

**DESAIN PENGKLASIFIKASI CITRA KUANTUM DENGAN
SKEMA DATA *RE-UPLOADING QUANTUM CONVOLUTION*
DAN DATA *RE-UPLOADING CLASSIFIER***

TUGAS AKHIR



Oleh
Eraraya Ricardo Muten NIM: 13316082

**PROGRAM STUDI TEKNIK FISIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI BANDUNG**

2021

**DESAIN PENGKLASIFIKASI CITRA KUANTUM DENGAN
SKEMA DATA *RE-UPLOADING QUANTUM CONVOLUTION*
DAN DATA *RE-UPLOADING CLASSIFIER***

TUGAS AKHIR

Diajukan untuk memenuhi syarat kelulusan tahap Pendidikan Strata-1 pada
Program Studi Teknik Fisika – Institut Teknologi Bandung

Oleh:

Eraraya Ricardo Muten 13316082

Dosen Pembimbing:

Ir. Nugraha, Ph.D.

Prof. Andriyan Bayu Suksmono M.T., Ph.D.

**PROGRAM STUDI TEKNIK FISIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI BANDUNG**

2021

ABSTRAK

DESAIN PENGKLASIFIKASI CITRA KUANTUM DENGAN SKEMA DATA RE-UPLOADING QUANTUM CONVOLUTION DAN DATA RE-UPLOADING CLASSIFIER

Oleh
Eraraya Ricardo Muten NIM: 13316082

(Program Studi Teknik Fisika)

Kebutuhan akan kekuatan komputasi terus meningkat seiring dengan semakin sulitnya permasalahan dalam dunia industri dan akademik yang perlu dipecahkan. Aplikasi mensimulasikan sistem kuantum besar seperti molekul, atau memecahkan sistem persamaan linear berskala besar, sangatlah mahal biaya komputasinya. Hal ini menjadi salah satu pemicu berkembangnya komputasi kuantum, metode komputasi baru yang memanfaatkan sifat-sifat dan teori sistem kuantum untuk mengolah informasi. Komputer kuantum berpotensi mampu memecahkan permasalahan tersebut dengan percepatan eksponensial.

Meski perkembangan komputer kuantum terbilang sangat pesat belakangan ini, tantangan dari segi teori maupun teknologi menjadi halangan dalam pembuatan komputer kuantum berskala besar. Komputer kuantum yang tersedia saat ini memiliki batasan seperti jumlah qubit yang terbatas dan operasi gerbang berderau yang membatasi jumlah gerbang yang dapat digunakan. Algoritma Kuantum Variasional (AKV) kemudian muncul menjadi salah satu strategi andalan yang menjanjikan dalam menghadapi batasan ini. Aplikasi di berbagai bidang yang memanfaatkan strategi ini sudah banyak diajukan, termasuk dalam bidang klasifikasi citra sebagai aplikasi pembelajaran mesin kuantum.

Penelitian ini mengajukan skema modifikasi dari *Data Re-uploading Classifier* (DRC) yang berbasis AKV untuk klasifikasi citra MNIST. Penggunaan metode reduksi *Principal Component Analysis* (PCA) dan pengklasifikasi DRC dengan representasi biner (DRC-RB) mencapai akurasi uji 99,7% pada klasifikasi 2 kelas, 96,5% pada klasifikasi 4 kelas, dan 86,25% pada klasifikasi 8 kelas, sebuah peningkatan akurasi jika dibandingkan penelitian AKV terkait sebelumnya. Penelitian ini juga mengajukan skema konvolusi kuantum berbasis DRC. Tanpa menggunakan PCA, konvolusi kuantum dan pengklasifikasi DRC-RB mampu mencapai akurasi uji 98,9% pada klasifikasi 2 kelas dan 89,5% pada klasifikasi 4 kelas, sepadan dengan hasil yang dicapai Jaringan Saraf Tiruan Konvolusi.

Kata kunci: algoritma kuantum variasional, klasifikasi citra, komputasi kuantum, konvolusi kuantum, pembelajaran mesin kuantum.

ABSTRACT

QUANTUM IMAGE CLASSIFIER DESIGN WITH DATA RE-UPLOADING QUANTUM CONVOLUTION AND DATA RE-UPLOADING CLASSIFIER SCHEME

By

Eraraya Ricardo Muten NIM: 13316082

(Engineering Physics Program)

The need for computational power keeps increasing as industry and academia's problems get harder and harder to solve. Applications such as simulation of large quantum systems like molecules or solving large linear systems can be very expensive in computational cost. This has become one of the reasons for quantum computing development, a computational method that employs characteristics and theories of quantum systems for information processing. Quantum computers promise us exponential speed-up for these kinds of problems.

Although quantum computers' development has been growing rapidly in recent years, the theoretical and technological challenges remain a barrier for a large-scale quantum computer. Quantum computers that exist today have severe limitations, such as limited qubits and limited gate operations due to noise in the processes. Variational Quantum Algorithms (VQA) have arisen to be one of the promising strategies in dealing with these limitations. Applications across the fields that employ this strategy have been proposed, including image classification as quantum machine learning applications.

This research proposed a modification scheme of the VQA-based Data Re-uploading Classifier (DRC) for MNIST classification. A binary, four-class, and eight-class classification task reached 99.7%, 96.5%, and 86.25% of testing accuracy, respectively, using the Principal Component Analysis (PCA) for dimensionality reduction and Data Re-uploading Classifier with binary representation (DRC-BR) for classification. An improvement in accuracy compared to the previous related VQA works. This research also proposed a DRC-based quantum convolution scheme. Without using PCA, quantum convolution with DRC-BR classifier for binary and four-class classification task achieved 98.9% and 89.5% of testing accuracy, respectively. A respectable result compared to the classical Convolutional Neural Network with the same amount of parameters.

Keywords: *image classification, quantum computing, quantum convolution, quantum machine learning, variational quantum algorithms.*

**DESAIN PENGKLASIFIKASI CITRA KUANTUM DENGAN
SKEMA DATA RE-UPLOADING QUANTUM CONVOLUTION
DAN DATA RE-UPLOADING CLASSIFIER**

HALAMAN PENGESAHAN

Oleh
Eraraya Ricardo Muten NIM: 13316082

(Program Studi Teknik Fisika)

Institut Teknologi Bandung

Menyetujui
Tim Pembimbing

8 Februari 2021

Pembimbing 1



Ir. Nugraha, Ph.D.
NIP: 196401151989021001

Pembimbing 2



Prof. Andriyan Bayu Suksmono
M.T., Ph.D.
NIP: 196607051996031002

KATA PENGANTAR

Dengan mengucapkan puji dan syukur kepada Tuhan Yang Maha Esa atas berkat dan rahmat-Nya telah berhasil diselesaikan penelitian tugas akhir ini dengan judul **“Desain Pengklasifikasi Citra Kuantum dengan Skema Data Re-uploading Quantum Convolution dan Data Re-uploading Classifier”**. Laporan tugas akhir ini ditulis untuk memenuhi salah satu persyaratan kelulusan mata kuliah TF4092 – Tugas Akhir serta sebagai pemenuhan kewajiban tahap S1 Program Studi Teknik Fisika di Institut Teknologi Bandung.

Penulis memilih tema komputasi kuantum untuk tugas akhir dengan harapan meningkatkan keawasan civitas academica Institut Teknologi Bandung khususnya, dan akademisi dan teknokrat di seluruh Indonesia pada umumnya, akan pentingnya komputasi kuantum yang saat ini sedang berkembang sangat pesat di negara-negara luar. Tugas akhir ini mungkin tugas akhir dengan tema komputasi kuantum yang pertama di jenjang S1 Teknik Fisika ITB, namun penulis berharap ini tidak akan jadi yang terakhir.

Penulis tidak lupa untuk mengucapkan terima kasih yang sebesar-besarnya kepada pihak-pihak yang telah membantu proses penggerjaan penelitian tugas akhir ini, baik secara langsung maupun tidak langsung, khususnya kepada:

1. Kedua orang tua dan keluarga penulis yang senantiasa memberikan bantuan materi dan moril kepada penulis.
2. Bapak Ir. Nugraha, Ph.D. dan Bapak Prof. Andriyan Bayu Suksmono M.T., Ph.D. selaku dosen pembimbing yang telah memberikan sumber daya dan dengan sabar dan penuh perhatian membantu proses penggerjaan tugas akhir lewat diskusi dan saran-saran yang penulis terima.
3. Bapak Dr. Agung Budiyono yang dengan antusias bersedia mendengarkan presentasi dan laporan-laporan kemajuan dari penulis selama proses penggerjaan tugas akhir, dan dengan ramah dan rendah hati memberikan masukan-masukan yang sangat konstruktif.

4. Elbert Timothy Lasiman, 13318002, yang telah rela berdiskusi dengan penulis terkait tugas akhir dan membantu dalam proses visualisasi data.
5. Bapak Dr.-Ing. Ir. Parsaulian Ishaya Siregar, Dipl., Bapak Dr. Ir. Mohammad Kemal Agusta, S.T., M.Eng., dan Bapak Narendra Kurnia Putra, ST., MT., Ph.D. selaku koordinator mata kuliah TF4092 – Tugas Akhir yang telah memberikan saya ilmu dan masukan yang diperlukan dalam pembuatan laporan tugas akhir ini.
6. Maria Schuld, peneliti pembelajaran mesin kuantum yang menjadi sosok inspirasi bagi penulis, atas pemberian buku yang ia tulis, *Supervised Learning with Quantum Computers*, kepada penulis dan juga atas karyakarya ilmiahnya yang banyak memberikan ilmu kepada penulis.
7. Segean tim Xanadu Quantum Technologies Inc. yang telah membagikan sumber-sumber pembelajaran dalam jaringan secara gratis, terutama tutorial pembelajaran mesin kuantum yang ada pada situs web PennyLane.
8. Rekan-rekan IBM Quantum Qiskit Advocate dan segenap tim IBM Quantum atas diskusi, masukan, dan sumber-sumber pembelajaran komputasi kuantum yang membantu penulis ketika proses mempelajari komputasi kuantum.

Penulis menyadari masih banyak kekurangan dalam penelitian tugas akhir ini. Penulis berharap dan dengan senang hati untuk menerima kritik dan saran konstruktif dari pembaca agar kedepannya kegiatan penelitian yang dilakukan penulis dapat menjadi semakin baik lagi.

Depok, 6 Februari 2021

Eraraya Ricardo Muten

DAFTAR ISI

ABSTRAK	i
ABSTRACT	ii
KATA PENGANTAR.....	iv
DAFTAR ISI.....	vi
DAFTAR GAMBAR.....	ix
DAFTAR TABEL	xiii
DAFTAR SINGKATAN DAN LAMBANG	xiv
1 BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Tujuan Penelitian.....	3
1.3 Penelitian Terkait	4
1.4 Batasan Penelitian	6
1.5 Sistematika Penulisan.....	6
2 BAB II TINJAUAN PUSTAKA	8
2.1 Dasar Komputasi Kuantum	8
2.1.1 Qubit.....	8
2.1.2 <i>Bloch Sphere</i>	9
2.1.3 Gerbang Kuantum dan Sirkuit Kuantum.....	10
2.1.4 Pengukuran.....	13
2.2 Jaringan Saraf Tiruan (JST)	14
2.3 Reduksi Dimensi Citra	17
2.3.1 <i>Principal Component Analysis</i> (PCA)	17
2.3.2 Konvolusi dan <i>Pooling</i>	20
2.4 Pengklasifikasi Kuantum Sebagai Aplikasi AKV.....	24
2.4.1 Penyematan Fitur Kuantum (PKF).....	25
2.4.2 Evolusi Unitari dengan Sirkuit Kuantum Variasional (SKV)	27
2.4.3 Pengukuran Nilai Ekspektasi.....	28
2.4.4 <i>Data Re-uploading Classifier</i> (DRC).....	29
2.5 Algoritma Optimisasi	32
2.5.1 <i>Adaptive Moment Estimation</i> (Adam).....	32

2.5.2 Laju Pembelajaran Meluruh	33
2.6 Gradien Kuantum	34
3 BAB III PERANCANGAN ARSITEKTUR SISTEM	36
3.1 Skema Reduksi Dimensi Citra MNIST	37
3.1.1 Reduksi dengan PCA.....	39
3.1.2 Reduksi dengan Konvolusi Kuantum.....	39
3.2 Skema Pengklasifikasi Kuantum.....	44
3.2.1 <i>Data Re-uploading Classifier</i> (DRC).....	44
3.2.2 DRC dengan Representasi Biner (DRC-RB)	49
3.3 Rangkuman Skema Pengklasifikasi yang Diuji	51
3.3.1 Pengklasifikasi DRC dengan Reduksi PCA.....	52
3.3.2 Pengklasifikasi DRC-RB dengan Reduksi PCA	52
3.3.3 Pengklasifikasi DRC-RB dengan Reduksi Konvolusi Kuantum (DRQConv)	53
4 BAB IV HASIL DAN PEMBAHASAN.....	59
4.1 <i>Explained Variance (EV)</i> PCA.....	59
4.2 Performa Pengklasifikasi DRC dengan Reduksi PCA	61
4.2.1 Klasifikasi 2 Kelas.....	61
4.2.2 Klasifikasi 4 Kelas.....	66
4.3 Performa Pengklasifikasi DRC-RB dengan Reduksi PCA	69
4.3.1 Klasifikasi 4 Kelas.....	69
4.3.2 Klasifikasi 8 Kelas.....	71
4.4 Performa Pengklasifikasi DRC-RB dengan Reduksi Konvolusi Kuantum	72
4.4.1 Klasifikasi 2 Kelas.....	73
4.4.2 Klasifikasi 4 Kelas.....	76
4.5 Citra Hasil Konvolusi Kuantum	79
4.5.1 Konvolusi Pertama	80
4.5.2 Konvolusi Kedua	81
5 BAB V KESIMPULAN DAN SARAN	83
5.1 Kesimpulan.....	83
5.2 Saran.....	84

6 DAFTAR PUSTAKA	85
A. LAMPIRAN A GRAFIK AKURASI DAN NILAI BIAYA SELAMA PROSES LATIH	89
B. LAMPIRAN B REPRESENTASI <i>BLOCH SPHERE</i> SEBELUM DAN SETELAH PROSES LATIH	92

DAFTAR GAMBAR

Gambar 2.1 Representasi <i>Bloch sphere</i> dari sebuah qubit dengan keadaan kuantum Ψ . Gambar disadur dari [25]. ..	9
Gambar 2.2 Contoh beberapa gerbang kuantum dengan representasi sirkuit (Gate) dan matriksnya (Matrix). Gambar disadur dari [27]. ..	11
Gambar 2.3 Proses komputasi dari sebuah simpul dalam JST. Gambar disadur dari [28] dengan sedikit perubahan pada <i>activation function</i>	14
Gambar 2.4 Contoh arsitektur sebuah JST dengan 3 lapisan tersembunyi. Disadur dari [29] dengan lokalisasi ke Bahasa Indonesia. ..	15
Gambar 2.5 Ilustrasi operasi konvolusi pada citra berkanal 3 (RGB). Gambar disadur dari [31]. ..	21
Gambar 2.6 Ilustrasi operasi <i>pooling</i> dengan filter berukuran 2×2 dengan <i>stride</i> = 2 pada salah satu kanal citra berdimensi 4×4 . Gambar disadur dari [31]. ..	22
Gambar 2.7 Contoh struktur algoritma JST Konvolusi. Gambar disadur dari [31]. ..	23
Gambar 2.8 Diagram alir algoritma Pengklasifikasi Kuantum secara umum.....	24
Gambar 2.9 Skematik sirkuit metode penyematan amplitudo. ..	26
Gambar 2.10 Skematik sirkuit metode penyematan sudut.....	27
Gambar 2.11 Contoh <i>ansatz</i> SKV sederhana yang menggunakan 4 qubit. ..	27
Gambar 2.12 Diagram alir sederhana dari algoritma DRC.....	30
Gambar 2.13 Skematik sirkuit DRC 1 qubit.....	31
Gambar 2.14 Skematik sirkuit DRC 1 qubit dengan jumlah lapisan sebanyak L . ..	32
Gambar 2.15 Ilustrasi pengaruh pemilihan nilai laju pembelajaran dalam proses optimisasi parameter θ untuk meminimumkan nilai fungsi biaya $J(\theta)$. Gambar disadur dari [37] dengan lokalisasi ke Bahasa Indonesia....	33
Gambar 3.1 Diagram alir skema penelitian.....	36
Gambar 3.2 Diagram aliran data. ..	37
Gambar 3.3 Contoh sampel citra dari himpunan data MNIST[21].....	38
Gambar 3.4 Skematik <i>ansatz</i> sirkuit pertama yang akan digunakan untuk DRQConv.....	42

Gambar 3.5 Skematik <i>ansatz</i> sirkuit kedua yang akan digunakan untuk DRQConv.....	43
Gambar 3.6 Skematik <i>ansatz</i> sirkuit ketiga yang akan digunakan untuk DRQConv.	44
Gambar 3.7 Visualisasi keadaan kuantum $ 0\rangle$ dan $ 1\rangle$ yang merepresentasikan dua kelas berbeda.....	46
Gambar 3.8 Visualisasi keempat vektor keadaan kuantum dalam <i>Bloch sphere</i> untuk merepresentasikan 4 kelas berbeda. Ujung-ujung vektor merupakan titik-titik sudut dari sebuah tetrahedral sama sisi.	47
Gambar 4.1 Grafik nilai <i>EV</i> untuk 100 komponen prinsip setelah transformasi himpunan data 2 kelas dengan PCA.....	59
Gambar 4.2 Grafik nilai <i>EV</i> untuk 100 komponen prinsip setelah transformasi himpunan data 4 kelas dengan PCA.....	60
Gambar 4.3 Grafik nilai <i>EV</i> untuk 100 komponen prinsip setelah transformasi himpunan data 8 kelas dengan PCA.....	60
Gambar 4.4 Grafik nilai akurasi pengklasifikasi selama proses latih pada tiap <i>epoch</i> . (a) 3 komponen prinsip, (b) 6 komponen prinsip.....	61
Gambar 4.5 Grafik nilai biaya pengklasifikasi selama proses latih pada tiap <i>epoch</i> . (a) 3 komponen prinsip, (b) 6 komponen prinsip.....	62
Gambar 4.6 Tabulasi capaian akurasi terbaik dari evaluasi (a) himpunan data latih dan (b) himpunan data uji dalam representasi <i>heatmap</i> . Angka 1 bermakna tercapainya 100% akurasi.....	63
Gambar 4.7 Representasi I. <i>Bloch sphere</i> dan II. <i>heatmap</i> dari keadaan kuantum keluaran DRC untuk (a) himpunan data latih dan (b) himpunan data uji sebelum proses latih, dan (c) himpunan data latih dan (d) himpunan data uji setelah proses latih.....	64
Gambar 4.8 Tabulasi capaian akurasi terbaik dari evaluasi (a) himpunan data latih dan (b) himpunan data uji dalam representasi <i>heatmap</i>	67
Gambar 4.9 Representasi I. <i>Bloch sphere</i> dan II. <i>heatmap</i> dari keadaan kuantum keluaran DRC untuk (a) himpunan data latih dan (b) himpunan data uji sebelum proses latih, dan (c) himpunan data latih dan (d) himpunan data uji setelah proses latih.....	68

Gambar 4.10 Representasi <i>Bloch sphere</i> keadaan kuantum keluaran DRC-RB dari I. evaluasi data latih dan II. evaluasi data uji. Baris atas dan bawah menunjukkan keadaan sebelum dan sesudah proses latih.....	69
Gambar 4.11 Representasi <i>heatmap</i> keadaan kuantum keluaran DRC-RB dari I. evaluasi data latih dan II. evaluasi data uji. Baris atas dan bawah menunjukkan keadaan sebelum dan sesudah proses latih.....	70
Gambar 4.12 Representasi <i>Bloch sphere</i> keadaan kuantum keluaran DRC-RB dari I. evaluasi data latih dan II. evaluasi data uji. Baris atas dan bawah menunjukkan keadaan sebelum dan sesudah proses latih.....	71
Gambar 4.13 Grafik (a) nilai akurasi dan (b) nilai biaya pengklasifikasi selama proses latih pada tiap <i>epoch</i>	73
Gambar 4.14 Representasi <i>heatmap</i> dari keadaan kuantum keluaran DRC-RB dengan DRQConv I (I), DRQConv II (II), dan DRQConv II + ent (III) untuk himpunan data (a) latih dan (b) uji sebelum proses latih, dan himpunan data (c) latih dan (d) uji setelah proses latih.....	75
Gambar 4.15 Representasi <i>Bloch sphere</i> evaluasi data I. latih dan II. uji sebelum proses latih (baris atas) dan setelah proses latih (baris bawah).....	77
Gambar 4.16 Sampel data MNIST berukuran 27 x 27.....	80
Gambar 4.17 Citra keluaran konvolusi kuantum pertama dari sirkuit (a) DRQConv I, (b) DRQConv II, dan (c) DRQConv II + ent.....	80
Gambar 4.18 Citra keluaran konvolusi kuantum kedua dari sirkuit (a) DRQConv I, (b) DRQConv II, dan (c) DRQConv II + ent.....	81
Gambar A.1 Grafik nilai akurasi dan nilai biaya selama proses latih pada kombinasi reduksi PCA (a) 6 komponen prinsip, (b) 12 komponen prinsip, dan (c) 18 komponen prinsip dengan pengklasifikasi DRC untuk klasifikasi 4 kelas.	89
Gambar A.2 Grafik nilai akurasi dan nilai biaya selama proses latih pada kombinasi reduksi PCA dengan pengklasifikasi DRC-RB untuk (a) klasifikasi 4 kelas (18 komponen prinsip, 4 lapisan) dan (b) klasifikasi 8 kelas (27 komponen prinsip, 4 lapisan).	90

Gambar A.3 Grafik nilai akurasi dan nilai biaya selama proses latih pada kombinasi (a) DRQConv I, (b) DRQConv II, dan (c) DRQConv II + ent dengan pengklasifikasi DRC-RB untuk klasifikasi 4 kelas.....	91
Gambar B.1 Representasi <i>Bloch sphere</i> untuk kasus klasifikasi 2 kelas sebelum (baris atas) dan setelah (baris bawah) proses latih dengan <i>ansatz</i> sirkuit I. DRQConv I, II. DRQConv II, III. DRQConv II + ent. (a) dan (c) adalah evaluasi data latih, (b) dan (d) adalah evaluasi data uji.	92
Gambar B.2 Representasi <i>Bloch sphere</i> untuk kasus klasifikasi 4 kelas pada evaluasi data I. latih dan II. uji sebelum proses latih (baris atas) dan setelah proses latih (baris bawah) menggunakan <i>ansatz</i> sirkuit DRQConv II.	93
Gambar B.3 Representasi <i>Bloch sphere</i> untuk kasus klasifikasi 4 kelas pada evaluasi data I. latih dan II. uji sebelum proses latih (baris atas) dan setelah proses latih (baris bawah) menggunakan <i>ansatz</i> sirkuit DRQConv II + ent.	94

DAFTAR TABEL

Tabel 3.1 Spesifikasi pengujian kombinasi pengklasifikasi DRC dengan reduksi PCA.....	52
Tabel 3.2 Spesifikasi pengujian kombinasi pengklasifikasi DRC-RB dengan reduksi PCA.....	53
Tabel 3.3 Spesifikasi pengujian kombinasi pengklasifikasi DRC-RB dengan reduksi DRQConv untuk klasifikasi 2 kelas.....	54
Tabel 3.4 Spesifikasi pengujian JST Konvolusi 2 kelas.....	55
Tabel 3.5 Spesifikasi pengujian kombinasi pengklasifikasi DRC-RB dengan reduksi DRQConv untuk klasifikasi 4 kelas.....	57
Tabel 3.6 Spesifikasi pengujian JST Konvolusi 4 kelas.....	58
Tabel 4.1 Tabulasi nilai akurasi terbaik pada evaluasi data latih dan data uji untuk setiap <i>ansatz</i> sirkuit. Disertakan juga hasil dari JST Konvolusi sebagai referensi.....	74
Tabel 4.2 Tabulasi nilai akurasi terbaik pada evaluasi data latih dan data uji untuk setiap <i>ansatz</i> sirkuit. Disertakan juga hasil dari JST Konvolusi sebagai referensi.....	78

DAFTAR SINGKATAN DAN LAMBANG

Singkatan	Nama	Halaman Pemakaian Pertama
Adam	<i>Adaptive Moment Estimation</i>	16
AKV	Algoritma Kuantum Variasional	3
DRC	<i>Data Re-uploading Classifier</i>	6
DRC-RB	DRC dengan Representasi Biner	7
DRQConv	<i>Data Re-uploading Quantum Convolution</i>	40
ent	<i>Entanglement</i>	44
EV	<i>Explained Variance</i>	18
JST	Jaringan Saraf Tiruan	3
MSE	<i>Mean Squared Error</i>	28
NISQ	<i>Noisy Intermediate Scale Quantum</i>	5
RGB	<i>Red Green Blue</i>	21
PCA	<i>Principal Component Analysis</i>	5
PFK	Penyematan Fitur Kuantum	25
ReLU	<i>Rectified Linear Unit</i>	16
SKV	Sirkuit Kuantum Variasional	3

LAMBANG

$ \Psi\rangle, \psi\rangle$	Keadaan kuantum	8
ϕ, θ, ω	Parameter sudut gerbang kuantum parametrik	12
O	<i>Observable</i> pengukuran	13
λ	Nilai eigen, hasil pengukuran	13
$\langle O \rangle$	Nilai ekspektasi <i>observable</i> O	13
a	Fungsi aktivasi	14
w	Parameter bobot masukan	14
b	Parameter bias	14
x	Sampel data	14
X	Himpunan data	14
n	Indeks fitur sampel data	14
N	Jumlah fitur sampel data	14
y	Keluaran simpul JST, hasil pengukuran sirkuit kuantum	14
σ	Standar deviasi populasi	17
μ	Rata-rata	18
m	Indeks sampel data	18
M	Jumlah sampel data	18
D	Matriks himpunan data	18
K	Jumlah kanal citra	18
H	Dimensi horizontal citra	18
V	Dimensi vertikal citra	18
ϵ	Konstanta stabilitas numerik	18
C	Matriks kovarians	19
v	Vektor eigen	19

P	Matriks komponen prinsip	19
p	Jumlah komponen prinsip	19
EV	<i>Explained Variance</i>	20
F	Jumlah filter	21
ρ	Ukuran filter	21
W	Matriks bobot masukan	21
s	<i>Stride</i>	21
χ	Nilai intensitas piksel	21
A	Amplitudo fungsi gelombang	25
$binary$	Fungsi konversi bilangan bulat ke bilangan biner	25
Q	Jumlah total qubit	26
U	<i>Ansatz</i> sirkuit	28
$\vec{\theta}$	Vektor parameter	28
J	Fungsi biaya	28
q	Indeks qubit	30
l	Indeks lapisan	30
L	Jumlah total lapisan	32
t	Langkah optimisasi	32
η	Laju pembelajaran	32
V_{θ_G}	Estimasi momen pertama dari $\frac{\partial}{\partial \theta_G} J(\vec{\theta})$	32
S_{θ_G}	Estimasi momen kedua dari $\frac{\partial}{\partial \theta_G} J(\vec{\theta})$	32
τ	Konstanta laju peluruhan	33
Δ	Konstanta langkah peluruhan	33
ζ	$= \rho \times \rho$	40
\vec{W}	Vektor parameter bobot masukan	41
\vec{b}	Vektor parameter bias	41
\vec{y}_{pred}	Vektor prediksi keluaran	45
α	Parameter bobot keluaran	45
C	Jumlah kelas	46
\vec{y}_{true}	Vektor label	46
sum	Fungsi sumasi elemen vektor	46

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pada tahun 1965, Gordon E. Moore, salah satu pendiri dan mantan Pejabat Eksekutif Tertinggi Intel, memprediksi secara empirik bahwa jumlah transistor pada sirkuit terpadu komputer akan berlipat ganda jumlahnya setiap tahun, prediksi ini dikenal dengan sebutan hukum Moore[1]. Tetapi keilmuan mekanika kuantum yang mulai berkembang pesat sejak abad 20 mengatakan bahwa partikel fisis mulai tidak mematuhi hukum-hukum mekanika dan elektrodinamika klasik ketika partikel mencapai dibawah ukuran tertentu. Selain itu, panas yang dihasilkan oleh transistor yang terlalu padat di wilayah yang kecil akan menyebabkan malfungsi pada transistor itu sendiri[2]. Faktanya, dalam beberapa tahun terakhir jumlah transistor dalam sirkuit terpadu komputer mulai mengalami saturasi. Dari tahun 1971 sampai 2012, sirkuit terpadu yang dibuat Intel masih mengikuti hukum Moore dengan baik[3], namun pada tahun-tahun berikutnya, kenaikan jumlah transistor tiap tahunnya tidak lagi mengikuti hukum Moore. Mulai tahun 2016, industri sirkuit terpadu di seluruh dunia tidak lagi menjadikan hukum Moore sebagai landasan utama dalam rencana riset dan pengembangan mereka[2]. Kekuatan komputasi dari sebuah komputer sangat bergantung pada seberapa banyak transistor yang dapat masuk dalam sirkuit terpadu, sehingga permasalahan ini dapat menyebabkan kekuatan komputasi berhenti bertambah.

Resiko stagnasi kekuatan komputasi dan kebutuhan kekuatan komputasi untuk simulasi baik dalam dunia industri dan akademik yang terus bertambah menyebabkan dimulainya pencarian alternatif teknologi komputasi[4]. Richard Feynmann, dalam kuliah singkatnya pada Mei 1981 di California Institute of Technology, pernah mengatakan bahwa untuk dapat mensimulasikan alam yang bersifat kuantum maka dibutuhkan pula alat komputasi yang memiliki sifat kuantum[5], kata-kata ini dipercaya sebagai cikal bakal mulai digunakannya istilah komputasi kuantum, yaitu cara-cara pengolahan informasi yang dapat

dilakukan menggunakan hukum-hukum dan sistem mekanika kuantum[6]. Seiring kemajuan-kemajuan yang terjadi dalam teknologi kuantum, ketertarikan pada komputasi kuantum semakin meningkat. Pada tahun 1994, Peter Shor secara analitik menunjukkan komputasi kuantum mampu melakukan pemfaktoran bilangan prima yang besar dengan percepatan eksponensial jika dibandingkan dengan solusi komputasi klasik terbaik yang ada saat ini[7]. Penemuan ini menjadi pemicu penting yang meningkatkan perkembangan komputer kuantum secara pesat karena pemfaktoran bilangan prima yang besar sangat penting dalam aplikasi kriptografi[6].

Mulai bermunculan makalah-makalah penelitian yang mengajukan algoritma kuantum untuk menyelesaikan permasalahan yang terkenal sulit dan mahal biaya komputasinya pada komputer klasik. Menjadi hal yang lumrah jika salah satu ketertarikan pada komputasi kuantum adalah aplikasi komputasi kuantum dalam pembelajaran mesin. Metode pembelajaran mesin klasik mampu mengenali pola statistik dalam data sekaligus menghasilkan kembali data dengan pola tersebut, mereka mampu mengenali apa yang mereka mampu produksi. Mekanika kuantum terkenal dapat memproduksi pola-pola tidak wajar dalam data, yang sulit direproduksi dengan komputer klasik, sehingga dari observasi sebelumnya, komputer kuantum berpotensi mampu mengenali pola-pola data yang sulit dikenali secara klasik[8]. Pada tahun 2009, Harrow, Hassidim, dan Lloyd mengajukan algoritma kuantum, biasa dikenal dengan nama algoritma HHL, yang dapat menyelesaikan persamaan linear dengan percepatan eksponensial jika dibandingkan solusi komputasi klasik[9]. Kemudian pada tahun 2014, Lloyd et al. mengajukan algoritma versi kuantum dari *Principal Component Analysis* yang memiliki percepatan eksponensial jika dibandingkan apa yang bisa dilakukan komputer klasik[10].

Meski kemajuan dalam bidang komputasi kuantum terbilang pesat, dengan pemain-pemain besar seperti IBM, Google, Amazon, dan Microsoft, dan puluhan *start-up* lainnya berlomba-lomba menjadi yang pertama membangun komputer kuantum, keadaan perangkat keras komputer kuantum yang tersedia saat ini masih sangat jauh dari yang dibutuhkan untuk mengeksekusi algoritma yang terlalu

kompleks. Kondisi komputer kuantum saat ini masih terbatas pada jumlah qubit yang sedikit (sekitar 50 qubit) dan jumlah operasi gerbang kuantum yang terbatas karena masih memiliki derau dalam prosesnya, John Preskill menciptakan istilah untuk mengacu pada era komputer kuantum saat ini yaitu *Noisy Intermediate Scale Quantum* (NISQ)[11]. Algoritma-algoritma kuantum yang telah disebutkan sebelumnya masih tidak memungkinkan untuk dijalankan pada komputer kuantum saat ini.

Batasan ini kemudian menjadi pendorong berkembangnya Algoritma Kuantum Variasional (AKV), yaitu algoritma kuantum yang memanfaatkan algoritma optimisasi klasik untuk mengoptimisasi parameter pada sirkuit kuantum variasional (SKV) agar dapat menghasilkan keadaan kuantum tertentu[12]. Algoritma ini memiliki struktur yang mirip sekali dengan jaringan saraf tiruan (JST) klasik yaitu memanfaatkan algoritma optimisasi untuk mengoptimisasi parameter di dalam suatu *black box* agar dapat menghasilkan keluaran tertentu. Metode ini berkembang pesat dan telah banyak ditemukan aplikasi-aplikasi di berbagai bidang. Dalam pembelajaran mesin, telah ditemukan adaptasi algoritma HHL menggunakan SKV[13], telah diajukan juga algoritma-algoritma yang menggunakan SKV untuk klasifikasi data[14].

Metode AKV yang memiliki banyak potensi namun terbilang masih belia membuka pintu peluang-peluang eksplorasi baik dalam pengembangan untuk aplikasi baru atau peningkatan dan perbaikan dari aplikasi yang telah ada. Penelitian tugas akhir ini berfokus pada pengembangan dan perbaikan aplikasi metode AKV untuk klasifikasi citra.

1.2 Tujuan Penelitian

Tujuan penelitian pada tugas akhir ini adalah merancang pengklasifikasi citra MNIST dengan komputasi kuantum berbasis metode AKV. Terdapat beberapa target sebagai berikut:

1. Algoritma masih dalam skala NISQ, yaitu algoritma yang menggunakan jumlah qubit terbatas (sekitar 15-50 qubit) dan jumlah gerbang terbatas, terutama gerbang 2-qubit.

2. Mencapai akurasi klasifikasi biner yang lebih baik dari penelitian-penelitian terkait sebelumnya (Subbab 1.3).
3. Algoritma dapat diekspansi ke klasifikasi banyak kelas.

Dan sasaran yang ingin dicapai dari penelitian tugas akhir ini adalah sebagai berikut:

1. Mengimplementasikan algoritma kuantum *Data Re-uploading Classifier* untuk klasifikasi citra MNIST.
2. Merancang dan mengimplementasikan arsitektur pengklasifikasi yang merupakan modifikasi dari *Data Re-uploading Classifier*.

Merancang dan mengimplementasikan arsitektur sirkuit kuantum yang mampu melakukan proses konvolusi citra serupa dengan proses konvolusi pada komputasi klasik.

1.3 Penelitian Terkait

Beberapa upaya dalam mengklasifikasikan himpunan data MNIST dengan komputasi kuantum telah dilakukan sebelumnya. Dari studi literatur, penulis menemukan beberapa makalah penelitian (*paper*) yang cukup sering disitasi dalam bidang akademik pembelajaran mesin kuantum. Penulis juga menemukan sebuah thesis mahasiswa S2 di luar negeri yang bertujuan serupa dengan penelitian ini. Berikut ini adalah rangkuman hasil dari penelitian-penelitian terkait tersebut:

1. E. Farhi dan H. Neven dalam makalah penelitiannya yang berjudul *Classification with Quantum Neural Networks on Near Term Processors*[15][16] telah berhasil melakukan klasifikasi biner dengan sirkuit kuantum. Arsitektur yang mereka ajukan membutuhkan 17 qubit dengan 32 parameter yang perlu dioptimisasi. Dimensi citra direduksi dengan metode interpolasi bilinear dari 28 x 28 menjadi 4 x 4. Terdapat kelemahan pada reduksi dimensi dengan metode ini, yaitu dua citra yang berbeda (bahkan berbeda labelnya) dapat tereduksi menjadi citra yang sama. Penulis makalah tersebut juga mengakui dalam makalahnya bahwa dari 5500 sampel label pertama dan 5500 sampel label kedua, hanya terdapat 797 sampel label pertama dan 617 sampel label kedua yang unik, bahkan ditemukan terdapat 197 sampel unik yang berasal dari dua label yang

berbeda (artinya terdapat sampel-sampel data dari kelas yang berbeda yang tereduksi menjadi citra yang sama). Akurasi yang berhasil tercapai adalah sebesar 88% pada evaluasi data latih dan 90% pada evaluasi data uji, hasil yang relatif terbilang rendah jika dibandingkan dengan JST klasik yang umumnya mampu mencapai 99% akurasi.

2. Skolik et al. dalam makalah penelitian berjudul *Layerwise learning for quantum neural networks*[17] mengajukan skema proses latih pengklasifikasi kuantum yang diberi nama *layerwise learning*. Metode reduksi dimensi citra yang digunakan sama dengan salah satu metode yang juga digunakan pada penelitian ini, yaitu *Principal Component Analysis* (PCA). Arsitektur yang mereka ajukan membutuhkan jumlah qubit sebanyak fitur yang dimiliki oleh sampel data. Pada makalah tersebut, digunakan sebanyak 10 komponen prinsip dari hasil PCA sehingga dibutuhkan 10 qubit. Hasil klasifikasi biner yang didapatkan mencapai akurasi sebesar 73% pada evaluasi data uji, nilai yang lebih rendah dari makalah sebelumnya.
3. Mardirosian dalam thesis S2-nya yang berjudul *Quantum-enhanced Supervised Learning with Variational Quantum Circuits*[18] mengajukan arsitektur pengklasifikasi biner yang juga membutuhkan jumlah qubit sebanyak jumlah fitur sampel data, namun dengan sirkuit yang berbeda. Metode reduksi dimensi citra yang digunakan juga adalah PCA. Hasil akurasi klasifikasi biner yang menggunakan 2 qubit (2 komponen prinsip) mencapai 96% pada evaluasi data latih dan 89% pada evaluasi data uji. Kemudian pengujian dengan 3 qubit (3 komponen prinsip) mencapai akurasi 97% pada evaluasi data latih dan 90% pada evaluasi data uji. Hasil ini lebih baik daripada makalah kedua namun tidak jauh berbeda dari makalah pertama, relatif masih rendah jika dibandingkan dengan JST klasik.

Dari makalah-makalah penelitian ini, terlihat hasil akurasi klasifikasi dari skema atau arsitektur yang diajukan masih kurang memuaskan. Kemudian, penggunaan jumlah qubit bersifat linear terhadap fitur data, semakin banyak fitur

data maka semakin banyak qubit yang dibutuhkan, ini menjadi masalah karena pada era komputasi kuantum NISQ jumlah qubit masih sangat terbatas.

Pada tahun 2020, Pérez-Salinas et al. mengajukan skema pengklasifikasi kuantum yang secara teori cukup menggunakan 1 qubit terlepas berapapun jumlah fitur sampel data, skema ini diberi nama *Data Re-uploading Classifier* (DRC)[19]. Penelitian ini akan berfokus pada utilisasi skema tersebut untuk klasifikasi MNIST. Penulis juga mengajukan beberapa modifikasi terhadap skema tersebut yang berdampak pada kenaikan akurasi.

1.4 Batasan Penelitian

Batasan penelitian yang digunakan pada penelitian ini adalah sebagai berikut:

1. Model komputasi kuantum yang digunakan adalah Komputasi Kuantum Model Gerbang.
2. Penelitian tidak meninjau sistem pengklasifikasi hibrid, yaitu sistem pengklasifikasi yang menggunakan sebagian sirkuit kuantum dan sebagian Jaringan Saraf Tiruan klasik.
3. Algoritma ditulis ke dalam kode sumber dalam bahasa pemrograman Python.
4. Algoritma kuantum dieksekusi menggunakan simulator *statevector* komputasi kuantum PennyLane[20] pada komputer klasik.
5. Simulasi komputasi kuantum berlangsung secara ideal tanpa penggunaan derau buatan (*simulated noise*).
6. Himpunan data citra yang digunakan adalah himpunan data MNIST[21].
7. Optimisasi parameter sirkuit kuantum dilakukan dengan modul Keras[22] dan TensorFlow[23].
8. *Principal component analysis* dilakukan dengan modul Scikit-learn[24].

1.5 Sistematika Penulisan

Laporan penelitian tugas akhir ini disusun dengan sistematika sebagai berikut:

BAB 1 PENDAHULUAN

Bab ini terdiri dari latar belakang, tujuan penelitian, penelitian terkait, batasan penelitian, dan sistematika penulisan.

BAB 2 TINJAUAN PUSTAKA

Bab ini berisi konsep-konsep dasar yang digunakan, dimulai dari penjelasan singkat dasar-dasar komputasi kuantum, JST, dan metode reduksi dimensi citra klasik. Kemudian diikuti dengan penjelasan mengenai pengklasifikasi kuantum dan algoritma optimisasi yang digunakan, serta metode menghitung gradien sirkuit kuantum jika sirkuit dijalankan pada komputer kuantum sesungguhnya.

BAB 3 PERANCANGAN ARSITEKTUR SISTEM

Bab ini berisi rancangan-rancangan skema yang digunakan dalam penelitian, termasuk di dalamnya rancangan skema reduksi dimensi citra dengan PCA dan konvolusi kuantum, dan rancangan skema pengklasifikasi kuantum dengan DRC dan DRC representasi biner (DRC-RB). Kemudian diikuti dengan rangkuman kombinasi skema yang diuji dalam penelitian beserta spesifikasinya.

BAB 4 HASIL DAN PEMBAHASAN

Bab ini berisi pengolahan hasil numerik yang diperoleh dari penelitian beserta pembahasan dan analisisnya. Termasuk di dalamnya hasil *explained variance* dari PCA, performa pengklasifikasi DRC dengan reduksi PCA, performa pengklasifikasi DRC-RB dengan reduksi PCA, performa pengklasifikasi DRC-RB dengan reduksi konvolusi kuantum, dan citra hasil konvolusi kuantum.

BAB 5 KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dan saran dari penelitian tugas akhir yang telah dilakukan.

BAB II

TINJAUAN PUSTAKA

2.1 Dasar Komputasi Kuantum

2.1.1 Qubit

Qubit adalah elemen komputasi terkecil dalam komputer kuantum. Secara fisis, qubit merupakan sistem mekanika kuantum dengan dua keadaan kuantum (baik yang berpasangan secara natural maupun yang berpasangan akibat dikontrol), misalnya spin elektron (up & down), polarisasi foton (vertikal & horizontal), dan tingkat energi partikel (ground state & excited state pertama). Secara matematis, dua keadaan qubit ini direpresentasikan sebagai dua basis vektor yang ortonormal. Qubit dapat dituliskan dalam bentuk notasi vektor/matriks maupun notasi Dirac/braket. Basis vektor yang paling umum digunakan adalah basis vektor $|0\rangle$ dan $|1\rangle$ sebagai berikut:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \quad (2.1)$$

Basis vektor ini membentuk ruang vektor qubit atau yang sering disebut sebagai ruang Hilbert.

Untuk merepresentasikan keadaan lebih dari satu qubit, keadaan total/gabungannya dihitung dengan perkalian tensor. Misal jika terdapat dua qubit, $|0\rangle$ dan $|1\rangle$, maka keadaan totalnya dapat dituliskan secara matematis sebagai berikut:

$$|01\rangle = |0\rangle \otimes |1\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}. \quad (2.2)$$

Qubit memiliki dua fitur yang membuatnya berbeda dari bit klasik, yaitu superposisi dan *entanglement*. Berbeda dari bit yang setiap waktunya hanya dapat berada dalam keadaan 0 atau 1, qubit dapat berada dalam dua keadaan sekaligus yang disebut sebagai superposisi. Qubit $|\Psi\rangle$ yang berada dalam keadaan superposisi dituliskan sebagai

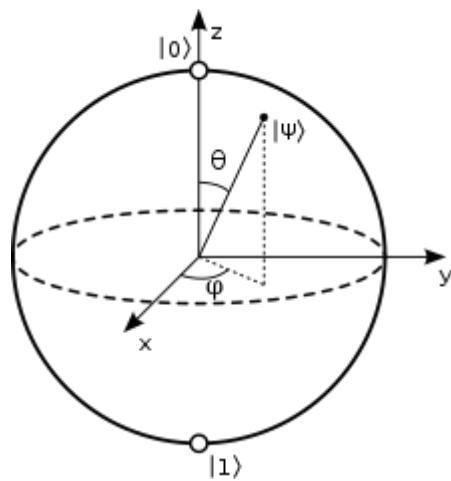
$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle \quad (2.3)$$

dengan α dan β merupakan bilangan kompleks dimana $|\alpha|^2 + |\beta|^2 = 1$.

Entanglement adalah peristiwa ketika dua atau lebih qubit berinteraksi sedemikian rupa sehingga keadaan gabungan/total dari qubit-qubit tersebut tidak dapat lagi didekomposisi untuk masing-masing qubit. Keadaan salah satu qubit bergantung pada keadaan qubit lainnya, tidak dapat lagi berdiri sendiri-sendiri. Jika diberikan suatu keadaan kuantum $|\Psi_{ent}\rangle$ yang merupakan keadaan gabungan dari N qubit, keadaan kuantum tersebut dikatakan ter-*entangle* dengan penuh jika tidak ada pasangan $\{|Q_1\rangle, |Q_2\rangle, \dots, |Q_N\rangle\} \in \{|0\rangle, |1\rangle\}$ yang memenuhi $|\Psi_{ent}\rangle = |Q_1\rangle \otimes |Q_2\rangle \otimes \dots \otimes |Q_N\rangle$.

2.1.2 Bloch Sphere

Untuk memudahkan visualisasi sebuah qubit yang menempati ruang Hilbert, representasi *Bloch sphere* sering digunakan. *Bloch sphere* adalah representasi geometri 2-sphere dengan jari-jari 1 dari keadaan kuantum sebuah qubit. Setiap titik pada permukaan bola merepresentasikan setiap keadaan kuantum murni yang mungkin dimiliki oleh sebuah qubit. Kutub utara dan kutub selatan pada permukaan bola biasanya dipilih untuk merepresentasikan vektor basis $|0\rangle$ dan $|1\rangle$.



Gambar 2.1 Representasi *Bloch sphere* dari sebuah qubit dengan keadaan kuantum $|\Psi\rangle$.
Gambar disadur dari [25].

Berdasarkan Gambar 2.1, keadaan kuantum $|\psi\rangle$ dapat dituliskan sebagai

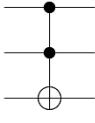
$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi} \sin\left(\frac{\theta}{2}\right)|1\rangle \quad (2.4)$$

Pada makalah penelitian ini, visualisasi data dan keadaan kuantum pada *Bloch sphere* dilakukan menggunakan modul QuTiP[26].

2.1.3 Gerbang Kuantum dan Sirkuit Kuantum

Analog dengan komputer klasik, komputer kuantum juga menggunakan gerbang-gerbang operasi untuk mengontrol dan mengubah keadaan qubit. Gerbang kuantum secara matematika dapat dideskripsikan sebagai matriks unitari. Perubahan dari keadaan kuantum mula-mula menjadi keadaan kuantum lainnya oleh gerbang kuantum sering disebut sebagai evolusi unitari. Implementasi fisis (perangkat keras) dari gerbang-gerbang kuantum ini bergantung pada implementasi fisis dari qubitnya. Tiap-tiap implementasi qubit memiliki cara masing-masing untuk menciptakan gerbang-gerbang kuantumnya.

Gerbang kuantum dapat beroperasi pada satu atau lebih qubit. Sebuah algoritma kuantum dapat dieksekusi oleh sebuah sirkuit kuantum, yaitu susunan dari beberapa gerbang kuantum yang beroperasi pada satu atau lebih qubit.

Operator	Gate(s)	Matrix
Pauli-X (X)	 	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Y (Y)		$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
Pauli-Z (Z)		$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Hadamard (H)		$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
Phase (S, P)		$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$
$\pi/8$ (T)		$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$
Controlled Not (CNOT, CX)		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
Controlled Z (CZ)		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$
SWAP		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Toffoli (CCNOT, CCX, TOFF)		$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$

Gambar 2.2 Contoh beberapa gerbang kuantum dengan representasi sirkuit (Gate) dan matriksnya (Matrix). Gambar disadur dari [27].

Keadaan qubit setelah dioperasikan dengan sebuah gerbang kuantum adalah hasil perkalian matriks gerbang kuantum dengan keadaan kuantum mula-mula dari qubit tersebut. Misalkan operasi gerbang Hadamard (H) pada qubit $|0\rangle$ akan menghasilkan sebuah qubit dalam keadaan superposisi berikut:

$$H|0\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}} \left(\begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) = \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle. \quad (2.5)$$

Jika kemudian gerbang 2-qubit CNOT beroperasi pada qubit $\frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle$ dan qubit $|0\rangle$, akan dihasilkan

$$CNOT((H|0\rangle) \otimes |0\rangle) = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle. \quad (2.6)$$

yang dikenal sebagai keadaan Bell, keadaan ter-*entangle* dengan penuh yang paling sederhana. Gerbang CNOT adalah gerbang yang paling umum digunakan untuk men-*entangle* dua qubit. Titik hitam pada representasi skematik gerbang CNOT menandakan kontrol, dan lambang ‘tanda tambah’ di dalam lingkaran menandakan target. Gerbang CNOT akan membalik keadaan kuantum qubit target dari $|0\rangle$ menjadi $|1\rangle$ dan sebaliknya apabila qubit kontrol berada dalam keadaan $|1\rangle$.

Terdapat juga gerbang kuantum yang bersifat parametrik, yaitu operasinya bergantung pada parameter yang ditetapkan pada gerbang tersebut. Contoh-contoh gerbang parametrik satu parameter misalnya gerbang RX, RY, dan RZ dengan bentuk matriks sebagai berikut:

$$\begin{aligned} RX(\phi) &= e^{-\frac{i\phi\sigma_x}{2}} = \begin{bmatrix} \cos\left(\frac{\phi}{2}\right) & -i\sin\left(\frac{\phi}{2}\right) \\ -i\sin\left(\frac{\phi}{2}\right) & \cos\left(\frac{\phi}{2}\right) \end{bmatrix} \\ RY(\phi) &= e^{-\frac{i\phi\sigma_y}{2}} = \begin{bmatrix} \cos\left(\frac{\phi}{2}\right) & -\sin\left(\frac{\phi}{2}\right) \\ \sin\left(\frac{\phi}{2}\right) & \cos\left(\frac{\phi}{2}\right) \end{bmatrix} \\ RZ(\phi) &= e^{-\frac{i\phi\sigma_z}{2}} = \begin{bmatrix} e^{-\frac{i\phi}{2}} & 0 \\ 0 & e^{\frac{i\phi}{2}} \end{bmatrix}. \end{aligned} \quad (2.7)$$

Dalam visualisasi Bloch sphere, gerbang $RX(\phi)$, $RY(\phi)$, dan $RZ(\phi)$ masing-masing merotasikan vektor qubit sebesar sudut ϕ terhadap sumbu putar x, y, dan z. Kemudian terdapat juga gerbang parametrik tiga parameter yang merotasikan vektor qubit pada tiga sumbu putar dengan nilai sudut yang berbeda-beda. Misalnya gerbang rotasi $R(\phi, \theta, \omega)$ yang operasinya bergantung pada parameter ϕ , θ , dan ω dengan bentuk matriks sebagai berikut:

$$R(\phi, \theta, \omega) = \begin{bmatrix} e^{-i(\phi+\omega)/2}\cos(\theta/2) & -e^{i(\phi-\omega)/2}\sin(\theta/2) \\ e^{-i(\phi-\omega)/2}\sin(\theta/2) & e^{i(\phi+\omega)/2}\cos(\theta/2) \end{bmatrix}. \quad (2.8)$$

Dalam representasi *Bloch sphere*, gerbang ini merotasikan sebuah qubit sebesar ϕ terhadap sumbu z, diikuti dengan rotasi sebesar θ terhadap sumbu y, dan diakhiri dengan rotasi sebesar ω terhadap sumbu z.

2.1.4 Pengukuran

Setelah proses komputasi selesai dilakukan, qubit-qubit yang terlibat dalam komputasi perlu diukur untuk mendapatkan hasil komputasi. Bit klasik tidak akan berubah nilainya ketika sebelum diukur maupun setelah diukur, misal jika representasi bit 1 adalah tegangan 5 volt dan bit 0 adalah tegangan 0 volt, maka bit 1 akan memiliki tegangan 5 volt baik sebelum maupun setelah diukur. Qubit akan kehilangan keadaan superposisinya dan berubah menjadi salah satu keadaan basis vektornya ketika kita mengukur nilai qubit tersebut, peristiwa ini dikenal dengan istilah "keruntuhan fungsi gelombang". Ini artinya, setiap qubit hanya dapat diukur satu kali karena setelah pengukuran qubit tersebut memiliki keadaan yang berbeda dengan sebelum pengukuran. Selain itu, pengukuran dua qubit terpisah dalam keadaan superposisi yang memiliki keadaan kuantum sama, misalnya seperti pada persamaan (2.3), dapat memberikan hasil pengukuran yang berbeda karena qubit tersebut dapat memberikan hasil pengukuran $|0\rangle$ atau $|1\rangle$.

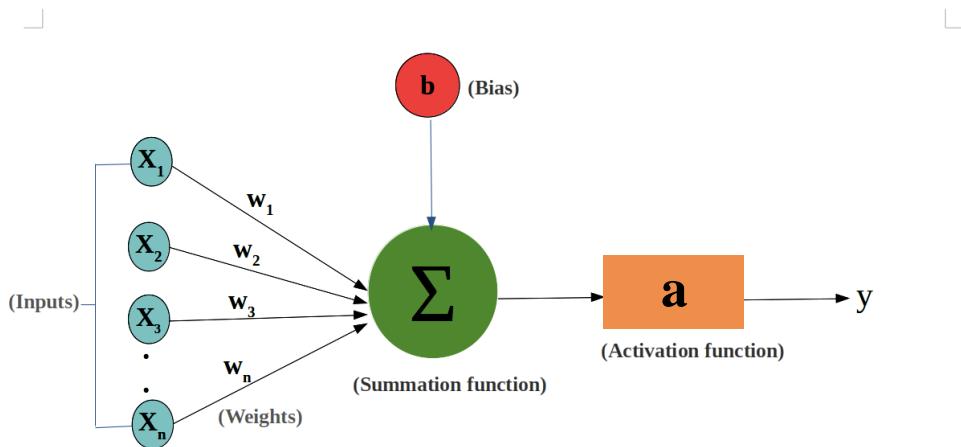
Oleh karena itu, pengukuran perlu dilakukan beberapa kali untuk mendapatkan statistik hasil pengukuran untuk kemudian dihitung nilai ekspektasinya. Pengukuran nilai ekspektasi dari qubit dilakukan terhadap suatu *observable* kuantum O yang direpresentasikan sebagai matriks Hermitian. *Observable* yang paling umum digunakan dalam komputasi kuantum adalah gerbang Pauli. Dari statistik pengukuran ini akan didapatkan estimasi nilai λ yang nilai idealnya secara matematis dapat dituliskan sebagai

$$\lambda = \langle \Psi | O | \Psi \rangle = \langle O \rangle. \quad (2.9)$$

Namun karena penelitian ini menggunakan simulator ideal, keadaan kuantum $|\Psi\rangle$ direpresentasikan dalam bentuk matriks oleh komputer klasik sehingga nilai hasil pengukuran λ dapat diperoleh secara eksak dengan perkalian matriks berdasarkan persamaan (2.9).

2.2 Jaringan Saraf Tiruan (JST)

JST adalah suatu algoritma komputasi yang didesain untuk memperoleh relasi utama antar data dalam sebuah himpunan data. JST terinspirasi dari cara kerja neuron pada otak. Sebuah JST tersusun atas serangkaian neuron-neuron buatan yang sering disebut sebagai simpul, struktur terkecil dalam JST. Ilustrasi dari sebuah simpul dapat dilihat pada Gambar 2.3.



Gambar 2.3 Proses komputasi dari sebuah simpul dalam JST. Gambar disadur dari [28] dengan sedikit perubahan pada *activation function*.

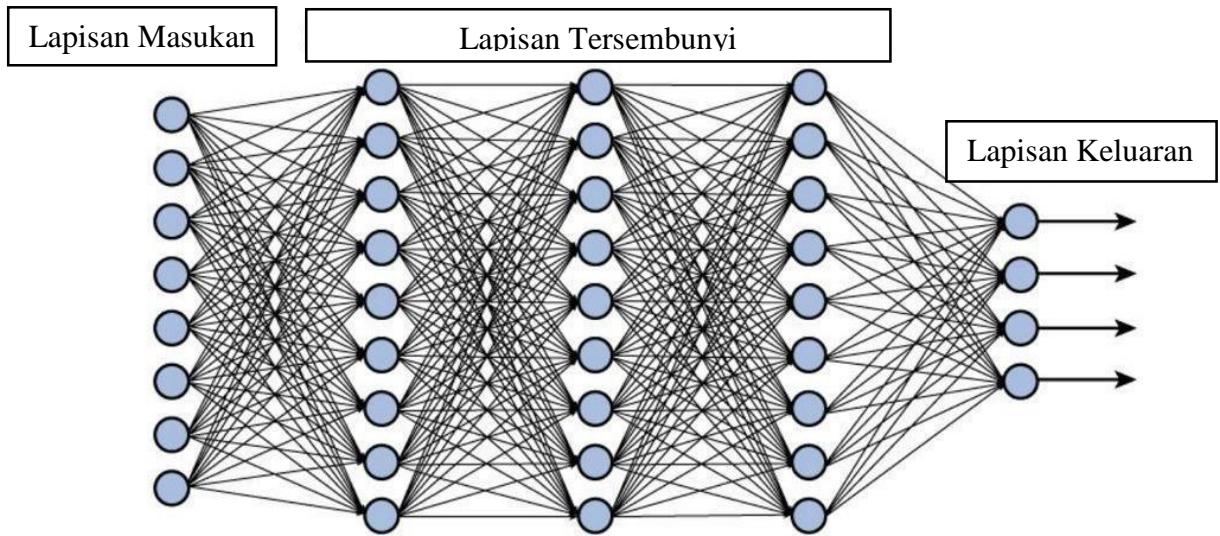
Misalkan sebuah sampel data x yang memiliki fitur sebanyak N dari himpunan data X dilewatkan pada sebuah simpul sebagai masukan (*input*). Maka keluaran dari simpul adalah sebagai berikut:

$$y = a \left(\left[\sum_{n=1}^N w_n x_n \right] + b \right) \quad (2.10)$$

dengan w dan b adalah parameter yang disebut bobot (*weight*) dan bias, a merupakan fungsi aktivasi (*activation function*), yaitu sebuah fungsi non-linear.

Rangkaian beberapa simpul yang menerima masukan yang sama disebut sebagai lapisan. Suatu lapisan dapat menerima keluaran dari lapisan sebelumnya. Susunan berantai dari lapisan-lapisan inilah yang disebut sebagai JST. Secara umum, lapisan dalam JST tergolong dalam 3 kategori yaitu lapisan masukan,

lapisan tersembunyi, dan lapisan keluaran. Lapisan masukan merupakan lapisan yang terbentuk atas sebuah sampel data x dengan simpul sebanyak fitur dalam sampel data tersebut. Lapisan tersembunyi terdiri atas simpul-simpul seperti pada persamaan (2.10). Jumlah simpul dalam lapisan tersembunyi dapat dipilih sesuai dengan kebutuhan. Lapisan tersembunyi ini dapat diulangi beberapa kali sebanyak yang dibutuhkan. Lapisan keluaran merupakan lapisan yang memberikan hasil algoritma JST. Gambar 2.4 menunjukkan ilustrasi dari contoh arsitektur JST.



Gambar 2.4 Contoh arsitektur sebuah JST dengan 3 lapisan tersembunyi. Disadur dari [29] dengan lokalisasi ke Bahasa Indonesia.

Untuk kasus klasifikasi, keluaran dari setiap simpul pada lapisan keluaran merupakan nilai prediksi untuk setiap kelas. Sehingga untuk kasus klasifikasi biner (2 kelas) biasanya lapisan keluaran hanya memiliki satu simpul dengan fungsi aktivasi yang biasanya digunakan adalah fungsi *sigmoid* sebagai berikut:

$$a(x) = \frac{1}{1 + e^{-x}} \quad (2.11)$$

dengan x adalah masukan ke simpul pada lapisan terakhir. Fungsi *sigmoid* menghasilkan keluaran yang berada dalam rentang (0, 1) sehingga kelas pertama dapat direpresentasikan dengan hasil $< 0,5$ dan kelas kedua direpresentasikan dengan hasil $\geq 0,5$.

Untuk lapisan tersembunyi, fungsi aktivasi yang umum digunakan adalah *Rectified Linear Unit* (ReLU), yang secara matematis dapat dituliskan sebagai berikut:

$$a_i(x) = \begin{cases} x, & \text{jika } x > 0 \\ 0, & \text{jika } x \leq 0 \end{cases}. \quad (2.12)$$

Sedangkan pada kasus klasifikasi banyak kelas, diperlukan satu simpul untuk setiap kelas pada lapisan keluaran. Fungsi aktivasi yang biasanya digunakan adalah fungsi *softmax* yang dapat dituliskan sebagai berikut:

$$a_i(\mathbf{x}) = \frac{e^{x_i}}{\sum_{j=1}^C e^{x_j}}, \quad \mathbf{x} = (x_1, x_2, \dots, x_C)^T \in \mathbb{R}^C \quad (2.13)$$

dengan a_i adalah keluaran dari simpul ke- i pada lapisan keluaran, C adalah jumlah kelas, dan \mathbf{x} adalah vektor yang terbentuk dari kumpulan hasil proses sumasi setiap simpul pada lapisan terakhir (sebelum memasuki fungsi aktivasi). Nilai keluaran tiap simpul merepresentasikan persentase keyakinan dari model untuk mengklasifikasikan sampel data ke dalam kelas tertentu. Misalkan keluaran simpul pertama dari lapisan keluaran memiliki nilai tertinggi, artinya model mengklasifikasikan sampel data tersebut ke dalam kelas pertama.

Keluaran dari lapisan keluaran kemudian digunakan untuk menghitung suatu fungsi biaya J . Fungsi biaya adalah sebuah fungsi yang nilainya menunjukkan seberapa baik performa klasifikasi dari JST, semakin kecil nilai fungsi biaya menunjukkan semakin baik performa klasifikasinya. Ketika fungsi biaya mencapai nilai terkecil yang mungkin, maka JST akan memiliki performa akurasi klasifikasi 100%.

Parameter w dan b dioptimisasi dengan suatu algoritma optimisasi, misalnya optimisasi Adam yang dibahas pada subbab 2.5, untuk memminimumkan fungsi biaya ini. Proses memminimumkan fungsi biaya dengan suatu algoritma optimisasi disebut sebagai proses latih/*training*. Setelah proses latih dirasa cukup, JST biasanya diuji performanya dengan himpunan data baru yang tidak digunakan untuk proses latih, biasa disebut himpunan data uji, dengan cara mengklasifikasikan semua sampel data di dalam himpunan data baru ini dengan JST tanpa

mengoptimisasi parameternya (semua parameter JST dibuat konstan). Proses mengevaluasi performa JST dengan himpunan data uji tanpa optimisasi parameter disebut sebagai proses uji/*testing*.

2.3 Reduksi Dimensi Citra

Setiap nilai piksel pada tiap kanal dalam sebuah sampel citra merupakan fitur dari sampel tersebut. Sehingga data citra yang jumlah pikselnya relatif kecil sekalipun dapat memiliki jumlah fitur yang sangat banyak. Oleh sebab itu, untuk mengklasifikasikan citra, umumnya dilakukan proses *downsampling* untuk mereduksi dimensi dari fitur citra tersebut.

2.3.1 Principal Component Analysis (PCA)

PCA adalah teknik reduksi dimensi dengan memproyeksikan data ke arah yang memaksimalkan variansi fitur dan membuat kovariansi antar fitur menjadi nol. Jika tedapat himpunan data X yang merupakan matriks berukuran $M \times N$ dengan M (baris) adalah total sampel data dan N (kolom) adalah jumlah fitur data, PCA mencari kombinasi linear dari kolom matriks X dengan variansi maksimum[30].

Data dengan dimensi fitur yang tinggi sering kali memiliki banyak fitur-fitur yang mubazir, yaitu fitur-fitur yang tidak membantu meningkatkan performa klasifikasi karena fitur-fitur tersebut memiliki korelasi yang tinggi dengan fitur-fitur lainnya. Korelasi memiliki hubungan linear dengan kovariansi, semakin tinggi kovariansi antar dua fitur, maka semakin tinggi pula korelasi antara kedua fitur tersebut. Persamaan (2.14) menunjukkan hubungan ini.

$$\text{Korelasi} = \frac{\text{Kovariansi}(A, B)}{\sigma_A \sigma_B} \quad (2.14)$$

Oleh sebab itu, dengan memproyeksikan data pada arah vektor yang membuat kovariansi menjadi nol, hasil proyeksi ini akan menyisakan fitur-fitur yang tidak saling berkorelasi.

Selain itu, fitur yang umumnya menentukan klasifikasi data adalah fitur-fitur dengan variansi yang tinggi. Fitur dengan variansi rendah tidak mengandung nilai perbedaan yang cukup untuk dijadikan patokan dalam membedakan sampel

data ke dalam dua atau lebih kelas yang berbeda. Sehingga, setelah membuat kovariansi antar data bernilai nol, fitur-fitur dapat diseleksi berdasarkan nilai variansinya. Fitur dengan nilai variansi yang terlalu kecil relatif terhadap variansi fitur lainnya dapat dibuang dan tidak digunakan untuk klasifikasi. Persentase variansi sebuah fitur terhadap total variansi tiap fitur biasa disebut sebagai *explained variance(EV)*.

Berikut ini adalah algoritma dari PCA untuk data citra:

1. Diberikan himpunan data citra X sebanyak M sampel dengan tiap sampel x memiliki kanal sebanyak K dan berukuran $H \times V$. Setiap sampel citra ini diratakan menjadi sebuah vektor dengan dimensi $N = K \times H \times V$.
2. Setiap vektor dari sampel ini kemudian disusun membentuk matriks D sebagai berikut:

$$D = \begin{bmatrix} x_1^1 & \dots & x_1^N \\ \vdots & \ddots & \vdots \\ x_M^1 & \dots & x_M^N \end{bmatrix}. \quad (2.15)$$

dengan x_m^n memiliki makna nilai fitur ke- n dari sampel data ke- m .

3. Rata-rata dan standar deviasi dari setiap fitur (kolom) pada matriks D kemudian dihitung dengan persamaan berikut:

$$\mu_j = \frac{\sum_{i=1}^M x_i^j}{M}, \quad \sigma_j = \sqrt{\frac{\sum_{i=1}^M (x_i^j - \mu_j)^2}{M}} \quad (2.16)$$

4. Setiap sampel kemudian dinormalisasi agar setiap fitur memiliki rata-rata 0 dan standar deviasi 1. ϵ adalah suatu konstanta bernilai kecil, biasanya 10^{-8} , untuk menjaga stabilitas numerik (menghindari pembagian dengan nol).

$$x_i^j = \frac{x_i^j - \mu_j}{\sigma_j + \epsilon} \quad (2.17)$$

5. Hitung matriks kovarians $C = \begin{bmatrix} C_{11} & \cdots & C_{1N} \\ \vdots & \ddots & \vdots \\ C_{N1} & \cdots & C_{NN} \end{bmatrix}$ dengan tiap komponen dalam matriks tersebut dihitung dengan persamaan untuk mencari kovariansi sebagai berikut:

$$C_{pq} = \frac{\sum_{i=1}^M (x_i^p - \mu_p)(x_i^q - \mu_q)}{M}. \quad (2.18)$$

6. Setelah didapatkan matriks C , akan dicari himpunan pasangan vektor eigen v dan nilai eigen λ dari matriks ini. Akan diperoleh sebanyak N pasangan vektor eigen dengan nilai eigen. Setiap vektor eigen ini menunjukkan arah fitur yang memaksimalkan variansi, setiap vektor juga saling ortogonal dengan vektor eigen lainnya sehingga himpunan data yang diproyeksikan pada vektor-vektor ini akan memiliki kovariansi nol. Vektor eigen kemudian diurutkan berdasarkan nilai dari nilai eigen pasangannya, dari yang terbesar sampai terkecil. Nilai eigen ini menunjukkan besar variansi data yang terwakili oleh arah vektor eigen pasangannya.

$$v_{urut} = \underbrace{\{v_1, v_2, v_3, \dots, v_N\}}_N \quad (2.19)$$

Vektor-vektor eigen ini juga disebut sebagai komponen-komponen prinsip.

7. Tidak semua vektor eigen berpasangan dengan nilai eigen yang cukup besar. Sebagian besar vektor eigen akan memiliki pasangan nilai eigen yang relatif kecil. Ini karena tidak semua fitur pada data merupakan fitur yang penting. Selanjutnya, pilih jumlah komponen prinsip yang ingin di pertahankan sebanyak $p < N$ mulai dari yang nilai eigennya paling besar. Kumpulan dari komponen prinsip ini kemudian disusun membentuk matriks P .

$$P = \underbrace{[v_1, v_2, v_3, \dots, v_p]}_p \quad (2.20)$$

8. Proyeksikan himpunan data pada matriks P . Hasil yang didapatkan merupakan matriks D' , yaitu matriks himpunan data baru dengan jumlah sampel tetap sebanyak M namun dengan jumlah fitur sebanyak p .

$$D' = D \cdot P \quad (2.21)$$

Explained variance(EV) milik fitur ke- i , dapat dihitung dengan persamaan berikut:

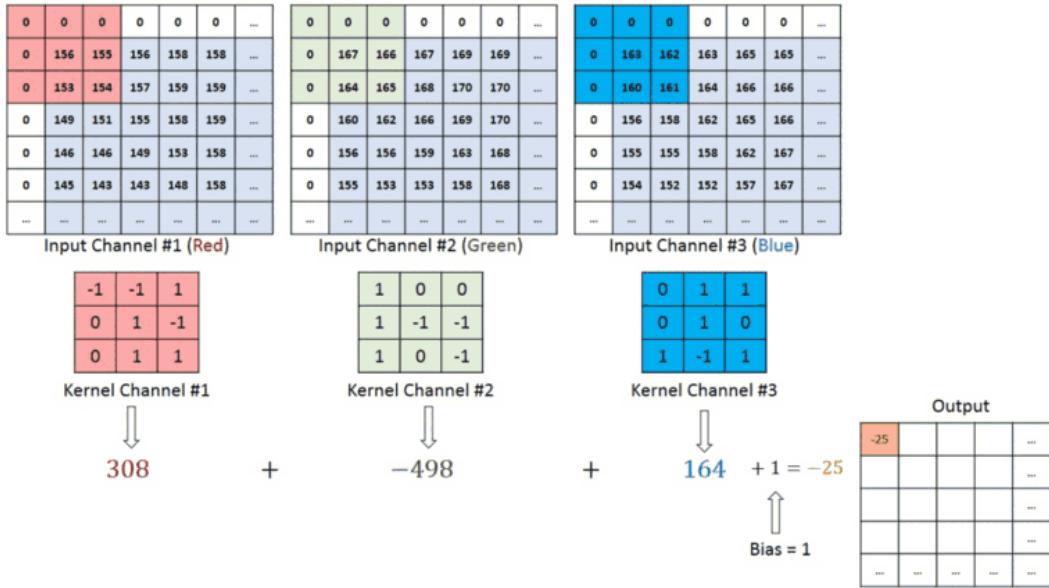
$$EV = \frac{\sigma_i^2}{\sum_{j=1}^N \sigma_j^2} \times 100\% = \frac{\lambda_i}{\sum_{j=1}^N \lambda_j} \times 100\%. \quad (2.22)$$

Nilai *EV* biasa dijadikan patokan dalam menentukan nilai p . Nilai p menentukan seberapa besar persentase variansi yang tertangkap oleh himpunan data baru (D').

Salah satu kelemahan PCA adalah jika setelah kovariansi antar fitur dibuat nol, seluruh fitur memiliki nilai *EV* yang relatif besar. Misalkan terdapat dua himpunan data dengan masing-masing 5 fitur. Himpunan data pertama memiliki *EV* tiap fitur sebagai berikut: [0,5; 0,3; 0,15; 0,03; 0,02]. Dan himpunan data kedua memiliki *EV* tiap fitur sebagai berikut: [0,245; 0,215; 0,19; 0,185; 0,165]. Pada himpunan data pertama, mudah sekali terlihat bahwa hanya 3 fitur pertama yang penting, sisa dua fitur lainnya dapat dibuang karena memiliki nilai *EV* yang jauh lebih kecil dibandingkan ketiga fitur pertama. Sehingga untuk himpunan data pertama, fitur data dapat direduksi dari 5 fitur menjadi 3 fitur dengan PCA. Namun untuk himpunan data kedua, tiap fitur memiliki nilai *EV* yang tidak jauh berbeda, tidak terdapat fitur yang memiliki *EV* yang sangat kecil. Himpunan data seperti ini jika direduksi dimensinya dengan PCA, himpunan data baru yang dihasilkan akan kehilangan variansi data yang cukup signifikan.

2.3.2 Konvolusi dan *Pooling*

Salah satu metode reduksi dimensi data citra yang juga sering digunakan adalah teknik konvolusi dan *pooling*. Teknik ini memungkinkan satu himpunan parameter untuk menangkap fitur-fitur penting dari citra, mulai dari fitur tingkat rendah, misalnya garis vertikal, garis horizontal, dan lingkaran, sampai ke fitur tingkat tinggi, misalnya gambar roda, mata, hidung, dan lain-lain. Sehingga teknik ini sering juga disebut proses ekstraksi fitur.

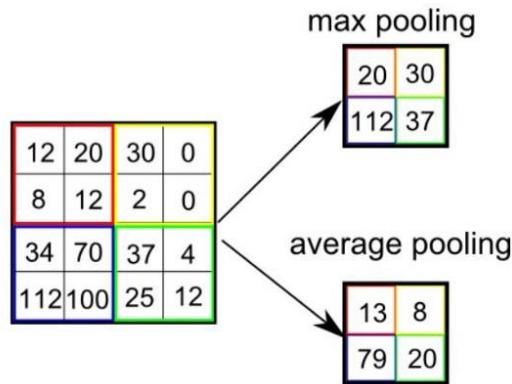


Gambar 2.5 Ilustrasi operasi konvolusi pada citra berkanal 3 (RGB). Gambar disadur dari [31].

Proses konvolusi dimulai dengan memilih jumlah filter F dan ukuran/dimensi tiap filter $\rho \times \rho$ yang ingin digunakan. Jumlah kanal dari data citra yang ingin dikonvolusi akan menentukan jumlah parameter filter yang diperlukan. Misalkan data citra memiliki jumlah kanal sebanyak K , maka dibutuhkan filter berupa matriks parameter bobot W dengan dimensi $\rho \times \rho \times K \times F$ dan sebuah parameter bias b . Setiap kanal citra akan dikonvolusi dengan matriks $\rho \times \rho \times 1$ yang berbeda. Konvolusi berlangsung dengan melakukan perkalian antar elemen matriks antara matriks filter dengan piksel-piksel pada citra kemudian hasil perkaliannya dijumlahkan. Matriks ini kemudian bergeser sejumlah piksel tertentu (ditentukan oleh parameter *stride s*) dan melakukan perkalian elemen matriks lagi. Proses ini diulangin sampai seluruh piksel pada citra telah mengalami perkalian antar elemen matriks dengan filter. Secara matematis, proses konvolusi untuk satu filter pada citra akan menghasilkan sebuah citra yang memiliki satu kanal dengan nilai tiap pikselnya sebagai berikut:

$$\chi'_{i,j} = b + \sum_{r=1}^K \sum_{p=0}^{\rho-1} \sum_{q=0}^{\rho-1} \chi_{si+p, sj+q, r} W_{p,q,r} \quad (2.23)$$

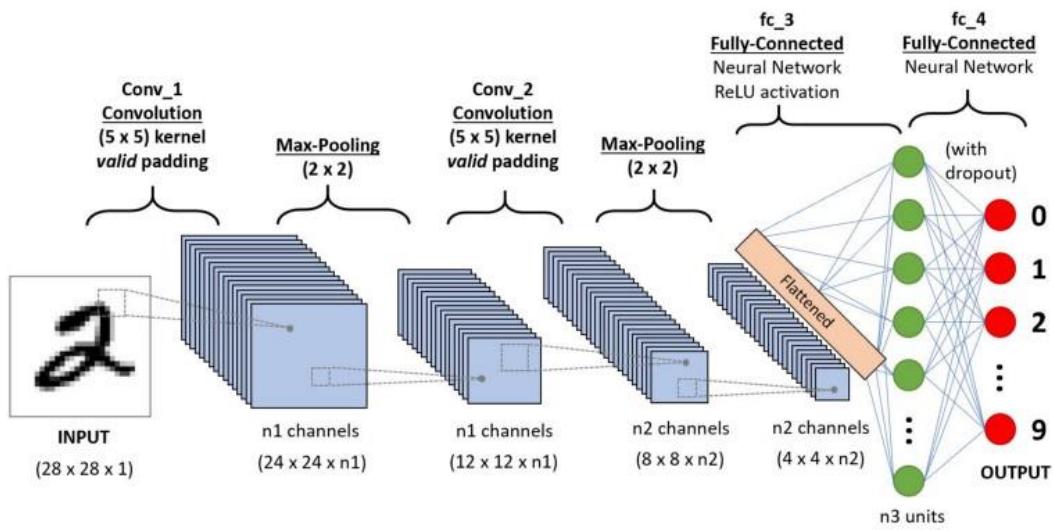
dengan $\chi'_{i,j}$ adalah nilai piksel citra hasil konvolusi pada koordinat (i,j) , $\chi_{i,j,k}$ adalah nilai piksel citra masukan pada koordinat (i,j) dan kanal ke- k , $W_{i,j,k}$ adalah nilai parameter bobot dengan koordinat (i,j,k) pada filter yang sedang digunakan, dan a adalah fungsi aktivasi yang biasanya non-linear. Koordinat piksel dimulai dari piksel paling pojok kiri atas dengan koordinat $(0, 0)$. Tanpa *padding*, jika citra masukan memiliki dimensi $H \times V$ dengan jumlah kanal K , maka citra hasil konvolusi citra masukan tersebut dengan filter berdimensi $\rho \times \rho \times K$ sebanyak F dengan *stride* s akan memiliki dimensi $\left(\frac{H-\rho}{s} + 1\right) \times \left(\frac{V-\rho}{s} + 1\right)$ dengan jumlah kanal sebanyak F .



Gambar 2.6 Ilustrasi operasi *pooling* dengan filter berukuran 2×2 dengan *stride* = 2 pada salah satu kanal citra berdimensi 4×4 . Gambar disadur dari [31].

Proses *pooling* juga dilakukan untuk mereduksi dimensi citra namun proses ini tidak memiliki parameter dan tidak melakukan komputasi perkalian antar elemen matriks sehingga beban komputasi lebih ringan daripada konvolusi. Sama seperti konvolusi, *pooling* diawali dengan menentukan ukuran filter $\rho \times \rho$. Bedanya dengan konvolusi adalah jumlah filter pada *pooling* sama dengan jumlah kanal citra masukan, sehingga setiap filter bekerja pada kanal yang berbeda-beda. Terdapat dua jenis *pooling* yang umum digunakan, yaitu *max pooling* dan *average pooling*. Pada *max pooling*, filter menghasilkan nilai keluaran berupa nilai piksel maksimum dari seluruh piksel yang tertutupi oleh filter. Sedangkan pada *average pooling*, filter menghasilkan nilai keluaran berupa rata-rata dari nilai seluruh piksel yang tertutupi oleh filter. Proses ini dilakukan berulang-ulang dengan tiap prosesnya menggeser filter sejauh sejumlah piksel tertentu yang ditentukan oleh

parameter *stride*. Ilustrasi proses *pooling* dapat dilihat pada Gambar 2.6. Citra hasil *pooling* akan memiliki jumlah kanal yang sama dengan jumlah kanal citra masukan. Jika citra masukan memiliki dimensi $H \times V$, maka hasil dari *pooling* dengan ukuran filter $\rho \times \rho$ dan *stride* s akan memiliki dimensi sebesar $\left(\frac{H-\rho}{s} + 1\right) \times \left(\frac{V-\rho}{s} + 1\right)$. *Pooling* mereduksi dimensi citra dengan hanya meneruskan fitur-fitur yang sifatnya dominan (*max pooling*), atau hanya meneruskan hasil rerata dari fitur-fitur yang berdekatan (*average pooling*).



Gambar 2.7 Contoh struktur algoritma JST Konvolusi. Gambar disadur dari [31].

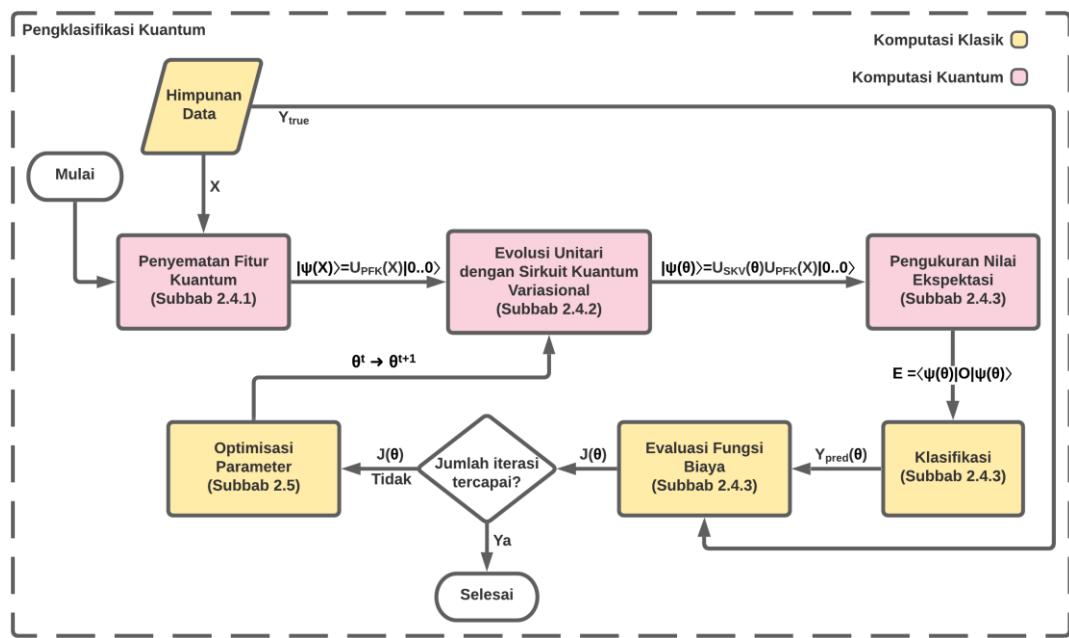
Teknik konvolusi dan *pooling* dapat dilakukan beberapa kali sampai dimensi citra sesuai dengan yang diinginkan. Matriks keluaran dari proses *pooling* terakhir dapat kemudian diratakan menjadi vektor (proses *flatten*) dan diumpulkan ke JST untuk melakukan klasifikasi citra. Gabungan proses konvolusi, *pooling*, dan algoritma JST disebut dengan algoritma JST Konvolusi. Gambar 2.7 menunjukkan ilustrasi contoh struktur dari JST Konvolusi. Pada JST Konvolusi, parameter bobot dan bias pada matriks filter konvolusi juga ikut dioptimisasi bersama dengan parameter bobot dan bias pada JST. JST Konvolusi memiliki performa klasifikasi yang lebih baik daripada algoritma JST biasa untuk himpunan data citra yang umumnya berdimensi tinggi [32]. Hal ini karena proses konvolusi dan *pooling* pada JST Konvolusi mampu mengekstraksi fitur-fitur yang dominan dan invariant secara translasi maupun rotasi sehingga fitur-fitur yang masuk ke JST hanyalah fitur-fitur

yang penting dalam klasifikasi. Ini memudahkan proses optimisasi parameter JST dalam meminimumkan fungsi biaya, menyebabkan performa klasifikasi JST yang lebih baik.

2.4 Pengklasifikasi Kuantum Sebagai Aplikasi AKV

AKV adalah istilah yang digunakan untuk mengacu pada algoritma kuantum yang memanfaatkan algoritma optimisasi klasik untuk mengoptimisasi parameter pada sirkuit kuantum variasional agar dapat menghasilkan keluaran tertentu[12]. Salah satu aplikasinya adalah untuk mengklasifikasikan data.

Pada komputasi klasik, algoritma JST umum digunakan untuk tugas-tugas klasifikasi. Algoritma ini menginspirasi algoritma-algoritma klasifikasi pada komputasi kuantum. Tiga bagian utama dalam algoritma kuantum untuk klasifikasi adalah penyematan fitur kuantum (serupa dengan lapisan masukan pada JST), sirkuit kuantum variasional (serupa dengan lapisan tersembunyi pada JST), dan pengukuran nilai ekspektasi (serupa dengan lapisan keluaran pada JST)[14].



Gambar 2.8 Diagram alir algoritma Pengklasifikasi Kuantum secara umum.

2.4.1 Penyematan Fitur Kuantum (PKF)

Karena sirkuit kuantum hanya dapat memproses keadaan kuantum, diperlukan suatu metode untuk menyematkan sampel data klasik seperti citra menjadi keadaan kuantum. Proses penyematan ini tidak trivial dan masih merupakan area riset yang aktif. Sampai saat ini, belum terdapat suatu metode rujukan ‘terbaik’ yang dijamin dapat selalu digunakan. Meski begitu, terdapat dua jenis penyematan yang umum digunakan yaitu penyematan amplitudo dan penyematan sudut.

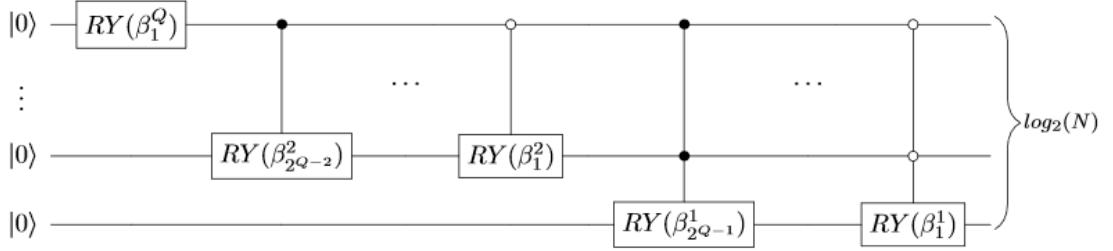
2.4.1.1 Penyematan Amplitudo

Metode ini menyematkan sampel data menjadi keadaan kuantum dengan cara menyematkan nilai dari setiap fitur pada sampel data ke dalam nilai-nilai amplitudo dari suatu keadaan kuantum. Misalkan terdapat sampel data x yang memiliki N fitur. Menggunakan metode penyematan amplitudo, keadaan kuantum yang merepresentasikan sampel data tersebut dapat dituliskan sebagai berikut[14]:

$$A_n = \frac{x^n}{\sqrt{\sum_{n=1}^N |x^n|^2}}$$
$$|x\rangle = \sum_{n=1}^N A_n |binary(n-1)\rangle \quad (2.24)$$

dengan $binary(n)$ adalah fungsi yang mengkonversi bilangan bulat n menjadi bentuk bilangan binernya dengan jumlah bit sebanyak $\log_2 N$, x^n adalah nilai fitur ke- n dari data sampel x , dan $|x\rangle$ adalah keadaan kuantum yang merepresentasikan sampel data x . Sebagai contoh, misalnya jumlah fitur sampel data x ada sebanyak $N = 4$ maka $binary(2) = 10$.

Keadaan kuantum $|x\rangle$ dapat diperoleh dengan skema sirkuit sebagai berikut[14,33]:



Gambar 2.9 Skematik sirkuit metode penyematan amplitudo.

dengan $Q = \log_2 N$ adalah jumlah qubit yang dibutuhkan dan nilai β didapat dari persamaan berikut:

$$\beta_j^g = 2 \arcsin \left(\frac{\sqrt{\sum_{l=1}^{2g-1} |A_{(2j-1)2^{g-1}+l}|^2}}{\sqrt{\sum_{l=1}^{2g} |A_{(j-1)2^g+l}|^2}} \right). \quad (2.25)$$

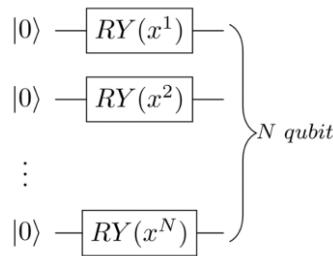
Gerbang RY yang tersambung pada titik hitam atau putih menunjukkan bahwa gerbang RY ini adalah gerbang RY terkontrol. Artinya, gerbang RY baru aktif operasinya jika qubit kontrol (qubit yang diberikan tanda titik hitam atau putih) berada dalam keadaan yang sesuai. Titik hitam akan mengaktifkan gerbang RY jika qubit kontrol berada dalam keadaan $|1\rangle$, sedangkan titik putih akan mengaktifkan gerbang RY jika qubit kontrol berada dalam keadaan $|0\rangle$. Keadaan kuantum keluaran dari sirkuit ini kemudian dapat diproses lebih lanjut dengan Sirkuit Kuantum Variasional untuk mendapatkan hasil klasifikasi.

2.4.1.2 Penyematan Sudut

Metode ini menyematkan nilai setiap fitur dari suatu sampel data x sebagai parameter sebuah gerbang parametrik satu parameter (RX, RY, atau RZ) yang bekerja pada sebuah qubit. Misalkan sampel data x memiliki fitur sebanyak N , jika penyematan sudut dilakukan dengan gerbang RY, maka keadaan kuantum yang merepresentasikan sampel data tersebut adalah sebagai berikut:

$$|x\rangle = RY(x^1) \otimes RY(x^2) \otimes \dots \otimes RY(x^N) \underbrace{|00 \dots 00\rangle}_N. \quad (2.26)$$

dengan x^n adalah nilai fitur ke- n dari data sampel x . Metode ini membutuhkan qubit sebanyak $Q = N$. Contoh skematik sirkuitnya terdapat pada Gambar 2.10.



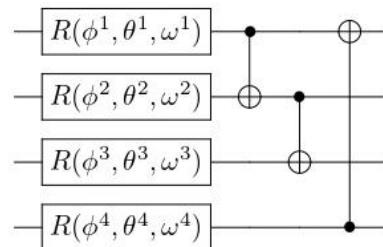
Gambar 2.10 Skematik sirkuit metode penyematan sudut.

Keadaan kuantum keluaran dari sirkuit ini kemudian dapat diproses lebih lanjut dengan Sirkuit Kuantum Variasional untuk mendapatkan hasil klasifikasi.

2.4.2 Evolusi Unitari dengan Sirkuit Kuantum Variasional (SKV)

SKV adalah sirkuit kuantum yang terdiri dari serangkaian gerbang parametrik (dan non-parametrik jika dibutuhkan) yang parameteranya divariasikan sedemikian rupa untuk mengevolusi keadaan kuantum masukan menjadi suatu keadaan kuantum keluaran yang akan memiliki hasil pengukuran tertentu. Keadaan kuantum keluaran metode penyematan yang dibahas pada subbab sebelumnya menjadi keadaan kuantum masukan untuk SKV. Dengan menggunakan SKV, evolusi unitari yang dilakukan pada keadaan kuantum masukan terkontrol oleh parameter-parameter gerbang kuantum di dalam SKV sehingga keadaan kuantum keluaran yang diinginkan dapat diperoleh dengan mencari nilai parameter-parameter yang sesuai.

Sama seperti metode penyematan, menentukan arsitektur sirkuit (biasa disebut *ansatz*) SKV untuk tugas klasifikasi tertentu tidaklah trivial. Belum terdapat suatu aturan rujukan yang mengatur hal-hal seperti jenis gerbang yang digunakan, jumlah gerbang, dan jumlah qubit yang diperlukan.



Gambar 2.11 Contoh *ansatz* SKV sederhana yang menggunakan 4 qubit.

Gambar 2.11 menunjukkan contoh *ansatz* SKV sederhana. *Ansatz* ini menggunakan 4 qubit dengan operasi gerbang R untuk tiap qubitnya, diikuti dengan serangkaian gerbang CNOT. Setiap gerbang R memiliki 3 parameter, sehingga total parameter variasi *ansatz* ini adalah 12 parameter. Serupa dengan JST, parameter-parameter ini kemudian dioptimisasi agar sirkuit dapat melakukan evolusi unitari yang menghasilkan hasil pengukuran yang diinginkan. Selain itu, SKV ini juga dapat diulangi beberapa kali, membentuk sirkuit yang lebih panjang dengan parameter yang lebih banyak, konsep yang serupa dengan pengulangan lapisan tersembunyi pada JST.

2.4.3 Pengukuran Nilai Ekspektasi

Agar dapat melakukan klasifikasi, *observable* pengukuran SKV harus dapat merepresentasikan kelas-kelas yang ada. Sebagai contoh, untuk klasifikasi biner (2 kelas), misalkan $|\Psi\rangle$ adalah keadaan kuantum keluaran SKV $U(\vec{\theta})$ yang menggunakan 1 qubit, *observable* $O = |0\rangle\langle 0| = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$ dapat digunakan.

$$|\Psi\rangle = U(\vec{\theta})|0\rangle$$

$$\lambda(\vec{\theta}) = \langle O \rangle = \langle \Psi | O | \Psi \rangle = \langle \Psi | 0 \rangle \langle 0 | \Psi \rangle = |\langle 0 | \Psi \rangle|^2 = |\langle 0 | U(\vec{\theta}) | 0 \rangle|^2. \quad (2.27)$$

Nilai ekspektasi yang akan didapat dari pengukuran ini akan berada dalam rentang $[0, 1]$ sehingga hasil pengukuran dapat dianggap merepresentasikan kelas pertama jika hasil pengukuran $< 0,5$, dan dianggap kelas kedua jika hasil pengukuran $\geq 0,5$, mirip seperti fungsi aktivasi *sigmoid* pada JST.

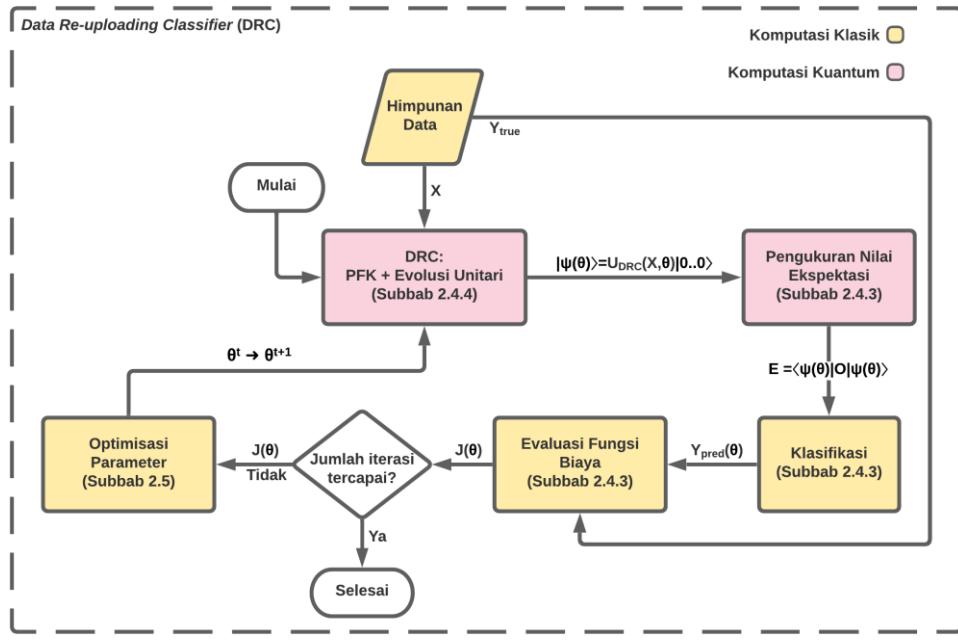
Kemudian, fungsi biaya yang digunakan haruslah merupakan fungsi dari hasil pengukuran. Melanjutkan contoh klasifikasi biner, karena nilai pengukuran berada dalam rentang $[0, 1]$ dan himpunan data hanya terdiri atas 2 kelas (0 atau 1), maka salah satu fungsi biaya yang dapat digunakan misalnya adalah *mean squared error* (MSE) sebagai berikut:

$$J(\vec{\theta}) = \frac{1}{M} \sum_{m=1}^M (\lambda(\vec{\theta})_m - y_m)^2 \quad (2.28)$$

dengan M adalah jumlah total sampel data, y_m adalah label sampel data ke- m (nilainya 0 atau 1), dan $\lambda(\vec{\theta})_m$ adalah nilai ekspektasi hasil pengukuran SKV yang menerima keadaan kuantum dari penyematkan sampel data ke- m . Terlihat fungsi biaya J disini serupa dengan fungsi biaya pada JST, keduanya sama-sama merupakan fungsi dari vektor parameter $\vec{\theta}$. Perbedaannya, vektor parameter disini memiliki komponen berupa parameter-parameter pada SKV. Sehingga untuk melatih SKV agar dapat melakukan klasifikasi, yang perlu dilakukan hanyalah mengoptimasi $\vec{\theta}$ yang meminimumkan $J(\vec{\theta})$, sama seperti JST. Algoritma optimisasi yang digunakan untuk JST dapat juga digunakan untuk mengoptimalkan parameter SKV.

2.4.4 Data Re-uploading Classifier (DRC)

Pada pengklasifikasi kuantum, dibutuhkan setidaknya dua sirkuit kuantum yang berbeda, yang pertama untuk menyematkan sampel data ke dalam keadaan kuantum, dan yang kedua untuk mengevolusi keadaan kuantum tersebut menjadi keadaan kuantum baru yang akan menghasilkan nilai pengukuran yang diinginkan. Arsitektur seperti ini membuat pengklasifikasi kuantum membutuhkan banyak qubit dan banyak gerbang, seperti yang terjadi pada riset-riset terkait yang telah dipaparkan pada Subbab 1.3. Makalah penelitian yang mengajukan ide-ide untuk menggabungkan penyematkan fitur kuantum dan evolusi unitari ke dalam satu sirkuit yang sama kemudian mulai bermunculan[34,35]. Salah satunya adalah Pérez-Salinas et al.[19] yang mengajukan arsitektur efisien yang menggabungkan penyematkan sudut dan evolusi unitari ke dalam satu sirkuit SKV yang sama. Arsitektur ini dikenal dengan nama DRC. Gambar 2.12 menunjukkan diagram alir sederhana dari DRC, bandingkan dengan diagram alir sederhana dari algoritma pengklasifikasi kuantum pada umumnya pada Gambar 2.8.



Gambar 2.12 Diagram alir sederhana dari algoritma DRC.

Untuk menjelaskan bentuk dari arsitektur DRC ini, diperlukan sebuah notasi sebagai berikut untuk memudahkan penulisan:

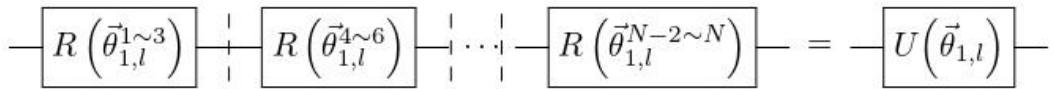
$$\begin{aligned}\vec{\theta}_{q,l}^{n \sim n+2} &= (\theta_{q,l}^n, \theta_{q,l}^{n+1}, \theta_{q,l}^{n+2}) \\ \vec{\theta}_{q,l} &= (\theta_{q,l}^1, \theta_{q,l}^2, \theta_{q,l}^3, \dots, \theta_{q,l}^N)\end{aligned}\quad (2.29)$$

dengan indeks q dan l menandakan nomor qubit dan nomor lapisan, indeks n menandakan indeks fitur (dimulai dari 1 sampai N dengan N adalah jumlah fitur sampel data). Notasi ini berlaku untuk setiap bab dan subbab selanjutnya.

Misalkan terdapat sampel data x dengan jumlah fitur N , nilai dari $\theta_{q,l}^n$ didapat dengan persamaan

$$\theta_{q,l}^n = w_{q,l}^n x^n + b_{q,l}^n \quad (2.30)$$

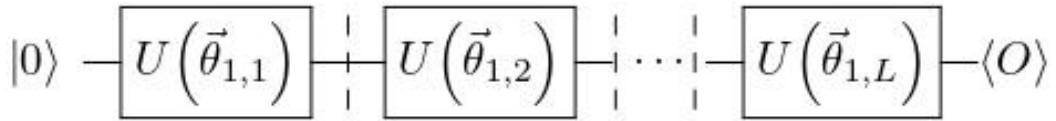
dengan w dan b adalah parameter bobot dan bias sehingga tiap $\vec{\theta}_{q,l}^{n \sim n+2}$ mengandung tiga nilai yang berbeda. $\vec{\theta}_{q,l}^{n \sim n+2}$ menjadi parameter masukan untuk sebuah gerbang parametrik tiga parameter, misalnya gerbang R.



Gambar 2.13 Skematik sirkuit DRC 1 qubit.

Gambar 2.13 merupakan skematik DRC 1 qubit menggunakan gerbang R. Dari skematik ini terlihat karena gerbang R adalah gerbang parametrik tiga parameter, maka untuk N fitur dibutuhkan $\frac{N}{3}$ gerbang. Jika N bukan kelipatan 3, Pérez-Salinas et al. menyarankan untuk menambah fitur data sampel tersebut dengan fitur-fitur bernilai nol sampai total fitur menjadi kelipatan 3. Sebagai contoh, jika sampel data memiliki 6 fitur, maka sampel data ini dibagi ke dalam 2 subsampel data, dengan subsampel data pertama mengandung fitur 1 sampai 3 dan subsampel data kedua mengandung fitur 4 sampai 6. Jika sampel data memiliki 7 fitur, maka sampel data ini dibagi ke dalam 3 subsampel data, dengan subsampel data pertama mengandung fitur 1 sampai 3, subsampel data kedua mengandung fitur 4 sampai 6, dan subsampel ketiga mengandung fitur 7 dan dua angka nol. Dengan arsitektur seperti ini, secara teori 1 qubit cukup untuk menyematkan berapapun jumlah fitur yang dimiliki oleh sampel data.

Dari persamaan (2.30) terlihat bahwa $\theta_{q,l}^n$ sudah mengandung variabel sampel data x sehingga DRC tidak membutuhkan sirkuit lainnya untuk menyematkan fitur kuantum, fitur disematkan pada sirkuit yang sama dengan sirkuit SKV yang melakukan evolusi unitari. Satu blok sirkuit DRC seperti pada Gambar 2.13 dapat diulangi membentuk beberapa lapisan seperti pada Gambar 2.14. Skema ini diberi nama “*data re-uploading*” karena sampel data disematkan ke dalam gerbang-gerbang kuantum pada tiap blok sirkuit DRC, sehingga jika blok ini diulangi beberapa kali, maka sampel data ini disematkan ke dalam sirkuit kuantum beberapa kali, sebanyak lapisan yang digunakan. Ini adalah perbedaan penting lainnya antara skema DRC dengan pengklasifikasi pada umumnya yang hanya menyematkan data ke sirkuit kuantum satu kali saat proses PFK. Keadaan kuantum keluaran sirkuit ini adalah $|\Psi_{DRC}\rangle = U(\vec{\theta}_{1,L})|0\rangle$ sehingga nilai ekspektasi pengukuran dari *observable* O adalah $\langle O \rangle = \langle \Psi_{DRC} | O | \Psi_{DRC} \rangle$.



Gambar 2.14 Skematik sirkuit DRC 1 qubit dengan jumlah lapisan sebanyak L .

2.5 Algoritma Optimisasi

Untuk mencari parameter optimal, baik dalam JST maupun SKV, diperlukan suatu algoritma optimisasi. Salah satu algoritma optimisasi yang memiliki performa yang sangat baik adalah *Adaptive Moment Estimation* (Adam)[36].

2.5.1 Adaptive Moment Estimation (Adam)

Adam adalah algoritma optimisasi pengembangan dari *gradient descent*. Tidak seperti *gradient descent* yang hanya menggunakan nilai gradien terakhir untuk menentukan besarnya pergeseran parameter, Adam juga memanfaatkan nilai-nilai gradien terdahulu. Pengaruh dari nilai-nilai terdahulu ini diwakilkan oleh sebuah nilai hasil dari *exponentially weighted moving average*. Misalkan terdapat fungsi $J(\vec{\theta})$ yang salah satu parameternya adalah θ_G (θ_G merupakan salah satu komponen dari vektor parameter $\vec{\theta}$). Untuk meminimumkan fungsi $J(\vec{\theta})$, algoritma perbaruan parameter θ_G dengan optimisasi Adam adalah sebagai berikut:

1. Inisialisasi awal beberapa variabel sebagai berikut:

$$V_{\theta_G} = 0, \quad S_{\theta_G} = 0, \quad t = 0. \quad (2.31)$$

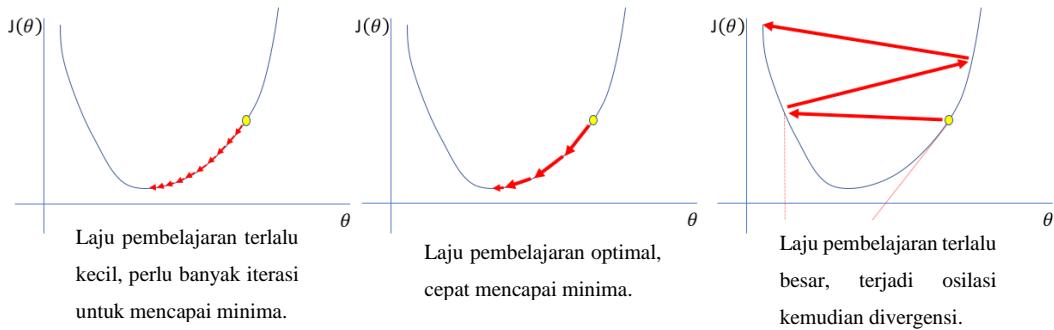
2. Untuk setiap langkah optimisasi, lakukan komputasi sebagai berikut:

$$\begin{aligned} t &= t + 1 \\ V_{\theta_G} &= \frac{\beta_1 V_{\theta_G} + (1 - \beta_1) \frac{\partial}{\partial \theta_G} J(\vec{\theta})}{1 - (\beta_1)^t} \\ S_{\theta_G} &= \frac{\beta_2 S_{\theta_G} + (1 - \beta_2) \left(\frac{\partial}{\partial \theta_G} J(\vec{\theta}) \right)^2}{1 - (\beta_2)^t} \\ \theta_G &= \theta_G - \eta \frac{V_{\theta_G}}{\sqrt{S_{\theta_G}} + \epsilon} \end{aligned} \quad (2.32)$$

dengan t adalah langkah optimisasi yang telah dilakukan dan η adalah laju pembelajaran. V_{θ_G} adalah estimasi momen pertama dan S_{θ_G} adalah estimasi momen kedua dari nilai $\frac{\partial}{\partial \theta_G} J(\vec{\theta})$. Untuk penelitian ini, ditetapkan $\beta_1 = 0.9$ dan $\beta_2 = 0.999$. Parameter $\epsilon = 10^{-7}$ disini hanya berfungsi untuk menjaga stabilisasi numerik, mencegah terjadinya pembagian dengan nol.

2.5.2 Laju Pembelajaran Meluruh

Pada proses latih/optimisasi parameter, konstanta laju pembelajaran menentukan seberapa cepat pembelajaran berlangsung. Laju pembelajaran yang besar dapat mempercepat penurunan fungsi biaya. Namun, laju pembelajaran yang besar juga memiliki kekurangan, yaitu meningkatkan resiko terjadinya osilasi atau bahkan divergensi (keluar dari lembah minima karena perubahan parameter yang terlalu besar) yang menyebabkan algoritma optimisasi gagal mencapai minima.



Gambar 2.15 Ilustrasi pengaruh pemilihan nilai laju pembelajaran dalam proses optimisasi parameter θ untuk meminimumkan nilai fungsi biaya $J(\theta)$. Gambar disadur dari [37] dengan lokalisasi ke Bahasa Indonesia.

Untuk dapat mencapai minima dengan relatif cukup cepat dan mengurangi resiko terjadinya osilasi atau divergensi, teknik laju pembelajaran meluruh dapat digunakan. Teknik ini mengecilkan nilai dari laju pembelajaran seiring iterasi yang dilakukan, sehingga memungkinkan untuk memulai proses latih dengan nilai laju pembelajaran yang relatif besar. Pada penelitian ini, algoritma laju pembelajaran meluruh yang digunakan sebagai berikut:

1. Tentukan nilai laju pembelajaran awal η_0 , konstanta laju peluruhan τ , dan konstanta langkah peluruhan Δ .

2. Pada setiap langkah optimisasi, lakukan komputasi berikut:

$$\begin{aligned} t &= t + 1 \\ \eta &= \eta_0 \times \tau^{\left\lfloor \frac{t}{\Delta} \right\rfloor} \end{aligned} \quad (2.33)$$

dengan $\left\lfloor \frac{t}{\Delta} \right\rfloor$ adalah fungsi untuk membulatkan nilai t/Δ ke bilangan bulat terdekat (yang lebih kecil dari t/Δ) dan t adalah langkah optimisasi yang telah dilakukan, dimulai dari nol.

2.6 Gradien Kuantum

Kebanyakan algoritma optimisasi membutuhkan nilai gradien dari fungsi yang ingin diminimumkan terhadap parameter yang akan dioptimisasi. Pada JST klasik, hal ini sangat mudah dilakukan karena komputer klasik mampu membuat salinan dari keluaran tiap simpul pada tiap lapisan JST kemudian gradien fungsi biaya J terhadap suatu parameter dapat diperoleh secara perlahan-lahan dengan memanfaatkan aturan rantai pada turunan. Algoritma ini biasa disebut sebagai *backpropagation*.

Pada komputasi kuantum, *backpropagation* dapat digunakan hanya jika komputasi dijalankan pada simulator dengan komputer klasik. Pada komputer kuantum, keadaan kuantum setelah melalui lapisan atau gerbang tertentu tidak dapat diakses nilainya, karena pengukuran akan menyebabkan runtuhnya keadaan kuantum. Satu-satunya keluaran yang dapat diukur hanyalah keluaran pada akhir dari sirkuit kuantum. Untuk itu, diperlukan suatu algoritma untuk mendapatkan nilai gradien tanpa perlu melakukan pengukuran di tengah-tengah sirkuit.

Aturan pergeseran parameter[38] memungkinkan didapatkannya nilai dari gradien tanpa perlu mengetahui nilai dari keluaran lapisan atau gerbang di tengah-tengah sirkuit. Misalkan terdapat fungsi biaya J yang salah satu parameternya adalah θ_G dari suatu gerbang U_G di dalam suatu sirkuit kuantum, dengan aturan pergeseran parameter, nilai gradien fungsi J terhadap θ_G adalah sebagai berikut:

$$J(\theta_G) = \langle \psi | U_G^\dagger(\theta_G) \Lambda U_G(\theta_G) | \psi \rangle \quad (2.34)$$

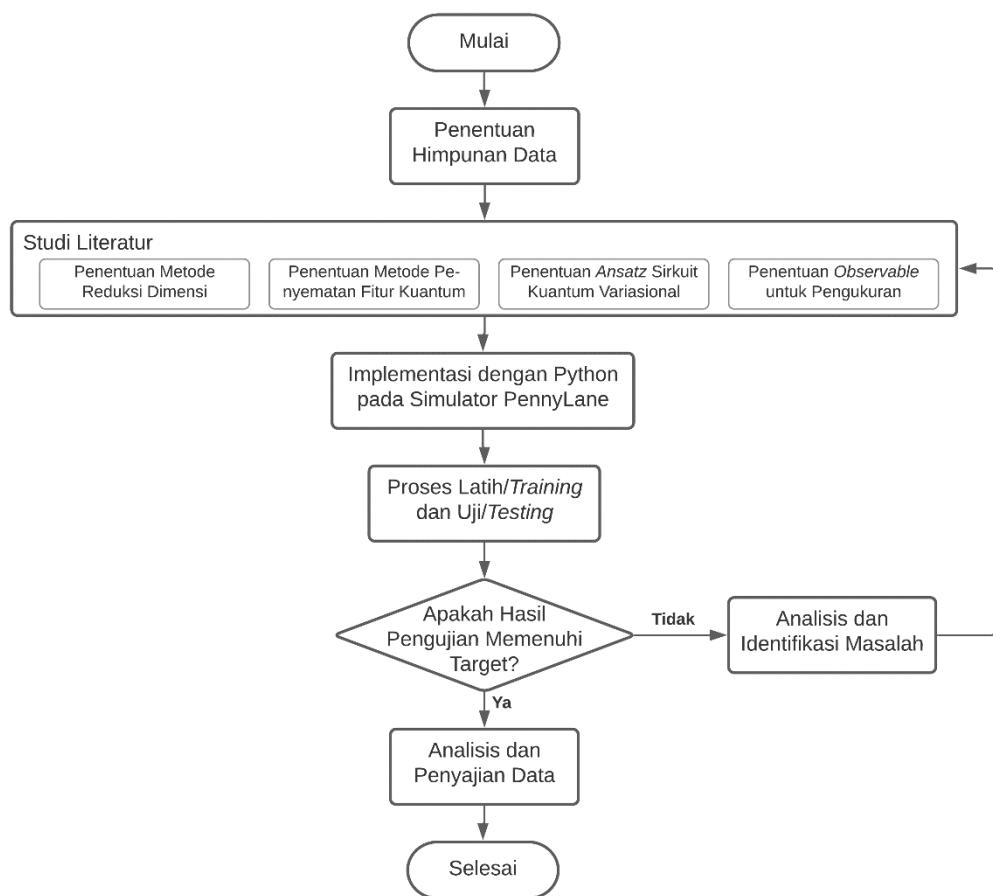
$$\frac{\partial}{\partial \theta_G} J(\theta_G) = r \left[J\left(\theta_G + \frac{\pi}{4r}\right) - J\left(\theta_G - \frac{\pi}{4r}\right) \right] \quad (2.35)$$

dengan r adalah suatu konstanta yang bergantung pada jenis gerbang kuantum dari U_G , Λ adalah serangkaian gerbang yang letaknya setelah gerbang U_G beserta *observable* pengukuran O dan $|\psi\rangle$ adalah keadaan kuantum hasil operasi dari serangkaian gerbang yang letaknya sebelum gerbang U_G pada keadaan kuantum $|00 \dots 00\rangle$.

BAB III

PERANCANGAN ARSITEKTUR SISTEM

Perancangan arsitektur sistem pada penelitian ini berlangsung secara iteratif sebagaimana terlihat pada Gambar 3.1. Iterasi dihentikan apabila hasil pengujian yang diperoleh memenuhi target yang dipaparkan pada tujuan penelitian. Gambar 3.2 menunjukkan aliran data pada penelitian ini.



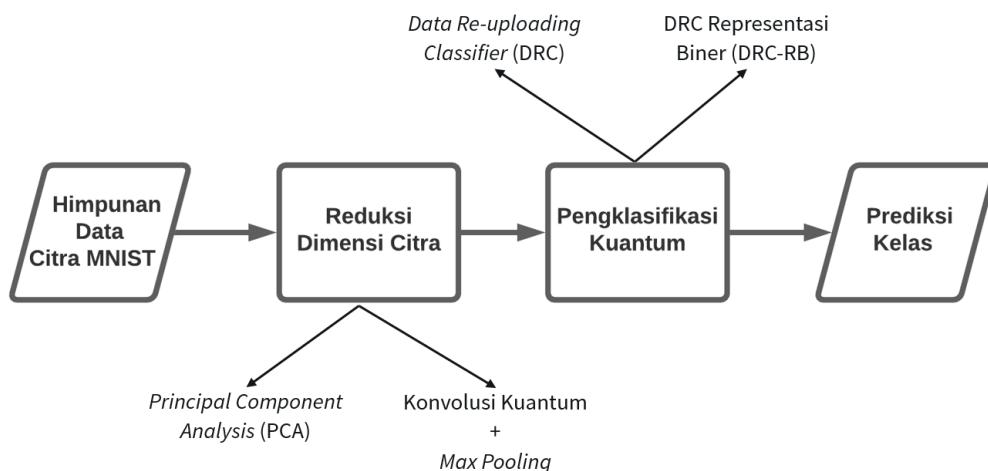
Gambar 3.1 Diagram alir skema penelitian.

Penelitian dimulai dari studi literatur terkait metode-metode klasifikasi menggunakan komputasi kuantum. Literatur yang digunakan adalah makalah penelitian (*paper*) dan *textbook*. Tujuan utama dari tahap ini adalah untuk merancang arsitektur awal yang akan digunakan. Studi literatur juga bertujuan untuk mendapatkan pengetahuan tentang seberapa baik performa klasifikasi

algoritma-algoritma pengklasifikasi kuantum yang sudah ada saat ini dan hal-hal apa saja yang dapat diperbaiki (tertuang dalam Subbab 1.3).

Setelah didapatkan arsitektur yang dirasa sesuai dari studi literatur, arsitektur ini diimplementasikan menjadi program dalam bahasa Python dan diuji coba dengan simulator PennyLane. Penelitian ini akan berfokus pada eksplorasi dan modifikasi arsitektur DRC untuk klasifikasi himpunan data citra MNIST 2 kelas, 4 kelas, dan 8 kelas.

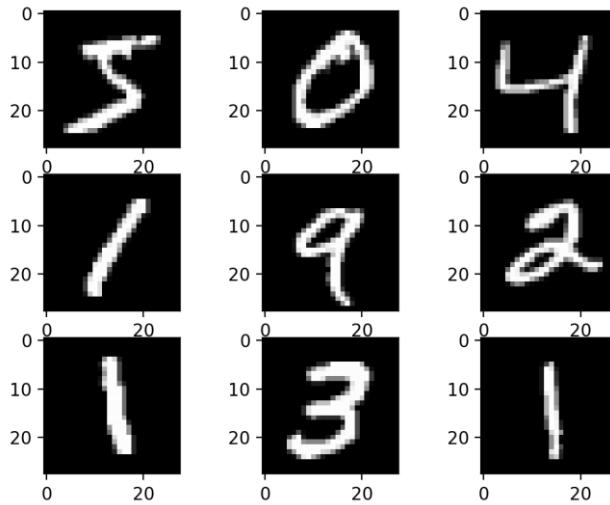
Hasil dari uji coba program kemudian dievaluasi dan dianalisis. Jika hasil sudah mencapai target tujuan penelitian, maka penelitian dilanjutkan dengan analisis hasil dan penyajian data untuk laporan penelitian. Jika hasil belum mencapai target, dilakukan analisis dan identifikasi masalah untuk kemudian dicari solusinya melalui studi literatur tambahan atau diskusi dengan dosen pembimbing.



Gambar 3.2 Diagram aliran data.

3.1 Skema Reduksi Dimensi Citra MNIST

Himpunan data MNIST adalah himpunan data citra angka tulisan tangan dari 0 sampai 9 (10 kelas) dengan ukuran dimensi 28 x 28 piksel (784 fitur). Citra angka pada himpunan data ini hanya memiliki satu kanal (citra skala keabuan) dengan nilai intensitas piksel berada di antara 0 sampai 255. Untuk penelitian ini, nilai intensitas piksel dinormalisasi ke dalam *range* 0 sampai 1.



Gambar 3.3 Contoh sampel citra dari himpunan data MNIST[21].

Terdapat total 60000 sampel data latih dan 10000 sampel data uji. Penelitian ini hanya menggunakan subsampel dari himpunan data ini, yaitu sebanyak 200 sampel data latih tiap kelas. Untuk pengujian, jumlah sampel data bergantung pada jumlah kelas yang diklasifikasi. Data latih adalah data yang digunakan dalam proses latih, yaitu proses optimisasi parameter sirkuit. Sedangkan data uji adalah data yang tidak digunakan dalam proses latih dan hanya digunakan dalam proses evaluasi, yaitu proses menghitung akurasi dan total nilai biaya dari pengklasifikasi tanpa melakukan optimisasi parameter sirkuit. Evaluasi dengan data uji umumnya lebih menggambarkan performa pengklasifikasi yang sesungguhnya karena ada resiko pengklasifikasi mengalami *overfitting* terhadap data latih saat proses latih yang menyebabkan evaluasi akurasi pengklasifikasi pada data latih sangat tinggi namun relatif rendah saat dievaluasi dengan data uji.

Pada klasifikasi biner (2 kelas), digunakan 500 sampel data uji tiap kelas, sedangkan untuk klasifikasi banyak kelas (lebih dari 2), digunakan 50 sampel data uji tiap kelas. Hal ini dilakukan karena klasifikasi biner relatif sangat mudah dibandingkan dengan klasifikasi banyak kelas. Jika jumlah sampel data ujinya terlalu sedikit, sulit membedakan performa akurasi antar model. Misalkan jika hanya terdapat 10 sampel data uji, maka besar kemungkinan dua model dengan arsitektur berbeda memperoleh skor akurasi yang sama. Tetapi jika terdapat total

1000 sampel data uji, maka kemungkinan kedua model ini memperoleh skor akurasi yang sama sangatlah kecil. Klasifikasi banyak kelas relatif sulit dan kecil kemungkinan dua model dengan arsitektur berbeda bisa memiliki skor akurasi yang persis sama sehingga jumlah sampel data uji tidak perlu sebanyak klasifikasi biner.

Penelitian ini menggunakan 2 metode untuk men-*downsampling* data MNIST agar dimensi fitur data berkurang. Yang pertama adalah dengan PCA, dan yang kedua adalah dengan konvolusi kuantum.

3.1.1 Reduksi dengan PCA

Dengan algoritma PCA sebagaimana dijelaskan dalam Subbab 2.3.1, nilai μ dan σ untuk setiap fitur, dan matriks P seperti pada persamaan (2.20) diperoleh dari himpunan data latih. Nilai komponen prinsip p divariasikan untuk melihat bagaimana performa klasifikasi pada berbagai nilai p . Jumlah komponen prinsip yang besar membuat data mengandung variansi yang semakin besar (sampel data semakin mudah dibedakan) namun *trade-off*-nya adalah diperlukan gerbang kuantum yang semakin banyak. Himpunan data hasil PCA, D' , kemudian akan digunakan untuk proses latih.

Pada proses pengujian, himpunan data uji dinormalisasi berdasarkan persamaan (2.17) dengan menggunakan nilai μ dan σ yang diperoleh dari himpunan data latih. Himpunan data uji kemudian diproyeksikan pada matriks P yang diperoleh dari himpunan data latih untuk mendapatkan himpunan data uji hasil PCA. Himpunan data uji ini lalu diumpulkan pada pengklasifikasi kuantum untuk mendapatkan skor akurasi ujinya.

3.1.2 Reduksi dengan Konvolusi Kuantum

Metode PCA umumnya baik digunakan pada himpunan data yang variasi antar sampel data dengan kelas samanya rendah. Untuk himpunan data MNIST, penulis berhipotesis metode PCA cukup dapat diandalkan karena variasi antar sampel dalam data tulisan angka seharusnya tidak terlalu tinggi. Misalkan angka 0, walaupun dapat dituliskan dengan berbagai cara, namun kurang lebihnya akan selalu berupa bentuk oval. Namun metode ini tentu tidak akan cocok jika digunakan untuk mereduksi dimensi citra yang variasi sampelnya tinggi dan non-linear,

misalnya citra kucing. Citra dua kucing yang berbeda memiliki variasi yang sangat tinggi, meskipun keduanya sama-sama memiliki kelas kucing. Bahkan perbedaan pada latar belakang saja dapat meningkatkan variasi antar sampel, meskipun untuk kucing yang sama. Penggunaan PCA untuk data seperti ini kemungkinan dapat membuat data kehilangan nilai variansi yang cukup signifikan. Oleh sebab itu untuk klasifikasi citra, JST klasik tidak mengandalkan reduksi dimensi dengan PCA, melainkan dengan lapisan konvolusi dan *pooling* sebagaimana dibahas dalam Subbab 2.3.2.

Meskipun penelitian ini dibatasi hanya menggunakan himpunan data MNIST yang dapat direduksi oleh PCA dengan baik, namun penulis mengajukan suatu algoritma baru untuk mereduksi dimensi citra dengan SKV yang terinspirasi dari lapisan konvolusi dan arsitektur DRC. Proses komputasi algoritma ini mirip dengan konvolusi namun sirkuit arsitektur DRC digunakan sebagai filernya. Penulis menyebut arsitektur ini dengan nama *Data Re-uploading Quantum Convolution* (DRQConv).

Algoritma DRQConv untuk citra 1 kanal adalah sebagai berikut:

1. Tentukan jumlah piksel $\zeta = \rho \times \rho$ pada citra masukan yang akan diambil menjadi masukan ke sirkuit kuantum. Jumlah piksel ini ibarat ukuran filter pada konvolusi klasik.
2. Tentukan *ansatz* sirkuit $U(\vec{\theta})$ yang akan berperan sebagai filter dan *observable* pengukuran O yang akan digunakan. *Ansatz* sirkuit ini harus dapat menerima parameter masukan minimal sebanyak ζ parameter. Jumlah qubit dimana *ansatz* sirkuit ini bekerja tidak diatur karena bergantung pada pemilihan *ansatz* sirkuit itu sendiri, yang penting adalah semua qubit memiliki keadaan awal $|0\rangle$, sehingga keadaan awal total semua qubit adalah $|00 \dots 00\rangle$.
3. Mulai proses konvolusi. Sama seperti konvolusi klasik, *window* dengan ukuran $\rho \times \rho$ bergeser sedikit demi sedikit sampai seluruh piksel pada citra tertutupi oleh *window* ini. Seberapa jauh pergeseran *window* ditentukan oleh parameter *stride* s .

4. Setiap kali *window* ini menutupi region piksel tertentu, ambil nilai semua piksel yang tertutupi tersebut dan ratakan menjadi sebuah vektor $\vec{x} = (x^1, x^2, \dots, x^\zeta)$. Sematkan \vec{x} ke dalam $U(\vec{\theta})$ seperti pada DRC, operasikan $U(\vec{\theta})$ pada qubit $|00 \dots 00\rangle$, lalu ukur dengan observable O . Persamaannya sebagai berikut:

$$\begin{aligned}\vec{\theta} &= \vec{W} \odot \vec{x} + \vec{b} = (w^1 x^1 + b^1, w^2 x^2 + b^2, \dots, w^\zeta x^\zeta + b^\zeta) \\ |\Psi\rangle &= U(\vec{\theta}) |00 \dots 00\rangle \\ \lambda &= \langle \Psi | O | \Psi \rangle\end{aligned}\tag{3.1}$$

dengan \vec{W} dan \vec{b} adalah vektor parameter, \odot adalah operasi perkalian antar elemen matriks/vektor (*Hadamard product*).

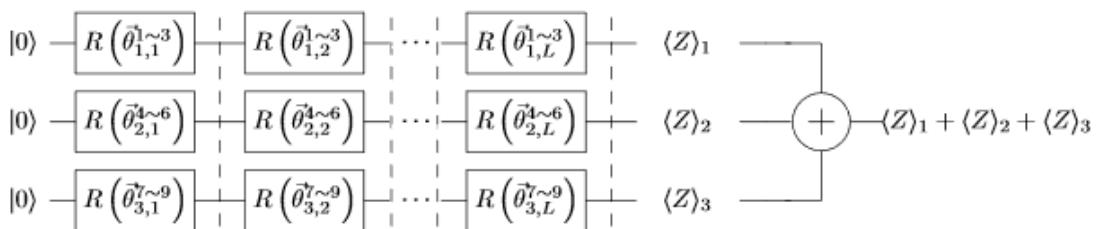
5. Nilai λ ini adalah nilai keluaran dari proses konvolusi pada piksel-piksel yang tertutupi. Lanjutkan proses konvolusi hingga semua piksel pada citra tertutupi oleh *window*. Susun nilai λ dengan urutan yang sama seperti susunan nilai hasil konvolusi pada konvolusi klasik.
6. Citra keluaran proses konvolusi ini akan memiliki dimensi yang berkurang dari dimensi citra masukannya. Jika sampel citra masukan memiliki dimensi $H \times V$, citra keluarannya akan memiliki dimensi sebesar $\left(\frac{H-\rho}{s} + 1\right) \times \left(\frac{V-\rho}{s} + 1\right)$, sama seperti konvolusi klasik yang dijabarkan pada Subbab 2.3.2.
7. Citra keluaran ini dapat dikonvolusi lebih jauh lagi agar dimensi citra semakin kecil, atau bisa diratakan dan kemudian diumpulkan ke DRC sebagai fitur dari sampel citra untuk diklasifikasi.
8. Parameter \vec{W} dan \vec{b} akan ikut dioptimisasi bersama dengan seluruh parameter DRC saat proses latih.

Algoritma ini dapat diekspansi untuk citra banyak kanal dengan menambahkan jumlah *ansatz* sirkuit $U(\vec{\theta})$ sebanyak kanal citra masukan dan proses konvolusi dilakukan untuk tiap kanal. Pada penelitian ini hanya akan digunakan algoritma DRQConv untuk citra masukan 1 kanal karena citra MNIST hanya memiliki 1 kanal.

Penelitian ini menguji coba 3 macam *ansatz* sirkuit berbeda untuk $U(\vec{\theta})$ yang masing-masing akan dibahas dalam Subbab 3.1.2.1, 3.1.2.2, dan 3.1.2.3. Beberapa hal yang dibuat tetap untuk ketiga *ansatz* adalah sebagai berikut:

1. MNIST akan dipotong sisi kanan dan bawahnya sebesar 1 piksel, sehingga dari awalnya berukuran 28×28 akan menjadi berukuran 27×27 . Hal ini dilakukan hanya agar citra masukan memiliki ukuran yang pas selama proses konvolusi. Pemotongan ujung kanan dan bawah sebesar 1 piksel tidak akan berpengaruh pada data karena tulisan tangan terletak di tengah-tengah citra sehingga hanya latar belakang hitam yang akan terpotong.
2. Parameter ζ dipilih sebesar $\zeta = 3 \times 3 = 9$ sehingga ukuran *window* dalam proses konvolusi adalah sebesar 3×3 . Parameter *stride* dipilih sebesar $s = 2$. Dengan ukuran ini, artinya 9 nilai piksel citra akan menjadi masukan ke *ansatz* sirkuit $U(\vec{\theta})$ setiap kali *window* menutupi piksel.
3. Konvolusi akan dilakukan dua kali. Konvolusi pertama mengubah dimensi citra dari 27×27 menjadi 13×13 . Kemudian konvolusi kedua mengubah dimensi citra dari 13×13 menjadi 6×6 . Setelah itu akan dilakukan *max pooling* dengan *stride* $s = 2$ dan ukuran filter 2×2 sehingga dimensi citra berkurang dari 6×6 menjadi 3×3 . Citra kemudian diratakan menjadi sebuah vektor berdimensi 9 yang akan diumpulkan pada DRC untuk diklasifikasi.
4. Notasi $\vec{\theta}_{q,l}^{n \sim n+2}$ yang digunakan pada Subbab 3.1.2.1, 3.1.2.2, dan 3.1.2.3 menggunakan aturan yang sama seperti notasi DRC pada persamaan (2.29) dan (2.30).

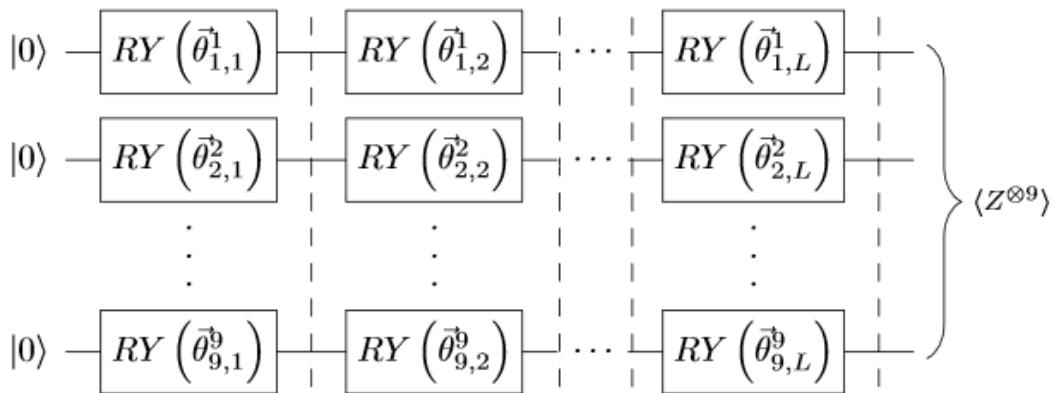
3.1.2.1 DRQConv dengan Gerbang R 3 Qubit (DRQConv I)



Gambar 3.4 Skematik *ansatz* sirkuit pertama yang akan digunakan untuk DRQConv.

Skematik untuk *ansatz* pertama yang akan digunakan dapat dilihat pada Gambar 3.4. *Ansatz* pertama ini terdiri atas 3 gerbang rotasi R yang masing-masing bekerja pada 3 qubit berkeadaan kuantum $|0\rangle$ berbeda yang saling independen. Kemudian deretan 3 gerbang rotasi R ini dapat diulangi sampai sebanyak L lapisan. Keadaan kuantum masing-masing qubit setelah operasi dengan sirkuit ini kemudian diukur nilai ekspektasinya terhadap *observable* gerbang Z. Nilai ekspektasi hasil pengukuran dari ketiga qubit kemudian dijumlahkan.

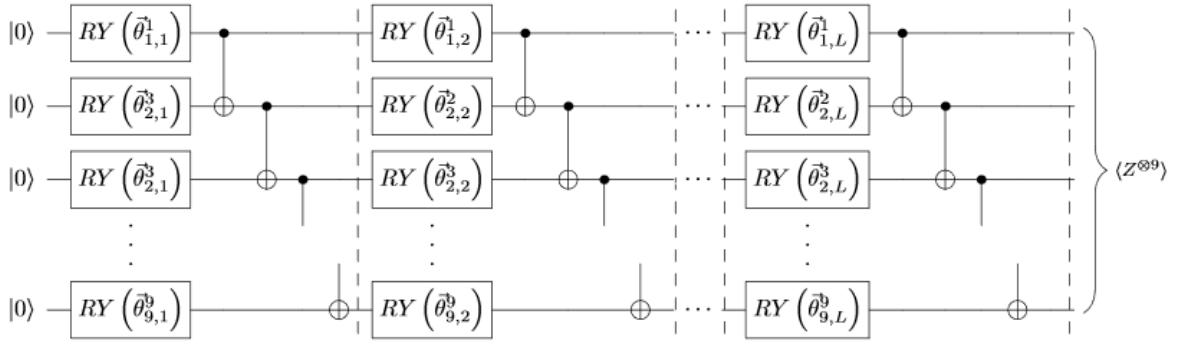
3.1.2.2 DRQConv dengan Gerbang RY 9 Qubit (DRQConv II)



Gambar 3.5 Skematik *ansatz* sirkuit kedua yang akan digunakan untuk DRQConv.

Skematik untuk *ansatz* kedua yang akan digunakan dapat dilihat pada Gambar 3.5. *Ansatz* kedua ini terdiri atas 9 gerbang rotasi RY yang masing-masing bekerja pada 9 qubit berkeadaan kuantum $|0\rangle$ berbeda yang saling independen. Kemudian deretan 9 gerbang RY ini dapat diulangi sampai sebanyak L lapisan. Keadaan kuantum setelah operasi dengan sirkuit ini kemudian diukur nilai ekspektasinya terhadap *observable* perkalian tensor dari 9 gerbang Z ($Z^{\otimes 9} = \underbrace{Z \otimes Z \otimes \dots \otimes Z}_9$).

3.1.2.3 Konvolusi Kuantum dengan Gerbang RY 9 Qubit dan *Entanglement* (DRQConv II + ent)



Gambar 3.6 Skematik *ansatz* sirkuit ketiga yang akan digunakan untuk DRQConv.

Skematik untuk *ansatz* ketiga yang akan digunakan dapat dilihat pada Gambar 3.6. *Ansatz* ketiga ini serupa dengan *ansatz* kedua yang telah dibahas sebelumnya, namun pada tiap lapisan gerbang RY disisipkan gerbang CNOT yang men-*entangle* dua qubit berdekatan. Qubit pertama mengontrol qubit kedua, qubit kedua mengontrol qubit ketiga, dan seterusnya hingga qubit ke delapan mengontrol qubit ke sembilan. Sehingga keadaan kesembilan qubit setelah operasi sirkuit ini menjadi ter-*entangle* dan tidak independen lagi.

3.2 Skema Pengklasifikasi Kuantum

Setelah melalui proses reduksi dimensi (baik dengan PCA atau DRQConv), fitur dari sampel data yang tersisa kemudian diumpulkan ke pengklasifikasi kuantum. Secara umum, pengklasifikasi kuantum yang akan digunakan adalah dengan skema DRC sebagaimana dijelaskan pada Subbab 2.4.4. Namun, terdapat beberapa modifikasi yang juga dilakukan dalam penelitian ini. Subbab 3.2.1 dan 3.2.2 akan menjelaskan secara detail terkait skema pengklasifikasi kuantum yang akan digunakan.

3.2.1 Data Re-uploading Classifier (DRC)

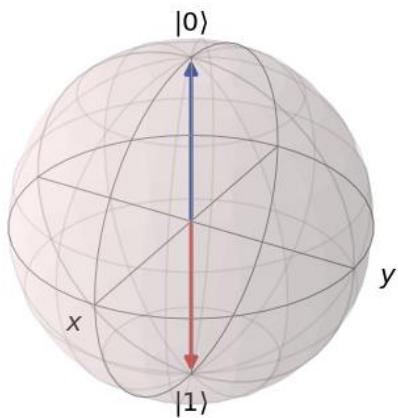
Skema pengklasifikasi kuantum pertama yang digunakan pada penelitian ini adalah skema DRC tanpa modifikasi. Hanya akan digunakan 1 qubit baik untuk klasifikasi biner maupun klasifikasi banyak kelas. Yang membedakan klasifikasi biner dengan klasifikasi banyak kelas adalah *observable* pengukuran yang digunakan.

3.2.1.1 Klasifikasi Biner

Untuk klasifikasi biner, tiap sampel data yang diumpulkan pada DRC akan diukur dua kali dengan dua *observable* yang berbeda. *Observable* pertama adalah $O_0 = |0\rangle\langle 0| = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$ dan *observable* kedua adalah $O_1 = |1\rangle\langle 1| = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$. Sehingga setiap sampel data akan menghasilkan keluaran berupa vektor dua komponen dimana tiap komponennya merupakan hasil pengukuran dengan tiap *observable*. Kemudian, hasil pengukuran ini dikalikan dengan parameter α untuk setiap *observable* yang digunakan. Parameter α ini akan dioptimisasi selama proses latih. Secara matematis keluaran vektor untuk sampel data ke- m dapat dituliskan sebagai berikut:

$$\vec{y}_{pred_m} = \begin{bmatrix} \alpha_0 \langle O_0 \rangle \\ \alpha_1 \langle O_1 \rangle \end{bmatrix}. \quad (3.2)$$

Setiap sampel data memiliki label, karena pada kasus klasifikasi biner hanya terdapat dua kelas, maka label kelas pertama (0) direpresentasikan dengan vektor $\vec{y}_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ dan label kelas kedua (1) direpresentasikan dengan vektor $\vec{y}_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$. Ini artinya, untuk sampel data berlabel \vec{y}_0 , diinginkan agar DRC dapat menghasilkan keadaan kuantum yang sedekat mungkin dengan $|0\rangle$ untuk memaksimalkan nilai ekspektasi $\langle O_0 \rangle$. Sebaliknya, untuk sampel data berlabel \vec{y}_1 , diinginkan agar DRC dapat menghasilkan keadaan kuantum sedekat mungkin dengan $|1\rangle$ untuk memaksimalkan nilai ekspektasi $\langle O_1 \rangle$.



Gambar 3.7 Visualisasi keadaan kuantum $|0\rangle$ dan $|1\rangle$ yang merepresentasikan dua kelas berbeda.

Fungsi biaya yang akan diminimumkan adalah fungsi MSE sebagai berikut:

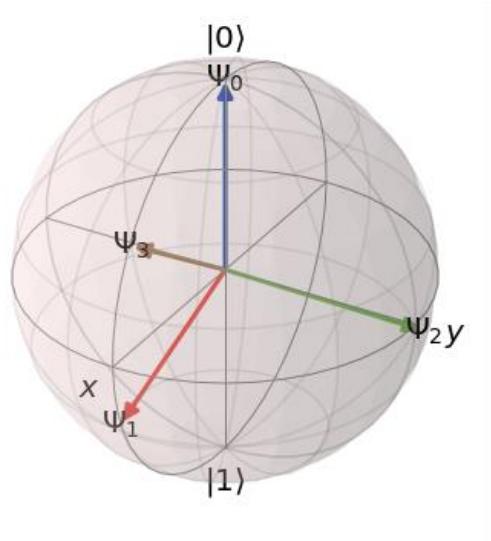
$$J(\vec{y}_{pred}, \vec{y}_{true}) = \frac{1}{C \times M} \sum_{m=1}^M \text{sum} \left\{ \left(\vec{y}_{pred_m} - \vec{y}_{true_m} \right)^2 \right\} \quad (3.3)$$

dengan $C = 2$ adalah jumlah kelas, \vec{y}_{pred} dan \vec{y}_{true} adalah sekumpulan himpunan vektor \vec{y}_{pred_m} dan \vec{y}_{true_m} untuk setiap sampel data dalam himpunan data, \vec{y}_{true_m} adalah label sampel data ke- m yang dapat bernilai \vec{y}_0 atau \vec{y}_1 , M adalah jumlah total sampel data dalam himpunan data, dan $\text{sum}\{\vec{v}\}$ adalah fungsi sumasi elemen vektor \vec{v} .

Pada proses uji, baik klasifikasi biner maupun banyak kelas, DRC dianggap mengklasifikasikan sampel data x_m ke kelas i , dimana i adalah indeks dari komponen vektor keluaran \vec{y}_{pred_m} dengan nilai tertinggi. Gambar 3.7 menunjukkan visualisasi keadaan kuantum $|0\rangle$ dan $|1\rangle$ yang merepresentasikan masing-masing kelas. Terlihat luasan area *Bloch sphere* terbagi dua, setengah bola bagian atas termasuk ke dalam kelas pertama dan setengah bola bagian bawah termasuk ke dalam kelas kedua. Jika keadaan kuantum yang dihasilkan DRC dari suatu sampel data jatuh pada daerah setengah bola bagian atas, keadaan kuantum ini lebih dekat dengan $|0\rangle$ daripada $|1\rangle$ sehingga $\langle O_0 \rangle > \langle O_1 \rangle$ dan sampel data ini akan diklasifikasikan sebagai kelas pertama.

3.2.1.2 Klasifikasi Banyak Kelas

Serupa dengan konsep klasifikasi biner, pada klasifikasi banyak kelas DRC juga diharapkan untuk dapat menghasilkan keadaan kuantum yang berbeda-beda untuk setiap kelas. Agar optimisasi parameter lebih mudah, perlu dipilih keadaan-keadaan kuantum yang saling ortogonal maksimal pada *Bloch sphere*. Pada klasifikasi biner, pemilihan keadaan kuantum untuk mewakili tiap kelas cenderung mudah, cukup dengan keadaan kuantum $|0\rangle$ dan $|1\rangle$ karena kedua keadaan kuantum ini sudah saling ortogonal maksimal. Untuk klasifikasi banyak kelas, pemilihan keadaan kuantum untuk mewakili tiap kelas tidak trivial.



Gambar 3.8 Visualisasi keempat vektor keadaan kuantum dalam *Bloch sphere* untuk merepresentasikan 4 kelas berbeda. Ujung-ujung vektor merupakan titik-titik sudut dari sebuah tetrahedral sama sisi.

Sebagai contoh, untuk klasifikasi 4 kelas, dapat menggunakan bentuk bangun ruang tetrahedral sama sisi di dalam *Bloch sphere* sebagai acuan[19]. Setiap titik sudut tetrahedral menyentuh permukaan *Bloch sphere*. Titik-titik sudut ini menjadi vektor-vektor keadaan kuantum yang mewakili keempat kelas. Gambar 3.8 menunjukkan visualisasi keempat keadaan kuantum ini di dalam *Bloch sphere*. Secara matematis, keadaan kuantum keempat vektor tersebut dapat dituliskan sebagai berikut:

$$|\Psi_0\rangle = |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (3.4)$$

$$|\Psi_1\rangle = \begin{bmatrix} 0,577350 \\ 0,816497 \end{bmatrix}$$

$$|\Psi_2\rangle = \begin{bmatrix} 0,288675 - 0,50j \\ 0,408248 + 0,707107j \end{bmatrix}$$

$$|\Psi_3\rangle = \begin{bmatrix} 0,288675 + 0,50j \\ 0,408248 - 0,707107j \end{bmatrix}$$

dengan j adalah unit imajiner.

Kemudian, karena keadaan kuantum untuk mewakili keempat kelas telah terdapat pada persamaan (3.4), maka yang kita inginkan adalah agar DRC dapat menghasilkan keadaan kuantum yang sesuai dengan label sampel datanya. Misalkan jika sampel data pertama memiliki label kelas 0, maka yang kita inginkan adalah agar DRC menghasilkan keadaan kuantum yang sedekat mungkin dengan $|\Psi_0\rangle$ dan sejauh mungkin dengan keadaan kuantum $|\Psi_1\rangle$, $|\Psi_2\rangle$, dan $|\Psi_3\rangle$.

Sehingga *observable* pengukuran yang kita inginkan adalah *observable* yang dapat mengukur seberapa dekat keadaan kuantum keluaran DRC dengan keempat keadaan kuantum pada persamaan (3.4). Yang paling mudah adalah dengan mengukur fidelitas keadaan kuantum keluaran DRC terhadap keempat keadaan kuantum tersebut dengan *observable* sebagai berikut:

$$\begin{aligned} O_0 &= |\Psi_0\rangle\langle\Psi_0| \\ O_1 &= |\Psi_1\rangle\langle\Psi_1| \\ O_2 &= |\Psi_2\rangle\langle\Psi_2| \\ O_3 &= |\Psi_3\rangle\langle\Psi_3|. \end{aligned} \tag{3.5}$$

Keluaran dari DRC akan diukur empat kali dengan empat *observable* ini. Sehingga setiap sampel data akan menghasilkan keluaran berupa vektor empat komponen dimana tiap komponennya merupakan hasil pengukuran dengan tiap *observable*. Dan sama seperti pada klasifikasi biner, hasil pengukuran dengan tiap *observable* dikalikan dengan parameter α yang akan dioptimisasi selama proses latih. Secara matematis keluaran vektor untuk sampel data ke- m dapat dituliskan sebagai berikut:

$$\vec{y}_{pred_m} = \begin{bmatrix} \alpha_0\langle O_0 \rangle \\ \alpha_1\langle O_1 \rangle \\ \alpha_2\langle O_2 \rangle \\ \alpha_3\langle O_3 \rangle \end{bmatrix}. \tag{3.6}$$

Persamaan untuk menentukan vektor representasi label *ground truth* untuk keempat kelas adalah sebagai berikut:

$$\vec{y}_i = \begin{bmatrix} \langle \Psi_i | O_0 | \Psi_i \rangle \\ \langle \Psi_i | O_1 | \Psi_i \rangle \\ \langle \Psi_i | O_2 | \Psi_i \rangle \\ \langle \Psi_i | O_3 | \Psi_i \rangle \end{bmatrix} \quad (3.7)$$

sehingga didapatkan vektor representasi label *ground truth* untuk tiap kelas (0, 1, 2, dan 3) sebagai berikut:

$$\vec{y}_0 = \begin{bmatrix} 1 \\ 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}, \quad \vec{y}_1 = \begin{bmatrix} 1/3 \\ 1 \\ 1/3 \\ 1/3 \end{bmatrix}, \quad \vec{y}_2 = \begin{bmatrix} 1/3 \\ 1/3 \\ 1 \\ 1/3 \end{bmatrix}, \quad \vec{y}_3 = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \\ 1 \end{bmatrix}. \quad (3.8)$$

Fungsi biaya yang akan diminimumkan adalah fungsi MSE sebagai berikut:

$$J(\vec{y}_{pred}, \vec{y}_{true}) = \frac{1}{C \times M} \sum_{m=1}^M \text{sum} \left\{ \left(\vec{y}_{pred_m} - \vec{y}_{true_m} \right)^2 \right\} \quad (3.9)$$

dengan $C = 4$ adalah jumlah kelas, \vec{y}_{true_m} adalah label sampel data ke- m yang dapat bernilai \vec{y}_0 , \vec{y}_1 , \vec{y}_2 , atau \vec{y}_3 .

3.2.2 DRC dengan Representasi Biner (DRC-RB)

Terlihat persiapan skema DRC untuk klasifikasi banyak kelas tidaklah mudah. Bukan hanya itu, penulis berhipotesis bahwa performa akurasi klasifikasi skema DRC akan semakin turun (dengan laju yang cukup drastis) seiring dengan penambahan jumlah kelas. Hal ini disebabkan karena luas area bagian *Bloch sphere* yang semakin sempit untuk tiap kelas seiring bertambahnya jumlah kelas. Pada klasifikasi biner, area *Bloch sphere* hanya terbagi ke dalam dua region, tiap kelas memiliki luas area sebesar setengah permukaan bola. Dengan area seluas ini, sangat mudah bagi DRC untuk menemukan himpunan parameter yang dapat memetakan semua sampel data ke dalam region *Bloch sphere* yang benar. Tetapi pada pembagian region *Bloch sphere* untuk 4 kelas (Gambar 3.8) terlihat bahwa sekarang tiap kelas memiliki luas region setengah kali dari luas pada klasifikasi biner karena

luas satu bola harus dibagi ke dalam 4 region. Tentunya akan semakin sulit bagi algoritma optimisasi untuk menemukan himpunan parameter yang dapat memetakan semua sampel data ke dalam region yang benar karena luas region yang semakin sempit.

Oleh karena itu, pada penelitian ini penulis mengajukan modifikasi pada skema DRC untuk klasifikasi banyak kelas yaitu dengan memanfaatkan representasi bilangan biner. Skema modifikasi ini penulis sebut sebagai *Data Re-uploading Classifier* dengan Representasi Biner (DRC-RB).

Daripada membagi *Bloch sphere* sebuah qubit ke dalam region-region kecil, DRC-RB menambahkan jumlah qubit untuk mengakomodasi tambahan kelas. DRC-RB menggunakan jumlah qubit sebanyak $Q = \log_2 C$ secara independen untuk mengklasifikasikan data ke dalam C kelas, dimana setiap qubit berperan sebagai pengklasifikasi biner. Dengan begini, performa klasifikasi tiap qubit dapat tetap terjaga (atau setidaknya tidak menurun drastis) karena setiap *Bloch sphere* hanya dibagi ke dalam dua region saja seperti pada kasus klasifikasi 2 kelas.

Misalkan untuk mengklasifikasi $C = 4$ kelas, DRC-RB membutuhkan qubit sebanyak $Q = \log_2 4 = 2$. Kemudian, setiap label kelas dicari representasi binernya dengan jumlah bit sebanyak jumlah qubit yang digunakan sebagai berikut:

$$\begin{aligned} \text{label 0} &= 00 \\ \text{label 1} &= 01 \\ \text{label 2} &= 10 \\ \text{label 3} &= 11. \end{aligned} \tag{3.10}$$

Dengan skema DRC pada masing-masing qubit, qubit pertama bertanggung jawab mengklasifikasikan sampel data dengan nomor pada bit pertama sebagai kelasnya, qubit kedua bertanggung jawab mengklasifikasikan sampel data dengan nomor pada bit kedua sebagai kelasnya. Sehingga, qubit pertama berperan sebagai pengklasifikasi biner yang memisahkan sampel data berlabel 0 dan 1 dengan sampel data berlabel 2 dan 3, qubit kedua berperan sebagai pengklasifikasi biner untuk memisahkan sampel data berlabel 0 dan 2 dengan sampel data berlabel 1 dan 3. Dengan begini, setiap qubit kembali hanya berperan sebagai pengklasifikasi biner,

namun kombinasi kedua qubit akan mampu mengklasifikasikan data ke dalam 4 kelas.

Fungsi biaya yang akan diminimumkan untuk kasus 4 kelas adalah sebagai berikut:

$$\begin{aligned}
 J(\vec{y}_{pred}, \vec{y}_{true}) &= \frac{1}{Q \times C \times M} \sum_{q=1}^{Q=2} \sum_{m=1}^M \text{sum} \left\{ \left(\vec{y}_{pred_{q,m}} - \vec{y}_{true_{q,m}} \right)^2 \right\} \\
 \vec{y}_{pred_{q,m}} &= \begin{bmatrix} \alpha_{0q} \langle O_0 \rangle_q \\ \alpha_{1q} \langle O_1 \rangle_q \end{bmatrix} \\
 O_0 = |0\rangle\langle 0| &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad O_1 = |1\rangle\langle 1| = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \\
 \vec{y}_0 &= \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \vec{y}_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}
 \end{aligned} \tag{3.11}$$

dengan indeks q dan m menandakan keluaran dari qubit ke- q dan sampel data ke- m , $\vec{y}_{true_{q,m}}$ adalah label pada posisi qubit ke- q untuk sampel data ke- m yang dapat bernilai \vec{y}_0 (representasi label 0) atau \vec{y}_1 (representasi label 1).

Pada proses uji, misalkan untuk sampel data ke- m , qubit pertama menghasilkan $\vec{y}_{pred_{1,m}}$ dengan nilai komponen tertingginya berada pada indeks kedua kemudian qubit kedua menghasilkan $\vec{y}_{pred_{2,m}}$ dengan nilai komponen tertingginya berada pada indeks pertama. Artinya qubit pertama mengklasifikasikan sampel data ke dalam label 1 dan qubit kedua mengklasifikasikan sampel data ke dalam label 0. Kombinasi kedua qubit menghasilkan klasifikasi 10, yang adalah label 2 berdasarkan persamaan (3.10).

3.3 Rangkuman Skema Pengklasifikasi yang Diuji

Karena pada penelitian ini akan dilakukan uji coba numerik beberapa kombinasi antara skema reduksi dimensi citra dengan skema pengklasifikasi kuantum, subbab ini akan merangkum kombinasi seperti apa saja yang akan dilakukan pada penelitian ini untuk memudahkan pembaca memahami hasil-hasil pada Bab 4. Selain itu, akan dirangkumkan juga spesifikasi-spesifikasi penting yang digunakan dalam uji coba numerik.

3.3.1 Pengklasifikasi DRC dengan Reduksi PCA

Untuk kombinasi ini, akan dilakukan uji coba klasifikasi $C = 2$ kelas (angka 0 dan angka 1) dan $C = 4$ kelas (angka 0, angka 1, angka 2, dan angka 3) dengan spesifikasi tertera pada Tabel 3.1.

Tabel 3.1 Spesifikasi pengujian kombinasi pengklasifikasi DRC dengan reduksi PCA.

Jumlah sampel data latih	200 sampel/kelas
Jumlah sampel data uji	500 sampel/kelas untuk 2 kelas, 50 sampel/kelas untuk 4 kelas
<i>Epoch</i>	10 iterasi
Ukuran <i>batch</i>	32 sampel
Algoritma Optimisasi	Adam
Laju pembelajaran	0,1 (konstan)
Jumlah komponen prinsip PCA (p)	Divariasikan
Jumlah qubit untuk DRC (Q)	1
Jumlah lapisan DRC (L)	Divariasikan
Total parameter yang perlu dioptimisasi	$2 \times L \times p + C$
Fungsi biaya	MSE

3.3.2 Pengklasifikasi DRC-RB dengan Reduksi PCA

Untuk kombinasi ini, akan dilakukan uji coba klasifikasi $C = 4$ kelas (angka 0, angka 1, angka 2, dan angka 3) dan $C = 8$ kelas (angka 0, angka 1, angka 2, angka 3, angka 4, angka 5, angka 6, dan angka 7) dengan spesifikasi tertera pada Tabel 3.2.

Tabel 3.2 Spesifikasi pengujian kombinasi pengklasifikasi DRC-RB dengan reduksi PCA.

Jumlah sampel data latih	200 sampel/kelas
Jumlah sampel data uji	50 sampel/kelas
<i>Epoch</i>	10 iterasi
Ukuran <i>batch</i>	32 sampel untuk 4 kelas, 64 sampel untuk 8 kelas
Algoritma Optimisasi	Adam
Laju pembelajaran	0,1 (konstan) untuk 4 kelas, laju pembelajaran meluruh untuk 8 kelas dengan $\eta_0 = 0,1$, $\tau = 0,95$, dan $\Delta = 50$
Jumlah komponen prinsip PCA (p)	18 untuk 4 kelas, 27 untuk 8 kelas
Jumlah qubit untuk DRC (Q)	$\log_2 C$
Jumlah lapisan DRC (L)	4
Total parameter yang perlu dioptimisasi	$(2 \times L \times p + 2) \times \log_2 C$
Fungsi biaya	MSE

Pada kombinasi ini tidak dilakukan pengujian klasifikasi 2 kelas karena skema pengklasifikasi DRC-RB akan menjadi sama dengan skema pengklasifikasi DRC untuk klasifikasi 2 kelas.

3.3.3 Pengklasifikasi DRC-RB dengan Reduksi Konvolusi Kuantum (DRQConv)

Untuk kombinasi ini, akan dilakukan uji coba klasifikasi $C = 2$ kelas (angka 0 dan angka 1) dan $C = 4$ kelas (angka 0, angka 1, angka 2, dan angka 3) dengan 3 *ansatz* sirkuit (DRQConv I, DRQConv II, DRQConv II + ent) yang berbeda untuk konvolusi kuantumnya.

Spesifikasi untuk klasifikasi 2 kelas tertera pada Tabel 3.3. Perlu diingat bahwa pada klasifikasi 2 kelas, tidak ada perbedaan antara skema pengklasifikasi DRC dengan DRC-RB. Penulis juga melakukan uji coba JST Konvolusi yang nilai akurasinya digunakan sebagai referensi dengan spesifikasi tertera pada Tabel 3.4.

Tabel 3.3 Spesifikasi pengujian kombinasi pengklasifikasi DRC-RB dengan reduksi DRQConv untuk klasifikasi 2 kelas.

Jumlah sampel data latih	200 sampel/kelas
Jumlah sampel data uji	500 sampel/kelas
<i>Epoch</i>	10 iterasi
Ukuran <i>batch</i>	32 sampel
Algoritma Optimisasi	Adam
Laju pembelajaran	0,1 (konstan)
Jumlah Proses Konvolusi	2 kali konvolusi. Citra awal 27 x 27 menjadi 13 x 13 setelah konvolusi pertama dan menjadi 6 x 6 setelah konvolusi kedua
Jumlah qubit untuk DRQConv (Q)	DRQConv I: 3, DRQConv II: 9, DRQConv II + ent: 9
Jumlah lapisan DRQConv (L) pada tiap proses konvolusi	1
Jumlah operasi <i>pooling</i>	1 kali <i>max pooling</i> setelah konvolusi kedua dengan ukuran filter 2 x 2 dan <i>stride</i> 2. Ukuran citra menjadi 3 x 3 setelah operasi <i>pooling</i>
Jumlah qubit untuk DRC (Q)	1
Jumlah lapisan DRC (L)	1
Total parameter yang perlu dioptimisasi	Parameter DRQConv: $18 + 18 = 36$ Parameter DRC: $18 + 2 = 20$ Total parameter: 56
Fungsi biaya	MSE

Tabel 3.4 Spesifikasi pengujian JST Konvolusi 2 kelas.

Jumlah sampel data latih	200 sampel/kelas
Jumlah sampel data uji	500 sampel/kelas
<i>Epoch</i>	10 iterasi
Ukuran <i>batch</i>	32 sampel
Algoritma Optimisasi	Adam
Laju pembelajaran	0,1 (konstan)
Jumlah Proses Konvolusi	2 kali konvolusi. Citra awal 27 x 27 menjadi 13 x 13 setelah konvolusi pertama dan menjadi 6 x 6 setelah konvolusi kedua
Spesifikasi filter	Ukuran filter 3 x 3 dengan <i>strides</i> 2. Digunakan 2 filter pada konvolusi pertama dan 1 filter pada konvolusi kedua. Fungsi aktivasi yang digunakan adalah ReLU
Jumlah operasi <i>pooling</i>	1 kali <i>max pooling</i> setelah konvolusi kedua dengan ukuran filter 2 x 2 dan <i>stride</i> 2. Ukuran citra menjadi 3 x 3 setelah operasi <i>pooling</i>
Jumlah lapisan dan simpul JST	Keluaran <i>max pooling</i> kemudian diratakan dan diumpulkan ke lapisan keluaran JST dengan jumlah simpul 2 dan fungsi aktivasi <i>softmax</i>
Total parameter yang perlu dioptimisasi	Parameter Konvolusi: $18 + 18 = 36$ Parameter JST: 20 Total parameter: 56
Fungsi biaya	<i>Categorical crossentropy</i>

Untuk klasifikasi 4 kelas, sedikit modifikasi dilakukan untuk representasi biner pada skema pengklasifikasi DRC-RB. Pada Subbab 3.2.2, telah diberikan penjabaran skema DRC-RB untuk klasifikasi 4 kelas. Disana dijelaskan bagaimana menggunakan 2 qubit untuk memprediksi 4 kelas dalam representasi biner seperti pada persamaan (3.10). Di pengujian ini, setelah citra direduksi dengan konvolusi dan *pooling*, citra diklasifikasikan dengan DRC-RB menggunakan 4 qubit independen, dimana masing-masing qubit bertanggung jawab mengklasifikasikan apakah sampel data termasuk ke dalam salah satu kelas atau tidak. Sehingga yang berbeda disini hanyalah representasi biner tiap-tiap kelasnya. Representasi biner keempat kelas tersebut menjadi sebagai berikut:

$$\begin{aligned}
 \text{label 0} &= 0111 \\
 \text{label 1} &= 1011 \\
 \text{label 2} &= 1101 \\
 \text{label 3} &= 1110.
 \end{aligned} \tag{3.12}$$

Pada proses pengujian, misalkan citra x_m adalah sampel data angka 0, setelah citra dikonvolusi dan di-*pooling*, berikut adalah contoh keluaran dari setiap qubit:

$$\begin{aligned}\vec{y}_{pred_{0,m}} &= \begin{bmatrix} 0.85 \\ 0.1 \end{bmatrix}, & \vec{y}_{pred_{1,m}} &= \begin{bmatrix} 0.55 \\ 0.4 \end{bmatrix} \\ \vec{y}_{pred_{2,m}} &= \begin{bmatrix} 0.21 \\ 0.9 \end{bmatrix}, & \vec{y}_{pred_{3,m}} &= \begin{bmatrix} 0.04 \\ 0.98 \end{bmatrix}. \end{aligned} \quad (3.13)$$

Dari contoh numerik ini, terlihat bahwa qubit pertama (indeks 0) mengklasifikasikan sampel ke dalam kelas 0, qubit kedua (indeks 1) mengklasifikasikan sampel ke dalam kelas 0, qubit ketiga (indeks 2) mengklasifikasikan sampel ke dalam kelas 1, qubit terakhir (indeks 3) mengklasifikasikan sampel ke dalam kelas 1. Kombinasi keempat qubit ini menghasilkan 0011, kombinasi yang tidak terdapat pada persamaan (3.12). Sehingga, saat pengujian, dipilih hanya satu qubit yang mengklasifikasikan sampel ke dalam kelas 0, yaitu qubit dengan vektor keluaran yang paling dekat dengan vektor $\vec{y}_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, sisa qubit lainnya dianggap mengklasifikasikan sampel ke dalam kelas 1. Penentuan seberapa dekat vektor keluaran dengan vektor \vec{y}_0 dihitung dengan *squared error*. Pada contoh numerik ini, qubit pertama menghasilkan keluaran yang paling mendekati \vec{y}_0 sehingga qubit pertama mengklasifikasikan sampel ke dalam kelas 0, sisa qubit lainnya otomatis kelas 1, menghasilkan kombinasi 0111 yang adalah label 0 berdasarkan persamaan (3.12).

Modifikasi ini dilakukan agar sirkuit DRC dapat mengakomodasi lebih banyak parameter tanpa perlu terlalu banyak menambahkan jumlah gerbang. Dari iterasi uji coba, penulis menemukan tugas klasifikasi 4 kelas jauh lebih sulit daripada klasifikasi 2 kelas sehingga sirkuit membutuhkan lebih banyak parameter untuk menghasilkan performa klasifikasi yang baik pada klasifikasi 4 kelas. Spesifikasi untuk pengujian DRC-RB (dengan sedikit modifikasi pada representasi binernya) tertera pada Tabel 3.5. Dilakukan juga uji coba JST Konvolusi 4 kelas dengan spesifikasi pada Tabel 3.6 yang akurasinya digunakan sebagai referensi.

Tabel 3.5 Spesifikasi pengujian kombinasi pengklasifikasi DRC-RB dengan reduksi DRQConv untuk klasifikasi 4 kelas.

Jumlah sampel data latih	200 sampel/kelas
Jumlah sampel data uji	50 sampel/kelas
<i>Epoch</i>	10 iterasi x 2 proses latih
Ukuran <i>batch</i>	32 sampel
Algoritma Optimisasi	Adam
Laju pembelajaran	0,1 (konstan) pada proses latih pertama, laju pembelajaran meluruh pada proses latih kedua dengan $\eta_0 = 0,1$, $\tau = 0,95$, dan $\Delta = 25$
Jumlah Proses Konvolusi	2 kali konvolusi. Citra awal 27 x 27 menjadi 13 x 13 setelah konvolusi pertama dan menjadi 6 x 6 setelah konvolusi kedua
Jumlah qubit untuk DRQConv (Q)	DRQConv I: 3, DRQConv II: 9, DRQConv II + ent: 9
Jumlah lapisan DRQConv (L) pada tiap proses konvolusi	2
Jumlah operasi <i>pooling</i>	1 kali <i>max pooling</i> setelah konvolusi kedua dengan ukuran filter 2 x 2 dan <i>stride</i> 2. Ukuran citra menjadi 3 x 3 setelah operasi <i>pooling</i>
Jumlah qubit untuk DRC (Q)	4
Jumlah lapisan DRC (L)	2
Total parameter yang perlu dioptimisasi	Parameter DRQConv: $2 \times (18 + 18) = 72$ Parameter DRC: $4 \times (2 \times 18 + 2) = 152$ Total parameter: 224
Fungsi biaya	MSE

Tabel 3.6 Spesifikasi pengujian JST Konvolusi 4 kelas.

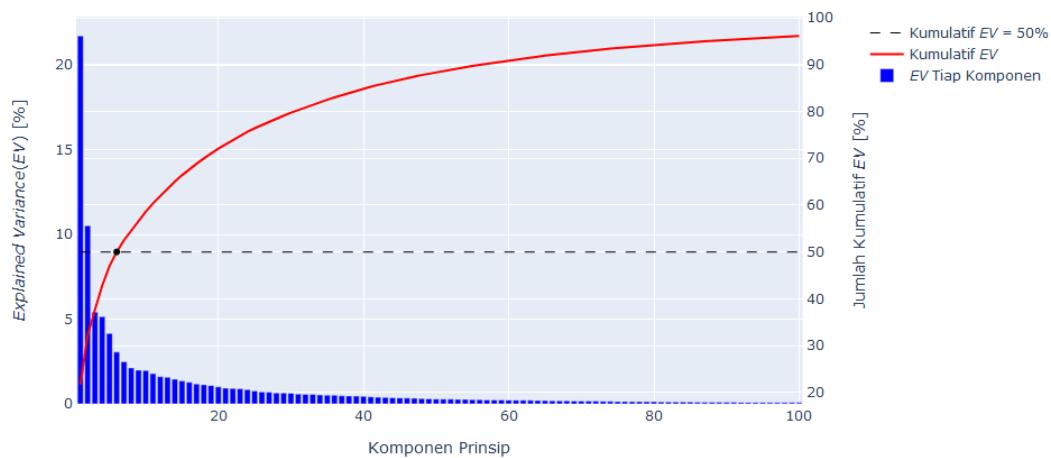
Jumlah sampel data latih	200 sampel/kelas
Jumlah sampel data uji	50 sampel/kelas
<i>Epoch</i>	10 iterasi x 2 proses latih
Ukuran <i>batch</i>	32 sampel
Algoritma Optimisasi	Adam
Laju pembelajaran	0,1 (konstan) pada proses latih pertama, laju pembelajaran meluruh pada proses latih kedua dengan $\eta_0 = 0,1$, $\tau = 0,95$, dan $\Delta = 25$
Jumlah Proses Konvolusi	2 kali konvolusi. Citra awal 27×27 menjadi 13×13 setelah konvolusi pertama dan menjadi 6×6 setelah konvolusi kedua
Spesifikasi filter	Ukuran filter 3×3 dengan <i>strides</i> 2. Digunakan 4 filter pada konvolusi pertama dan 1 filter pada konvolusi kedua. Fungsi aktivasi yang digunakan adalah ReLU
Jumlah operasi <i>pooling</i>	1 kali <i>max pooling</i> setelah konvolusi kedua dengan ukuran filter 2×2 dan <i>stride</i> 2. Ukuran citra menjadi 3×3 setelah operasi <i>pooling</i>
Jumlah lapisan dan simpul JST	Keluaran <i>max pooling</i> kemudian diratakan dan diumpulkan ke lapisan tersembunyi dengan jumlah simpul 11 (fungsi aktivasi ReLU) dan lapisan keluaran dengan jumlah simpul 4 (fungsi aktivasi <i>softmax</i>)
Total parameter yang perlu dioptimisasi	Parameter Konvolusi: $36 + 36 = 72$ Parameter JST: $110 + 48 = 158$ Total parameter: 230
Fungsi biaya	<i>Categorical crossentropy</i>

BAB IV

HASIL DAN PEMBAHASAN

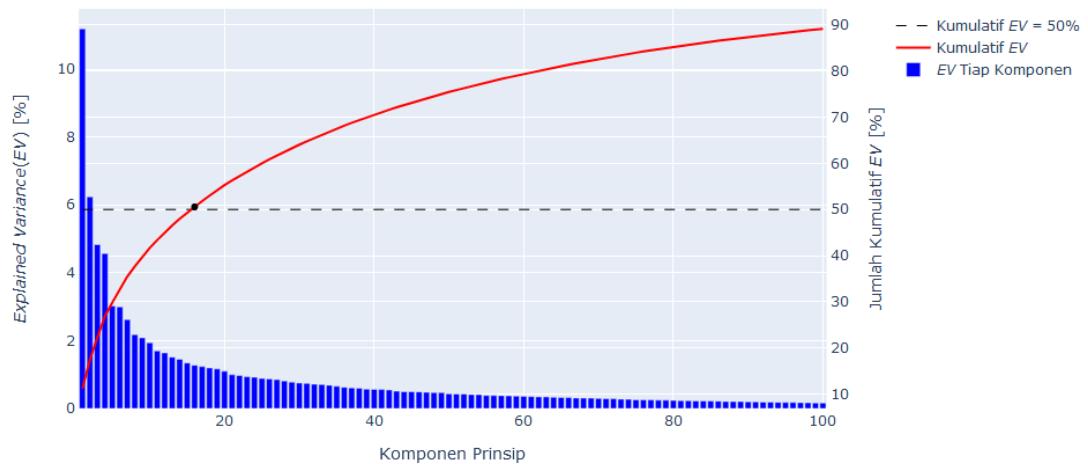
4.1 *Explained Variance (EV) PCA*

Sebelum menentukan berapa banyak jumlah komponen prinsip yang akan digunakan pada pengujian, analisis nilai *EV* dilakukan terlebih dahulu untuk himpunan data MNIST 2 kelas (angka 0 dan 1), 4 kelas (angka 0, 1, 2, dan 3), dan 8 kelas (angka 0-7).

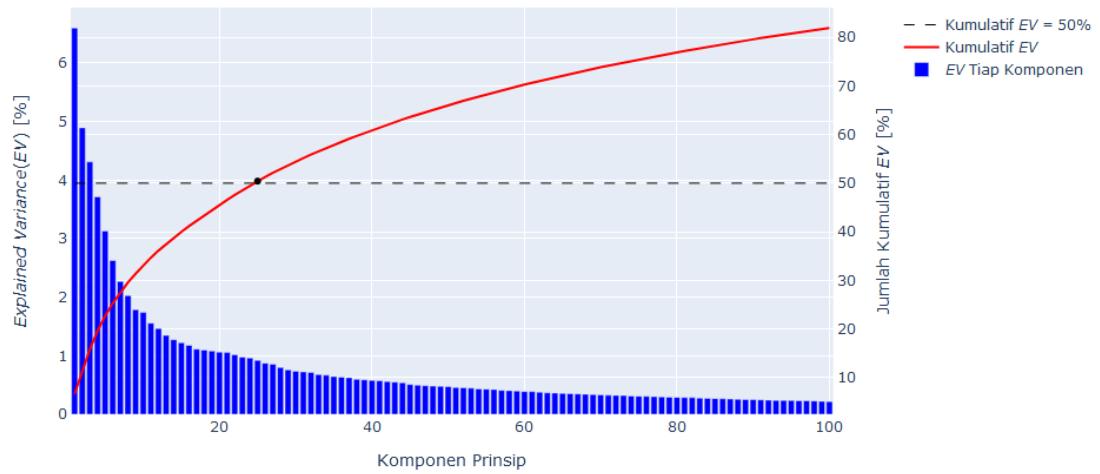


Gambar 4.1 Grafik nilai *EV* untuk 100 komponen prinsip setelah transformasi himpunan data 2 kelas dengan PCA.

Gambar 4.1 menunjukkan grafik nilai *EV* (bar berwarna biru) 100 komponen prinsip (fitur) dengan nilai *EV* tertinggi setelah himpunan data 2 kelas ditransformasi dengan PCA. Kurva merah menunjukkan jumlah kumulatif dari nilai *EV*, titik hitam pada kurva merah menunjukkan titik pertama kali jumlah kumulatif $EV \geq 50\%$. Terlihat bahwa hanya 3 fitur pertama saja yang memiliki *EV* di atas 5%. Kemudian terlihat juga bahwa hanya dengan 6 fitur saja jumlah kumulatif *EV*-nya sudah mencapai lebih dari 50%. Ini menunjukkan hanya dengan menggunakan 6 komponen prinsip, himpunan data baru sudah mewakili setidaknya lebih dari 50% variansi himpunan data. Mengingat jumlah fitur citra MNIST ada sebanyak $28 \times 28 = 784$, reduksi PCA menyisakan 6 fitur dengan tetap mempertahankan 50% variansi data menunjukkan PCA dapat digunakan untuk mereduksi dimensi fitur himpunan data MNIST dengan efektif.



Gambar 4.2 Grafik nilai *EV* untuk 100 komponen prinsip setelah transformasi himpunan data 4 kelas dengan PCA.



Gambar 4.3 Grafik nilai *EV* untuk 100 komponen prinsip setelah transformasi himpunan data 8 kelas dengan PCA.

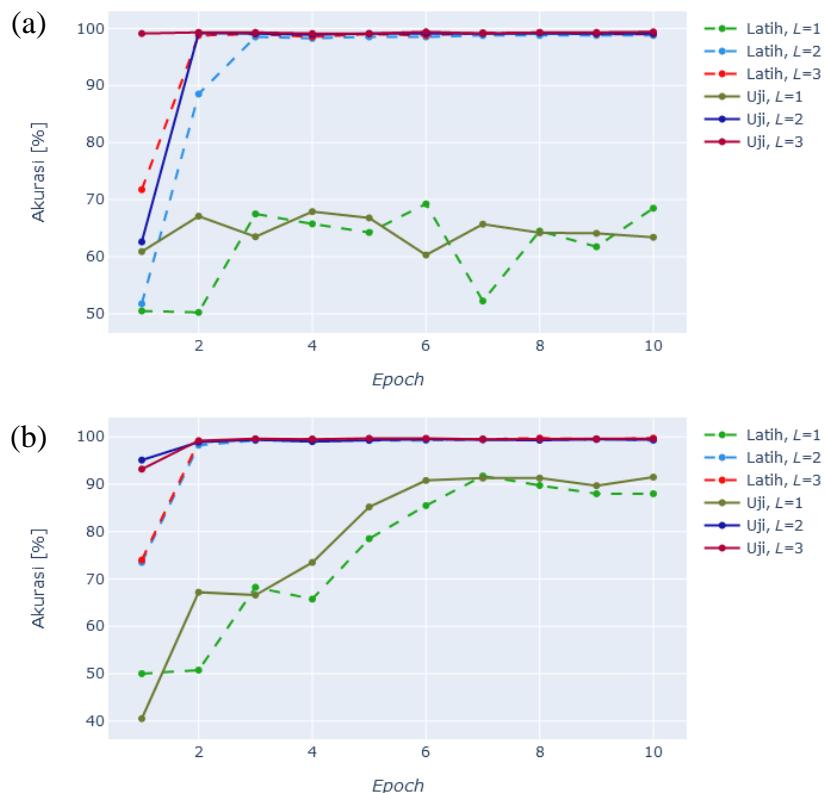
Gambar 4.2 dan Gambar 4.3 menunjukkan nilai *EV* untuk himpunan data 4 dan 8 kelas. Terlihat bahwa distribusi nilai *EV* antar komponen prinsip masih serupa dengan himpunan data 2 kelas, yaitu besar nilainya di sebagian kecil komponen prinsip awal saja. Namun seiring bertambahnya kelas, distribusi nilai *EV* semakin menyebar, terlihat dari berkurangnya nilai *EV* untuk komponen-komponen prinsip di awal tersebut. Hal ini wajar mengingat himpunan data akan semakin bervariasi seiring dengan bertambahnya jumlah kelas. Meski begitu, himpunan data hasil transformasi tetap mampu menangkap lebih dari 50% variansi data dengan jumlah fitur kurang dari 30. Lebih dari 50% variansi data tercapai dengan menggunakan

16 komponen prinsip pada himpunan data 4 kelas dan 25 komponen prinsip pada himpunan data 8 kelas.

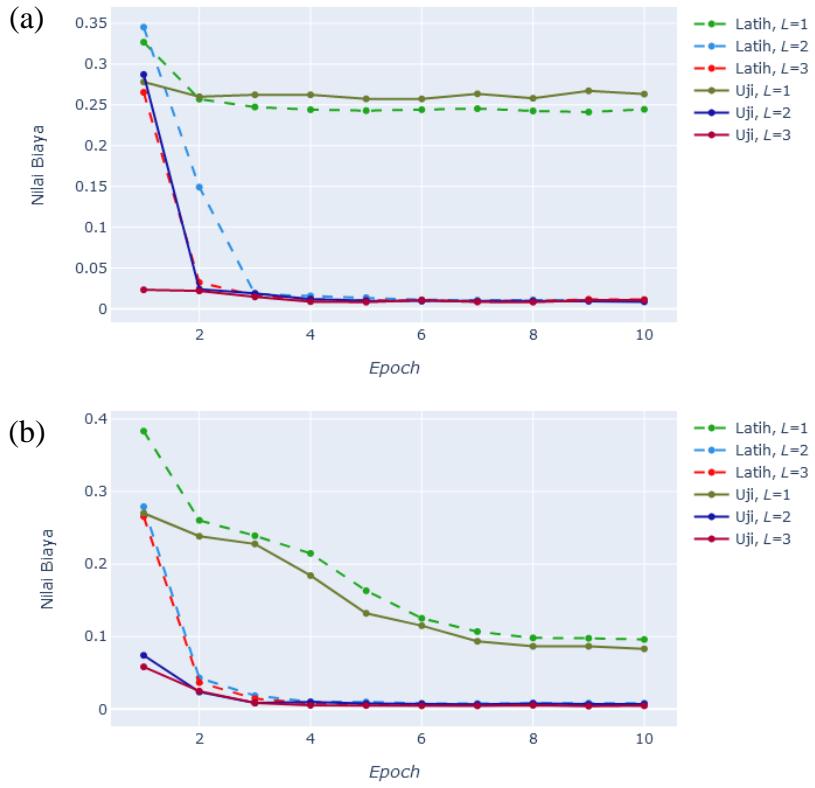
Dari hasil-hasil ini, jumlah komponen prinsip yang akan digunakan pada uji coba penelitian ini adalah sebanyak yang dibutuhkan hingga mencapai kumulatif EV minimal 50%, dan beberapa jumlah komponen prinsip lainnya yang lebih rendah sebagai perbandingan. Untuk 2 kelas, akan digunakan 3 dan 6 komponen prinsip. Untuk 4 kelas, akan digunakan 6, 12, dan 18 komponen prinsip. Angka 18 dipilih karena adalah bilangan bulat terdekat dari 16 (titik tercapainya 50% kumulatif EV) yang habis dibagi 3. Jumlah komponen dipilih kelipatan 3 karena skema DRC membutuhkan 3 fitur sebagai masukan pada tiap gerbangnya. Kemudian untuk 8 kelas, akan digunakan 27 komponen prinsip dengan alasan yang sama.

4.2 Performa Pengklasifikasi DRC dengan Reduksi PCA

4.2.1 Klasifikasi 2 Kelas



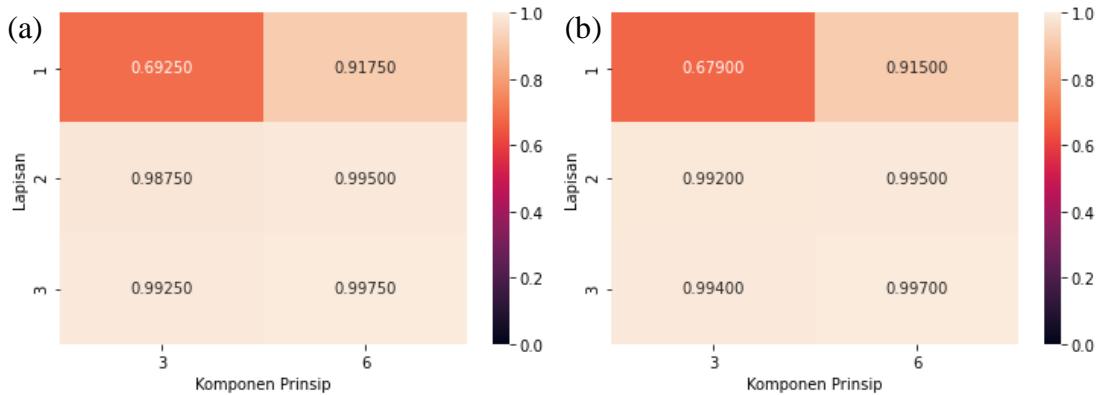
Gambar 4.4 Grafik nilai akurasi pengklasifikasi selama proses latih pada tiap *epoch*. (a) 3 komponen prinsip, (b) 6 komponen prinsip.



Gambar 4.5 Grafik nilai biaya pengklasifikasi selama proses latih pada tiap epoch. (a) 3 komponen prinsip, (b) 6 komponen prinsip.

Hasil nilai akurasi dan nilai biaya dari pengklasifikasi DRC dengan himpunan data 3 dan 6 komponen disajikan pada Gambar 4.4 dan Gambar 4.5. Warna yang berbeda menunjukkan perbedaan jumlah lapisan DRC yang digunakan. Garis putus-putus menunjukkan hasil evaluasi dengan himpunan data latih, sedangkan garis tegas menunjukkan hasil evaluasi dengan himpunan data uji.

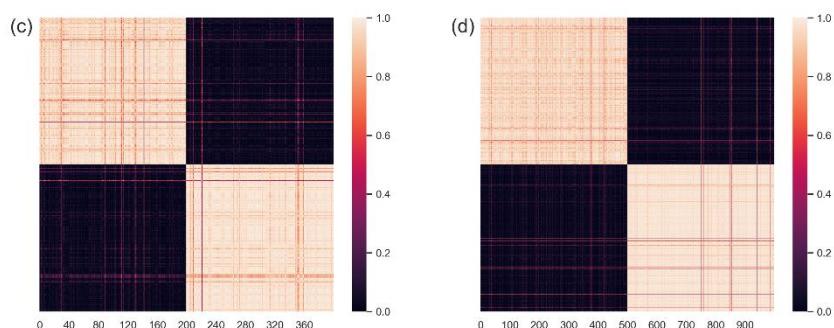
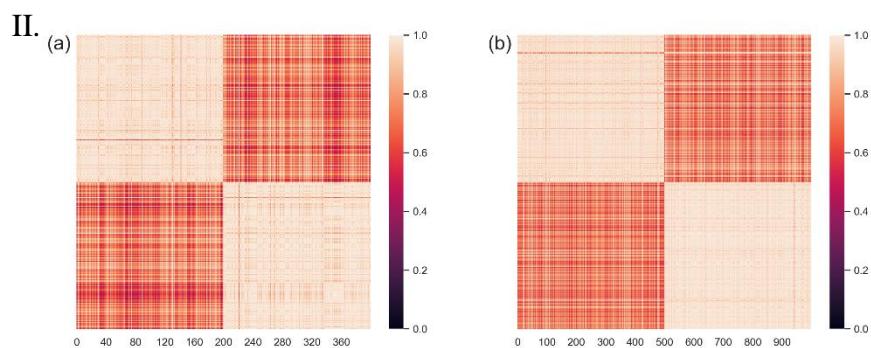
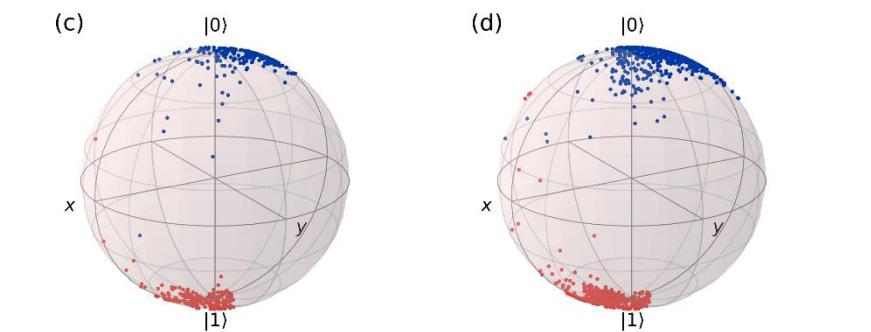
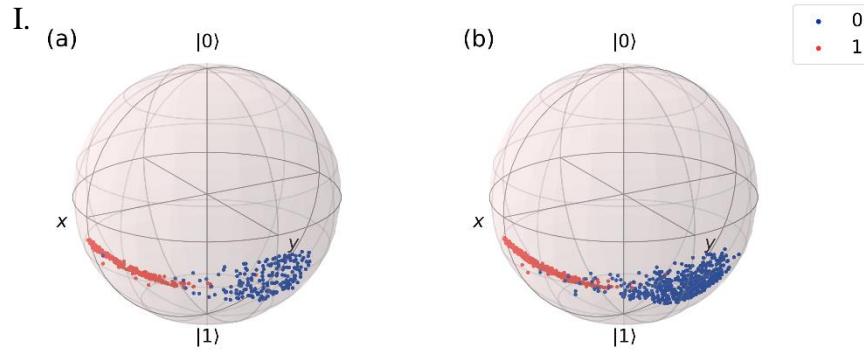
Dari hasil ini, terlihat bahwa penambahan lapisan DRC meningkatkan akurasi lebih signifikan daripada penambahan komponen prinsip. 3 komponen prinsip dengan 2 lapisan DRC memiliki akurasi yang lebih baik daripada 6 komponen prinsip yang hanya menggunakan 1 lapisan DRC. Hasil ini menunjukkan unsur “*data re-uploading*” pada DRC berperan besar dalam peningkatan akurasi klasifikasi.



Gambar 4.6 Tabulasi capaian akurasi terbaik dari evaluasi (a) himpunan data latih dan (b) himpunan data uji dalam representasi *heatmap*. Angka 1 bermakna tercapainya 100% akurasi.

Gambar 4.6 menunjukkan capaian akurasi terbaik tiap konfigurasi komponen prinsip dan jumlah lapisan yang digunakan dalam jangkauan 10 *epoch*/iterasi. Tabulasi ini memberikan gambaran yang lebih jelas akan pengaruh dari penambahan jumlah komponen prinsip dan jumlah lapisan yang digunakan terhadap akurasi klasifikasi.

Kembali terlihat penambahan lapisan lebih signifikan dalam meningkatkan akurasi daripada penambahan jumlah komponen. Pada evaluasi himpunan data uji, peningkatan akurasi dari 3 komponen prinsip dan 1 lapisan ke 3 komponen prinsip dan 2 lapisan sebesar 0,313 (31,3%). Sedangkan peningkatan akurasi dari 3 komponen prinsip dan 1 lapisan ke 6 komponen prinsip dan 1 lapisan hanya sebesar 0,236 (23,6%).



Gambar 4.7 Representasi I. *Bloch sphere* dan II. *heatmap* dari keadaan kuantum keluaran DRC untuk (a) himpunan data latih dan (b) himpunan data uji sebelum proses latih, dan (c) himpunan data latih dan (d) himpunan data uji setelah proses latih.

Representasi *Bloch sphere* dari keadaan kuantum keluaran DRC untuk setiap sampel data dapat terlihat pada Gambar 4.7 bagian I. Sebelum proses latih, keadaan kuantum dari sampel – sampel berada pada posisi yang sembarang di dalam *Bloch sphere*. Persebaran sampel data agak terpisah antara kelas angka 0 dan kelas angka 1, namun posisinya sembarang. Setelah proses latih, sampel data kelas angka 0 berkumpul di sekitar keadaan kuantum $|0\rangle$ dan sampel data kelas angka 1 berkumpul di sekitar keadaan kuantum $|1\rangle$, sesuai dengan yang diinginkan. Karena hanya terdapat 2 kelas, permukaan *Bloch sphere* hanya terbagi dua. Setiap sampel yang jatuh pada setengah atas permukaan bola akan diklasifikasikan ke kelas angka 0, dan ke kelas angka 1 untuk yang jatuh pada setengah bawah permukaan bola. Terdapat sedikit sampel data yang tidak pada bagian yang sesuai, hal ini karena pengklasifikasi tidak mencapai akurasi 100%.

Gambar 4.7 bagian II menunjukkan fidelitas antar keadaan kuantum tiap sampel dalam representasi *heatmap*. Fidelitas adalah ukuran seberapa mirip dua keadaan kuantum, yang dapat dihitung nilainya dengan persamaan berikut:

$$Fidelitas_{i,j} = |\langle \Psi_i | \Psi_j \rangle|^2. \quad (4.1)$$

Karena keadaan kuantum sebuah qubit secara matematis adalah vektor 2 dimensi dengan panjang 1, maka nilai maksimal dari fidelitas adalah satu, yaitu saat kedua keadaan kuantum sama persis, dan nilai minimalnya adalah nol, yaitu saat kedua keadaan kuantum saling ortogonal. Tiap piksel dengan koordinat (i,j) pada *heatmap* memiliki nilai $Fidelitas_{i,j}$ sebagai nilai intensitasnya.

Secara teori, sebelum proses latih, nilai fidelitas akan bernilai di sekitaran 1,0, karena sampel data apapun yang masuk ke DRC akan menghasilkan keadaan kuantum yang kurang lebih sama. Namun pada himpunan data ini, dapat terlihat sampel data dengan kelas yang berbeda sudah sedikit terpisah distribusinya pada visualisasi *Bloch sphere* sebelum proses latih. Sehingga *heatmap* yang dihasilkan tidak semua memiliki intensitas sekitar 1,0.

Kemudian setelah proses latih, secara teori nilai fidelitas akan bernilai sekitar 1,0 untuk dua keadaan kuantum yang memiliki kelas yang sama karena

proses latih mengoptimisasi parameter DRC untuk menghasilkan keadaan kuantum yang kurang lebih sama untuk sampel-sampel data dengan kelas yang sama. Sedangkan nilai fidelitas akan bernilai sekitar 0.0 untuk dua keadaan kuantum yang kelasnya berbeda karena proses latih membuat DRC menghasilkan keadaan kuantum yang jauh berbeda untuk sampel-sampel data dengan kelas berbeda.

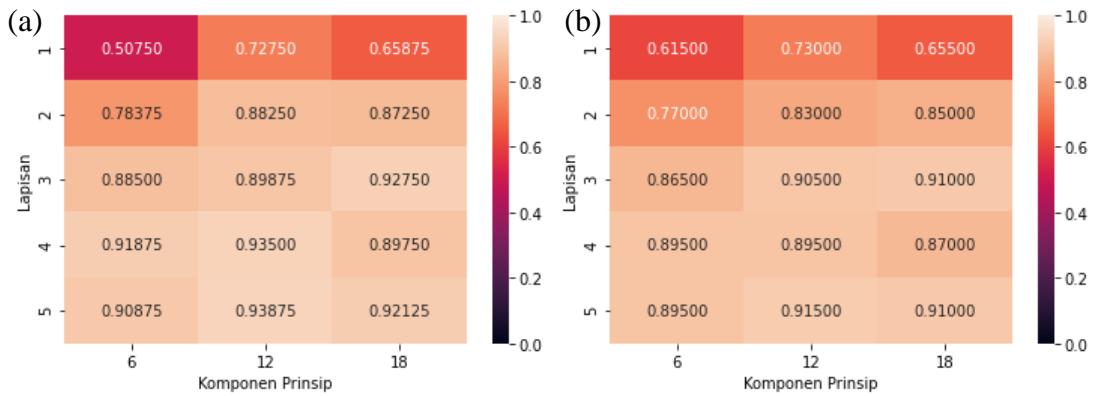
Terlihat visualisasi *heatmap* setelah proses latih terbagi ke dalam empat region, dua region berwarna terang (intensitas sekitar 1,0) dan dua region berwarna gelap (intensitas sekitar 0,0). Karena hanya ada 2 kelas, keseluruhan himpunan data terbagi dalam 2 kelas, setengah pertama kelas angka 0 dan setengah berikutnya kelas angka 1. Region terang terbentuk karena fidelitas dihitung menggunakan dua keadaan kuantum yang memiliki kelas yang sama, sedangkan region gelap terbentuk karena fidelitas dihitung menggunakan dua keadaan kuantum yang kelasnya berbeda.

Semakin baik pemisahan region terang dan gelap, dan semakin jauh perbedaan selisih intensitas rata-rata antar dua region ini, menunjukkan presisi klasifikasi yang semakin baik pula. Untuk klasifikasi 2 kelas, terlihat pemisahan region terang dan gelap sudah sangat baik, meskipun masih terdapat garis-garis terang tipis pada region gelap dan terdapat juga garis-garis gelap tipis pada region terang. Hal ini terjadi karena pengklasifikasi memang tidak mencapai presisi dengan ketepatan 100%, sama seperti pada visualisasi *Bloch sphere*.

Dari uji coba ini terbukti bahwa skema DRC jauh lebih baik dalam mengklasifikasikan himpunan data MNIST dibandingkan beberapa skema yang diajukan penelitian terkait sebelumnya (Subbab 1.3). DRC dengan mudah mencapai akurasi lebih dari 99% hanya dengan 1 qubit.

4.2.2 Klasifikasi 4 Kelas

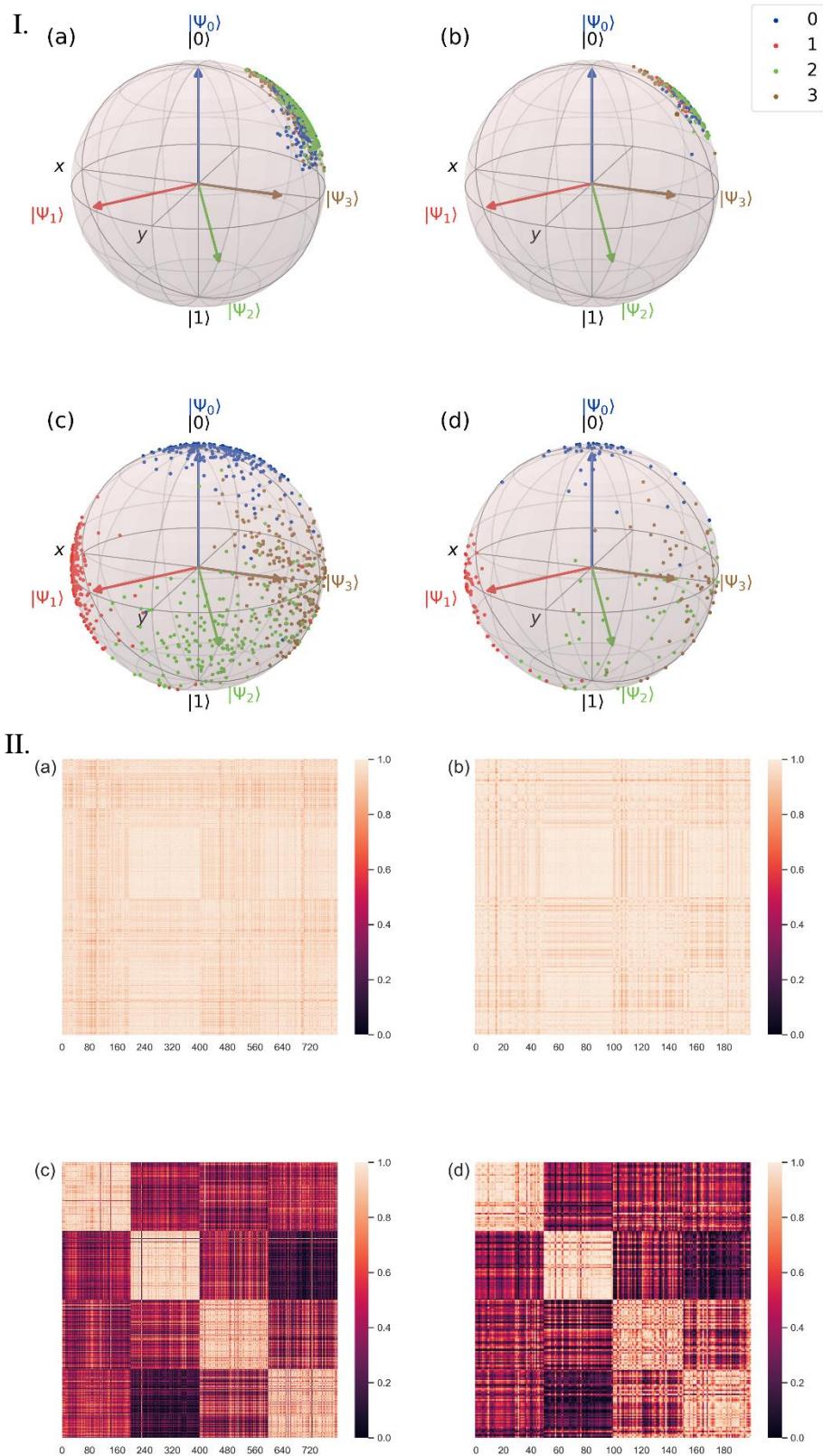
Kecenderungan yang terlihat pada klasifikasi 2 kelas tetap konsisten pada kasus klasifikasi 4 kelas. Gambar 4.8 menunjukkan akurasi klasifikasi terbaik yang didapatkan dari beberapa konfigurasi jumlah komponen prinsip dan jumlah lapisan DRC yang digunakan.



Gambar 4.8 Tabulasi capaian akurasi terbaik dari evaluasi (a) himpunan data latih dan (b) himpunan data uji dalam representasi *heatmap*.

Kembali terlihat bahwa penambahan jumlah lapisan lebih signifikan daripada penambahan jumlah komponen prinsip. Dengan hanya menggunakan 6 komponen prinsip, jika lapisan diulang sebanyak 5 kali, pengklasifikasi mampu mencapai akurasi 0,895 (89,5%) pada evaluasi data uji. Sedangkan meskipun jumlah komponen prinsip ditingkatkan sampai 18 komponen, akurasi yang dicapai hanya 0,655 (65,5%) untuk jumlah lapisan 1.

Dibandingkan dengan klasifikasi 2 kelas, secara umum klasifikasi 4 kelas mencapai akurasi yang lebih rendah. Klasifikasi 2 kelas mencapai akurasi lebih dari 99% dengan mudah, sedangkan klasifikasi 4 kelas maksimum hanya berhasil mencapai akurasi 91,5% meski dengan jumlah komponen prinsip dan jumlah lapisan yang lebih banyak. Hal ini disebabkan karena pada klasifikasi 4 kelas, *Bloch sphere* sebuah qubit yang awalnya hanya terbagi ke dalam dua region pada klasifikasi 2 kelas, sekarang harus terbagi menjadi 4 region untuk klasifikasi 4 kelas. Gambar 4.9 bagian I (c) dan (d) menunjukkan *Bloch sphere* qubit yang terbagi ke dalam 4 region setelah proses latih. Garis panah dengan warna yang berbeda-beda menunjukkan keadaan kuantum representasi tiap kelas. Luas permukaan yang dimiliki tiap region menjadi semakin sempit, sehingga probabilitas keadaan kuantum jatuh pada region yang salah menjadi semakin besar. Algoritma optimisasi parameter juga menjadi semakin sulit dalam menemukan himpunan parameter yang dapat memetakan semua sampel data ke dalam 4 region yang berbeda.



Gambar 4.9 Representasi I. *Bloch sphere* dan II. *heatmap* dari keadaan kuantum keluaran DRC untuk (a) himpunan data latih dan (b) himpunan data uji sebelum proses latih, dan (c) himpunan data latih dan (d) himpunan data uji setelah proses latih.

Visualisasi *heatmap* setelah proses latih juga menunjukkan pemisahan region gelap terang yang tidak sebaik pada klasifikasi 2 kelas. Banyak terdapat garis-garis terang pada region gelap dan begitu juga sebaliknya.

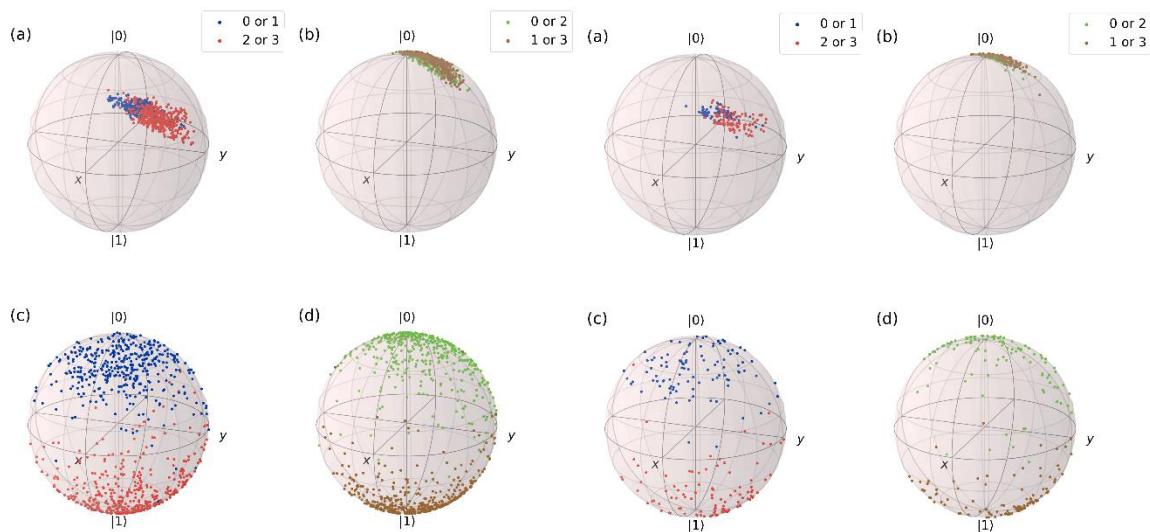
Meski skema ini mampu mencapai akurasi klasifikasi 4 kelas sebesar 91,5%, nilai ini merupakan penurunan yang cukup drastis dari hasil yang diperoleh pada klasifikasi 2 kelas. Penurunan akurasi ini akan semakin memburuk seiring dengan bertambahnya jumlah kelas. Oleh sebab itu penulis mengajukan skema DRC-RB, modifikasi yang berusaha mengatasi permasalahan ini.

4.3 Performa Pengklasifikasi DRC-RB dengan Reduksi PCA

4.3.1 Klasifikasi 4 Kelas

Pada DRC-RB, digunakan 2 qubit untuk klasifikasi 4 kelas, dimana masing-masing qubit berperan sebagai pengklasifikasi biner. Gambar 4.10 menunjukkan representasi *Bloch sphere* dari qubit pertama ((a) dan (c)) dan qubit kedua ((b) dan (d)) hasil evaluasi pengklasifikasi DRC-RB dengan 18 komponen prinsip dan 4 lapisan.

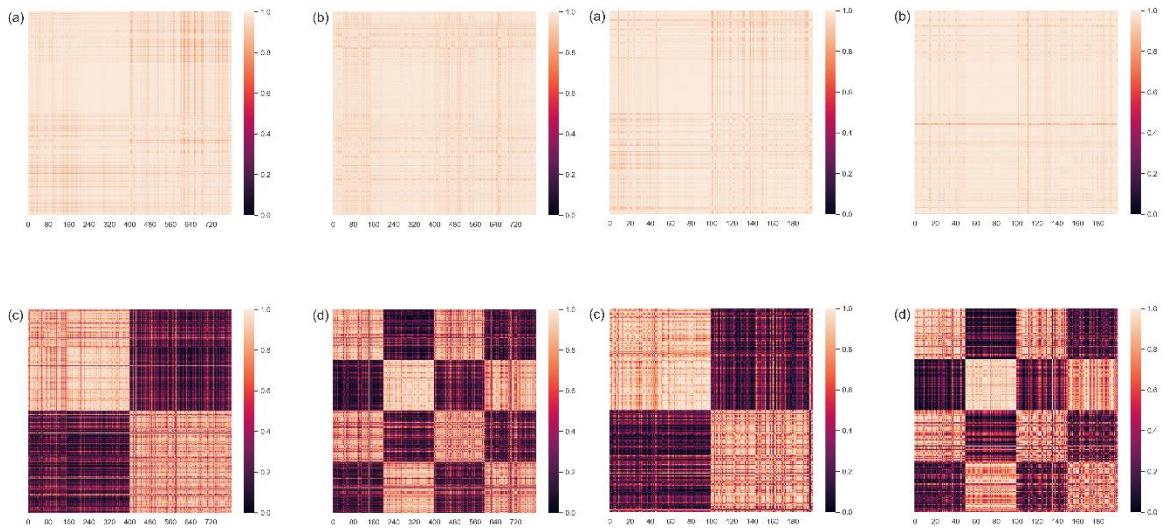
II.



Gambar 4.10 Representasi *Bloch sphere* keadaan kuantum keluaran DRC-RB dari I. evaluasi data latih dan II. evaluasi data uji. Baris atas dan bawah menunjukkan keadaan sebelum dan sesudah proses latih.

Keadaan kuantum setelah proses latih pada klasifikasi 2 kelas (Gambar 4.7 bagian I) terlihat terkonsentrasi pada keadaan kuantum $|0\rangle$ dan $|1\rangle$. Pada klasifikasi

4 kelas, Gambar 4.10 menunjukkan keluaran keadaan kuantum yang lebih menyebar meskipun setelah proses latih, tidak terkonsentrasi pada $|0\rangle$ dan $|1\rangle$. Meski begitu, akurasi klasifikasi setiap qubit masih sangat baik, mencapai lebih dari 95%. Hal ini karena *Bloch sphere* tiap qubit hanya terbagi ke dalam 2 region, setengah permukaan bola atas dan setengah permukaan bawah. Tidak masalah jika sampel data menyebar, selama sampel data tersebut berada pada region permukaan bola yang tepat, maka hasil klasifikasi akan tetap akurat.



Gambar 4.11 Representasi *heatmap* keadaan kuantum keluaran DRC-RB dari I. evaluasi data latih dan II. evaluasi data uji. Baris atas dan bawah menunjukkan keadaan sebelum dan sesudah proses latih.

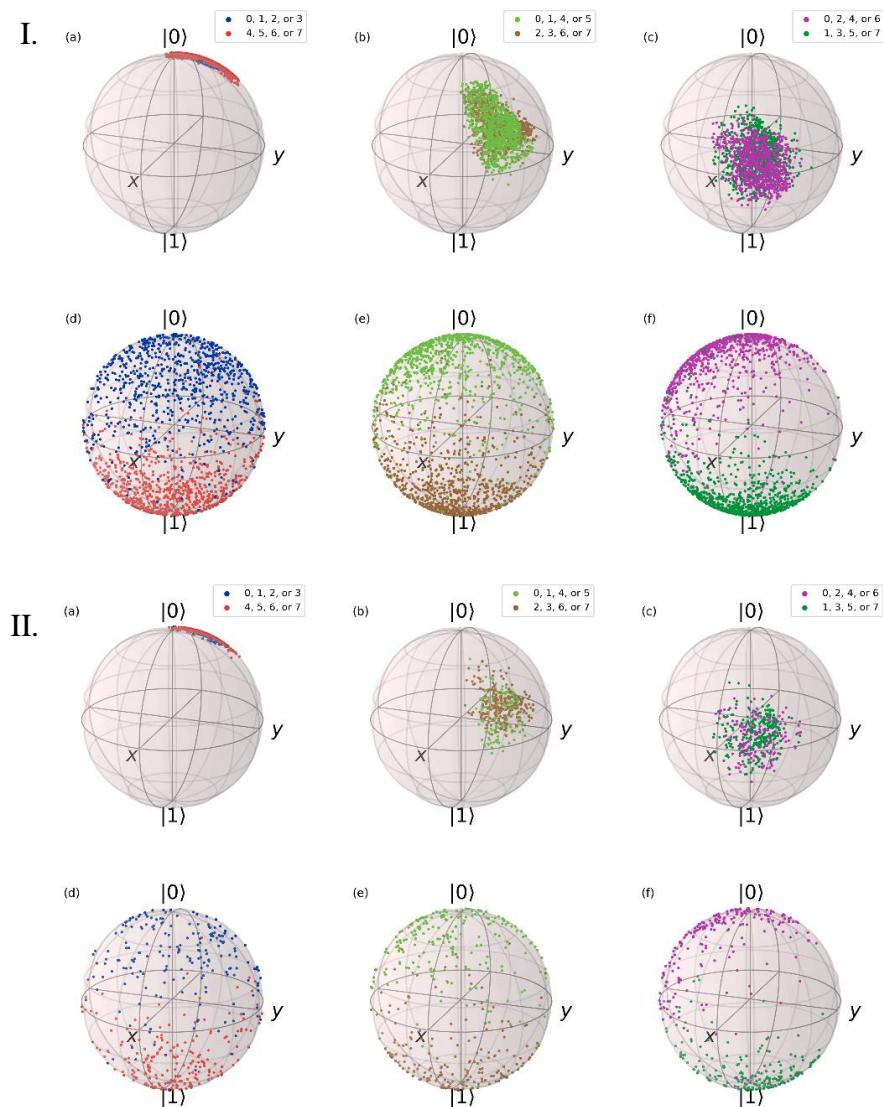
Gambar 4.11 menunjukkan representasi *heatmap*-nya. Jika dibandingkan dengan *heatmap* pada klasifikasi 2 kelas (Gambar 4.7 bagian II), pemisahan region terang gelapnya tidak sebaik pada klasifikasi 2 kelas, namun masih tetap lebih baik daripada klasifikasi 4 kelas yang tidak menggunakan skema DRC-RB (Gambar 4.9 bagian II). Pola gelap terang qubit pertama berbeda dengan qubit kedua karena qubit pertama menggolongkan kelas angka 0 dan 1 ke label 0 dan menggolongkan kelas angka 2 dan 3 ke label 1, sedangkan qubit kedua menggolongkan kelas angka 0 dan 2 ke label 0 dan menggolongkan kelas angka 1 dan 3 ke label 1, sebagaimana tertera pada Subbab 3.2.2.

Secara keseluruhan, skema DRC-RB ini berhasil mencapai akurasi terbaik sebesar 95,625% pada evaluasi data latih dan 96,5% pada evaluasi data uji, naik

masing-masing sebesar 1,75% dan 5% jika dibandingkan dengan performa terbaik pada skema DRC biasa. Hasil ini membuktikan akurasi klasifikasi dapat meningkat dengan mengurangi jumlah kelas yang harus diklasifikasi oleh setiap qubitnya.

4.3.2 Klasifikasi 8 Kelas

Untuk klasifikasi 8 kelas, skema DRC-RB membutuhkan 3 qubit. Representasi *Bloch sphere* dari qubit pertama ((a) dan (d)), qubit kedua ((b) dan (e)), dan qubit ketiga ((c) dan (f)) dengan 27 komponen prinsip dan 4 lapisan dapat terlihat pada Gambar 4.12.



Gambar 4.12 Representasi *Bloch sphere* keadaan kuantum keluaran DRC-RB dari I. evaluasi data latih dan II. evaluasi data uji. Baris atas dan bawah menunjukkan keadaan sebelum dan sesudah proses latih.

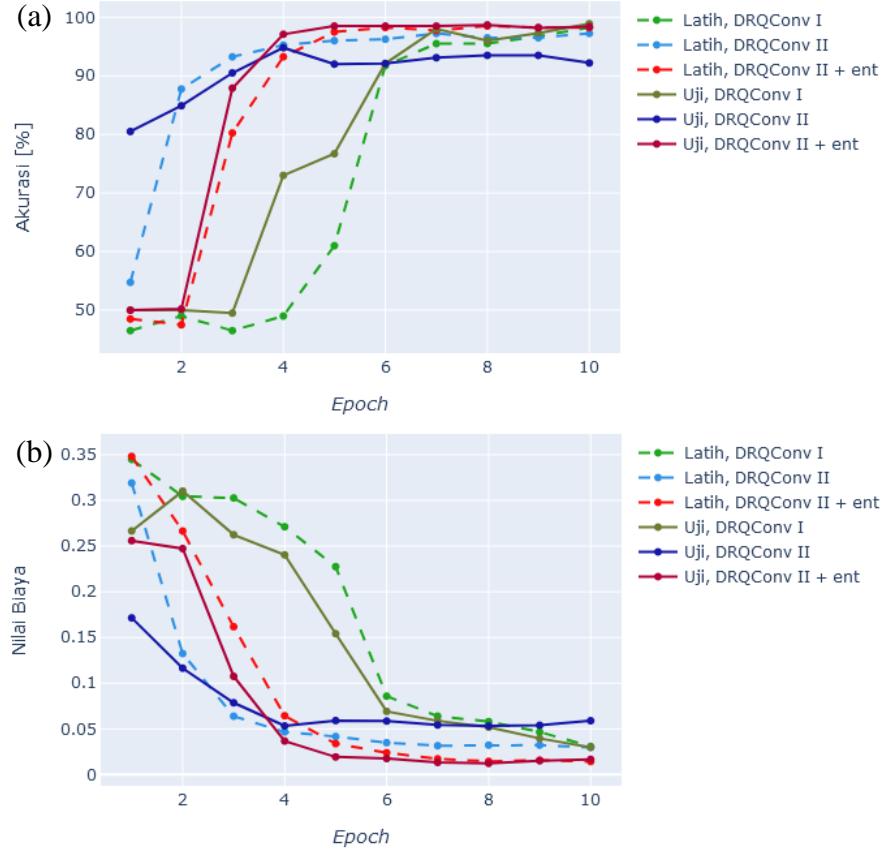
Seiring dengan bertambahnya kelas, terlihat pengklasifikasi semakin sulit memetakan keadaan kuantum secara terkonsentrasi pada luasan permukaan bola yang sempit. Persebaran keadaan kuantum dari sampel-sampel data semakin melebar jika dibandingkan dengan klasifikasi 4 kelas (Gambar 4.10). Jika klasifikasi 8 kelas dilakukan dengan skema DRC biasa, maka hasilnya tidak akan baik karena semua sampel data harus dapat dipetakan secara terkonsentrasi ke 8 region yang berbeda pada sebuah permukaan bola.

Dengan skema DRC-RB, tidak masalah jika keadaan kuantum tidak terkonsentrasi dengan baik sebagaimana telah dijelaskan dalam pembahasan pada Subbab 4.3.1. Pada klasifikasi 8 kelas, skema DRC-RB berhasil mencapai akurasi sebesar 87,75% pada evaluasi data latih, dan 86,25% pada evaluasi data uji.

4.4 Performa Pengklasifikasi DRC-RB dengan Reduksi Konvolusi Kuantum

Telah terlihat dari hasil dan pembahasan sebelumnya bahwa PCA masih cocok digunakan pada himpunan data MNIST. Meski begitu, PCA kemungkinan akan gagal dalam mereduksi dimensi citra-citra yang lebih kompleks dari MNIST sehingga penulis mengeksplorasi metode konvolusi dengan komputasi kuantum.

4.4.1 Klasifikasi 2 Kelas



Gambar 4.13 Grafik (a) nilai akurasi dan (b) nilai biaya pengklasifikasi selama proses latih pada tiap epoch.

Gambar 4.13 menunjukkan tren nilai akurasi dan nilai biaya selama proses latih untuk tiap *ansatz* sirkuit yang digunakan untuk konvolusi. Ketiganya menggunakan pengklasifikasi DRC-RB 1 lapis (untuk kasus klasifikasi 2 kelas, skema DRC-RB dan DRC yang biasa menjadi sama).

Terlihat sirkuit DRQConv II (garis biru) cenderung cepat naik akurasinya (dan turun nilai biayanya) namun cepat juga mencapai tunak dan akurasi berhenti naik sejak *epoch* kelima. Sedangkan sirkuit DRQConv I (garis hijau) adalah yang paling lambat proses pertambahan akurasinya, namun masih terus naik setelah *epoch* kelima dan melebihi akurasi latih yang menggunakan sirkuit DRQConv II setelah *epoch* keenam. Sirkuit DRQConv II + ent (garis merah) mencapai nilai akurasi (paling tinggi) dan nilai biaya (paling rendah) yang terbaik di antara

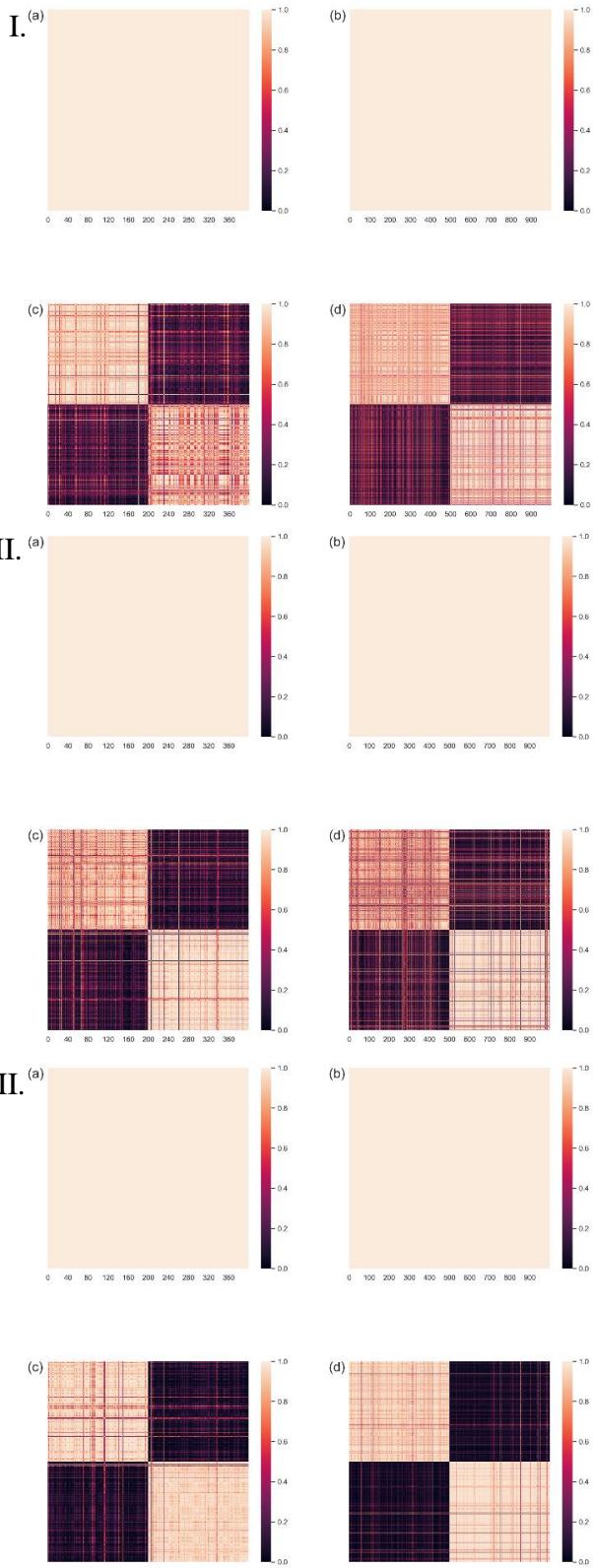
ketiganya. Sirkuit ini sedikit lebih lambat pertambahan akurasinya daripada sirkuit DRQConv II namun sirkuit ini mampu mencapai nilai akurasi yang lebih tinggi.

Tabel 4.1 Tabulasi nilai akurasi terbaik pada evaluasi data latih dan data uji untuk setiap *ansatz* sirkuit. Disertakan juga hasil dari JST Konvolusi sebagai referensi.

Ansatz Sirkut Konvolusi Kuantum	Akurasi Latih	Akurasi Uji
DRQConv I	98,250%	98,90%
DRQConv II	97,250%	94,80%
DRQConv II + ent	98,50%	98,70%
JST Konvolusi 2 kelas	99,750%	99,70%

Tabel 4.1 menunjukkan nilai akurasi terbaik untuk tiap *ansatz* sirkuit. Hasil akurasi JST Konvolusi juga diberikan sebagai nilai referensi. JST Konvolusi yang digunakan memiliki 56 parameter, jumlah yang sama dengan metode lainnya agar perbandingan dapat dibuat seadil mungkin.

Terlihat DRQConv II menghasilkan nilai akurasi yang lebih rendah dibandingkan sirkuit lain. DRQConv I dan DRQConv II + ent menghasilkan akurasi yang kurang lebih sama. Metode klasik masih lebih unggul kurang lebih sekitar 1%.

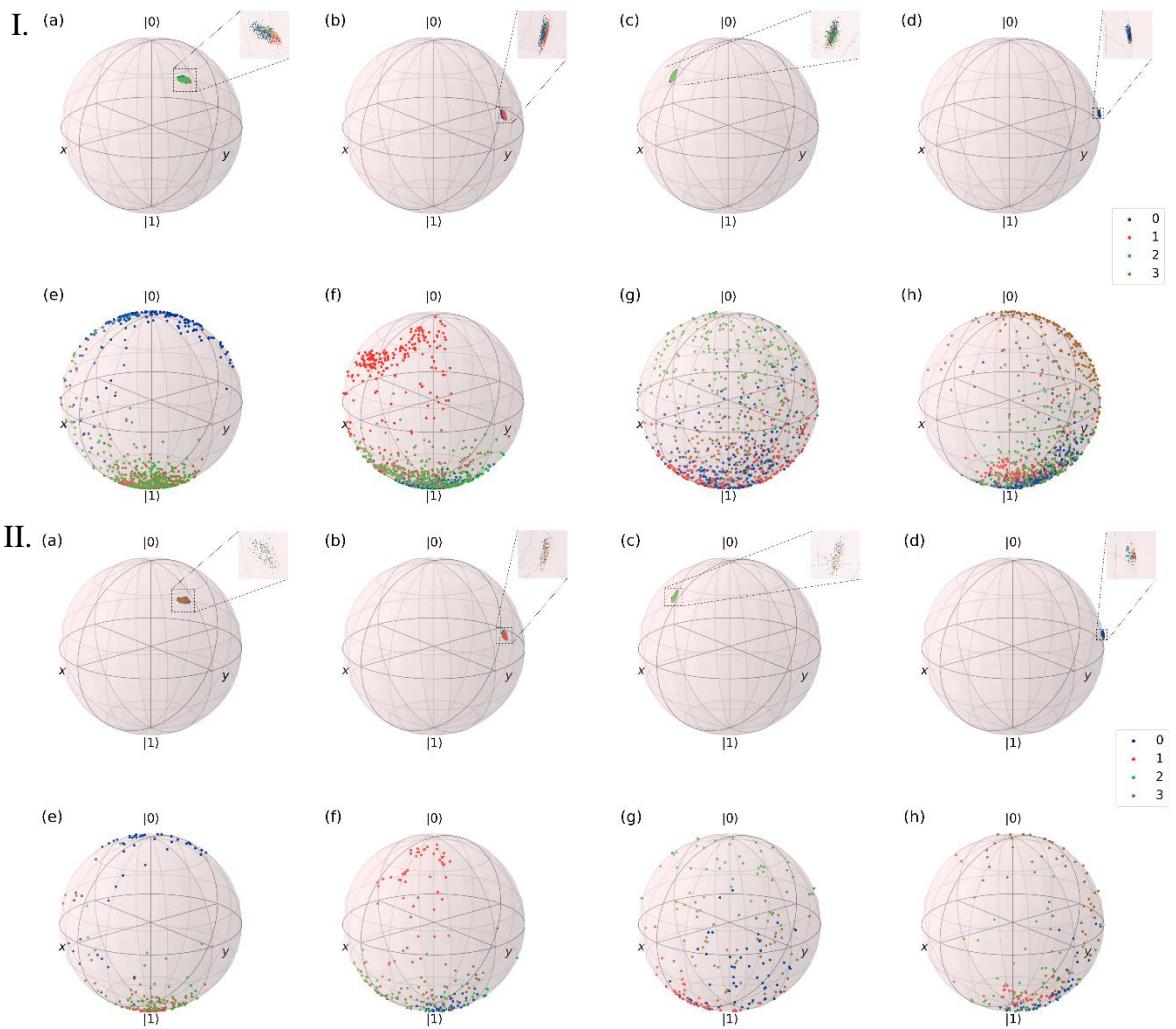


Gambar 4.14 Representasi *heatmap* dari keadaan kuantum keluaran DRC-RB dengan DRQConv I (I), DRQConv II (II), dan DRQConv II + ent (III) untuk himpunan data (a) latih dan (b) uji sebelum proses latih, dan himpunan data (c) latih dan (d) uji setelah proses latih.

Gambar 4.14 menunjukkan adanya *trade-off* antara akurasi dan presisi. Sirkuit DRQConv II menghasilkan separasi kelas (tingkat presisi) yang lebih baik daripada DRQConv I meskipun akurasinya lebih rendah. Sirkuit DRQConv II dengan *entanglement* secara konsisten menghasilkan akurasi dan presisi yang paling baik.

4.4.2 Klasifikasi 4 Kelas

Sedikit modifikasi dilakukan pada DRC-RB untuk klasifikasi 4 kelas sebagaimana dijelaskan pada Subbab 3.3.3. Setiap qubit berperan sebagai pengklasifikasi biner yang menentukan apakah suatu sampel data termasuk ke dalam kelas angka tertentu atau tidak. Gambar 4.15 menunjukkan representasi *Bloch sphere* keadaan kuantum keluaran DRC-RB untuk setiap qubit. *Ansatz* sirkuit untuk konvolusi yang digunakan pada Gambar 4.15 adalah DRQConv I. Representasi untuk *ansatz* sirkuit lainnya tidak jauh berbeda distribusinya sehingga tidak ditampilkan namun dapat dilihat dalam Lampiran B.



Gambar 4.15 Representasi *Bloch sphere* evaluasi data I. latih dan II. uji sebelum proses latih (baris atas) dan setelah proses latih (baris bawah).

Gambar 4.15 menunjukkan perbedaan keadaan kuantum keluaran DRC-RB antara sebelum dan sesudah proses latih untuk qubit pertama ((a) dan (e)), qubit kedua ((b) dan (f)), qubit ketiga ((c) dan (g)), dan qubit keempat ((d) dan (h)).

Setelah proses latih, terlihat hanya 1 warna saja yang dominan berada di region setengah atas permukaan bola. Ini menunjukkan sirkuit kuantum yang telah dioptimasi berhasil memisahkan sampel data ke dalam dua region *Bloch sphere*. Sebagai contoh, sampel berwarna biru merupakan sampel data berlabel angka 0. Qubit pertama adalah qubit yang bertugas untuk memetakan sampel data berlabel angka 0 ke keadaan kuantum $|0\rangle$ dan sampel data lainnya ke keadaan kuantum $|1\rangle$. Oleh sebab itu warna biru dominan berada di bagian setengah atas permukaan bola

pada qubit pertama setelah proses latih, begitu juga pada qubit lainnya untuk warna-warna lainnya.

Kemudian terlihat juga distribusi warna hijau dan cokelat pada qubit ketiga dan keempat setelah proses latih lebih menyebar daripada warna biru dan merah pada qubit pertama dan kedua. Ini menunjukkan pengklasifikasi lebih sulit mengklasifikasikan angka 2 dan 3 daripada angka 0 dan 1. Hal ini menurut penulis adalah hal yang wajar mengingat angka 2 dan 3 bentuknya relatif mirip dibandingkan dengan angka 0 dan 1 yang sangat berbeda dari angka lainnya. Perihal ini akan menjadi lebih jelas pada Subbab 4.5 dimana akan diperlihatkan citra keluaran konvolusi kuantum.

Tabel 4.2 Tabulasi nilai akurasi terbaik pada evaluasi data latih dan data uji untuk setiap *ansatz* sirkuit. Disertakan juga hasil dari JST Konvolusi sebagai referensi.

Ansatz Sirkut Konvolusi Kuantum	Akurasi Latih	Akurasi Uji
DRQConv I	89,0%	89,50%
DRQConv II	85,1250%	85,0%
DRQConv II + ent	90,0%	86,50%
JST Konvolusi 4 kelas	87,8750%	88,50%

Konsisten dengan hasil yang diperoleh pada klasifikasi 2 kelas, akurasi yang menggunakan sirkuit DRQConv II untuk konvolusi adalah yang paling rendah di antara yang lainnya. Kemudian sirkuit DRQConv I dan DRQConv II + ent mampu mencapai akurasi yang relatif seimbang dengan JST Konvolusi 4 kelas, dengan sirkuit DRQConv I selalu konsisten mencapai akurasi yang lebih tinggi dari JST Konvolusi 4 kelas baik pada evaluasi data latih maupun uji, meskipun hanya sekitar 1%. Perlu diingat bahwa JST Konvolusi 4 kelas yang digunakan disini memiliki parameter yang sedikit lebih banyak dari yang lainnya, yaitu 230 parameter, sedangkan yang lain hanya memiliki 224 parameter. Penulis tidak dapat menyamakan jumlah parameter dengan persis karena algoritma yang berbeda, ini adalah jumlah terdekat yang dapat diusahakan penulis.

Dari hasil yang diperoleh pada klasifikasi 2 kelas dan 4 kelas, terlihat bahwa arsitektur sirkuit DRQConv I dan DRQConv II + ent menghasilkan hasil yang terbaik. Hasil ini menunjukkan bahwa penggunaan jumlah qubit yang banyak tidak serta merta akan selalu meningkatkan akurasi, desain arsitektur sirkuit dan bagaimana data disematkan juga berpengaruh pada akurasi akhir. DRQConv II yang menggunakan 9 qubit tidak lebih baik daripada DRQConv I yang hanya menggunakan 3 qubit.

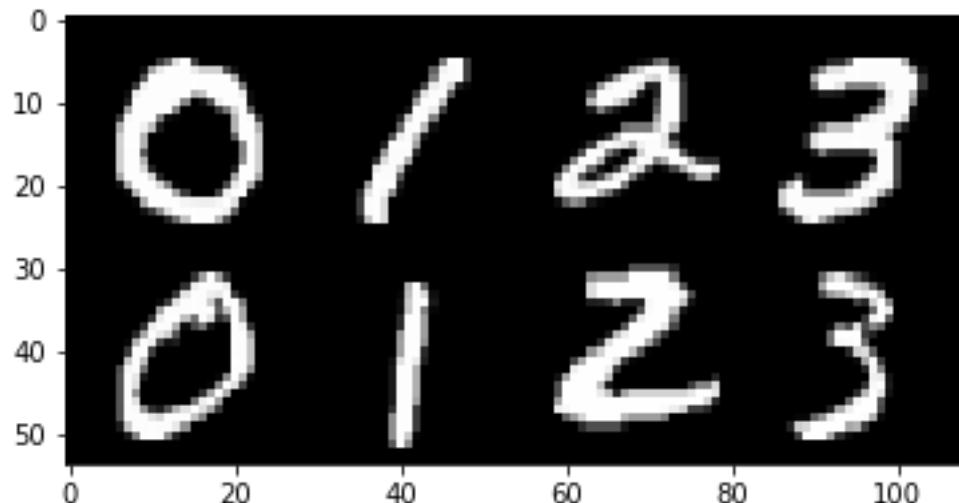
Kemudian, terdapat indikasi penggunaan gerbang CNOT yang meng-*entangle* dua qubit berperan positif dalam meningkatkan akurasi dan presisi. Perbedaan antara DRQConv II dan DRQConv II + ent hanya terletak pada penggunaan gerbang CNOT pada tiap lapisannya saja dan terlihat DRQConv II + ent secara konsisten selalu lebih tinggi akurasinya dan lebih baik pemisahan antar kelasnya (presisi) daripada DRQConv II. Penulis berhipotesis ini terjadi karena gerbang CNOT membuat sirkuit menjadi mampu mengakses keadaan-keadaan kuantum yang lebih kompleks sehingga sirkuit memiliki lebih banyak opsi dalam merepresentasikan sampel data ke dalam keadaan kuantum. Diperlukan penelitian yang lebih mendalam dan berfokus pada hal ini kedepannya untuk dapat menjawab pertanyaan ini.

Terakhir, sistem yang murni hanya menggunakan sirkuit kuantum mulai dari konvolusi (DRQConv) hingga klasifikasi (DRC-RB) mampu menyamai hasil akurasi klasifikasi JST Konvolusi klasik, bahkan dengan jumlah parameter yang sedikit lebih rendah. Meski hasil ini tidak serta merta memberikan kesimpulan bahwa sistem klasifikasi dengan komputasi kuantum lebih baik, namun hasil ini dapat memberikan *proof of concept* dan bukti numerik bahwa sistem klasifikasi dengan komputasi kuantum setidaknya dapat menyamai performa sistem klasik. Dibutuhkan penelitian yang lebih mendalam terkait hal ini.

4.5 Citra Hasil Konvolusi Kuantum

Setiap proses konvolusi selesai dilakukan pada sebuah sampel citra, keluaran dari proses ini adalah sebuah sampel citra baru yang memiliki ukuran yang lebih kecil. Gambar 4.16 menunjukkan beberapa sampel data citra MNIST dari

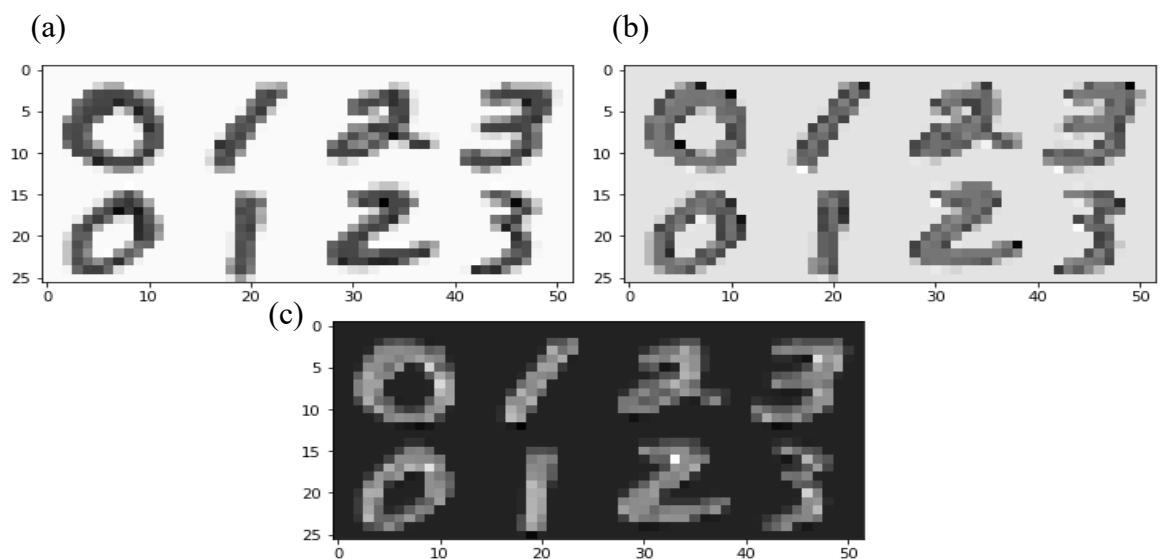
himpunan data latih. Subbab ini akan memperlihatkan hasil konvolusi kuantum pada kasus klasifikasi 4 kelas dari sampel data pada Gambar 4.16 setelah proses latih.



Gambar 4.16 Sampel data MNIST berukuran 27 x 27.

4.5.1 Konvolusi Pertama

Konvolusi pertama mereduksi citra dari 27 x 27 menjadi 13 x 13. Gambar 4.17 menunjukkan citra keluaran konvolusi pertama dari tiap *ansatz* sirkuit yang digunakan.

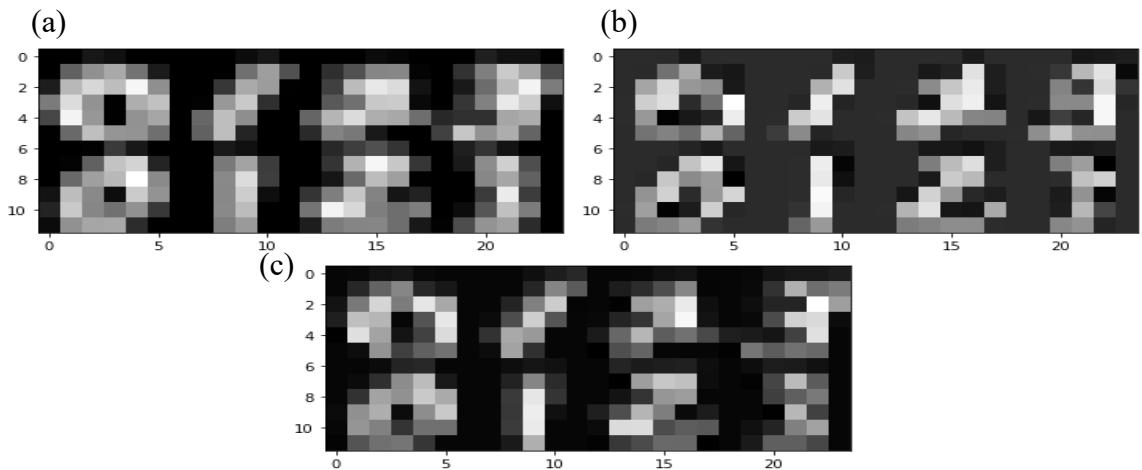


Gambar 4.17 Citra keluaran konvolusi kuantum pertama dari sirkuit (a) DRQConv I, (b) DRQConv II, dan (c) DRQConv II + ent.

Hal menarik yang dapat terlihat adalah karakteristik dari citra yang berbeda-beda untuk masing-masing sirkuit. DRQConv I seperti membalik nilai intensitas dari piksel pada citra, hitam menjadi putih dan putih menjadi hitam. DRQConv II mirip dengan DRQConv I namun dengan intensitas yang lebih keabuan baik pada latar belakang yang putih maupun tulisan angka yang hitam. DRQConv II + ent mempertahankan latar belakang berwarna hitam dan menurunkan intensitas tulisan angka menjadi keabuan.

4.5.2 Konvolusi Kedua

Konvolusi kedua mereduksi citra dari 13×13 menjadi 6×6 . Gambar 4.18 menunjukkan citra keluaran konvolusi kedua dari tiap *ansatz* sirkuit yang digunakan.



Gambar 4.18 Citra keluaran konvolusi kuantum kedua dari sirkuit (a) DRQConv I, (b) DRQConv II, dan (c) DRQConv II + ent.

Setelah menghasilkan keluaran dengan karakteristik yang berbeda-beda pada konvolusi pertama, ketiga sirkuit malah menghasilkan keluaran yang serupa pada konvolusi kedua. Ini menandakan representasi citra seperti pada Gambar 4.18 merupakan representasi yang cukup optimal bagi pengklasifikasi DRC-RB karena 3 sirkuit konvolusi yang berbeda menghasilkan keluaran yang serupa setelah proses latih. Penulis berhipotesis bahwa hal ini juga menunjukkan tiap sirkuit mampu dilatih menuju titik optimal, namun tiap sirkuit kemungkinan terjebak pada lokal

minima yang berbeda-beda tergantung karakteristik arsitektur sirkuitnya sehingga menghasilkan nilai akurasi yang berbeda-beda.

Terlihat juga angka 0 dan angka 1 memiliki karakteristik yang secara konsisten selalu muncul pada keluaran konvolusi kedua terlepas dari *ansatz* sirkuit yang digunakan. Pada angka 0, beberapa piksel yang terletak di tengah-tengah citra selalu memiliki intensitas yang bernilai nol. Dan pada angka 1, citra terbagi ke dalam 3 region, region tengah dominan piksel berintensitas lebih dari nol, region kiri dan kanan dominan piksel berintensitas nol. Sedangkan untuk angka 2 dan 3, sulit menemukan pola atau karakteristik yang selalu muncul secara konsisten pada semua sampel. Tidak terdapatnya pola yang konsisten menyebabkan parameter sirkuit menjadi semakin sulit dioptimisasi terhadap sampel data tersebut. Hal ini menjelaskan kenapa pada Gambar 4.15 sampel warna hijau (angka 2) dan cokelat (angka 3) pada qubit ketiga dan keempat lebih menyebar daripada sampel biru (angka 0) dan merah (angka 1) pada qubit pertama dan kedua.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

1. Telah dilakukan implementasi algoritma kuantum DRC untuk klasifikasi 2 dan 4 kelas himpunan data MNIST dengan hanya menggunakan 1 qubit. Akurasi terbaik dicapai sebesar 99,750% pada evaluasi data latih dan 99,70% pada evaluasi data uji untuk klasifikasi 2 kelas dengan menggunakan 6 komponen prinsip dan 3 lapisan. Hasil ini jauh lebih baik daripada hasil-hasil klasifikasi 2 kelas pada pekerjaan terkait sebelumnya, baik dari segi akurasi yang tercapai dan dari segi jumlah qubit yang diperlukan (lebih sedikit qubit diperlukan). Algoritma yang sama juga dapat diekspansi untuk klasifikasi 4 kelas, mencapai akurasi terbaik sebesar 93,875% pada evaluasi data latih dan 91,50% pada evaluasi data uji dengan menggunakan 12 komponen prinsip dan 5 lapisan.
2. Telah diajukan skema arsitektur DRC-RB, arsitektur modifikasi dari algoritma kuantum DRC untuk klasifikasi kelas lebih dari 2. Hasil implementasi dengan 2 qubit menunjukkan peningkatan akurasi untuk klasifikasi 4 kelas dengan 18 komponen prinsip dan 4 lapisan sebesar 1,75% menjadi 95,625% pada evaluasi data latih dan 5% menjadi 96,5% pada evaluasi data uji jika dibandingkan dengan hasil dari algoritma DRC biasa. Algoritma DRC-RB juga telah diimplementasikan untuk klasifikasi 8 kelas dengan 3 qubit menggunakan 27 komponen prinsip dan 4 lapisan, mencapai akurasi terbaik sebesar 87,75% pada evaluasi data latih dan 86,25% pada evaluasi data uji.
3. Telah diajukan skema arsitektur DRQConv, yaitu proses konvolusi citra dengan sirkuit kuantum yang serupa dengan konvolusi klasik. Skema ini telah berhasil mereduksi dimensi citra. 3 *ansatz* sirkuit yang berbeda telah diimplementasikan. Dengan reduksi dimensi DRQConv (tanpa PCA) sebanyak 2 kali konvolusi dengan masing-masing 1 lapisan, pengklasifikasi DRC-RB dengan 1 lapisan mampu mencapai akurasi terbaik di atas 98% dengan hanya menggunakan 3 atau 9 qubit untuk DRQConv (tergantung

ansatz yang digunakan) dan 1 qubit untuk DRC-RB. Hasil ini jauh lebih baik daripada hasil-hasil klasifikasi 2 kelas pada pekerjaan terkait sebelumnya dari segi akurasi. Skema ini telah diimplementasikan juga untuk klasifikasi 4 kelas. Dari hasil klasifikasi 2 kelas dan 4 kelas, *ansatz* sirkuit terbaik untuk DRQConv adalah DRQConv I karena menghasilkan akurasi yang relatif tinggi dengan penggunaan qubit yang paling sedikit (3 qubit). Reduksi dimensi citra dengan sirkuit DRQConv I dan pengklasifikasi DRC-RB menghasilkan akurasi terbaik sebesar 98,250% pada evaluasi data latih dan 98,90% pada evaluasi data uji untuk klasifikasi 2 kelas dengan 2 kali konvolusi DRQConv I masing-masing 1 lapisan dan 1 lapisan DRC-RB, kemudian menghasilkan 89% pada evaluasi data latih dan 89,5% pada evaluasi data uji untuk klasifikasi 4 kelas dengan 2 kali konvolusi DRQConv I masing-masing 2 lapisan dan 2 lapisan DRC-RB.

5.2 Saran

1. Melakukan pengujian numerik klasifikasi 2 kelas dengan pengklasifikasi DRC-RB dan dengan reduksi dimensi PCA dan DRQConv untuk himpunan data yang lebih kompleks dari MNIST untuk melihat bagaimana performa algoritma pada citra yang lebih kompleks.
2. Melakukan pengujian numerik dan analisis matematika yang berfokus pada variasi jumlah qubit yang digunakan untuk melihat bagaimana penambahan jumlah qubit (yang berimplikasi pada penambahan jumlah dimensi dalam ruang Hilbert) berdampak pada akurasi klasifikasi.
3. Melakukan pengujian numerik dan analisis matematika yang berfokus pada variasi jumlah gerbang 2-qubit yang digunakan untuk melihat dampak dari penggunaan gerbang 2-qubit (yang berimplikasi pada peningkatan kompleksitas keadaan kuantum yang dapat diakses oleh sirkuit) pada akurasi klasifikasi.
4. Merancang algoritma atau desain skema yang dapat mengukur seberapa kuat suatu algoritma/sirkuit pengklasifikasi kuantum dalam melakukan klasifikasi jika dibandingkan dengan pengklasifikasi klasik.

DAFTAR PUSTAKA

- [1] Moore GM. Cramming more components onto integrated circuits. *Electronics* 1965;38:114.
- [2] Waldrop MM. More Than Moore. *Nature* 2016;530:144–7. <https://doi.org/10.1038/530144a>.
- [3] Lee CY. Transistor Degradations in Very Large-Scale-Integrated CMOS Technologies. *Very-Large-Scale Integr* 2018. <https://doi.org/10.5772/intechopen.68825>.
- [4] Grumblng E, Horowitz M. Quantum Computing: Progress and Prospects. vol. 9781461418. Washington, D.C.: National Academies Press; 2019. <https://doi.org/10.17226/25196>.
- [5] Feynman RP. Simulating physics with computers. *Int J Theor Phys* 1982;21:467–88. <https://doi.org/10.1007/BF02650179>.
- [6] Nielsen MA, Chuang IL. Quantum Computation and Quantum Information. 2010.
- [7] Shor PW. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev* 1999;41:303–32. <https://doi.org/10.1137/S0036144598347011>.
- [8] Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N, Lloyd S. Quantum machine learning. *Nature* 2017;549:195–202. <https://doi.org/10.1038/nature23474>.
- [9] Harrow AW, Hassidim A, Lloyd S. Quantum algorithm for linear systems of equations. *Phys Rev Lett* 2009;103:150502. <https://doi.org/10.1103/PhysRevLett.103.150502>.
- [10] Lloyd S, Mohseni M, Rebentrost P. Quantum principal component analysis. *Nat Phys* 2014;10:631–3. <https://doi.org/10.1038/NPHYS3029>.
- [11] Preskill J. Quantum Computing in the NISQ era and beyond. *Quantum* 2018;2. <https://doi.org/10.22331/q-2018-08-06-79>.

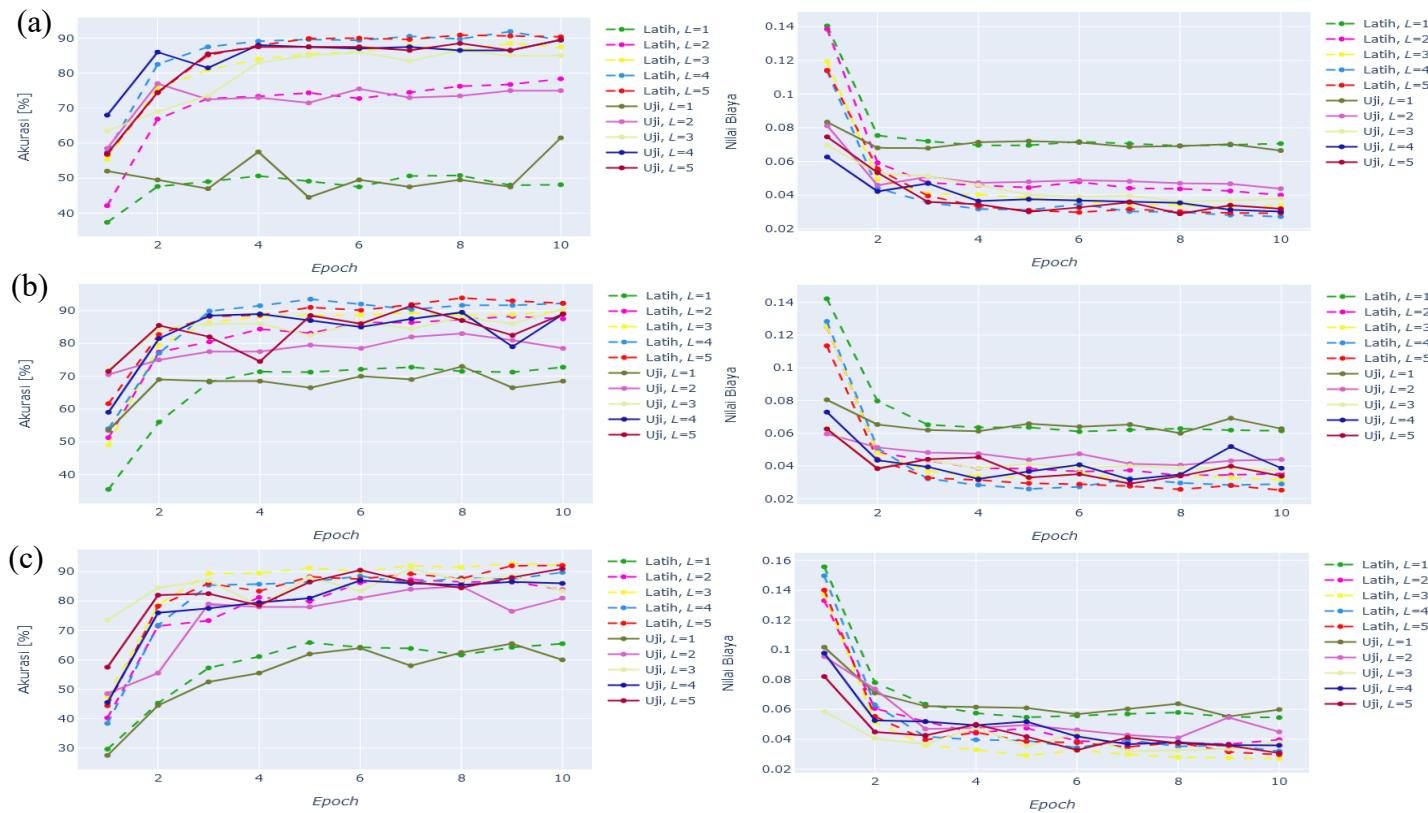
- [12] Cerezo M, Arrasmith A, Babbush R, Benjamin SC, Endo S, Fujii K, et al. Variational Quantum Algorithms 2020.
- [13] Bravo-Prieto C, LaRose R, Cerezo M, Subasi Y, Cincio L, Coles PJ. Variational Quantum Linear Solver 2019.
- [14] Schuld M, Petruccione F. Supervised Learning with Quantum Computers. 2018. <https://doi.org/10.1007/978-3-319-96424-9>.
- [15] Farhi E, Neven H. Classification with Quantum Neural Networks on Near Term Processors 2018:1–21.
- [16] MNIST classification | TensorFlow Quantum n.d. <https://www.tensorflow.org/quantum/tutorials/mnist> (diakses 6 Februari 2021).
- [17] Skolik A, McClean JR, Mohseni M, van der Smagt P, Leib M. Layerwise learning for quantum neural networks. *Quantum Mach Intell* 2021;3:5. <https://doi.org/10.1007/s42484-020-00036-4>.
- [18] Mardirosian S. Quantum-enhanced Supervised Learning with Variational Quantum Circuits. Leiden University, 2019.
- [19] Pérez-Salinas A, Cervera-Lierta A, Gil-Fuster E, Latorre JI. Data re-uploading for a universal quantum classifier. *Quantum* 2020;4:226. <https://doi.org/10.22331/q-2020-02-06-226>.
- [20] Bergholm V, Izaac J, Schuld M, Gogolin C, Alam MS, Ahmed S, et al. PennyLane: Automatic differentiation of hybrid quantum-classical computations 2018:1–15.
- [21] LeCun Y, Cortes C. MNIST handwritten digit database 2010.
- [22] Chollet F, others. Keras 2015.
- [23] Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. n.d.
- [24] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et

- al. Scikit-learn: Machine Learning in Python. *J Mach Learn Res* 2011;12:2825–30.
- [25] Bloch sphere - Wikipedia n.d. https://en.wikipedia.org/wiki/Bloch_sphere#cite_note-1 (diakses 31 Januari 2021).
- [26] Johansson JR, Nation PD, Nori F. QuTiP: An open-source Python framework for the dynamics of open quantum systems. *Comput Phys Commun* 2012;183:1760–72. <https://doi.org/10.1016/j.cpc.2012.02.021>.
- [27] Quantum Logic Gates - Quantum logic gate - Wikipedia n.d. https://en.wikipedia.org/wiki/Quantum_logic_gate#/media/File:Quantum_L_Logic_Gates.png (diakses 30 Januari 2021).
- [28] Weights and Bias in a Neural Network | Towards Data Science n.d. <https://towardsdatascience.com/whats-the-role-of-weights-and-bias-in-a-neural-network-4cf7e9888a0f> (diakses 31 Januari 2021).
- [29] Parmar R. Training Deep Neural Networks. Deep Learn Accessories, Toward Data Sci n.d. <https://towardsdatascience.com/training-deep-neural-networks-9fdb1964b964> (diakses 1 Februari 2021).
- [30] Jolliffe IT, Cadima J. Principal component analysis: a review and recent developments. *Philos Trans R Soc A Math Phys Eng Sci* 2016;374:20150202. <https://doi.org/10.1098/rsta.2015.0202>.
- [31] Sumit S. A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way n.d. <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53> (diakses 16 Maret 2020).
- [32] Erdas A, Arslan E, Ozturkcan B, Yildiran U. A study on object classification using deep convolutional neural networks and comparison with shallow networks. 2018 6th Int. Conf. Control Eng. Inf. Technol. CEIT 2018, Institute of Electrical and Electronics Engineers Inc.; 2018. <https://doi.org/10.1109/CEIT.2018.8751754>.

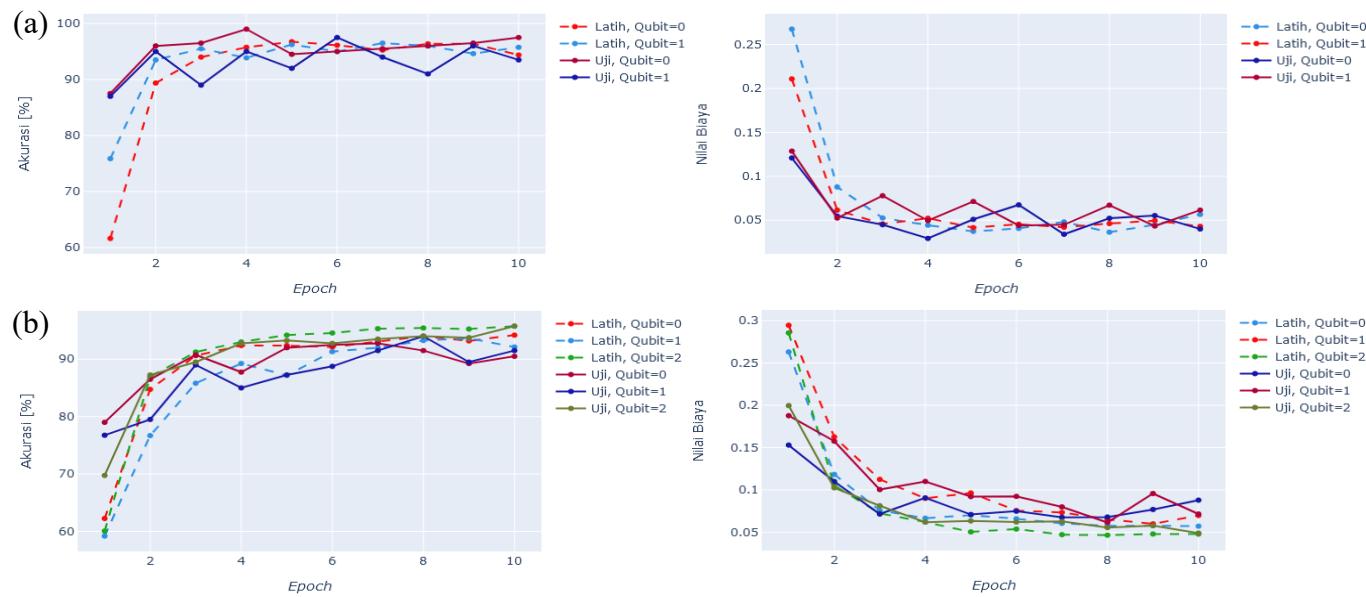
- [33] Mottonen M, Vartiainen JJ, Bergholm V, Salomaa MM. Transformation of quantum states using uniformly controlled rotations. *Quantum Inf Comput* 2004;5:467–73.
- [34] Lloyd S, Schuld M, Ijaz A, Izaac J, Killoran N. Quantum embeddings for machine learning 2020.
- [35] Nghiêm NA, Chen SY-C, Wei T-C. A Unified Classification Framework with Quantum Metric Learning 2020.
- [36] Kingma DP, Ba J. Adam: A Method for Stochastic Optimization. In: Bengio Y, LeCun Y, editor. 3rd Int. Conf. Learn. Represent. {ICLR} 2015, San Diego, CA, USA, May 7-9, 2015, Conf. Track Proc., 2015.
- [37] Jordan J. Setting the learning rate of your neural network. n.d. <https://www.jeremyjordan.me/nn-learning-rate/> (diakses 1 Februari 2021).
- [38] Crooks GE. Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition 2019.

LAMPIRAN A

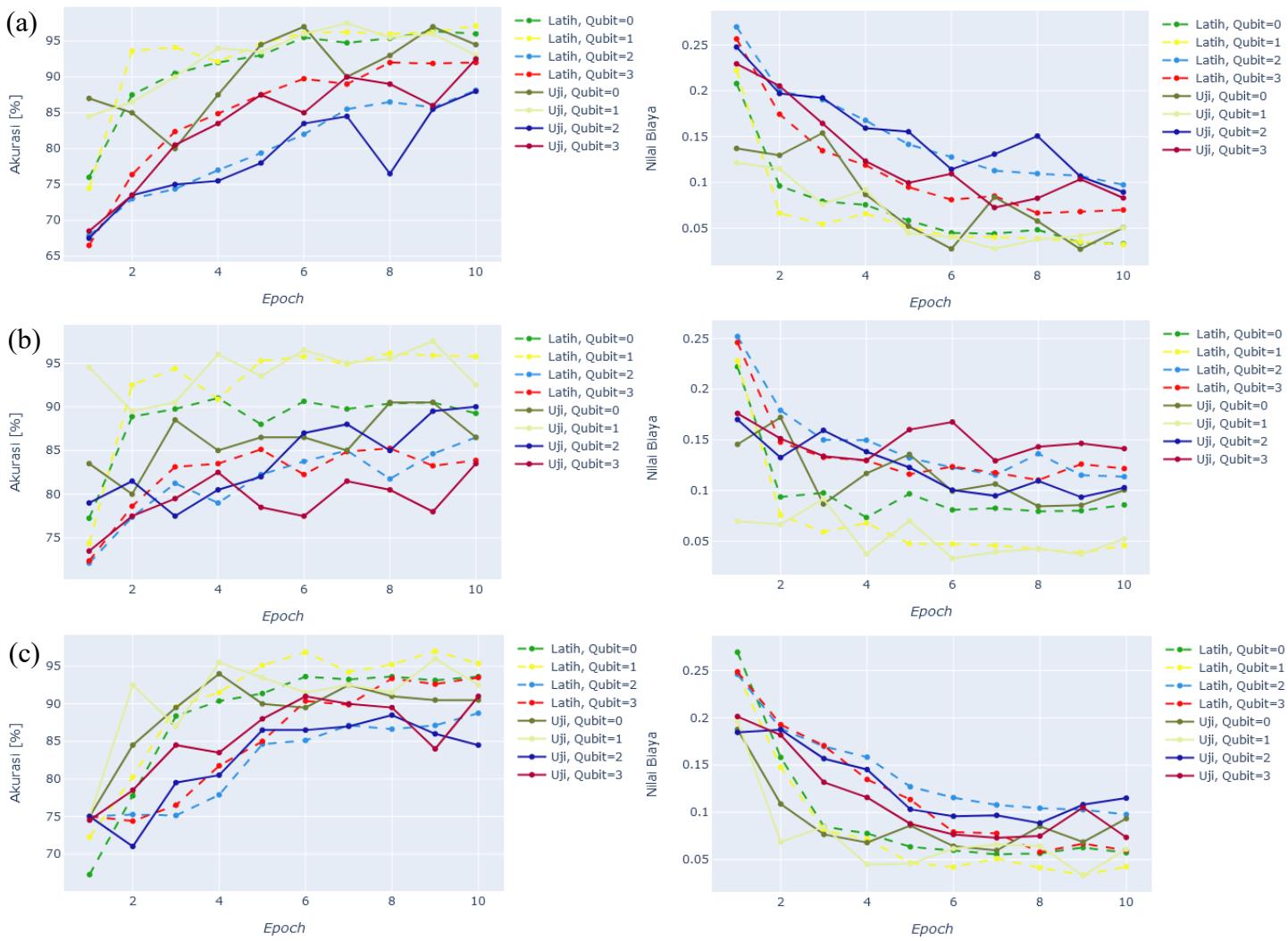
GRAFIK AKURASI DAN NILAI BIAYA SELAMA PROSES LATIH



Gambar A.1 Grafik nilai akurasi dan nilai biaya selama proses latih pada kombinasi reduksi PCA (a) 6 komponen prinsip, (b) 12 komponen prinsip, dan (c) 18 komponen prinsip dengan pengklasifikasi DRC untuk klasifikasi 4 kelas.



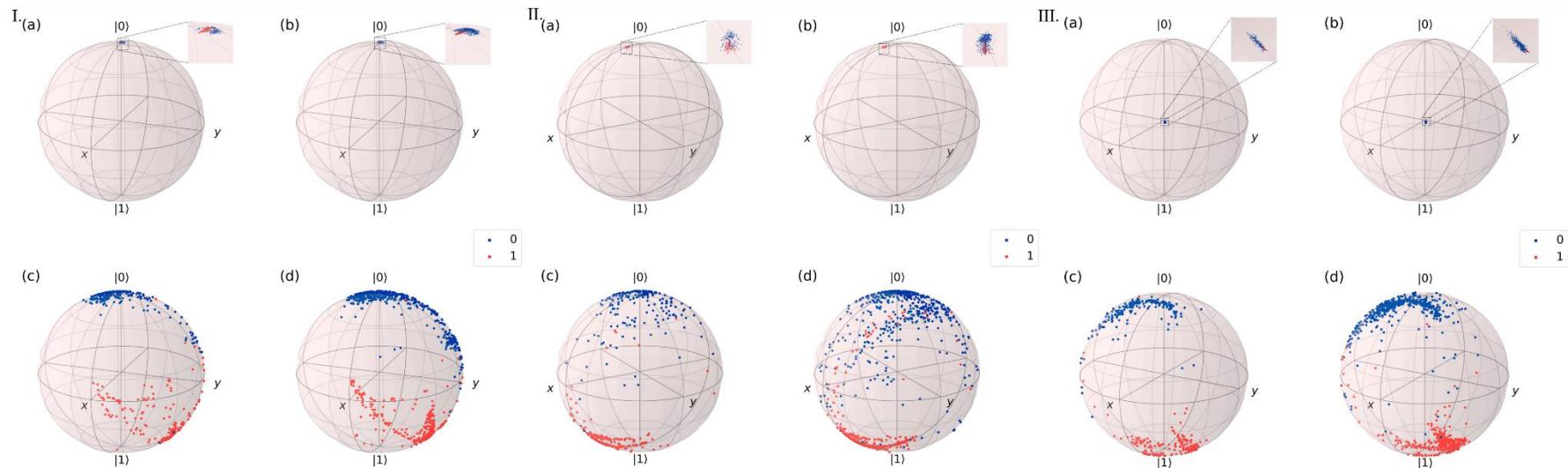
Gambar A.2 Grafik nilai akurasi dan nilai biaya selama proses latih pada kombinasi reduksi PCA dengan pengklasifikasi DRC-RB untuk (a) klasifikasi 4 kelas (18 komponen prinsip, 4 lapisan) dan (b) klasifikasi 8 kelas (27 komponen prinsip, 4 lapisan).



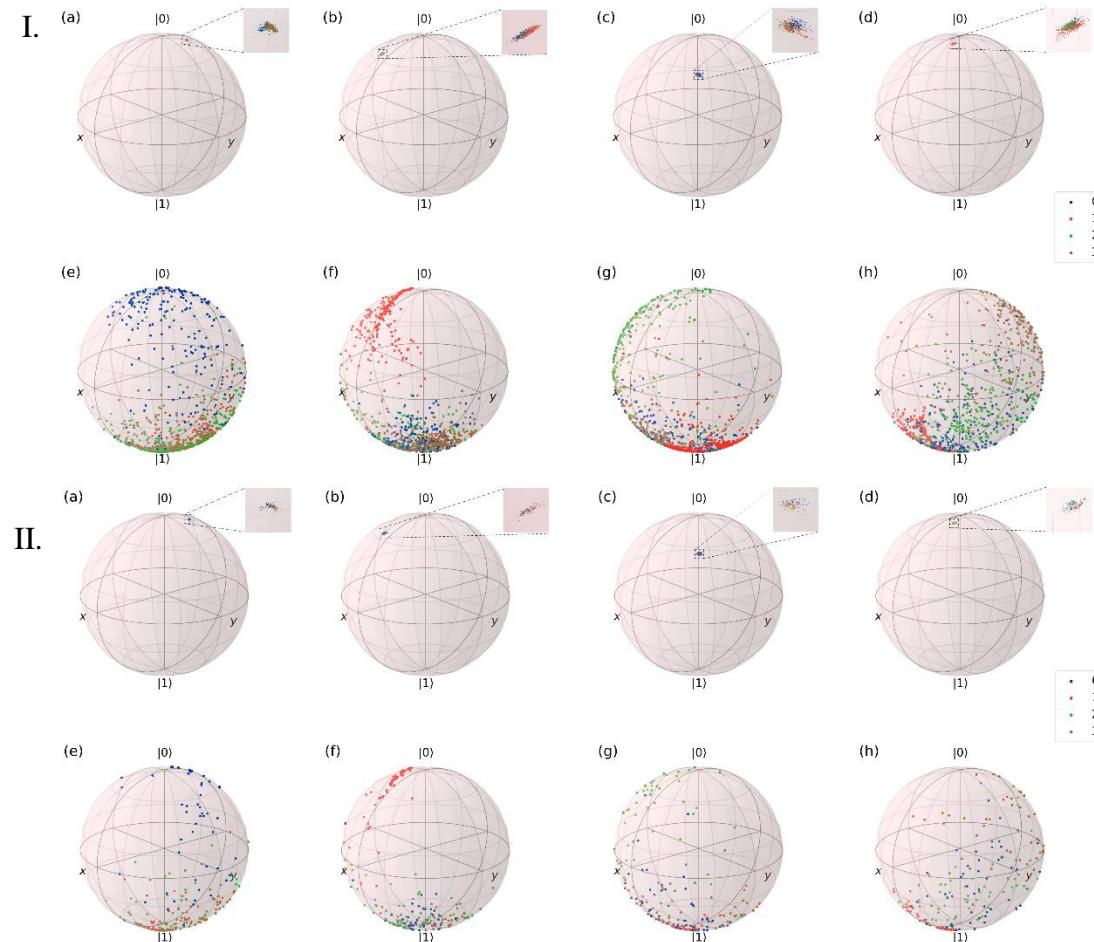
Gambar A.3 Grafik nilai akurasi dan nilai biaya selama proses latih pada kombinasi (a) DRQConv I, (b) DRQConv II, dan (c) DRQConv II + ent dengan pengklasifikasi DRC-RB untuk klasifikasi 4 kelas.

LAMPIRAN B

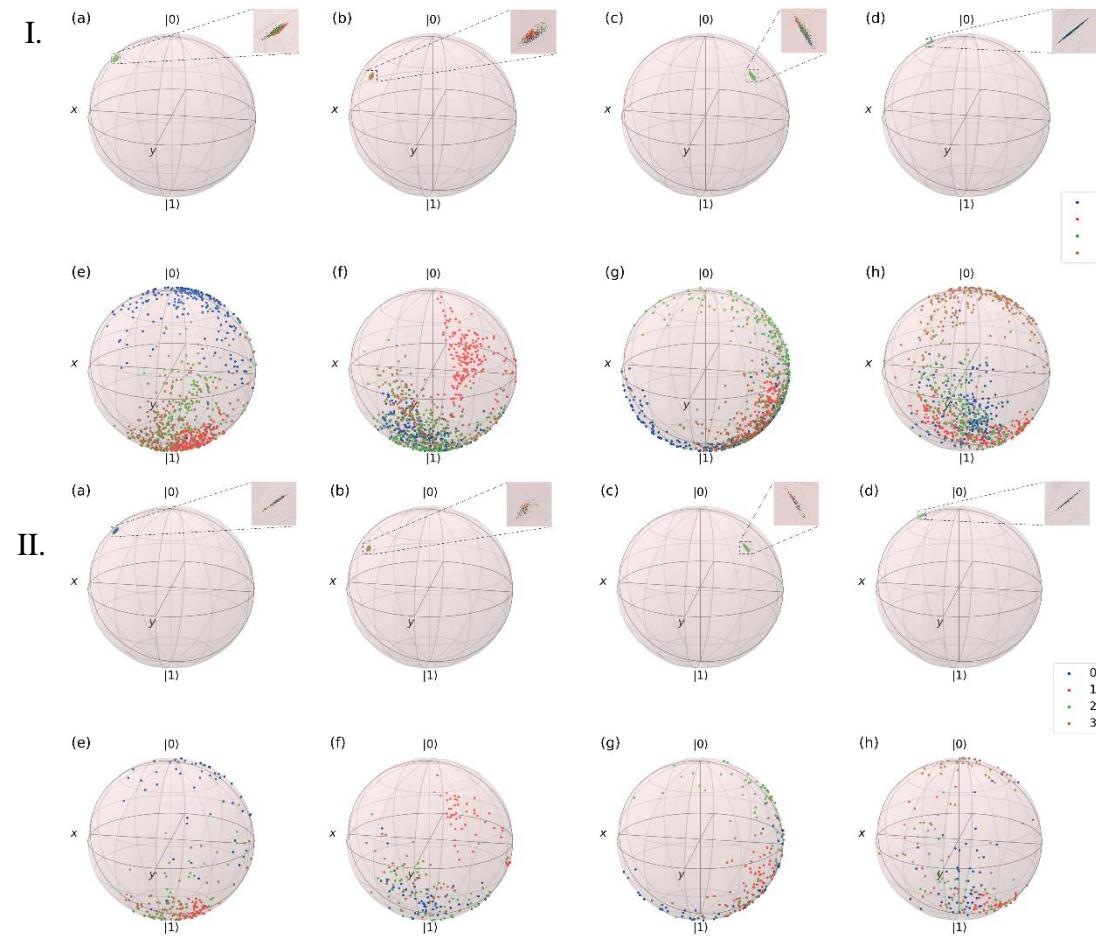
REPRESENTASI *BLOCH SPHERE* SEBELUM DAN SETELAH PROSES LATIH



Gambar B.1 Representasi *Bloch sphere* untuk kasus klasifikasi 2 kelas sebelum (baris atas) dan setelah (baris bawah) proses latih dengan *ansatz* sirkuit I. DRQConv I, II. DRQConv II, III. DRQConv II + ent. (a) dan (c) adalah evaluasi data latih, (b) dan (d) adalah evaluasi data uji.



Gambar B.2 Representasi *Bloch sphere* untuk kasus klasifikasi 4 kelas pada evaluasi data I. latih dan II. uji sebelum proses latih (baris atas) dan setelah proses latih (baris bawah) menggunakan *ansatz* sirkuit DRQConv II.



Gambar B.3 Representasi *Bloch sphere* untuk kasus klasifikasi 4 kelas pada evaluasi data I. latih dan II. uji sebelum proses latih (baris atas) dan setelah proses latih (baris bawah) menggunakan *ansatz* sirkuit DRQConv II + ent.