	Yu Zhu dkk [24]	M3U-CDVAE	95,72	80,54	83,43	78,54
4.	D.E.Alvarado-Carrillo dan O. S. Dalmau-Cedeño [25]	WA-Net	95.75	79.66	-	-
5.	E. Abdelmaksoud dkk [7]	U-Net	95.55	66.1	-	-
6.	O. O. Sule[67]	EEA U-Net	95,77	79,18	-	-
7.	J. Zhang dkk [27]	*U-Net*	94,77	70,92	-	-
8.	M.E.Gegundez-Arias dkk [67]	U-Net	95,47	-	-	-
9.	Y. Lv, H. Ma, J. Li, and S. Liu [8]	AA-U-Net	95,58	-	-	-
10.	Anita,Erwin, Bambang, dan Sinta[30]	U-Net	95,46	74,20	-	-
		*Ladder*Net	95,47	75,19	-	-
11.	L. Li dkk[31]	IterNet	95,74	77,91	-	-
12.	Q. Xu, Z. Ma, N. HE, and W. Duan [32]	DI-U-net	95,42	82,83	-	-
13.	Wardhana, dkk [80]	Patch-based multi-scale line detection	95,95	82,61	81,98	
14.	Y.Tang dkk[33]	ResWnet	95,54	81,60	-	-

15.	Di Li dkk[9]	ResU-Net	-	77,89	-	-
16.	Chang Wang dkk[81]	*Dense* U-net	95,11	79,86	-	-
17.	Alan Reyes-Figueroa dan Mariano Rivera[12]	V-Net	95,15	84	-	-
18.	Sinta Bella Agustina dkk[13]	VV-Net	96,27	84,38	75,95	66,28
19.	Proposal penelitian	ResVNet	**96,57**	**96,27**	**97,97**	**96,31**

Dalam tabel 4.14 terdapat hasil penelitian lain yang melakukan segmentasi pembuluh darah pada citra retina menggunakan dataset DRIVE. Hasilnya menunjukkan bahwa arsitektur ResVNet yang digunakan dalam penelitian ini mencapai tingkat *accuracy* tertinggi dan sangat baik dibandingkan dengan penelitian lain yang tercantum. Selain itu, dalam hal *sensitivity* , arsitektur ResVNet juga menunjukkan performa yang lebih baik dan mengungguli penelitian lain yang serupa. Penelitian ini juga melakukan perhitungan nilai *precision* dan *Jaccard score*, yang tidak dilakukan dalam penelitian lain yang dibandingkan. Hasil *precision* menunjukkan bahwa arsitektur ResVNet yang digunakan sangat baik untuk mengidentifikasi piksel-piksel positif dari keseluruhan prediksi positif, *jaccard score* yang dihasilkan juga sangat baik yang menunjukkan bahwa arsitektur ResVNet yang digunakan memiliki kemampuan yang sangat baik dalam mendeteksi antara *ground truth* dan prediksi. Perbandingan hasil segmentasi pembuluh darah menggunakan arsitektur ResVNet dengan penelitian lain pada dataset STARE akan ditampilkan dalam tabel 4.15.

Tabel 4.15. Perbandingan Hasil Arsitektur ResVNet Untuk Segmentasi Pembuluh Darah Citra Retina Pada Dataset STARE Dengan Penelitian Lain

No.	**Peneliti**	**Metode Segmentasi**	***Accuracy*** **(%)**	***Sensitivity*** **(%)**	***Precision*** **(%)**	***Jaccard score*** **(%)**
1.	D. E. Alvarado-Carrillo dan O. S. Dalmau-Cedeño [25]	WA-NET	96.65	77.67	-	-
2.	O. Çiçek A. Abdulkadir dkk [7]	U-NET	95.55	66.1	-	76,06
3.	J. Li, G dkk.[28]	GDF-NET	96.53	76.16	-	-
4.	Tomar dkk [82]	FANet	96,86	85,44	79,67	
5.	Manu Yan Lv, Hui Mai [8]	AA-U-NET	96.40	-	-	-
6.	M. S. Gargari, dkk[5]	ResU-Net	-	78,02	-	-
7.	DanYang dkk[83]	*Multi-scale feature fusion retinal vessel segmentation model based on U-Net (MSFFU-Net)*	95,37	77,21	-	-
8.	Hao-Chiang Shao dkk[84]	2Unet	-	78,48	-	-
9.	Chang Wang dkk[81]	Dense U-net	95,38	79,14	-	-
10.	Hongwei Ding dkk[85]	U-Net	96,62	78,87	-	-

11.	Y. Zhou dkk [86]	Patch based multi-scale line detection	95,95	82,61	81,98	
12.	Nasser Tamim dkk[87]	*The local intensity, morphological , morphology black-bottom-hat,maximum-moment of phase congruency, minimum-moment of phase congruency, features at five different scales*	96,32	78,06	76,02	74,60
13.	Q. Jin dkk[88]	DU-Net	96,41	-	-	-
14.	Alan Reyes-Figueroa dan Mariano Rivera[12]	V-Net	93,42	84,35	-	
15.	Sinta Bella Agustina dkk[13]	VV-Net	96,58	82,78	76,73	65,38
16.	(Proposal penelitian, 2024)	ResVNet	**96,71**	**96,28**	**98,14**	**96,48**

Dalam tabel 4.15 terdapat hasil penelitian lain yang melakukan segmentasi pembuluh darah pada citra retina menggunakan dataset STARE. Hasilnya menunjukkan bahwa arsitektur ResVNet yang digunakan dalam penelitian ini mencapai tingkat *accuracy* tertinggi dan sangat baik dibandingkan dengan penelitian lain yang tercantum. Selain itu, dalam hal *sensitivity* , arsitektur ResVNet juga menunjukkan performa yang lebih baik dan mengungguli penelitian lain yang serupa. Penelitian ini juga melakukan perhitungan nilai *precision* dan *Jaccard score*, yang tidak dilakukan dalam penelitian lain yang dibandingkan. Hasil *precision* menunjukkan bahwa arsitektur ResVNet sangat baik dalam

mengidentifikasi piksel-piksel positif dari keseluruhan prediksi positif. *Jaccard score* yang dihasilkan juga menunjukkan bahwa arsitektur ResVNet memiliki kinerja yang sangat baik dalam mendeteksi kesamaan antara *ground truth* dan prediksi.

Dari tabel 4.14 dan 4.15, terlihat bahwa arsitektur ResVNet yang digunakan secara signifikan unggul dalam hal *accuracy* dan kemampuan untuk mendeteksi pembuluh darah pada citra retina dibandingkan dengan metode-metode lain yang disajikan. Hasil ini mencerminkan potensi besar dari pendekatan yang diusulkan dalam penelitian ini. Keunggulan ini dapat disebabkan oleh inovasi dalam penggunaan arsitektur ResVNet. Penelitian ini juga mencakup pengukuran *precision* dan *Jaccard score*, yang memberikan informasi lebih detail tentang performa arsitektur dalam hal keakuratan dan sejauh mana segmentasi sesuai dengan area sebenarnya.