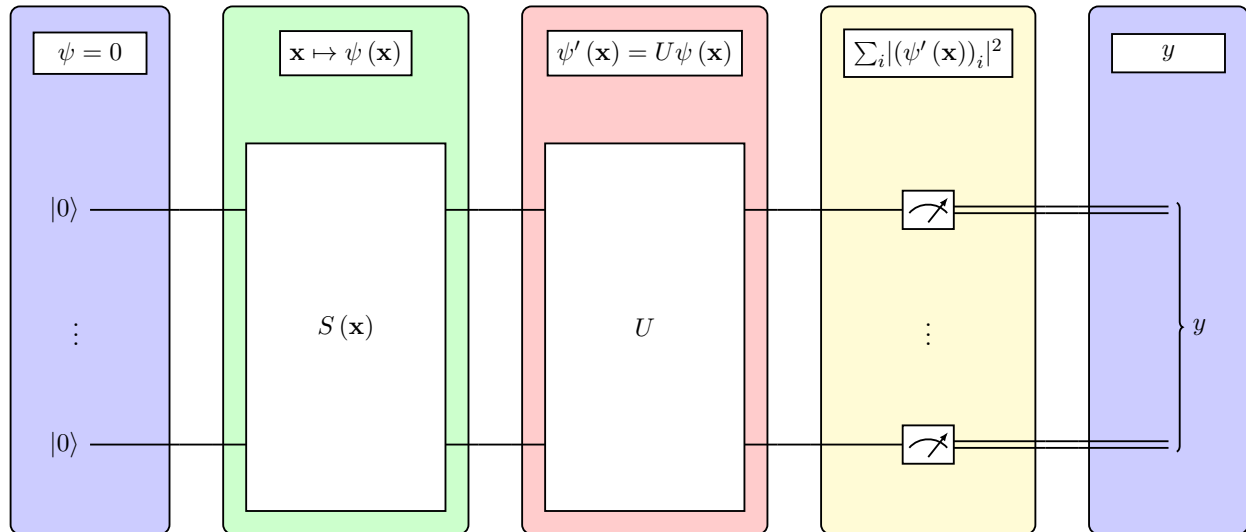




Maskininlärning för klassificering med en kvantdator

Prestandajämförelse mellan en klassisk stödvektormaskin, en kvantmekanisk variationell klassificerare och en kvantmekanisk kärnfunktionsuppskattare

Machine learning for classification with a quantum computer

Performance comparison between a classical support vector machine, a variational quantum classifier, and a quantum kernel estimator

Kandidatarbete i Teknisk fysik

Erik Broback, Victor Henkow, Oscar Liljenzin, Andreas Lund, Alex Maltesson & Emil Zaya

Institutionen för mikroteknologi och nanovetenskap

KANDIDATARBETE 2022

Maskininlärning för klassificering med en kvantdator

Prestandajämförelse mellan en klassisk stödvektormaskin, en kvantmekanisk variationell klassificerare och en kvantmekanisk kärnfunktionsuppskattare

Machine learning for classification with a quantum computer

Performance comparison between a classical support vector machine, a variational quantum classifier, and a quantum kernel estimator

Erik Broback
Victor Henkow
Oscar Liljenzin
Andreas Lund
Alex Maltesson
Emil Zaya



CHALMERS

Avdelningen för tillämpad kvantfysik
Institutionen för mikroteknologi och nanovetenskap
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2022

Maskininlärning för klassificering med en kvantdator
Prestandajämförelse mellan en klassisk stödvektormaskin, en kvantmekanisk variationell klassificerare
och en kvantmekanisk kärnfunktionsuppskattare
Erik Broback
Victor Henkow
Oscar Liljenzin
Andreas Lund
Alex Maltesson
Emil Zaya

© Erik Broback, Victor Henkow, Oscar Liljenzin, Andreas Lund, Alex Maltesson & Emil Zaya,
2022.

Handledare: David Fitzek & Anton Frisk Kockum, Institutionen för mikroteknologi och nanovetenskap
Examinator: Per Lundgren, Institutionen för mikroteknologi och nanovetenskap

Kandidatarbete 2022
Avdelningen för tillämpad kvantfysik
Institutionen för mikroteknologi och nanovetenskap
Chalmers tekniska högskola
412 96 Göteborg
Telefon +46 31 772 1000

Omslag: En schematisk förklaring av beräkningsprocessen i en kvantkretsalgorithm då klassisk data
skall klassificeras.

Typsatt i L^AT_EX
Göteborg, Sverige 2022

Sammandrag

I detta arbete implementeras och undersöks en klassisk och två kvantmekaniska maskininlärnings-algoritmer för att se om kvantalgoritmerna har någon fördel gentemot den klassiska. De algoritmer som undersöks är en stödvektormaskin (SVM), en kvantmekanisk variationell klassificerare (VQC) och en kvantmekanisk kärnfunktionsuppskattare (QKE). De kvantmekaniska algoritmerna testas i huvudsak med få (≤ 7) kvantbitar genom simuleringar i mjukvarubiblioteken PennyLane och Qiskit, men även till viss del på riktiga kvantdatorer från IBMQ. Efter implementering och testande med olika datamängder, kärnfunktioner och tillståndsförberedelser jämförs prestandan för de olika algoritmerna, genom bedömning av noggrannheten och körtiden för varje test. Resultaten visar både via noggrannhet och körtid att SVM:en presterar bäst, QKE:n presterar näst bäst och i ett fall bättre än SVM:en, och VQC:n presterar sämst. Slutsatsen är att denna jämförelse inte är till kvantalgoritmernas fördel, då den klassiska algoritmen redan löser klassificering väl. Förmodligen finns mer potential hos kvantalgoritmer när det kommer till att lösa andra problem som klassiska algoritmer inte kan hantera.

Abstract

In this study one classical and two quantum machine learning algorithms are implemented and investigated with the goal of determining if the quantum algorithms show any advantage compared to the classical one. The examined algorithms are a support vector machine (SVM), a variational quantum classifier (VQC) and a quantum kernel estimator (QKE). The quantum algorithms are mainly tested with few (≤ 7) qubits through simulations in the software libraries PennyLane and Qiskit, but also to some extent through the usage of real quantum computers from IBMQ. After implementation and testing for different data sets, kernel functions, and encodings the performance of the algorithms are compared, by assessment of the accuracy and runtime for each test. The results show, both in accuracy and runtime, that the SVM performs the best, the QKE performs second best and in one case even outperforms the SVM, while the VQC performs the worst. The conclusion taken from this is that the quantum algorithms are at a disadvantage in this comparison, since the classical algorithm already manages classification well. There is most likely more potential for quantum algorithms in solving other problems which classical algorithms struggle with.

Nyckelord: Maskininläring, Kvantdatorer, Kvantmaskininläring, Stödvektormaskin, Kvantmekanisk variationell klassificerare, Kvantmekanisk kärnfunktionsuppskattare.

Förord

Vi vill tacka våra handledare David Fitzek och Anton Frisk Kockum för all vägledning och hjälp som gjorde det möjligt att genomföra detta arbete. Vi är tacksamma för de konstruktiva kommentarerna och det uppmuntrande som vi fått genom arbetets gång.

Erik Broback, Victor Henkow, Oscar Liljenzin, Andreas Lund, Alex Maltesson & Emil Zaya

Göteborg, maj 2022

Ordlista

Egenskapsavbildning (eng. feature map) är en avbildning av lågdimensionella datapunkter till ett högdimensionellt egenskapsrum. 4, 11

Egenskapsrum (eng. feature space) är ett högdimensionellt rum där en egenskap hos datapunkterna motsvarar en eller flera axlar. viii, 2, 4

Korsvalidering är statistisk metod för att kunna testa förmågan hos en algoritm med högre säkerhet. Den bygger på att upprepat undersöka modellen med olika uppdelningar, för träning och testning, av datamängden. V, 13, 17–19

Kvantbit är en kvantdators motsvarighet till en klassisk bit. De kan uppvisa kvantmekaniska effekter som sammanflätning, vilket kan utnyttjas till att utföra snabbare beräkningar i vissa fall. iv, viii, 1, 5, 6, 8, 10, 14, 17, 18, 23, 24

Kvantmekanisk kärnfunktionsuppskattare (QKE) (eng. quantum kernel estimator) är en kvantmekanisk klassificeringsalgoritm som motsvarar en SVM förutom att kärnfunktionen beräknas med en kvantdator. iv, 1, 9, 11, 12, 16–19, 21–23, 26

Kvantmekanisk variationell klassificerare (VQC) (eng. variational quantum classifier) är en kvantmekanisk klassificeringsalgoritm som är designad för att fungera på NISQ-system. iv, 1, 9–12, 14, 16–18, 21–24, 26

Kärnfunktion är en generalisering av skalärprodukten. Det är ett centralt koncept för en SVM. iv, viii, 4, 5

NISQ (eng. Noisy Intermediate-Scale Quantum) är den teknologiska era som kvantdatorer befinner sig i idag, där antalet kvantbitar är begränsat och brus har stor påverkan. viii, 1, 9, 24

Nesterovmomentummetoden (NMM) är en variant av en momentummetod, vilken används för att minimera en funktion. 11, 15

Principalkomponentanalys (PCA) är en metod för att minska dimensionalitet hos en datamängd. 14, 17, 18, 21

Stödvektor är de datapunkter som används för att bestämma hyperplanet i en SVM. I, 2, 12

Stödvektormaskin (SVM) är en vanlig klassisk klassificeringsalgoritm som utnyttjar kärnfunktioner. iv, viii, 1, 2, 5, 7, 12, 14, 16–18, 20–22, 26

Tillståndsförberedelse är en avbildning av datapunkter till ett högdimensionellt Hilbertrum. Metoden används för att kunna använda klassisk data i en kvantdator. iv, 1, 7–9, 14, 21

Nomenklatur

$\mathbf{x}_i$	Datapunkt
y_i	Markeringen som hör till datapunkten $\mathbf{x}_i$
M	Antal datapunkter i en datamängd
N	Antal egenskaper hos en datapunkt
b	Bias
$\mathbf{w}$	Viktvektor
f	Klassificeringsfunktion
α_i	Lagrangemultiplikator
n	Antal kvantbitar i en krets
l	Antal lager i en kvantkrets
θ	Parametrar
S	Tillståndsförberedelseoperator
U	Unitär operator
K	Kärnfunktion
C	Kostnadsfunktion
ψ	Vågfunktion
$\mathcal{P}$	Sannolikhet
π	Sannolikhet som har blivit justerad av bias
ϕ	Egenskapsavbildning
T	Träningsmängd
V	Testmängd
k	Antal lager i korsvalidering
a	Noggrannhet

Innehåll

1	Inledning och syfte	1
2	Teori	1
2.1	Maskininlärning och klassificering	1
2.2	Stödvektormaskin	2
2.2.1	Matematisk formulering av en klassisk stödvektormaskin	2
2.2.2	Kärnfunktioner i en klassisk stödvektormaskin	4
2.2.3	Exempel på kärnfunktioner	5
2.3	Kvantkretsar	5
2.3.1	Kvanttillstånd i tvånivåsystem	5
2.3.2	Blochsfären	6
2.3.3	Logiska grindar i en kvantdator	6
2.3.4	Tillståndsförberedelse av klassiska data	7
2.3.5	Mätning av kvanttillstånd	8
2.4	Kvantmekanisk variationell klassificerare	9
2.4.1	Tillståndsförberedelse	9
2.4.2	Variationell krets	9
2.4.3	Mätning och tolkning	10
2.4.4	Optimering av algoritmen med kostnadsfunktion	10
2.5	Kvantmekanisk beräkning av kärnfunktion	11
3	Metod och utförande	12
3.1	Datamängder	12
3.2	Prestandautvärdering och jämförelse av maskininlärningsalgoritmer	13
3.2.1	Noggrannhet	13
3.2.2	Korsvalidering	13
3.2.3	Tid	13
3.3	Stödvektormaskinen	14
3.4	Tillståndsförberedelse	14
3.5	Kvantmekanisk variationell klassificerare	14
3.6	Kvantmekanisk beräkning av kärnfunktion	16
4	Resultat	17
4.1	Stödvektormaskin	17
4.2	Kvantmekanisk variationell klassificerare	17
4.2.1	Simuleringsresultat	17
4.2.2	IBMQ-resultat	18
4.3	Kvantmekanisk kärnfunktionsuppskattare	18
4.3.1	Simuleringsresultat	18
4.3.2	IBMQ-resultat	19
4.4	Jämförelse	19
4.4.1	Noggrannhet	19
4.4.2	Tid	20
5	Diskussion	20
5.1	Tolkning av resultat	20
5.1.1	Stödvektormaskin	20

5.1.2	Kvantmekanisk variationell klassificerare	21
5.1.3	Kvantmekanisk kärnfunktionsuppskattare och ad hoc	21
5.2	Jämförelse av algoritmerna	21
5.2.1	Simuleringar och principalkomponentanalys	21
5.2.2	Noggrannhet	22
5.2.3	Tid	22
5.3	Möjliga utökningar av arbetet	22
5.4	Implikationer för stora kvantdatorer	23
5.4.1	”Karga plåtår”-problemet	24
5.4.2	Brus för stora kvantkretsar	24
6	Samhälleliga och etiska aspekter	24
7	Slutsats	25
	Referenser	27
A	Härledning från primal- till dualproblem	I
B	Kvantgrindar	II
C	Kvantkretsar för tillståndsförberedelse	III
D	Parametrar för klassisk stödvektormaskin	III
E	Kod	IV
F	Resultat för kvantmekanisk variationell klassificerare och kvantmekanisk kärnfunktionsuppskattare	IV

Figurer

2.1	En tvådimensionell SVM	3
2.2	Dimensionshöjning i en stödvektor för att klassificera data	4
2.3	Beräkningsprocessen i en kvantkretsalgorithm då klassiska data skall klassificeras	5
2.4	Blochsferen, en representation av tvånivå-tillstånd	6
2.5	En klassisk NOT-grind	6
2.6	Exempel på en enkel kvantkrets	7
2.7	Kvantkretsens tillståndsförberedelse jämfört med SVM:ens egenskapsavbildning	7
2.8	En kvantmekanisk mätgrind	9
2.9	Beräkningsprocessen i en VQC	9
2.10	Ett exempel på ett lager i en variationell krets	10
2.11	Kvantkretsen i en QKE	11
3.1	Kvantkretsen som implementerades i den kvantmekaniska variationella klassificeraren .	15
C.1	Kretsar för tillståndsförberedelse.	III
F.1	Grafer över noggrannheten och kostandsfunktionen från VQC:n	IV
F.2	Fler grafer över noggrannheten och kostnadsfunktionen från VQC:n	V

Tabeller

2.1	Jämförelse mellan fyra tillståndsförberedelser	8
3.1	De fyra datamängderna som användes i rapporten	12
3.2	De kombinationer av tillståndsförberedelser och antal kvantbitar som testades för VQC:n	15
3.3	Några kombinationer av de antal kvantbitar som testades för QKE:n	16
4.1	Resultaten för SVM:en	17
4.2	Simulerade resultat för VQC:n	18
4.3	Resultat från IBMQ:s kvantdatorer för VQC:n	18
4.4	Simulerade resultat för QKE:n	19
4.5	Resultat på IBMQ:s kvantdatorer för QKE:n	19
4.6	Den bäst uppnådda noggrannheten för varje algoritm på varje datamängd.	20
4.7	Körtid för en SVM på en klassisk dator, och för en VQC och en QKE på en kvantdator.	20
B.1	Tabell över enkvantbitsgrindar med kretsrepresentation och matrisrepresentation . . .	II
B.2	Tabell över tvåkvantbitsgrindar med kretsrepresentation och matrisrepresentation . . .	II
D.1	Parametrar som gav bäst resultat för den SVM:en	III
F.1	Övriga resultat för QKE:n	V

1 Inledning och syfte

Maskininlärning är ett snabbt växande fält med många olika användningsområden. Det används bland annat idag för medicinska diagnoser [1], riskanalys [2], förutsägning av ekonomiska trender [3], bildigenkänning [4] samt i självkörande bilar från till exempel Tesla [5]. Med så många olika användningsområden finns det en motsvarande riklig mängd olika algoritmer som implementerar maskininlärning för att lösa olika problem. Ett sådant problem är klassificering, vilket syftar på att avgöra huruvida någonting tillhör en viss grupp eller ej. Denna typ av algoritmer används vanligen för datorseende, exempelvis ansiktsigenkänning eller kategorisering av text [6].

Ett annat område som befinner sig i ett tidigt utvecklingsstadium är kvantdatorer. Det är datorer som, istället för klassiska datorers binära ettor och nollor, använder kvanttillstånd som kan finna sig i en superposition av ettor och nollor [7]. Idag är kvantdatorer svåra att implementera, men i teorin ska de kunna utföra somliga beräkningar snabbare än dagens superdatorer. Det finns de som hävdar [7] att realiseringen av en kvantdator skulle innebära ett lika stort teknologiskt steg framåt som när mänskligheten tog steget från den mekaniska till den elektriska datorn.

Där dessa två forskningsområden möts uppstår kvantmaskininlärning, vilket är implementation av maskininlärningsalgoritmer på en kvantdator. Maskininlärningens många användningsområden tillsammans med kvantdatorers potential innebär att detta kan bli ett område med många tillämpningar i framtiden [8]. Dock befinner sig de kvantdatorer som används idag i den så kallade NISQ-eran (Noisy Intermediate-Scale Quantum) [9], där beräkningskraften är begränsad och brus har stor inverkan på beräkningar.

Denna rapport syftar till att jämföra en klassisk och två kvantmekaniska maskininlärningsalgoritmer som alla ämnar lösa klassificeringsproblem. Algoritmerna som implementeras i rapporten är en klassisk algoritm, en stödvektormaskin (SVM), vilken jämförs med två kvantmekaniska algoritmer anpassade för NISQ-system, kvantmekanisk variationell klassificerare (VQC) och kvantmekanisk kärnfunktionsuppskattare (QKE). Algoritmerna testas på fyra datamängder och olika parametrar. Kvantalgoritmerna testas även med olika tillståndsförberedelser och simuleras på en klassisk dator med hjälp av olika mjukvarubibliotek. De testas även till en mindre utsträckning på en riktig kvantdator. På grund av begränsningarna hos dagens kvantdatorer testas kvantalgoritmerna för ett litet antal kvantbitar (≤ 7). Prestandan hos de olika algoritmerna utvärderas och jämförs huvudsakligen genom noggrannheten i deras klassificering samt via algoritmernas körtider.

2 Teori

Här presenteras grundläggande koncept för att förstå de maskininlärningsalgoritmer som undersöks i rapporten. Först introduceras maskininlärning och klassificering, som sedan följs upp med beskrivningen av SVM:en. Efter det presenteras en principiell beskrivning av kvantkretsar och därefter de olika kvantalgoritmerna, VQC:n och QKE:n.

2.1 Maskininlärning och klassificering

Maskininlärning syftar på algoritmer som förbättrar genomförandet av en uppgift genom inlärning från datapunkter [10]. Dessa maskininlärningsalgoritmer bygger en modell med hjälp av en datamängd som representerar problemet som behöver lösas. Datamängden delas då oftast upp i en träningsmängd T

och en testmängd V , där testmängden används för att verifiera modellen. För att en maskininlärnings-algoritm skall vara meningsfull behöver det finnas ett samband mellan datapunkterna.

Datapunkterna kan här ses som koordinater i ett koordinatsystem, vilket visas i figur 2.1. Värdet på varje axel utgör ett värde för en viss egenskap hos datapunkten. Exempelvis kan en datapunkt vara en blomma och en egenskap hos datapunkten kan då vara blombladets tjocklek.

Ibland är varje datapunkt märkt med en viss klass. Det är en typ av klassificeringsproblem som kan lösas med maskininläring. Dessa problem kan lösas med väglett lärande (eng. supervised learning) [11], vilket är vad som har studerats i denna rapport. Då är syftet med algoritmen att kunna förutsäga klassen hos nya datapunkter genom att konstruera en klassificeringsmodell utifrån de märkta datapunkterna.

En speciell typ av klassificering är så kallad binär klassificering, vilket innebär att endast två klasser existerar. För denna klassificering kallas vanligen klasserna för positiv och negativ, alltså $\{\pm 1\}$.

Kostnadsfunktionen är en funktion som utgör ett mått på hur bra modellen förhåller sig till datapunkterna. Den använder sig både av förutsagda utdata från modellen och faktiska data från testmängden och beräknar hur mycket fel modellen hade i sin förutsägelse. Genom att minimera kostnadsfunktionen kan modellen förbättras [11]. Vanligtvis är det dock inte möjligt att få modellen att prestera perfekt eftersom träningsmängden är ändlig, således är kostnadsfunktionen sällan 0.

Det är viktigt att en maskininlärningsalgoritm är generaliserad, alltså anpassad till problemet som helhet snarare än de specifika datapunkterna som den tränas på. Avsaknad av generalisering kan orsakas av problem som överanpassning (eng. overfitting) [12]. Vid överanpassning har modellen anpassat sig så bra till varje punkt i träningsdatan att prestandan på det verkliga problemet blir dålig. Det finns också en risk för motsatsen, alltså att modellen blir underanpassad (eng. underfitting). Då har modellen anpassat sig för dåligt till varje datapunkt, vilket också ger dålig prestanda på det verkliga problemet. Målet är att bygga en modell som stämmer bra överens med nya data, och alltså undviker båda över- och underanpassning. Detta kräver en balansgång som i litteratur benämns bias-varians-dilemmat (eng. bias-variance tradeoff) [13].

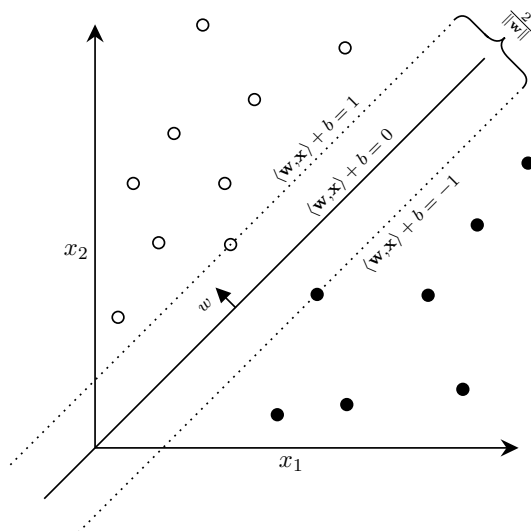
2.2 Stödvektormaskin

En SVM är en klassisk algoritm som kan användas för att lösa binära klassificeringsproblem genom att separera två klasser med ett hyperplan [6]. Se ett tvådimensionellt exempel i figur 2.1. SVM:en kan antingen utföra linjärklassificering i det ursprungliga rummet som i figur 2.1 eller icke-linjär-klassificering i ett implicit högdimensionellt rum, det så kallade egenskapsrummet (eng. feature space) som i figur 2.2b.

Träningen av en SVM består av att beräkna hyperplanet. Detta görs genom att använda sig av märkta punkter. Det visar sig från avsnitt 2.2.1 att endast punkterna som ligger närmast hyperplanet krävs för att bestämma det. Dessa punkter kallas stödvektorer. Beräkning av hyperplanet utgör ett optimeringsproblem vilket alltså är minimeringen av en kostnadsfunktion. Detta innefattar att hitta en viktvektor $\mathbf{w}$ och en bias b till hyperplanet som maximerar dess avstånd till alla stödvektorer. Detta brukar uttryckas som en maximering av den så kallade marginalen $d = 1/\|\mathbf{w}\|$.

2.2.1 Matematisk formulering av en klassisk stödvektormaskin

Matematiskt formuleras en SVM som ett kvadratiskt optimeringsproblem [6], där uppgiften är att klassificera en punkt $\mathbf{x} \in \mathbb{R}^N$ i en av två klasser. Detta görs med ett hyperplan, som parametreras av en viktvektor $\mathbf{w} \in \mathbb{R}^N$ och en bias $b \in \mathbb{R}$. Vid klassgränsen (eng. decision boundary) representeras hyperplanet matematiskt enligt $\mathcal{H} : \langle \mathbf{w}, \mathbf{x}_i \rangle + b = 0$. Detta illustreras som linjen i figur 2.1.



Figur 2.1: Ett exempel på en SVM i två dimensioner, där de svarta och vita punkterna illustrerar två olika klasser. Det separerande hyperplanet ges av den heldragna linjen. Punkterna som ligger på de två streckade linjerna är de punkter som ligger närmast hyperplanet och de är dessa som kallas stödvektorer.

Givet M datapunkter $\{(\mathbf{x}_i, y_i) : \mathbf{x}_i \in \mathbb{R}^N\}_{i=1}^M$, där $y_i = \pm 1$ beroende på om $\mathbf{x}_i$ kategoriseras som positiv eller negativ, skall en klassificeringsfunktion

$$f_{\mathbf{w},b}(\mathbf{x}) = \text{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b) \quad (2.1)$$

konstrueras. Detta görs för unika $\mathbf{w}$ och b , där sgn betecknar signumfunktionen. Klassificeringsfunktionen tilldelar klassen efter vilken sida av hyperplanet punkten befinner sig, så att

$$f_{\mathbf{w},b}(\mathbf{x}_i) = y_i, \quad \forall i = 1, \dots, M. \quad (2.2)$$

Om alla datapunkter antas klassificeras felfritt gäller det att

$$(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) y_i \geq 1, \quad \forall i = 1, \dots, M. \quad (2.3)$$

Hyperplanet $\mathcal{H}$ som söks är det som maximerar avståndet till stödvektorerna [14], det vill säga maximerar marginalen

$$d = \frac{1}{\|\mathbf{w}\|} = \sqrt{\frac{1}{\|\mathbf{w}\|^2}}. \quad (2.4)$$

Så $\mathcal{H}$ är det hyperplan som minimerar $\|\mathbf{w}\|^2$. Därav brukar $\mathcal{H}$ tas fram genom att lösa följande optimeringsproblem

$$\begin{aligned} \min_{\mathbf{w} \in \mathbb{R}^N, b \in \mathbb{R}} \tau(\mathbf{w}) &= \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{då } (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) y_i &\geq 1, \quad \forall i = 1, \dots, M, \end{aligned} \quad (2.5)$$

där τ är en så kallad målfunktion [6]. Det kan visas att detta är ekvivalent med att lösa följande dualproblem, se bilaga A,

$$\begin{aligned} \max_{\boldsymbol{\alpha} \in \mathbb{R}^n} L_d(\boldsymbol{\alpha}) &= \sum_{i=1}^M \alpha_i - \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ \text{då } 0 &= \sum_{i=1}^M \alpha_i y_i, \quad \alpha_i \geq 0, \quad \forall i = 1, \dots, M, \end{aligned} \quad (2.6)$$

där L_d kallas för Lagrangianen och α_i kallas för Lagrangemultiplikatorer. Från detta kan sedan ett uttryck för $f_{\mathbf{w},b}$ fås, nämligen

$$f_{\mathbf{w},b}(\mathbf{x}) = \text{sgn} \left(\sum_{i=1}^M y_i \alpha_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b \right). \quad (2.7)$$

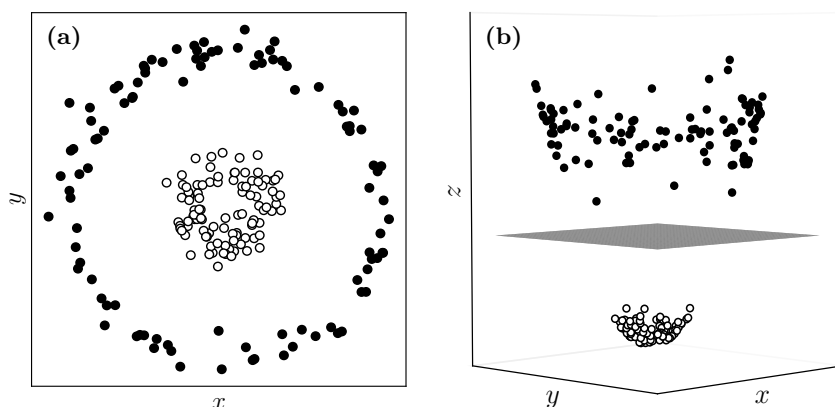
Notera här att $\alpha_i \neq 0$ endast för stödvektorer. Genom att numrera om punkterna så att de m första är stödvektorer fås slutligen

$$f_{\mathbf{w},b}(\mathbf{x}) = \text{sgn} \left(\sum_{i=1}^m y_i \alpha_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b \right), \quad (2.8)$$

varpå punkter kan klassificeras.

2.2.2 Kärnfunktioner i en klassisk stödvektormaskin

Det är inte alltid möjligt att klassificera punkter linjärt med ett hyperplan, exempelvis om punkter av den ena klassen omringar den andra som i figur 2.2a. För att lösa detta introduceras nya koncept, nämligen egenskapsrum och kärnfunktioner.



Figur 2.2: Ett exempel i (a) på en datamängd där punkter av ena klassen omringar den andra. Här kan inte klassificering ske linjärt. Istället avbildas punkterna med en så kallad egenskapsavbildning $\phi(x,y) = (x, y, x^2 + y^2)$ i (b) där ett hyperplan sedan kan användas för klassificering.

Uttrycket i (2.8) kan generaliseras med hjälp av en så kallad kärnfunktion K [15], exempelvis en linjär kärnfunktion

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle, \quad (2.9)$$

där ϕ är en så kallad egenskapsavbildning (eng. feature map) som avbildar punkterna till ett egenskapsrum [6]. Generaliseringen av (2.8) blir då

$$f_{\mathbf{w},b}(\mathbf{x}) = \text{sgn} \left(\sum_{i=1}^m y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b \right). \quad (2.10)$$

Observera att typen av egenskapsavbildning går att välja, vilket exemplifieras i avsnitt 2.2.3, och kärnfunktionen påverkas utefter detta. Kärnfunktionen är då det som skall beräknas för att bestämma hyperplanet i stödvektormaskinen.

Det kan noteras i (2.9) att de explicita vektorerna $\phi(\mathbf{x}_i)$ och $\phi(\mathbf{x}_j)$ inte krävs enskilt för kärnfunktionen, utan endast deras skalärprodukt [6]. Så om en annan avbildningsfunktion $\tilde{\phi}$ som kräver mindre beräkningskraft och

$$\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle = \langle \tilde{\phi}(\mathbf{x}_i), \tilde{\phi}(\mathbf{x}_j) \rangle, \quad \forall i, j = 1, \dots, n, \quad (2.11)$$

kan K tas fram genom att istället beräkna $\tilde{\phi}(\mathbf{x}_1), \dots, \tilde{\phi}(\mathbf{x}_n)$ och deras skalärprodukter. Då krävs det alltså inte att det högdimensionella egenskapsrummet beräknas explicit, vilket undviker tidskrävande beräkningar. Detta undvikande kallas i litteratur för kärntricket (eng. kernel trick) och ligger till grunden för stödvektormaskinens effektivitet [15].

2.2.3 Exempel på kärnfunktioner

I denna rapport kommer fyra olika kärnfunktioner att undersökas för en klassisk SVM, dels den linjära som togs upp i (2.9), men även den polynomiella

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle^d, \quad (2.12)$$

där $d \in \mathbb{Z}_+$, den Gaussiska, vilken i kontexten av en SVM även kallas radiell basfunktion (eng. radial basis function)

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right), \quad (2.13)$$

där $\sigma > 0$, samt sigmolda

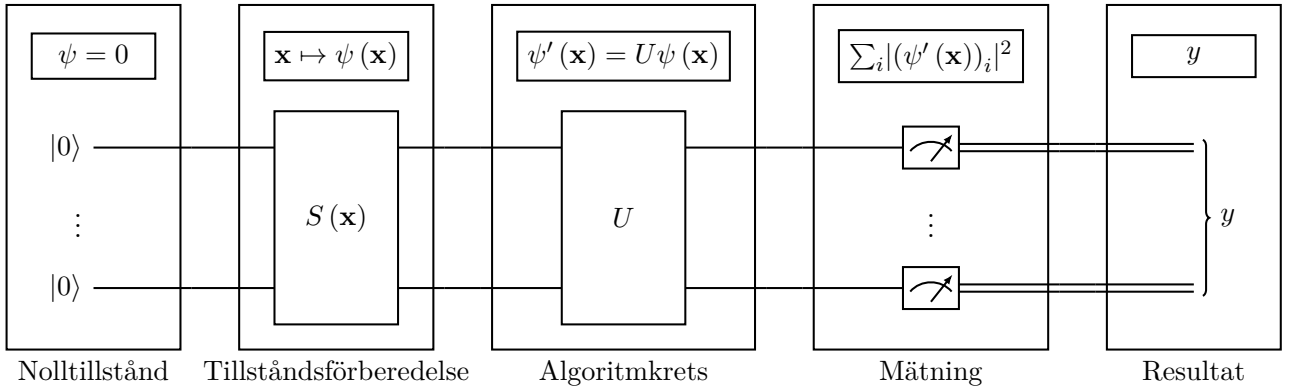
$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\kappa \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \vartheta), \quad (2.14)$$

där $\kappa > 0$ och $\vartheta < 0$ [6].

2.3 Kvantkretsar

Kvantalgoritmer i denna rapport kommer att implementeras genom simulering av så kallade kvantkretsar. Denna typ av krets kan liknas med den klassiska datorkretsen, men i kvantkretsen genomförs kvantmekaniska beräkningar på kvanttillstånd. Det är dessa kretsar som kommer möjliggöra klassificeringen i kvantalgoritmer.

För att ge en förståelse för kvantdatorer och kvantkretsar presenteras i detta avsnitt relevanta koncept inom kvantmekanik samt grundläggande principer för uppbyggnad av kvantkretsar. Detta innefattar tvånivåsystem, Blochsfären, logiska grindar samt dataöverföring in och ut ur en kvantkrets. En övergripande bild av uppbyggnaden av en kvantkretsalgorithm för klassificering ses i figur 2.3.



Figur 2.3: En schematisk förklaring av beräkningsprocessen i en kvantkretsalgorithm då klassiska data skall klassificeras, där S betecknar en tillståndsförberedelsekrets och U betecknar en algoritm.

2.3.1 Kvanttillstånd i tvånivåsystem

Kvantdatorer och kvantkretsar utnyttjar kvantbitar (eng. qubits), deras tillstånd och sammanflätning mellan dem för att utföra beräkningar. Kvantbitarna befinner sig i tvånivåtillstånd. Detta beskrivs av dess normerade kvantmekaniska vågfunktion ψ [16] enligt

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad |\alpha|^2 + |\beta|^2 = 1, \quad \alpha, \beta \in \mathbb{C}. \quad (2.15)$$

Ett tvånivåtillstånd kan också uttryckas med vektornotation på följande vis

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, |\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}. \quad (2.16)$$

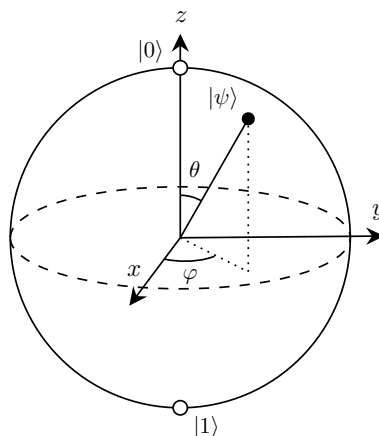
Om två separata kvanttillstånd, $|\psi\rangle$ och $|\phi\rangle$, ska sammanflätas till ett kvanttillstånd, $|\psi\phi\rangle$, blir det sammanslagna tillståndet tensorprodukten av de två individuella, $|\psi\phi\rangle = |\psi\rangle \otimes |\phi\rangle$. För ett högre antal kvantbiter sammanflätas dessa också enligt samma princip, $|\psi_1 \dots \psi_n\rangle = \bigotimes_{i=1}^n |\psi_i\rangle$.

2.3.2 Blochsfären

För att lättare visualisera vad som sker med ett tvånivå-kvantbitstillstånd införs Blochsfären. Som namnet antyder ritas en sfär upp utifrån tillståndet enligt

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right) |0\rangle + \exp(i\varphi) \sin\left(\frac{\theta}{2}\right) |1\rangle. \quad (2.17)$$

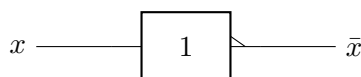
Operatorerna som beskrivs i avsnitt 2.3.3 kan alternativt beskrivas som rotationer av tillståndet i Blochsfären, se figur 2.4, vilket kan underlätta visualiseringen.



Figur 2.4: Blochsfären, en visuell representation av tvånivå-tillståndet i (2.17).

2.3.3 Logiska grindar i en kvantdator

För att förändra vågfunktionen hos en kvantbit i kvantkretsen används unitära operatorer som verkar på kvanttillståndet. När en krets modelleras brukar dessa operatorer ritas upp som logiska grindar. Dessa liknar de logiska grindar som bygger upp klassiska datorer. Ett exempel på en klassisk grind är NOT-grinden [17] i figur 2.5 som inverterar en 1:a till en 0:a och vice versa.



Figur 2.5: En klassisk NOT-grind, där $\bar{x}$ betecknar negationen av x , det vill säga den funktion som avbildar 0 på 1 och vice versa.

Vid en matematisk vektornotation representeras dessa operatorer som matriser. Med denna utökade definition av en logisk grind tillkommer det flera nya grindar som inte återfinns i det klassiska fallet. De vanligaste av dessa är Paulioperatorerna [18] som uttrycks på matrisform enligt

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \quad (2.18)$$

När ett tillstånd färdas genom en grind i ett kretsschema svarar det mot att vågfunktionen multipliceras med grindoperatoren. Exempelvis kan kretsen i figur 2.6 beskrivas av det algebraiska uttrycket $\sigma_x \sigma_z |1\rangle$.

$$|1\rangle \longrightarrow \boxed{Z} \longrightarrow \boxed{X} \longrightarrow -|0\rangle$$

Figur 2.6: En simpel krets där tillståndet $|1\rangle$ transformeras till $-|0\rangle$ genom att det först passerar förbi en "Pauli Z"-grind och sedan en "Pauli X"-grind.

Kretsen i figur 2.6 representerar alltså, i vektornotation, beräkningen

$$\sigma_x \sigma_z |1\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ -1 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \end{pmatrix} = -|0\rangle. \quad (2.19)$$

En operator representeras här av en matris. Matriser kan multipliceras med varandra för att bilda en ny gemensam matris och på samma sätt kan flera operatorer sättas ihop till en gemensam operator, U . Då varje operator är unitär kommer U också att vara det. Att slå samman operatorer på detta vis är en abstraktion som oftast görs för att generalisera samt få en överblick över hela kretsen, se figur 2.3.

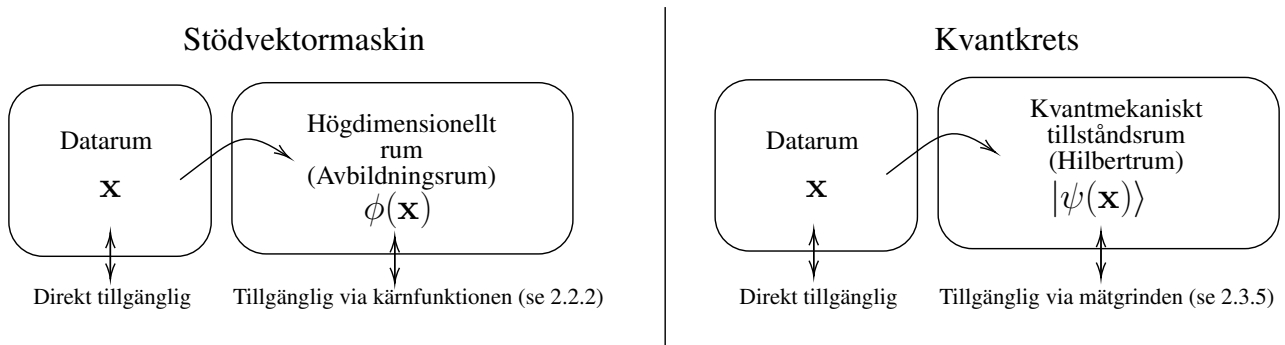
Ifall operatorerna A och B verkar på kvantbit $|a\rangle$ respektive $|b\rangle$ samtidigt kan operatorerna skrivas ihop som en operation på ett tillstånd av två kvantbiter enligt $A \otimes B$. På liknande vis kan ännu fler operatorer slås ihop för fler kvantbiter, på samma sätt som för kvanttillstånd, vilket kan ses i avsnitt 2.3.1.

Det finns fler viktiga kvantoperatorer som används i denna rapport. En sådan är Hadamardoperatoren som omvandlar ett grundtillstånd, alltså $|0\rangle$ eller $|1\rangle$, till ett superpositionstillstånd. Den kan även tolkas som ett basbyte från z till x och vice versa på Blochsfären, se avsnitt 2.3.2. Operatoren kan beskrivas som en matris och denna visas i tabell B.1.

Ett exempel på en operator som verkar på flera kvantbiter är CNOT-operatoren (eng. controlled NOT). Den agerar som en NOT-operator på den andra kvantbiten enbart om den första kvantbiten är $|1\rangle$ och gör ingenting om första kvantbiten är $|0\rangle$. CNOT-operatorns matris- och kretsrepresentation visas i tabell B.2.

2.3.4 Tillståndsförberedelse av klassiska data

Kvantalgoritmerna som används i denna rapport, vilka redovisas i avsnitt 2.4 och 2.5, analyserar klassiska data som representeras av numeriska värden och inte kvanttillstånd. För att möjliggöra detta måste inmatningsdata $\mathbf{x}$, N -dimensionella punkter, först avbildas till kvanttillstånd $|\psi(\mathbf{x})\rangle$ som kvantkretsen kan hantera. Denna process kallas tillståndsförberedelse (eng. encoding). Se figur 2.7 för en illustration av tillståndsförberedelse, samt en parallell till SVM:en.



Figur 2.7: Tillståndsförberedelse i en kvantkrets, som även liknas med egenskapsavbildning i en stödvektormaskin. Detta är en liknelse vars implikationer kan undersökas vidare, se [19].

Tillståndsförberedelse kan utföras på flera olika sätt, varav några kan ses i tabell 2.1. Dessa har olika för- och nackdelar beroende på datastorleken och kretsutseendet [20]. De tillståndsförberedelserna förverkligas med logiska grindar i olika konfigurationer, se figur C.1.

Tabell 2.1: Jämförelse mellan fyra tillståndsförberedelser för en datamängd som har M datapunkter med N egenskaper vardera. För varje tillståndsförberedelse anges antal kvantbitar som behöver användas, körtid för tillståndsförberedelsen samt strukturen på inmatningsdata. Tabellen kommer från [18], där tillståndsförberedelserna även förklaras i detalj.

Tillståndsförberedelse	Kvantbitar	Körtid	Inmatningsdata
Bas	N	$\mathcal{O}(MN)$	Binär
Amplitud	$\log_2 N$	$\mathcal{O}(MN)/\mathcal{O}(\log(MN))^*$	Kontinuerlig
Vinkel	N	$\mathcal{O}(N)$	Kontinuerlig
Hamiltonian	$\log_2 N$	$\mathcal{O}(MN)/\mathcal{O}(\log(MN))^*$	Kontinuerlig

* Det högra alternativet gäller bara under särskilda förhållanden, se [18] för detaljer.

Nedan presenteras de tillståndsförbereder som används i denna rapport i mer detalj.

Amplitudtillståndsförberedelse (eng. amplitude encoding) antar att $\mathbf{x} \in \mathbb{C}^{2^n}$ och är normaliserad [19], där n är antal kvantbitar. Se figur C.1a för kretsdetaljer. I Hilbertrummet representeras varje egenskap x_i med ett kvanttillstånd vars amplitud motsvarar elementen i inmatningsvektorn. Alltså krävs $\log_2 N$ kvantbitar. Tillståndsförberedelsen beskrivs matematiskt enligt

$$\mathbf{x} \mapsto |\psi(\mathbf{x})\rangle \text{ där } |\psi(\mathbf{x})\rangle = \sum_{i=0}^{2^n-1} x_{i+1} |i\rangle, \quad (2.20)$$

där $|i\rangle$ är den binära representationen av i [18]. Exempelvis blir $|i\rangle = |110\rangle$ då $n = 3$ och $i = 6$.

Vinkeltillståndsförberedelse (eng. angle encoding) antar att $\mathbf{x} \in \mathbb{R}^n$ utan krav på normalisering [19]. Se figur C.1b för kretsdetaljer. I Hilbertrummet roteras här varje kvantbit, exempelvis via Y-rotationsoperatorer som redovisas i tabell B.1, med ett värde motsvarande varje egenskap x_i i radianer. Således krävs N kvantbitar. Tillståndsförberedelsen beskrivs matematiskt enligt

$$\mathbf{x} \mapsto |\psi(\mathbf{x})\rangle \text{ där } |\psi(\mathbf{x})\rangle = \sum_{q_1, \dots, q_n=0}^1 \prod_{k=1}^n \cos(x_k)^{q_k} \sin(x_k)^{1-q_k} |q_1 \dots q_n\rangle. \quad (2.21)$$

IQP-tillståndsförberedelse (eng. instantaneous quantum polynomial style encoding) är en mer komplicerad tillståndsförberedelse som använder Hadamardgrindar, CNOT-sammanflätningar och Z-rotationer [21]. Se figur C.1d för kretsdetaljer. Det behöver inte vara just Z-rotationer utan X och Y fungerar också, men här används Z-rotationer och då kallas den även för ZZ-tillståndsförberedelse. Likt vinkeltillståndsförberedelse finns inga särskilda normaliseringskrav här och det krävs N kvantbitar. Tillståndsförberedelsen beskrivs matematiskt enligt

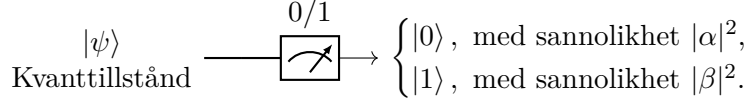
$$\mathbf{x} \mapsto (U_Z(\mathbf{x})H^{\otimes n})^l |0\rangle^{\otimes n} \text{ där } U_Z(\mathbf{x}) = \prod_{(i,j) \in S} R_{Z_i Z_j}(x_i x_j) \bigotimes_{k=1}^n R_z(x_k), \quad (2.22)$$

där l är kretsdjupet, alltså antal gånger som kretsen U_Z upprepas, och S är mängden par av kvantbitar som sammanflätas med R_{ZZ} -grinden. I denna rapport undersöks även ett specialfall av IQP-tillståndsförberedelse då $S = \emptyset$, vilket framöver kallas för Z-tillståndsförberedelse, se figur C.1c.

2.3.5 Mätning av kvanttillstånd

En grundläggande princip från kvantmekaniken är att det inte går att direkt observera ett kvanttillstånd [16]. När ett tillstånd observeras kommer det att kollapsa till ett av de möjliga egentillstånden.

I ett tvånivåsystem ger alltså en observation alltid antingen 0 eller 1, vilket på ett naturligt sätt kan uttryckas som en klassisk bit. För att klassificeringen efter kvantkretsen i praktiken skall kunna uppmätas måste tillståndet först manipuleras. Den här processen beskrivs med en logisk grind som visas i figur 2.8 med en tillhörande operator. För detaljer se [22].

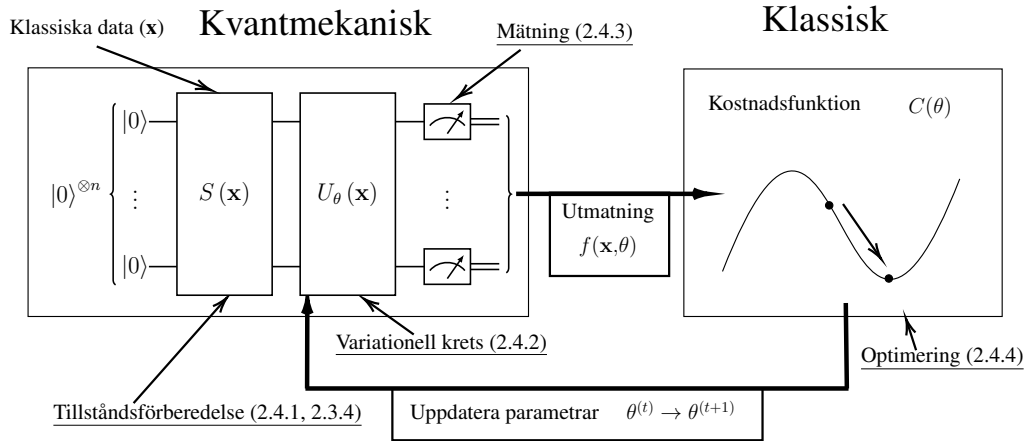


Figur 2.8: En tillståndsmätningssgrind, som används i en kvantkrets för att extrahera information ur kvantbitstillståndet. Kvanttillståndet kollapsar i denna operator till antingen 0 eller 1, alltså en klassisk bit. Där α och β definieras i (2.15).

Resultatet som fås från kvantkretsen är slumpartat och bör således tolkas probabilistiskt. Väntevärdet för tillståndet ges som summan av egenvärdena multiplicerat med sannolikheten för tillhörande egentillstånd $\langle\psi\rangle = \sum_i a_i |\psi_i|^2$. I praktiken måste väntevärdet uppskattas, vilket görs genom att köra kretsen flera gånger och sedan ta medelvärde av de erhållna resultaten [20]. Detta gäller för alla kvantkretsar som används i detta arbete.

2.4 Kvantmekanisk variationell klassificerare

En VQC är en maskininlärningsalgoritm som är framtagen för att kunna implementeras på dagens NISQ-kvantdatorer [18]. En av fördelarna med en variationell algoritm är att den kan lära sig att motverka systematiska fel, exempelvis då en grind överroterar ett tillstånd [20]. Tillvägagångssättet för en VQC kan sammanfattas i fyra steg [23], tillståndsförberedelse, en variationell krets, mätning och optimering, vilka visas i figur 2.9.



Figur 2.9: Schematisk bild av beräkningssprocess för klassificering med en VQC. En kvantkrets utför klassificering och dess utmatningsvärde används för att minimera en kostnadsfunktion som sedan uppdaterar parametrarna för kvantkretsen. Detta utgör en iterativ process, vilken körs tills optimerade parametrar erhålles.

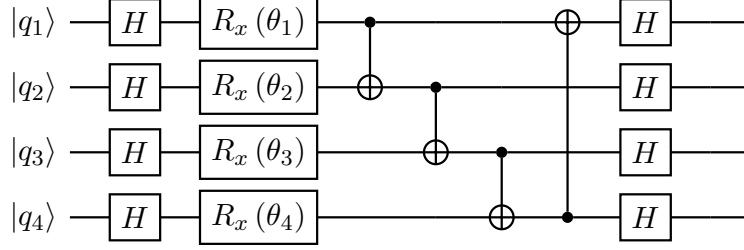
2.4.1 Tillståndsförberedelse

Tillståndsförberedelse för VQC:n är samma som för QKE:n. Därmed har detta samlats i avsnitt 2.3.4.

2.4.2 Variationell krets

Den variationella kretsen $U_\theta(\mathbf{x})$ är en central del av VQC:n där, efter fullständig träning, en datapunkt representerad av ett kvanttillstånd överförs till en gissning om punktens klassificering. Kretsen består

av flera lager $U = U_m \dots U_1$, där varje lager består av en och samma krets [23]. Dessa lager kan konstrueras på många olika sätt, men väsentligen är de ett antal parametriserade operatorer för att rotera tillstånden i Blochsfären, samt några CNOT-grindar eller liknande för att sammanfläta mellan kvantbitarna, se figur 2.10. Denna krets namnges i litteraturen som ansats (eng. ansatz) och valet av den utgör ett problem i sig.



Figur 2.10: Ett exempel på ett lager i en variationell krets, där de ingående kvantbitarna $|q_1\rangle$, $|q_2\rangle$, $|q_3\rangle$ och $|q_4\rangle$ roteras av parametriserade R_x -grindar och sammanflätas med CNOT-grindar.

Som det nämndes innan är det ett viktigt krav för dessa parametriserade operatorer att de är justerbara, eftersom det är nödvändigt för att få förbättringar i kretsen [20]. Detta visas i figur 2.9 under parameteruppdatering, där varje parametriserad operator är beroende av en parameter θ_i , vilken optimeras under minimering av en kostnadsfunktion. Detta redogörs för i avsnitt 2.4.4.

2.4.3 Mätning och tolkning

Kvanttillståndet som kommer ut från den variationella kretsen behöver mätas och bedömas. När de utgående kvanttillstånden passerar mätgrindar, enligt avsnitt 2.3.5, mäts tillstånden. Ett sätt att utföra denna mätning [20] är att hitta sannolikheten $\mathcal{P}$ för att den första kvantbiten q_1 är i tillstånd 1, vilket matematiskt blir

$$\mathcal{P}(q_1 = 1; \mathbf{x}, \theta) = \sum_{k=2^{n-1}+1}^{2^n} |U_\theta \psi(\mathbf{x})_k|^2. \quad (2.23)$$

Nästa steg är att justera resultatet genom att addera biasen b till den erhållna sannolikheten

$$\pi(\mathbf{x}; \theta, b) = \mathcal{P}(q_1 = 1; \mathbf{x}, \theta) + b. \quad (2.24)$$

Slutligen blir gissningen för klassificeringen

$$f(\mathbf{x}, \theta, b) = \begin{cases} 1 & \text{om } \pi(\mathbf{x}; \theta, b) > \frac{1}{2}, \\ 0 & \text{annars.} \end{cases} \quad (2.25)$$

2.4.4 Optimering av algoritmen med kostnadsfunktion

Träning av en VQC görs stegvis genom att ansätta en kostnadsfunktion (eng. cost function) [20]. Kostnadsfunktionen minimeras för att uppnå optimala parametrar som då ger flest korrekta klassificeringar. Själva kostnaden kan evalueras på flera olika sätt. I denna rapport används en specifik kostnadsfunktion, nämligen medelkvadratfelet (MSE) (eng. mean squared error),

$$C(\theta, b) = \text{MSE}(\hat{y}(\theta, b)) = \frac{1}{|T|} \sum_{\mathbf{x}_i \in T} (y_i - \hat{y}_i(\theta, b))^2, \quad (2.26)$$

där $\hat{y}_i(\theta, b)$ är en skattning av y_i , vilket är markeringen som hör till $\mathbf{x}_i$, och $|\cdot|$ betecknar kardinaliteten, det vill säga antal element.

Minimeringen av kostnadsfunktionen kan genomföras med den så kallade momentummetoden, vilket innebär att C minimeras med avseende på en eller flera variabler θ ,

$$\theta^{(t+1)} = \theta^{(t)} - v^{(t+1)}, \quad (2.27)$$

$$v^{(t+1)} = \mu v^{(t+1)} + \eta \nabla_{\theta} C(\theta^{(t)}), \quad (2.28)$$

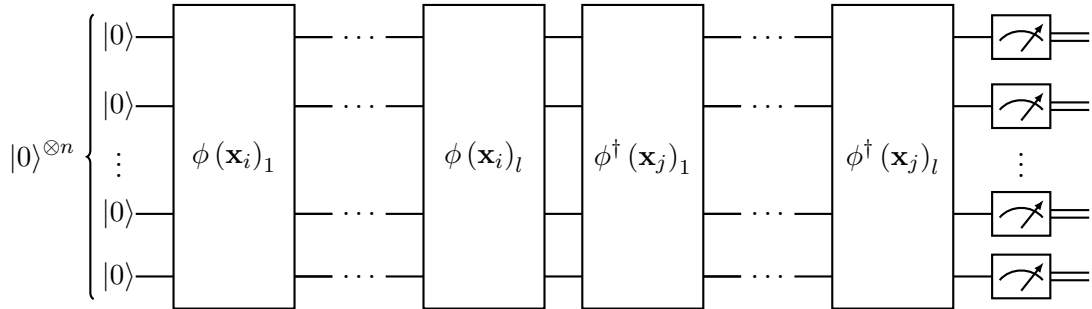
där η är steglängden och $\mu \in (0, 1)$ är en parameter [24]. Punkterna uppdateras i (2.27) utifrån en momentumterm $v^{(t+1)}$. Denna Metod för att minimera C presterar bättre än vanlig sjunkgradientmetod, vilken används i (2.28) för att beräkna momentumtermen. Metoden kan förbättras ytterligare genom att byta ut (2.28) mot

$$v^{(t+1)} = \mu v^{(t+1)} + \eta \nabla_{\theta} C(\theta^{(t)} - \mu v^{(t)}), \quad (2.29)$$

där momentumtermen beräknas med hjälp av Nesterovaccelererad gradientmetod istället, vilket ger en kvadratisk uppsnabbning gentemot vanlig sjunkgradientmetod [25]. Kombinationen av dessa brukar kallas för Nesterovmomentummetoden (NMM) [26].

2.5 Kvantmekanisk beräkning av kärnfunktion

Beräkning av en kärnfunktion kan för vissa problem vara väldigt kostsamt, eller otillräckligt för att hitta en lösning [27]. En alternativ metod för att beräkna kärnfunktionen är med en kvantkrets enligt figur 2.11. Detta är en QKE. Genom att beräkna en kärnfunktion med en kvantkrets som är svår att simulera klassiskt, kan en beräkningsmässig fördel uppnås över klassiska metoder [21].



Figur 2.11: Kretsen som utgör en QKE med n kvantbitar $|0\rangle^{\otimes n}$ och l lager. Exempel på avbildningskretsen ϕ där klassiska datapunkter $\mathbf{x}_i$ och $\mathbf{x}_j$ avbildas ses i figur C.1.

Kärnfunktionen beräknas med hjälp av en kvantmekanisk egenskapsavbildning $\phi(\mathbf{x})$ som används för att avbilda en märkt datapunkt $\mathbf{x} \in T$ till ett Hilbertrum $|\phi(\mathbf{x})\rangle$. För QKE:n gör egenskapsavbildningar samma sak som tillståndsförberedelser. En egenskapsavbildning $\phi(\mathbf{x})$ appliceras enligt figur 2.11 och kan göras med olika antal lager l . Det finns många olika val av $\phi(\mathbf{x})$. Exempel på avbildningar är Z-egenskapsavbildningen och ZZ-egenskapsavbildningen, som båda finns att se i figur C.1. Avbildningskretsarna här utgör samma koncept som ansatsen i VQC:n och det finns många olika kombinationer. Om antal lager $l = 1$ kan kvanttillstånderna skrivas $|\phi(\mathbf{x})\rangle = \phi(\mathbf{x})|0\rangle^{\otimes n}$.

Genom att applicera $\phi^\dagger(\mathbf{x}_j)\phi(\mathbf{x}_i)$ på initialtillståndet $|0\rangle^{\otimes n}$ och göra mätningar längs z -axeln, fås $K(\mathbf{x}_i, \mathbf{x}_j)$ som sannolikheten att mäta tillståndet $|0\rangle^{\otimes n}$. I praktiken beräknas kvoten mellan antalet uppmätta $|0\rangle^{\otimes n}$ -tillstånd och totala antalet körningar [21], [28]. Resultatet är en kärnfunktion

$$K(\mathbf{x}_i, \mathbf{x}_j) = |\langle \phi(\mathbf{x}_j)_1 \dots \phi(\mathbf{x}_j)_l | \phi(\mathbf{x}_i)_1 \dots \phi(\mathbf{x}_i)_l \rangle|^2. \quad (2.30)$$

Speciellt om $l = 1$ gäller

$$K(\mathbf{x}_i, \mathbf{x}_j) = |\langle \phi(\mathbf{x}_j) | \phi(\mathbf{x}_i) \rangle|^2 = |\langle 0 |^{\otimes n} \phi^\dagger(\mathbf{x}_j) \phi(\mathbf{x}_i) | 0 \rangle^{\otimes n}|^2. \quad (2.31)$$

Kärnfunktionen används sedan i en SVM, genom att först använda (2.6) för beräkningen av nya Lagrangemultiplikatorer, α_i och erhålla stödvektorer för att bestämma en klassificerare enligt (2.10).

3 Metod och utförande

I denna rapport undersöktes SVM:en och de två kvantalgoritmerna, VQC:n och QKE:n, för att konstruera en klassificerare. Dessa jämfördes sedan för att avgöra för- och nackdelar med tillvägagångssätten. Nedan redovisas hur de olika algoritmerna implementerades och hur de jämfördes. Koden som producerats i denna rapport hittas via en GitHub-länk i bilaga E.

3.1 Datamängder

Jämförelsen gjordes genom att studera algoritmernas klassificeringsförmåga på fyra datamängder: Iris plants dataset [29] (iris), Breast cancer wisconsin (diagnostic) dataset [29] (cancer), Forest covertypes [29] (forest) och Ad hoc [30] (ad hoc). Dessa datamängder redovisas i tabell 3.1. I tabellen är antal egenskaper och punkter angivna för de olika mängderna.

Tabell 3.1: De fyra datamängderna som användes i rapporten med antal egenskaper och punkter.

Datamängd	Antal egenskaper	Antal punkter
Iris	4	100
Cancer	30	569
Forest	54	500
Ad hoc	2–3	500

Dessa datamängder valdes för att testa algoritmernas förmåga att klassificera datapunkter i mängder med ökande klassificeringssvårighet. Framför allt iris men även cancer är datamängder där det finns ett tydligt mönster och brukar därför ingå i tester av maskininlärningsalgoritmer. I till exempel datamängden iris finns en tydlig separation mellan klasserna och således är dessa lätta att klassificera [29]. Iris är den mest lättklassificerade datamängden som studeras och ger alltså ett första mått på om klassificeringen fungerar. Datamängden cancer är mindre lättklassificerad, då sambandet mellan egenskaperna och klasserna är mer komplicerat, och testar alltså hur bra algoritmerna kan hantera besvärligare problem. Den tredje datamängden, forest, består av verkliga rådata och har inte behandlats för att tydliggöra något samband [29]. Detta gör denna datamängd svårklassificerad och visar hur bra algoritmerna hanterar verkliga problem.

Den sista mängden i tabellen, ad hoc, är en konstgjord datamängd utan någon anknytning till verkliga data. Denna datamängd valdes att tas med eftersom den konstruerats [31] för att vara en datamängd som QKE:n kan klassificera mer korrekt än andra algoritmer. Datamängden [30] kan genereras med både två och tre egenskaper. Härfter benämnes mängderna ad hoc 2 respektive ad hoc 3 efter hur många egenskaper datamängden har. Algoritmen som används för att konstruera denna datamängd fokuserar på tre viktiga egenskaper hos en datamängd, nämligen komplexiteten, dimensionaliteten och geometriska skillnader. För ad hoc är dessa egenskaper baserade på ZZ-tillståndsförberedelsen. En detaljerad beskrivning av algoritmen som används för att konstruera ad hoc finns i bilaga F i [31].

Då denna rapport endast fokuserar på binära klassificerare behövdes två av datamängderna, iris och forest, transformeras då dessa ursprungligen hade tre respektive sju klasser. Detta gjordes genom att endast använda de två första klasserna och bortse från resterande.

3.2 Prestandautvärdering och jämförelse av maskininlärningsalgoritmer

I detta avsnitt presenteras hur prestanda mättes för jämförelsen mellan maskininlärningsalgoritmerna.

3.2.1 Noggrannhet

Efter att algoritmerna kördes var noggrannheten ett grundläggande mått för att utvärdera dem. Från resultaten är det enkelt att bestämma antalet korrekt och inkorrekt klassificerade punkter. Utifrån detta kan sedan noggrannheten a bestämmas enligt [32]

$$a = \frac{\text{Antal korrekt klassificerade punkter}}{\text{Totalt antal punkter}}. \quad (3.1)$$

a är det primära måttet som användes för att jämföra algoritmernas prestanda. Denna parameter kunde tas fram för alla algoritmer och utgör ett tydligt mått på hur bra algoritmen var på att klassificera efter träning.

3.2.2 Korsvalidering

För att få ett säkrare mått på en algoritms förmåga att klassificera nya data användes korsvalidering (eng. cross validation) [33]. Detta innebär att datamängden delades upp i k delar (eng. folds). Efter detta tränas algoritmen på alla delar utom en och testas på den del som inte använts för träning. Denna process upprepades k gånger med en ny testdel för varje upprepning [34]. Noggrannheten (3.1) beräknades för algoritmen i varje iteration och ett medelvärde för algoritmen sammanställs enligt [35]

$$\bar{a} = \frac{1}{k} \sum_{i=1}^k a_i, \quad (3.2)$$

där $\bar{a}$ betecknar medelvärdet. Från detta kunde även en standardavvikelse för stickprovet beräknas [36] enligt

$$s = \sqrt{\frac{1}{k-1} \sum_{i=1}^k (a_i - \bar{a})^2}. \quad (3.3)$$

Alla tester som utfördes i denna rapport gjordes med tiofaldig korsvalidering, $k = 10$. Datamängden valdes att delas upp i 10 delar då det har påvisats [37] vara det bästa antalet för valideringstester.

3.2.3 Tid

Eftersom körtiderna för simulationerna inte kan antas vara proportionella mot motsvarande körtider på kvantdatorer kan inte simulationerna användas för att jämföra körtider. Istället beräknades körtiden för kvantkretsen utifrån tester på en riktig kvantdator. Från simulationerna bestämdes hur många gånger kvantkretsen skulle behöva köras för att slutföra träningen och klassifikationen. Iris-datamängden användes för båda algoritmerna då den kräver minst antal kvantbitar av de använda datamängderna. Z-tillståndsförberedelse användes för QKE:n och vinkeltillståndsförberedelse för VQC:n. Dessa tillståndsförberedelser kunde ändå jämföras då de liknar varandra. Ekvivalenta tester på kvantdatorer för algoritmerna visade hur långt tid det tog att köra igenom kvantkretsarna en gång. Körtiden beräknades därefter genom att ta antal körningar multiplicerat med tiden för en körning. Denna beräkning är inte exakt eftersom tiden varierade mellan körningar och testerna gjordes på olika kvantdatorer vid olika tillfällen, men ger storleksordningen av körtiden för de olika algoritmerna.

3.3 Stödvektormaskinen

SVM:en implementerades i mjukvarubiblioteket `sklearn` [38] och användes för att göra mätningar på olika datamängder. Den funktion som användes för klassificering var `SVC`. I `sklearn` fanns även inbyggt korsvalidering, `cross_val_score`, samt flera kärnfunktioner [39]. De kärnfunktioner som testades var en linjär, en polynomiell, en Gaussisk och en sigmoid, vilka ses i avsnitt 2.2.3. Alla kärnfunktioner testades på alla fyra datamängder som användes i denna rapport.

För samtliga kärnfunktioner undersöktes olika värden på deras parametrar. De värden på parametrarna som användes i resultaten var de som gav högst noggrannhet. För den linjära och Gaussiska kärnfunktionen utgör parametern C en regulariseringskonstant, och används för att bortse från vissa datapunkter i träningsmängden. Eftersom verkliga data kan missklassificeras är det ibland bra att bortse från dessa datapunkter för att inte få en överanpassad modell. Parametern $\gamma \propto 1/\sigma^2$ nämns i avsnitt 2.2.3 och används endast för den Gaussiska funktionen. Den beskriver radien för grupperingar av datapunkter. Parametern `coef0` används för sigmoidfunktionen och är samma som parametern ϑ vilken beskrivs i avsnitt 2.2.3.

3.4 Tillståndsförberedelse

Fyra typer av tillståndsförberedelse testades i denna rapport: amplitud-, vinkel-, Z- och ZZ-tillståndsförberedelse, se avsnitt 2.3.4 för detaljerade beskrivningar av dessa. För att använda amplitudtillståndsförberedelse krävs det av en datapunkt $\mathbf{x}_i$ att antalet egenskaper är en exakt tvåpotens, det vill säga $N = 2^n$ där n är antalet kvantbitar. Detta gällde inte för alla datamängder, se tabell 3.1, och löstes genom att lägga till egenskaper där alla värden var satta till noll, så kallad utfyllning (eng. padding). För att använda vinkel-, Z- eller ZZ-tillståndsförberedelse krävs det $n = N$ kvantbitar för att koda in N egenskaper. Då antalet egenskaper hos vissa av datamängderna, se tabell 3.1, översteg antalet kvantbitar som går att simulera, behövdes dessa transformeras på ett systematiskt sätt för att minska antalet egenskaper. Detta gjordes med principalkomponentanalys (PCA), vilket fanns implementerat i `sklearn` [38]. För att underlätta jämförelse anpassades reduceringen så att antalet använda kvantbitar för en datamängd var samma som användes vid amplitudtillståndsförberedelse för den datamängden.

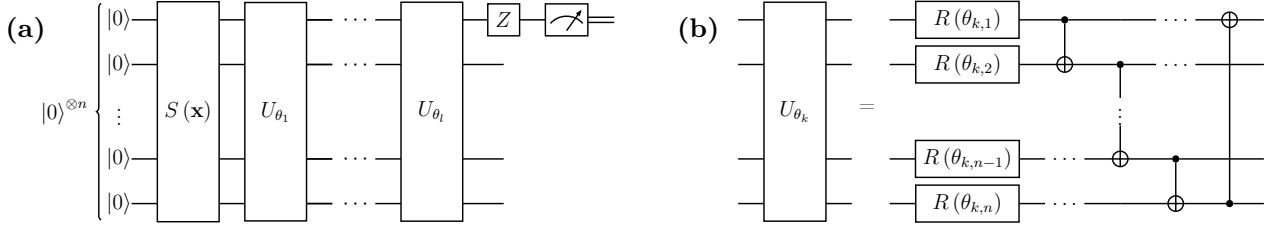
3.5 Kvantmekanisk variationell klassificerare

VQC:n implementerades och testades genom att simulera kvantkretsar med mjukvarubiblioteket PennyLane [40]. Två typer av tillståndsförberedelse testades på VQC:n: amplitud- och vinkeltillståndsförberedelse. De kombinationer som testades kan ses i tabell 3.2.

Tabell 3.2: De kombinationer av tillståndsförberedelser och antal kvantbitar som testades för VQC:n. För mer information om datamängderna se tabell 3.1. Notera att kombinationen ad hoc 2 och amplitud aldrig testas då detta skulle kräva en kvantbit, vilket ansatsen inte stödjer då CNOT-grindarna kräver minst två kvantbitar.

Datamängd	Tillståndsförberedelse	Antal kvantbitar
Iris	Amplitud	2
	Vinkel	4
Iris PCA	Vinkel	2
Cancer	Amplitud	5
Cancer PCA	Vinkel	5
Forest	Amplitud	6
Forest PCA	Vinkel	6
Ad hoc 2	Vinkel	2
Ad hoc 3	Amplitud	2
	Vinkel	3

VQC:n implementerades enligt kretsen i figur 3.1a. Samtliga kvantbitar initierades till $|0\rangle$ och passerade sedan genom en tillståndsförberedelsekrets S , där en klassisk datapunkt $\mathbf{x}$ kodades in genom antingen amplitud- eller vinkeltillståndsförberedelse, se figur C.1a respektive figur C.1b. Därefter passerades l lagerkretsar vilka var parametriserade och kunde tränas. Sist passerade den första kvantbiten genom en "Pauli Z"-grind och en mätning togs.



Figur 3.1: Kretsen som implementerades för VQC:n i (a) samt en förklaring av kretsen som utgjorde ett lager U_{θ_k} för $k = 1, \dots, l$ i (b). I ett lager U_{θ_k} tillämpades först en rotationsoperator på varje kvantbit och sedan sammanflätades de med hjälp av CNOT-grindar.

I figur 3.1b appliceras en rotation kring de tre kartesiska axlarna för varje kvantbit med avseende på en parameter $\theta_{k,m}$, $k = 1, \dots, l$, $m = 1, \dots, n$, där

$$\theta_{k,m} = \begin{pmatrix} \omega_{k,m,x} \\ \omega_{k,m,y} \\ \omega_{k,m,z} \end{pmatrix}. \quad (3.4)$$

Operatören kan representeras av matrisen

$$R(\theta_{k,m}) = R_x(\omega_{k,m,x}) R_y(\omega_{k,m,y}) R_z(\omega_{k,m,z}), \quad (3.5)$$

där $R_x(\omega_{k,m,x})$ är operatören som roterar kvanttillståndet med en vinkel $\omega_{k,m,x}$ kring x -axeln på Blochs-fären, vilken kan ses i figur 2.4. R_y och R_z definieras analogt. För respektive matris se tabell B.1.

För en märkt datapunkt $\mathbf{x}_i \in T$ med tillhörande y_i beräknades en skattning $\hat{y}_i$ enligt

$$\hat{y}_i = \text{sgn}(b + \langle q_1 | \sigma_z | q_1 \rangle), \quad (3.6)$$

där b är en bias och $|q_1\rangle$ är första kvantbiten efter den genomgått tillståndsförberedelse och lagerkrets, det vill säga $|q_1 \dots q_n\rangle = U_{\theta_l} \dots U_{\theta_1} S(\mathbf{x}_i) |0\rangle^{\otimes n}$. Kostnadsfunktionen i (2.26) kunde därefter beräknas och minimeras med NMM med steglängd $\eta = 0.1$ och $\mu = 0.9$. För att förbättra körtiden användes även en variant av en så kallad [41] stokastisk sjunkgradientmetod, där gradienten i (2.29) inte beräknas med

avseende på hela träningsmängden utan enbart ett stickprov. I denna rapport var storleken på detta stickprov fem för samtliga körningar för VQC:n. Stickprovet utfördes genom att ta fem punkter från träningsmängden på måfå. Tidigare hade ett frö såts i slumpalsgeneratoren för att få reproducerbara mätningar.

Ursprungsvärdet för biasen var $b = 0$ och ursprungsvikterna drogs pseudoslumpmässigt enligt $\omega \sim \mathcal{N}(0, 0.01^2)$ där återigen ett frö såts i slumpalsgeneratoren för att få reproducerbara mätningar.

Noggrannhet och kostnadsfunktion hos algoritmen varierade med antalet lager. Av denna anledning jämfördes egenskaperna när antalet lager varierades. Detta gjordes genom att successivt köra algoritmen med $1, 2, \dots, 10$ lager. För samtliga lager testades både amplitud- och vinkeltillståndsförberedelse med 100 iterationer.

Det gjordes även tester på en riktig kvantdator från IBMQ [42] för datamängderna iris och iris PCA. På grund av att körtiden var mycket lång för VQC:n gjordes träningen via simulering på en klassisk dator. Därefter utfördes en klassificering med de optimerade parametrarna på en riktig kvantdator. Algoritmen kördes med de antal lager som gav den högsta noggrannheten från simuleringarna. De kvantdatorer som användes var `ibmq_bemel` och `ibmq_lima`.

3.6 Kvantmekanisk beräkning av kärnfunktion

Konceptuellt är QKE:n lik en SVM förutom att kärnfunktionen beräknas med kvantkretsar. Dessa kvantkretsar simulerades genom implementering i mjukvarubiblioteken PennyLane [43] och Qiskit [44] med `statevector_simulator` [45], [46]. Klassificeringen testades med en klassisk SVM i sklearn [38]. Fyra typer av egenskapsavbildningar testades på QKE:n. Dessa var amplitud-, vinkel, Z- och ZZ-tillståndsförberedelse, se figur C.1. De olika kombinationer av datamängder, tillståndsförberedelse och antal kvantbitar som testades visas i tabell 3.2 och 3.3. Det gjordes även tester på en riktig kvantdator från IBMQ [42] med Z- och ZZ-tillståndsförberedelse för iris och ZZ-tillståndsförberedelse för ad hoc 2, då dessa tillståndsförberedelser var enkla att implementera på IBMQ:s kvantdatorer. De kvantdatorer som användes var `ibmq_manila` och `ibmq_santiago`.

Tabell 3.3: Några kombinationer av de antal kvantbitar som testades för QKE:n. Antal kvantbitar för samtliga datamängder gäller för både Z- och ZZ-tillståndsförberedelse. För mer information om datamängderna se tabell 3.1.

Datamängd	Antal kvantbitar
Iris	4
Cancer PCA	5
Forest PCA	6
Ad hoc 2	2
Ad hoc 3	3

Kretsen som implementerades för QKE:n med n kvantbitar och l lager visas i figur 2.11. Samtliga kvantbitar initierades till $|0\rangle$ och passerade först genom en egenskapsavbildande krets ϕ , se figur C.1, där en klassisk datapunkt $\mathbf{x}_i$ avbildades genom antingen amplitud-, vinkel-, Z- eller ZZ-tillståndsförberedelse l gånger. Därefter passerade kvantbitarna genom en identisk uppsättning av hermiteskt konjugerade egenskapsavbildade kretsar $\phi^\dagger$ där en annan klassisk datapunkt $\mathbf{x}_j$ avbildades l gånger.

Likt VQC:n användes rotationsgrindar för QKE-kretsen för varje kvantbit, detta förklaras i mer detalj i avsnitt 3.5. Den stora skillnaden mot VQC:n är att vinklarna som förs in i rotationsgrindarna inte beror på varierande parametrar, utan kommer istället från värdena på datapunkterna $\mathbf{x}_i$ och $\mathbf{x}_j$ i radianer. Genom mätningsgrindarna fås $K(\mathbf{x}_i, \mathbf{x}_j)$ enligt (2.31).

Amplitud- och vinkeltillståndsförberedelse implementerades med de inbyggda funktionerna `AmplitudeEmbedding` och `AngleEmbedding` i PennyLane [43] medan Z- och ZZ-tillståndsförberedelse med de in-

byggda funktionerna `ZFeatureMap` och `ZZFeatureMap` i Qiskit [44]. Kärnfunktionen beräknades genom att köra kretsen för alla par av datapunkter $\mathbf{x}_i, \mathbf{x}_j$ och användes därefter på samma sätt i en SVM med funktionen `SVC` i `sklearn` [39], vilket beskrivs i avsnitt 3.3.

4 Resultat

Här presenteras de noggrannheter och körtider som har erhållits från algoritmernas testkörningar. Resultaten för SVM:en kommer först att presenteras, följt av VQC:n och sist QKE:n.

4.1 Stödvektormaskin

Resultatet från SVM-testerna redovisas i form av noggrannheten för varje test, vilket ses i tabell 4.1. Se tabell D.1 för optimala parametervärden för SVM:en. Notera i tabellen att om medelvärdet av noggrannheten är nära 1 och standardavvikelsen är låg så tyder det på att algoritmen var bra anpassad till datan, medan om medelvärdet är nära 0,5 och standardavvikelsen är hög så var algoritmen dåligt anpassad.

Tabell 4.1: Resultaten för SVM:en med korsvalidering då $k = 10$. Noggrannhet beskrivs som medelvärde $\pm$ standardavvikelsen.

Kärnfunktion Datamängd	Linjär	Polynomiell	Gaussisk	Sigmoid
Iris	1,000 $\pm$ 0,000	1,000 $\pm$ 0,000	1,000 $\pm$ 0,000	0,933 $\pm$ 0,133
Cancer	0,933 $\pm$ 0,067	0,813 $\pm$ 0,098	0,953 $\pm$ 0,052	0,873 $\pm$ 0,096
Forest	0,767 $\pm$ 0,095	0,693 $\pm$ 0,033	0,813 $\pm$ 0,05	0,693 $\pm$ 0,033
Ad hoc 2	0,567 $\pm$ 0,076	0,560 $\pm$ 0,068	0,837 $\pm$ 0,081	0,603 $\pm$ 0,081

4.2 Kvantmekanisk variationell klassificerare

VQC:n testades för kombinationer av datamängder och tillståndsförberedelser. Noggrannheten och kostnaden presenteras i tabellerna nedan, där kostnaden beräknats med avseende på hela datamängden och noggrannheten med avseende på testmängden. I varje fall undersöktes olika antal lager för kretsen och bara de lager som gav högst noggrannhet anges här. Resterande resultat finns i figur F.1 och figur F.2.

4.2.1 Simuleringsresultat

Antalet kvantbitar varierades beroende på tillståndsförberedelsen, datamängden samt om datamängden är reducerad med hjälp av PCA. Resultatet för dessa tester presenteras som noggrannheten a i tabell 4.2. Precis som för SVM:en bedöms algoritmen vara bra anpassad till datan då noggrannhet är nära 1 och standardavvikelsen är låg, och dåligt anpassad då noggrannhet är nära 0,5 och standardavvikelsen hög.

Tabell 4.2: Resultat för VQC:n med korsvalidering då $k = 10$. Kostnad och noggrannhet beskrivs som medelvärde $\pm$ standardavvikelse. Antal lager upp till 10 testades och de resultat som anges är det antal som hade lägst kostnad efter 100 iterationer.

Datamängd	Tillståndsförberedelse	Antal lager	Kostnad	Noggrannhet
Iris	Amplitud	3	$0,403 \pm 0,016$	$1,000 \pm 0,000$
	Vinkel	7	$0,017 \pm 0,001$	$1,000 \pm 0,000$
Iris PCA	Vinkel	10	$0,020 \pm 0,004$	$1,000 \pm 0,000$
Cancer	Amplitud	1	$0,813 \pm 0,017$	$0,671 \pm 0,079$
Cancer PCA	Vinkel	8	$0,947 \pm 0,011$	$0,600 \pm 0,092$
Forest	Amplitud	1	$1,009 \pm 0,013$	$0,498 \pm 0,061$
Forest PCA	Vinkel	7	$1,004 \pm 0,017$	$0,524 \pm 0,088$
Ad hoc 2	Vinkel	7	$0,268 \pm 0,015$	$0,498 \pm 0,053$
Ad hoc 3	Amplitud	5	$0,301 \pm 0,020$	$0,492 \pm 0,045$
	Vinkel	7	$0,272 \pm 0,015$	$0,498 \pm 0,046$

4.2.2 IBMQ-resultat

Några datamängder testades på IBM:s kvantdatorer med amplitud- eller vinkeltillståndsförberedelse. Noggrannhet och specifikationer för dessa tester redogörs för i tabell 4.3. Notera att noggrannheten för dessa resultat är mestadels lägre men ändå mycket nära de resultat som erhöles för motsvarande simulerade testeter i tabell 4.2.

Tabell 4.3: Resultat för VQC:n på IBMQ:s kvantdatorer med korsvalidering då $k = 10$. Kostnad och noggrannhet beskrivs som medelvärde $\pm$ standardavvikelse.

Datamängd	Kvantdator	Tillståndsförberedelse	Antal lager	Kostnad	Noggrannhet
Iris	IBMQ Belem	Amplitud	3	$0,869 \pm 0,025$	$0,810 \pm 0,179$
	IBMQ Lima	Vinkel	7	$0,631 \pm 0,015$	$0,980 \pm 0,042$
Iris PCA	IBMQ Belem	Vinkel	10	$0,625 \pm 0,055$	$0,920 \pm 0,103$

4.3 Kvantmekanisk kärnfunktionsuppskattare

För QKE:n togs resultat både genom att simulera kvantkretsarna med mjukvarubiblioteken PennyLane och Qiskit, samt genom att köra kretsen på en riktig kvantdator från IBMQ. Resultaten presenteras i tabellerna nedan.

4.3.1 Simuleringsresultat

QKE:n testades för flera olika kombinationer av datamängder och tillståndsförberedelser. Antalet kvantbitar varierades också beroende på tillståndsförberedelsen, datamängden samt om datamängden är reducerad med hjälp av PCA. Resultatet för dessa tester presenteras som noggrannheten a i tabell 4.4. Precis som för SVM:en och VQC:n bedömes algoritmen vara bra anpassad till datamängden då noggrannhet är nära 1 och standardavvikelsen är låg, och dåligt anpassad för hög standardavvikelse och noggrannhet nära 0,5.

Tabell 4.4: Resultat för QKE:n med korsvalidering då $k = 10$. Noggrannhet beskrivs som medelvärde $\pm$ standardavvikelse. Alla mätningar som gjordes syns inte här, de resterande återfinns i tabell F.1.

Datamängd	Tillståndsförberedelse	Antal kvantbitar	Noggrannhet
Iris	Amplitud	2	$1,000 \pm 0,000$
	Vinkel	4	$1,000 \pm 0,000$
	ZZ		$0,886 \pm 0,107$
Cancer**	Amplitud	5	$0,840 \pm 0,067$
Cancer PCA**	Vinkel	5	$0,949 \pm 0,033$
	ZZ		$0,537 \pm 0,084$
Forest	Amplitud	6	$0,646 \pm 0,046$
Forest PCA	Vinkel	6	$0,797 \pm 0,067$
	ZZ		$0,520 \pm 0,076$
Ad hoc 2*	ZZ	2	$1,000 \pm 0,000$
	Z		$0,770 \pm 0,037$
Ad hoc 2	Amplitud	1	$0,434 \pm 0,056$
Ad hoc 3*	ZZ	3	$0,790 \pm 0,055$
	Z		$0,696 \pm 0,039$
Ad hoc 3	Vinkel	3	$0,507 \pm 0,052$

* Tillståndsförberedelsen har två lager.

** Datamängden skalas från $(-1, 1)$ och normaliseras.

4.3.2 IBMQ-resultat

För några datamängder testades QKE:n på IBMQ's kvantdatorer. Noggrannhet och specifikationer för dessa tester redogörs för i tabell 4.5. Notera att noggrannheten för dessa resultat är lägre men ändå mycket nära de resultat som erhöles för motsvarande simulerade tester i tabell 4.4.

Tabell 4.5: Resultaten för QKE:n på IBMQ:s kvantdatorer med korsvalidering då $k = 10$. Noggrannhet beskrivs som medelvärde $\pm$ standardavvikelse.

Datamängd	Storlek	Kvantdator	Tillståndsförberedelse	Antal kvantbitar	Noggrannhet
Iris	100	IBMQ Santiago	Z	4	$1,000 \pm 0,000$
		IBMQ Manila			$1,000 \pm 0,000$
		IBMQ Santiago	ZZ		$0,871 \pm 0,119$
Ad hoc 2*	50	IBMQ Santiago	ZZ	2	$0,986 \pm 0,043$
	100				$0,993 \pm 0,021$

* Tillståndsförberedelsen har två lager.

4.4 Jämförelse

En sammanställd jämförelse mellan alla resultaten för noggrannhet och tid presenteras här.

4.4.1 Noggrannhet

De bästa noggrannheterna från alla resultat på alla datamängder presenteras i tabell 4.6. De fetstilta resultaten är de som är högst för varje datamängd. Notera att alla algoritmer kunde uppnå den maximala noggrannheten 1 för datamängden iris.

Tabell 4.6: Den bästa uppnådda noggrannheten för varje algoritm på varje datamängd, där det bästa resultatet för en given datamängd är fetstilt.

Datamängd	Noggrannhet		
	SVM	VQC	QKE
Iris	1,000 $\pm$ 0,000	1,000 $\pm$ 0,000	1,000 $\pm$ 0,000
Cancer	0,953 $\pm$ 0,053	0,671 $\pm$ 0,079	0,949 $\pm$ 0,033
Forest	0,813 $\pm$ 0,050	0,524 $\pm$ 0,088	0,797 $\pm$ 0,067
Ad hoc	0,837 $\pm$ 0,081	0,498 $\pm$ 0,053	1,000 $\pm$ 0,000

4.4.2 Tid

Körtiderna för de olika algoritmerna att klassificera iris-datat mängden presenteras via storleksordning i tabell 4.7. Tiden för de kvantmekaniska algoritmerna är tagna från testerna på riktiga kvantdatorer.

Tabell 4.7: Antal gånger kretsen behöver köras igenom, tiden per körning samt den totala körtidens storleksordning visas för klassificering av iris-datat mängden. SVM:en kördes på en klassisk dator, och VQC:n och QKE:n kördes på en riktig kvantdator.

Algoritm	Antal kretskörningar	Körtid per krets [s]	Total körtid [s]
SVM	ej	ej	$\sim 10^0$
QKE	18	≈ 120	$\sim 10^3$
VQC	205 000	≈ 5	$\sim 10^6$

5 Diskussion

I detta avsnitt kommer maskininlärningsalgoritmerna att tolkas och jämföras utgående från de erhållna resultaten som presenteras i avsnitt 4 och bilaga F. Det kommer även att diskuteras möjliga utökningar av rapporten för framtida projekt samt implikationer för användning av större kvantdatorer.

5.1 Tolkning av resultat

Här presenteras en diskussion om de olika algoritmernas individuella prestationer.

5.1.1 Stödvektormaskin

Bland de olika kärnfunktionerna som användes för SVM:en syns det att den Gaussiska kärnfunktionen ger mest fördelaktig klassificering för alla datamängder. Detta kan ha flera orsaker. Den Gaussiska kärnfunktionen utför beräkningar i ett oändligdimensionellt egenskapsrum R^∞ , till skillnad från exempelvis den polynomiella som räknar i rummet $\mathbb{R}^{p \times N}$ där p är graden av polynomet [15]. Att egenskapsrummet för den Gaussiska kärnfunktionen är oändligdimensionellt bör i princip göra det enklare för den att separera datapunkter, då varje ny dimension utgör en ny axel. En annan orsak till funktionens överlägsenhet i dessa tester kan vara dess optimeringsparameter. Parametern γ i den Gaussiska kärnfunktionen, se avsnitt 3.3, kan väljas optimalt och kan då bland annat minimera effekterna av över- och underanpassning [47].

Notera dock att även om den Gaussiska kärnfunktionen presterar bäst här, betyder det inte att den gör det generellt. Principiellt lämpar sig olika kärnfunktioner för olika datamängder. Exempelvis visar [48] att en datamängd över patienter med COVID-19 att den polynomiella kärnfunktionen kan ge högst noggrannhet.

5.1.2 Kvantmekanisk variationell klassificerare

Från VQC-resultaten framgår att noggrannheten generellt är låg för alla datamängder, förutom iris. Det kan även antas att noggrannheterna för VQC:n inte kommer att förbättras signifikant om algoritmen tillåts köra fler iterationer. Alltså är de erhållna noggrannheterna de bästa möjliga. Detta baseras på att kostnadsfunktionerna, som ses i bilaga F, planar ut, alltså att gradienten asymptotiskt går mot noll. Detta implicerar att kostnadsfunktionen når ett minimum och inte kommer att minska markant vid fler iterationer och därmed kommer inte heller noggrannheten att förbättras. Anledningar till VQC:ns sämre prestanda diskuteras i avsnitt 5.2.2.

Genom att studera noggrannheterna som erhållits från samma datamängd med användningen av olika tillståndsförberedelser syns det att valet av tillståndsförberedelse endast har en liten påverkan på resultatet. Till exempel är noggrannheterna för de två cancer-fallen ganska lika då de endast skiljer på andra decimalen. Detta innebär att båda tillståndsförberedelserna är ungefär lika välanpassade för att hantera dessa datamängder. Dock är det mycket möjligt att det finns tillfällen då någon tillståndsförberedelse är bättre anpassad för en datamängd, enligt ett analogt resonemang till det i avsnitt 5.1.1.

5.1.3 Kvantmekanisk kärnfunktionsuppskattare och ad hoc

QKE:n gav det bästa resultatet för ad hoc då den klassificerade upp till 100 % med ZZ-tillståndsförberedelse. Jämförelsevis var den bästa klassificering av ad hoc för SVM:en runt 83,7 % och VQC:n gav endast slutgiltiga noggrannheter som var under 50 %, det vill säga sämre än slumpen. QKE:ns överlägsna klassificering för "ad hoc"-datamängden stämmer överens med tidigare studier [21] och antas bero på strukturen av mängden. Detta eftersom "ad hoc"-datamängdens struktur är noggrant konstruerad så att datan skall vara separerbar av just den icke-linjära egenskapsavbildning som QKE:n använder. Se [31] för mer ingående matematiska detaljer. Detta är ett ytterst smalt förhållande som datan måste följa, så det anses vara högst osannolikt att en verklig datamängd skulle följa just denna strikta matematiska definition. Alltså är QKE:n troligtvis inte bättre än SVM:en i praktiska tillämpningar. Dock skall det noteras att QKE:n ändå inte ligger långt bakom SVM:en i dessa resultat.

5.2 Jämförelse av algoritmerna

Här diskuteras jämförelserna mellan de olika algoritmerna.

5.2.1 Simuleringar och principalkomponentanalys

För vissa tillståndsförberedelser var antalet egenskaper så stort att principalkomponentanalys (PCA) behövde användas. Detta tar, i teorin, bort de egenskaper hos datan som har minst påverkan för klassificeringen. Principiellt kommer detta alltid att ha en negativ påverkan på den slutgiltiga noggrannheten, då information alltid försvinner. Det syns dock i resultaten att de PCA-reducerade resultaten fortfarande är mycket nära de icke-reducerade resultaten. Detta tyder på att PCA-metoden gör sitt jobb effektivt, med en försumbar påverkan på resultatet.

Ett begränsat antal körningar, huvudsakligen med iris-datamängden, utfördes både som simulering och på riktiga kvantdatorer. Då simuleringarna förutsätter ett idealt system utan brus, bör i princip en riktig kvantdator alltid ge sämre resultat. Det ses dock i de resultat som både simulerades och kördes på en kvantdator att noggrannheten är sämre men inte skiljer sig mycket alls. Detta beror troligtvis på att detta är väldigt små system där bara en liten mängd brus tillkommer. För diskussion om brus på större system se avsnitt 5.4.2. Det skall nämnas att dessa mätningar är få och således bör inga större slutsatser dras. Notera också att olika tester utfördes på olika kvantdatorer, vilket alltså kan vara en felkälla i dessa mätningar.

5.2.2 Noggrannhet

Baserat på resultaten gav VQC:n allmänt den sämsta noggrannheten för alla datamängder. Ett exempel på detta är de slutgiltiga noggrannheterna för bröstcancer-datamängden som erhållits för VQC:n. Dessa noggrannheter kommer generellt inte över 70 %, jämfört med SVM:en och QKE:n som kan ge en noggrannhet upp till cirka 95 %. Den stora prestationsskillnaden mellan VQC:n och QKE:n kan vara grundad i principerna hur algoritmerna är uppbyggda. QKE:n är uppbyggd på sådant sätt att den, till skillnad från VQC:n, garanterat kommer hitta den optimala modellen [19]. Ett liknande argument kan föras för att förklara VQC:ns prestationsskillnad mot SVM:en. Således är QKE:n och SVM:en snabbare och effektivare än VQC:n i dessa tester.

En fördel med VQC:n kommer dock in i algoritmstrukturens flexibilitet, då mer allmänna variationella kvantalgoritmer (eng. variational quantum algorithms) kan lösa många fler problem än bara klassificering. Notera också att VQC:ns resultat, med ett bättre val av ansats och startparametrar, möjligtvis skulle kunna närma sig QKE:ns och SVM:ens noggrannhet. En annan fördel för VQC:n är att den skalar bättre med antalet datapunkter än QKE:n [19]. Vid stora datamängder bör då VQC:n vara bättre lämpad än QKE:n, se avsnitt 5.2.3.

Utifrån noggrannheten för de olika algoritmerna framgår det att SVM:en generellt ger bäst resultat, förutom för ad hoc då QKE:n hade högre noggrannhet med vissa tillståndsförberedelser. Att SVM:en allmänt ger den bästa noggrannheten skulle kunna bero på datamängdernas uppbyggnad, då "ad hoc"-fallet visar på att datamängdens struktur påverkar vilken algoritm som ger bäst klassificering. Men den klassiska algoritmens allmänna överlägsenhet är förmodligen att vänta i en sådan här jämförelse. Oavsett hur mycket potential kvantalgoritmer har är klassiska maskininlärningsalgoritmer fortfarande mycket effektiva och välutvecklade för problemlösning i dagens samhälle. Tillkommande processer som tillståndsförberedelse samt osäkerheter i mätningar och brus i kvantkretsar gör det svårt för kvantalgoritmer att konkurrera med en klassisk algoritm. Detta arbete tyder alltså på att kvantmekaniska klassificeringsalgoritmer konsekvent presterar sämre än klassiska klassificeringsalgoritmer. Så istället för att anpassa kvantalgoritmer till klassiska problem som redan kan lösas anser vi, likt många forskare [21], [49], [50], att det troligtvis finns mer potential i att försöka lösa problem som klassiska algoritmer inte kan lösa, exempelvis beräkningar på stora kvantdatamängder.

5.2.3 Tid

Det ses i avsnitt 4.4.2 att SVM:en är snabbast, QKE:n är näst snabbast och VQC:n är långsammast. Överlägsenheten hos SVM:en antas bero på liknande resonemang som tas upp under 5.2.2, då klassiska metoder är effektiva och välutvecklade.

Eftersom kärnfunktionen i QKE:n kräver parvisa beräkningar för alla datapunkter kommer antalet beräkningar att skalas sämre än för VQC:n som inte kräver det [19]. Denna effekt visar sig däremot mer tydligt vid större datamängder än de som användes i detta arbete. Därför är det rimligt att QKE:n presterar bättre än VQC:n i detta arbete.

För VQC:n testades 100 iterationer vilket var betydligt fler än vad som krävdes för att nå 100 % noggrannhet. Även om den hade nått 100 % noggrannhet efter en iteration hade den dock ändå varit 10 gånger långsammare än QKE:n och 10^4 gånger långsammare än SVM:en.

5.3 Möjliga utökningar av arbetet

Under arbetets gång framgick möjliga utökningar som skulle kunna utföras. Många av dessa utökningar utfördes inte, främst på grund av tidsbegränsningar. Dessa presenteras istället här som förslag till kommande arbeten i rekommenderad prioriteringsordning, med den högst prioriterade först.

Det här arbetet har fokuserat på jämförelsen mellan en klassisk algoritm, SVM, och två kvantalgo-

ritmer, VQC och QKE. En vidare undersökning skulle kunna studera fler algoritmer i jämförelsen, för ett bredare perspektiv på resultatet. I synnerhet hade en jämförelse med ett neuronnät [51] varit intressant, då denna algoritm har många konceptuella likheter till VQC:n. Eftersom båda algoritmer optimeras i iterationer och har olika antal lager hade iterationsberoende träningskurvor och lagerberoende träningskurvor kunnat jämföras mellan dessa, se bilaga F.

Vid undersökningen av VQC:n varierades ett flertal parametrar, men ansatsen varierades inte. Vid noggrant testande och optimerande av ansatsen hade VQC:n möjligtvis kunnat prestera bättre än i denna rapports resultat.

Den huvudsakliga parametern som betraktas för jämförelsen i denna rapport är noggrannheten. För en mer utförlig jämförelse skulle fler parametrar kunna användas för att jämföra algoritmerna. Exempelvis påstår en rapport [48] att utöver noggrannheten bör även känsligheten och specificiteten betraktas. Känsligheten kan ge ett mått på algoritmens förmåga att klassificera positiva punkter korrekt, medan specificiteten anger motsvarande för negativa punkter [32]. Dessa kan vara viktiga att betrakta i exempelvis sjukdomar där det är viktigare att klassificera sjuka rätt än att klassificera friska rätt.

Fyra stycken datamängder av olika karaktär användes i denna jämförelse. Fler datamängder hade kunnat användas i detta arbete för att ge en mer fullskalig jämförelse. Detta innefattar att testa fler klassiska datamängder. Dock är det kanske ännu mer intressant att också jämföra med kvantdatamängder som från början befinner sig i kvanttillstånd [49]. Dessa datamängder skulle troligtvis gynna kvantalgoritmer då tillståndsförberedelsen förenklas.

Endast fyra tillståndsförberedelser undersöks i denna rapport. Z- och ZZ-tillståndsförberedelse testades dock aldrig för VQC:n, vilket möjligtvis kunde ha gett algoritmen bättre resultat i vissa fall. Fler tillståndsförberedelser än dessa kan även undersökas, dock kan nya krav på datamängden då tillkomma vilket alltså möjligtvis kräver modifikationer i datamängdsvalet. En ytterligare sak som kan testas angående detta är upprepade tillståndsförberedelser, vilket i princip innebär att samma tillståndsförberedelse utförs flera gånger i flera lager [19]. Detta utfördes i liten utsträckning för två lager med QKE:n, men skulle kunna undersökas i större utsträckning.

För SVM:en valdes fyra kärnfunktioner för undersökningen. Detta skulle kunna utökas ytterligare för att hitta en kärnfunktion som möjligtvis presterar bättre än den Gaussiska. Detta hade möjligtvis kunnat förbättra SVM:ens resultat för bland annat "ad hoc"-datamängden, men förväntas troligtvis inte komma i nivå med QKE:n för denna datamängd.

5.4 Implikationer för stora kvantdatorer

I de simulerade testerna som utförts har antalet kvantbitar inte överstigit 7. Många projekt idag, som WACQT-projektet på Chalmers [52], siktar på att bygga stora kvantdatorer med $100 \leq$ kvantbitar. Det är därför relevant att diskutera förutsättningarna för stora system och hur resultaten bör tolkas utifrån ett sådant perspektiv.

Det bör noteras att en reproduktion av de beräkningar som gjorts i detta arbete med fler kvantbitar troligtvis inte hade gett bättre resultat. Datamängderna och algoritmerna som användes har anpassats för att inte kräva många kvantbitar. En utökning av antalet kvantbitar med samma metod hade alltså inte nödvändigtvis bidragit till någon förbättring. Istället krävs troligtvis nya datamängder och kanske också nya algoritmer för att få en förbättring av resultaten gentemot den klassiska algoritmen.

De problem som presenteras nedan visar dock på att det uppstår problem som försvårar beräkningar på stora kvantdatorer. Det som detta innebär är att resultaten i denna rapport är begränsade, då de inte representerar många svårigheter och utmaningar som uppstår i forskning idag kring realisering av maskininlärning på kvantdatorer.

5.4.1 ”Karga platåer”-problemet

I teorin är det alltid möjligt att optimera en VQC med tillräckligt många lager [18], dock finns det fall där kostnadsfunktionen är olämplig för träning. Karga platåer (eng. Barren Plateaus) är områden där gradienten av kostnadsfunktionen, för parameteroptimering av θ , är väldigt nära noll [53]. Eftersom lutningen utnyttjas i sökandet efter ett minimum kan detta innebära att sökandet blir långsamt och dyrt på beräkningskraft. Situationen förvärras ytterligare med de brusiga NISQ-datorer som används idag, då optimering på karga platåer blir känsligt för brus.

Matematiskt kan det visas [18] att ”karga platåer”-problemet ofta framträder då system med stora antal kvantbitar används. Ju större och slumpmässigare kretsen är, desto lägre blir gradienten av kostnadsfunktionen och karga platåer uppkommer. Detta innebär alltså en utmaning då algoritmer som VQC:n används på stora NISQ-kvantdatorer med många kvantbitar.

5.4.2 Brus för stora kvantkretsar

Vid användning av exempelvis amplitudtillståndsförberedelse krävs endast $\log_2 N$ kvantbitar, vilket till vis del löser problemet som presenteras i avsnitt 5.4.1. Dock krävs det ofta fortfarande i dessa fall desto fler logiska grindar, där amplitudtillståndsförberedelse vanligen kräver $\mathcal{O}(\exp N)$ logiska grindar i utbyte [18].

Problemet som uppstår är att NISQ-erans kvantdatorer har ett brus som påverkar mätningarna mer ju fler grindar som finns i systemet [54]. Således har stora kvantkretsar idag problem med detta då resultatsignalerna på dessa system blir minimala jämfört med bruset. Detta problem kan möjligtvis lösas i framtiden med en utveckling bort från NISQ, men i nuläget är detta en stor problematik för stora kvantdatorer.

6 Samhälleliga och etiska aspekter

Maskininlärning är en brett tillämpbar metod för att effektivisera och producera teknik som kan användas både i vardag och i forskning. En vardaglig tillämpning är algoritmer för självkörande bilar, som utvecklas av exempelvis Tesla [5], där en dator studerar information i dess omgivning och efter det beräknar en säker körrutin. Denna applikation kan tillåta människor med körförbud att fortfarande vara rörliga i samhället på ett säkert sätt.

Ett juridiskt problem som kan uppstå när maskiner i större utsträckning utför arbetsuppgifter självständigt, är vem som har ansvar för maskinen. Särskilt tydligt blir detta dilemma från perspektivet från det tidigare exemplet om självkörande bilar. Om en automatiserad bil skulle hamna i en olycka, uppstår frågan om det är passageraren eller tillverkaren som är ansvarig. En statlig rapport från Storbritannien [55] kom fram till att passagerare i självkörande fordon bör vara befriade från ansvar vid en eventuell olycka, men att så ej är fallet i dagsläget. En undersökning av svensk lag hävdar istället att passagerare i självkörande bilar är, och bör vara, ansvariga för fordonets framförande [56]. Undersökningen argumenterar för att så bör fallet vara så länge fordonet har förmågan att styras av en mänsklig förare även om bilen körde sig själv vid olyckan. Saknar bilen möjlighet att styras av en person, vilket kan bli fallet i framtiden, kan det snarare liknas vid till exempel en buss där passagerarna saknar ansvar för fordonets framfart.

Den maskininlärning som studerats i denna rapport försöker att lösa ett klassificeringsproblem. Den typen av maskininlärning används redan i sjukvården för att identifiera och kategorisera sjukdomar, som cancer. Utveckling inom detta område skulle därför kunna göra att fler patienters sjukdomar upptäcks snabbare och således ökar chansen att de får adekvat vård [57].

Tillämpningarna för maskininlärning på just kvantdatorer är i nuläget okänt, då området är tidigt i

sitt utvecklingsstadie. Vissa är optimistiska gällande teknikens kommande användningsområden och tror att kvantdatorer kommer innebära en förbättring i kapacitet, träningstid och -effektivitet [8], [58]. Å andra sidan påpekar bland annat en framgångsrik forskare inom området, Maria Schuld, att kvantdatorteknologin befinner sig i ett för tidigt stadie för att kunna utvärdera om kvantmaskininlärning har signifikanta fördelar gentemot klassiska metoder [19], [59]. Det är alltså svårt idag att bedöma hur stor påverkan för maskininlärning som denna rapport kan komma att ha.

En mer avancerad framställning av maskininlärning som möjligen kan bli implementerad på en kvantdator är artificiell intelligens (AI) som har blivit mycket kontroversiellt i många nyttjanden. Ett sådant område där AI:s integrering diskuteras är inom ansiktsgenkänning. Detta kan vara praktiskt och gynnsamt inom flera användningsområden, exempelvis som ett alternativ till lösenord för mobiltelefoner. Det finns dock risk att denna teknik skulle kunna användas för att spionera och göra intrång på folks anonymitet [6].

En annan kontroversiell debatt är frågan om artificiell generell intelligens (AGI), det vill säga AI som har förmågan att hantera alla typer av problem och anpassa sig till nya situationer [60]. Risken som vissa ser är att AI skulle bli så avancerad att människan inte längre kan kontrollera den. Till de som har förespråkats oro hör bland annat Bill Gates [61], Stephen Hawking [62] och Elon Musk [63]. Det finns dock de som är skeptiska till den risk som AI skulle innebära av olika anledningar. Vissa anser att AGI befinner sig för långt bort i utvecklingen för att innebära en stor risk i nuläget [64], [65]. Andra anser att utvecklingen mot okontrollerbarhet kommer att ske tillräckligt sakta för att människan skulle kunna stoppa den, om de så skulle önska [66]. Mark Zuckerberg anser att fördelarna med AI väger tyngre än de eventuella nackdelarna och förespråkar en snabbare utveckling [67].

Hos forskare och de som arbetar inom området är åsikterna inte lika starka om AGI. Det flesta ser det som en potentiell fara och anser att man bör vara vaksam, men att risken är låg och att det snarare är de stora konsekvenserna som motiverar förebyggande åtgärder. En enkät riktad till forskare från 2014 visar att endast 31 % av de anser att existensen av AGI skulle vara negativt för mänskligheten, men 76 % anser att det skulle ge upphov till negativa effekter [68]. En annan enkät från 2017 säger att endast 15 % anser att AGI skulle vara dåligt för mänskligheten men samtidigt anser 70 % att man bör ta hänsyn till risken med AGI och 47 % anser att man borde lägga mer resurser på att hantera problemet än i dagsläget [69]. Alltså har forskare en viss vaksamhet gällande AGI, men anser fortfarande överlag att det har potential att vara mer positivt än negativt.

Det bör noteras att maskininlärningsalgoritmerna i detta arbete användes på en relativt låg skala. Arbetet baseras i hög grad på existerande forskning när det kommer till utformning av algoritmer och kommer därför inte på kort sikt möjliggöra för algoritmer som inte hade varit tillgängliga utan arbetet. Resultaten förväntas istället kunna användas som rådgivande för vad efterkommande forskning bör fokuseras på, vilket innebär att senare arbete i större utsträckning blir ansvariga för att hantera de etiska implikationerna.

Enligt det som diskuterats ovan, och med de användningsområden som maskininlärning har, så finns potentiellt en stor positiv effekt av att utveckla maskininlärning, men även möjliga negativa konsekvenser. Maskininlärning är i grund och botten en form av beräkningskraft och det skulle därför vara i princip omöjligt att utveckla den så att det endast kunde användas för positiva syften. Totaleffekten av rapporten bedöms dock vara positivt för mänskligheten, då dess potentiella effekt kommer att vara att hjälpa till med att driva utvecklingen framåt, men den kommer inte själv leda till utveckling av nya och farliga typer av AI.

7 Slutsats

Från de resultat som presenterats ses inga fördelar med att använda kvantdatorer för att förbättra maskininlärning på verkliga datamängder. Beräkningstiden är betydligt längre för kvantalgoritmerna

på samtliga testade datamängder gentemot den klassiska algoritmen. Noggrannheterna är på liknande vis sämst för VQC:n och oftast bäst för SVM:en, utom för den konstgjorda datamängden ad hoc då QKE:n fick en högre noggrannhet. Det är dock oklart om "ad hoc"-strukturen kan likna någon datamängd kopplad till ett praktiskt problem. Men även om det inte har påvisats några praktiska fördelar för användning av kvantalgoritmer, så kan det fortfarande finnas andra möjligheter i problem som maskininlärning på en klassisk dator inte kan behandla. Istället för att försöka hantera problem som redan går att lösa med en klassisk dator rekommenderas framtida studier inom detta område att fokusera på uppgifter som idag saknar lösning.

Referenser

- [1] A. Thamara, M. Elersy, A. Sherif, H. Hassan, O. Abdelsalam och K. H. Almotairi, "A Novel Classification of Machine Learning Applications in Healthcare," i *2021 3rd IEEE Middle East and North Africa COMMunications Conference (MENACOMM)*, 2021, s. 80–85. DOI: 10.1109/MENACOMM50742.2021.9678232.
- [2] S. E. Mudiyansele, P. H. D. Nguyen, M. S. Rajabi och R. Akhavan, "Automated Workers' Ergonomic Risk Assessment in Manual Material Handling Using sEMG Wearable Sensors and Machine Learning," *Electronics*, årg. 10, nr 20, s. 2558, okt. 2021. DOI: 10.3390/electronics10202558.
- [3] S. Firouzi, X. Wang och A. Totonyfardmotlagh, "Machine Learning Forecasting of Foreign Exchange Markets Trend Based on Order Flow and US Economic News," i *2021 7th Annual International Conference on Network and Information Systems for Computers (ICNISC)*, 2021, s. 640–645. DOI: 10.1109/ICNISC54316.2021.00121.
- [4] J. Shen och T. Wu, *Learning Inception Attention for Image Synthesis and Image Recognition*, 2021. DOI: 10.48550/ARXIV.2112.14804.
- [5] The Tesla Team. "All Tesla Cars Being Produced Now Have Full Self-Driving Hardware." (2016), Tillgänglig: <https://www.tesla.com/blog/all-tesla-cars-being-produced-now-have-full-self-driving-hardware> (hämtad 2022-01-27).
- [6] B. Schölkopf och A. J. Smola, *Learning with Kernels : Support Vector Machines, Regularization, Optimization, and Beyond*. Ser. Adaptive Computation and Machine Learning Ser. Cambridge: MIT Press, 2001, s. 25–28, 34–35, 45, 196–199, 221–222, ISBN: 9780262256933.
- [7] A. Matuschak och M. A. Nielsen. "Quantum Computing for the Very Curious." (2019), Tillgänglig: <https://quantum.country/qcvc>.
- [8] D. Peral, J. Cruz-Benito och F. J. G. Peñalvo, "Systematic Literature Review: Quantum Machine Learning and its applications," 2022. DOI: 10.48550/ARXIV.2201.04093.
- [9] J. Preskill, "Quantum Computing in the NISQ era and beyond," *Quantum*, årg. 2, s. 79, aug. 2018. DOI: 10.22331/q-2018-08-06-79.
- [10] T. Mitchell, *Machine Learning*, ser. McGraw-Hill series in computer science. New York, NY: McGraw-Hill Professional, mars 1997, s. 1–10, ISBN: 0070428077.
- [11] J. R. Koza, F. H. Bennett, D. Andre och M. A. Keane, "Automated Design of Both the Topology and Sizing of Analog Electrical Circuits Using Genetic Programming," i *Artificial Intelligence in Design '96*, J. S. Gero och F. Sudweeks, utg., Dordrecht: Springer Netherlands, 1996, s. 151–170, ISBN: 978-94-009-0279-4. DOI: 10.1007/978-94-009-0279-4_9.
- [12] IBM Cloud Education. "Overfitting." (2021), Tillgänglig: <https://www.ibm.com/cloud/learn/overfitting> (hämtad 2022-04-17).

- [13] C. Oswald, "2.8 The Bias-Variance Trade-off," i *Python 3 and Data Analytics - Pocket Primer*. Mercury Learning och Information, 2021.
- [14] C. Cortes och V. Vapnik, "Support-vector networks," *Machine Learning*, årg. 20, nr 3, s. 273–297, sept. 1995. DOI: 10.1007/bf00994018.
- [15] T. Hofmann, B. Schölkopf och A. J. Smola, "Kernel methods in machine learning," *The Annals of Statistics*, årg. 36, nr 3, juni 2008. DOI: 10.1214/009053607000000677.
- [16] D. J. Griffiths och D. F. Schroeter, *Introduction to quantum mechanics*. Cambridge: Cambridge University Press, 2018, ISBN: 9781107189638.
- [17] R. Johansson, *Grundläggande datorteknik*. Lund: Studentlitteratur, 2016, s. 46, ISBN: 978 - 9144076508.
- [18] M. Schuld och F. Petruccione, *Machine Learning with Quantum Computers*. Cham: Springer Nature Switzerland AG, 2021, s. 100–103, 137, 158–159, 197–201, ISBN: 3030830977. DOI: 10.1007/978-3-030-83098-4.
- [19] M. Schuld, *Supervised quantum machine learning models are kernel methods*, 2021. DOI: 10.48550/ARXIV.2101.11020. Tillgänglig: <https://arxiv.org/abs/2101.11020>.
- [20] M. Schuld, A. Bocharov, K. M. Svore och N. Wiebe, "Circuit-centric quantum classifiers," *Physical Review A*, årg. 101, nr 3, mars 2020. DOI: 10.1103/physreva.101.032308.
- [21] V. Havlíček, A. D. Córcoles, K. Temme m.fl., "Supervised learning with quantum-enhanced feature spaces," *Nature*, årg. 567, s. 209–212, 2019. DOI: 10.1038/s41586-019-0980-2.
- [22] N. Mishra. "Understanding the basics of measurements in Quantum Computation." (2019), Tillgänglig: <https://towardsdatascience.com/understanding-basics-of-measurements-in-quantum-computation-4c885879eba0> (hämtad 2022-03-27).
- [23] R. Osodo. "Building a Quantum Variational Classifier Using Real-World Data." (2021), Tillgänglig: <https://medium.com/qiskit/building-a-quantum-variational-classifier-using-real-world-data-809c59eb17c2> (hämtad 2022-04-13).
- [24] B. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Computational Mathematics and Mathematical Physics*, årg. 4, nr 5, s. 1–17, jan. 1964. DOI: 10.1016/0041-5553(64)90137-5.
- [25] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $O\left(\frac{1}{k^2}\right)$," *Doklady Akademii Nauk SSSR*, årg. 269, nr 3, s. 543–547, 1983, ISSN: 0002-3264.
- [26] Y. Bengio, N. Boulanger-Lewandowski och R. Pascanu, "Advances in optimizing recurrent networks.," *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, s. 8624–8628, 2013, ISSN: 978-1-4799-0356-6.
- [27] P. Rebentrost, M. Mohseni och S. Lloyd, "Quantum Support Vector Machine for Big Data Classification," *Phys. Rev. Lett.*, årg. 113, s. 130503, 13 sept. 2014. DOI: 10.1103/PhysRevLett.113.130503.

-
- [28] Y. Liu, S. Arunachalam och K. Temme, "A rigorous and robust quantum speed-up in supervised machine learning," *Nature Physics*, årg. 17, nr 9, s. 1013–1017, juli 2021. DOI: 10.1038/s41567-021-01287-z.
- [29] "Dataset loading utilities." (2022), Tillgänglig: <https://scikit-learn.org/stable/datasets.html#dataset-loading-utilities> (hämtad 2022-04-17).
- [30] "Datasets." (2020), Tillgänglig: <https://github.com/Qiskit/qiskit-aqua/tree/main/qiskit/ml/datasets> (hämtad 2022-04-17).
- [31] H.-Y. Huang, M. Broughton, M. Mohseni m. fl., "Power of data in quantum machine learning," *Nature Communications*, årg. 12, nr 1, maj 2021. DOI: 10.1038/s41467-021-22539-9.
- [32] M. Hossin och M. Sulaiman, "A Review on Evaluation Metrics for Data Classification Evaluations," *International Journal of Data Mining and Knowledge Management Process (IJDMP)*, årg. 5, nr 2, s. 1–11, 2019. DOI: 10.5281/zenodo.3557376.
- [33] T. Wong och P. Yeh, "Reliable Accuracy Estimates from k-Fold Cross Validation.," *IEEE Transactions on Knowledge and Data Engineering*, årg. 32, nr 8, s. 1586–1594, aug. 2020. DOI: 10.1109/TKDE.2019.2912815.
- [34] M. Ojala och G. G. Garriga, "Permutation Tests for Studying Classifier Performance," *Journal of Machine Learning Research*, årg. 11, nr 62, s. 1833–1863, 2010. Tillgänglig: <http://www.jmlr.org/papers/volume11/ojala10a/ojala10a.pdf> (hämtad 2022-03-31).
- [35] I. Goodfellow, Y. Bengio och A. Courville, *Deep Learning*. MIT Press, 2016, s. 120. Tillgänglig: <http://www.deeplearningbook.org> (hämtad 2022-04-10).
- [36] J. Rice, *Mathematical statistics and data analysis*, 3. utg. London, England: Thomson Learning, 2007, s. 202–214.
- [37] R. Kohavi, "A study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection," 1995. Tillgänglig: <http://ai.stanford.edu/~ronnyk/accEst.pdf>.
- [38] F. Pedregosa, G. Varoquaux, A. Gramfort m. fl., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, årg. 12, s. 2825–2830, 2011.
- [39] L. Buitinck, G. Louppe, M. Blondel m. fl., "API design for machine learning software: experiences from the scikit-learn project," i *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, s. 108–122.
- [40] PennyLane dev team, "Variational classifier," 2021. Tillgänglig: https://pennylane.ai/qml/demos/tutorial_variational_classifier.html (hämtad 2022-04-11).
- [41] L. Bottou och O. Bousquet, "The Tradeoffs of Large Scale Learning," i *Proceedings of the 20th International Conference on Neural Information Processing Systems*, ser. NIPS'07, Vancouver, British Columbia, Canada: Curran Associates Inc., 2007, s. 161–168, ISBN: 9781605603520.
- [42] IBM Cloud Education. "Real quantum computers. Right at your fingertips.," Tillgänglig: <https://quantum-computing.ibm.com/> (hämtad 2022-04-27).

-
- [43] V. Bergholm, J. Izaac, M. Schuld m. fl., "PennyLane: Automatic differentiation of hybrid quantum-classical computations," 2018.
- [44] M. S. ANIS, H. Abraham, AduOffei m. fl., *Qiskit: An Open-source Framework for Quantum Computing*, 2021. DOI: 10.5281/zenodo.6419747.
- [45] PennyLane dev team, "Kernel-based training of quantum models with scikit-learn," 2021. Tillgänglig: https://pennylane.ai/qml/demos/tutorial_kernel_based_training.html (hämtad 2022-02-03).
- [46] Qiskit Machine Learning Development Team, "Quantum Kernel Machine Learning," 2021. Tillgänglig: https://qiskit.org/documentation/machine-learning/tutorials/03_quantum_kernel.html (hämtad 2022-02-17).
- [47] scikit-learn developers. "Support Vector Machines." (2022), Tillgänglig: <https://scikit-learn.org/stable/modules/svm.html> (hämtad 2022-05-03).
- [48] A. F. Rochim, K. Widyaningrum och D. Eridani, "Performance Comparison of Support Vector Machine Kernel Functions in Classifying COVID-19 Sentiment," i *2021 4th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, 2021, s. 224–228. DOI: 10.1109/ISRITI54043.2021.9702845.
- [49] R. Blume-Kohout, S. Croke och M. Zwolak, "Quantum data gathering," *Sci. Rep.*, årg. 3, nr 1, dec. 2013.
- [50] M. Schuld och N. Killoran, *Is quantum advantage the right goal for quantum machine learning?* 2022. DOI: 10.48550/ARXIV.2203.01340.
- [51] I. N. da Silva, D. H. Spatti, R. A. Flauzino, L. H. B. Liboni och S. F. dos Reis Alves, *Artificial Neural Networks*. Springer International Publishing, 2017. DOI: 10.1007/978-3-319-43162-8. Tillgänglig: <https://doi.org/10.1007/978-3-319-43162-8>.
- [52] "WACQT | Wallenberg Centre for Quantum Technology." (2022), Tillgänglig: <https://www.chalmers.se/en/centres/wacqt/Pages/default.aspx> (hämtad 2022-04-14).
- [53] Z. Holmes, K. Sharma, M. Cerezo och P. J. Coles, "Connecting Ansatz Expressibility to Gradient Magnitudes and Barren Plateaus," *PRX Quantum*, årg. 3, nr 1, jan. 2022. DOI: 10.1103/prxquantum.3.010313.
- [54] H. Yetis och M. Karakoes, "Investigation of Noise Effects for Different Quantum Computing Architectures in IBM-Q at NISQ Level," i *2021 25th International Conference on Information Technology (IT)*, 2021, s. 1–4. DOI: 10.1109/IT51528.2021.9390130.
- [55] Law Commission, "Automated Vehicles: joint report," *Law Com*, nr 404, 2022.
- [56] G. Ahlgren, "Självkörande fordon - problematiken kring förare och straffansvar," Kandidatarb. Juridiska fakulteten, Lunds universitet, 2017. Tillgänglig: <https://lup.lub.lu.se/luur/download?func=downloadFile&recordId=8930219&fileId=8933775> (hämtad 2022-04-11).

- [57] T. Han, J. Zhu, X. Chen m.fl., "Application of artificial intelligence in a real-world research for predicting the risk of liver metastasis in T1 colorectal cancer," *Cancer Cell International*, årg. 22, nr 1, 2022. DOI: <https://doi.org/10.1186/s12935-021-02424-7>.
- [58] F. Phillipson, "Quantum Machine Learning: Benefits and Practical Examples," i *1st International Workshop on the QuANtum SoftWare Engineering & pRogramming*, (Talavera de la Reina, Spanien, 12 febr. 2020), M. Piattini, G. Peterssen, R. Perez-Castillo, J. L. Hevia och M. A. Serrano, utg., ser. CEUR Workshop proceedings, Aachen, 2020, s. 51–56. Tillgänglig: <http://ceur-ws.org/Vol-2561/paper5.pdf> (hämtad 2022-04-11).
- [59] M. Schuld, *Why measuring performance is our biggest blind spot in quantum machine learning*, 2022. Tillgänglig: <https://pennylane.ai/blog/2022/03/why-measuring-performance-is-our-biggest-blind-spot-in-quantum-machine-learning/> (hämtad 2022-04-11).
- [60] H. Shevlin, K. Vold, M. Crosby och M. Halina, "The limits of machine intelligence: Despite progress in machine intelligence, artificial general intelligence is still a major challenge," *EMBO reports*, årg. 20, nr 10, 2019. DOI: <https://doi.org/10.15252/embr.201949177>.
- [61] K. Rawlinson, "Microsoft's Bill Gates insists AI is a threat," *BBC*, 2015. Tillgänglig: <https://www.bbc.com/news/31047780> (hämtad 2022-04-11).
- [62] S. Hawking, S. Russell, M. Tegmark och F. Wilczek, "Stephen Hawking: 'Transcendence looks at the implications of artificial intelligence - but are we taking AI seriously enough?'" *Independent*, 2014. Tillgänglig: <https://www.independent.co.uk/news/science/stephen-hawking-transcendence-looks-at-the-implications-of-artificial-intelligence-but-are-we-taking-ai-seriously-enough-9313474.html> (hämtad 2022-04-11).
- [63] B. Gomez, "Elon Musk warned about a 'Terminator'-like AI apocalypse – Now he is building a Tesla Robot," *CNBC*, 2021. Tillgänglig: <https://www.cnbc.com/2021/08/24/elon-musk-warned-of-ai-apocalypsenow-hes-building-a-tesla-robot.html> (hämtad 2022-05-10).
- [64] R. Brooks, "artificial intelligence is a tool, not a threat," *rethinking robots*, 2014. Tillgänglig: <https://web.archive.org/web/20141112130954/http://www.rethinkrobotics.com/artificial-intelligence-tool-threat/> (hämtad 2022-04-11).
- [65] M. Shermer, "Artificial Intelligence Is Not a Threat—Yet," *Scientific American*, 2017. Tillgänglig: <https://www.scientificamerican.com/article/artificial-intelligence-is-not-a-threat-mdash-yet> (hämtad 2022-04-11).
- [66] M. More. "Singularity Meets Economy." (1998), Tillgänglig: <https://web.archive.org/web/20210225111414/http://mason.gmu.edu/~rhanson/vc.html#more> (hämtad 2022-03-27).
- [67] P. Bhardwaj, "Mark Zuckerberg responds to Elon Musk's paranoia about AI: 'AI is going to... help keep our communities safe.'," *Buisness Insider*, 2018. Tillgänglig: <https://www.businessinsider.com/mark-zuckerberg-shares-thoughts-elon-musks-ai-2018-5?r=US&IR=T> (hämtad 2022-04-11).
- [68] V. C. Müller och N. Bostrom, "Future progress in artificial intelligence: A survey of expert opinion," i *Fundamental Issues of Artificial Intelligence*, ser. Synthese Library, V. C. Müller, utg., Springer, 2016, s. 553–571.

- [69] K. Grace, J. Salvatier, A. Dafoe, B. Zhang och O. Evans, "Viewpoint: When Will AI Exceed Human Performance? Evidence from AI Experts," *J. Artif. Int. Res.*, årg. 62, nr 1, s. 729–754, 2018, issn: 1076-9757. DOI: 10.1613/jair.1.11222.

A Härledning från primal- till dualproblem

Genom att utgå från samma optimeringsproblem som i (2.5), det vill säga det så kallade primala optimeringsproblemet

$$\begin{aligned} \min_{\mathbf{w}, b \in \mathbb{R}} \tau(\mathbf{w}) &= \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{då } (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) y_i &\geq 1, \quad \forall i = 1, \dots, M, \end{aligned} \quad (\text{A.1})$$

kan från detta ett så kallat dualt optimeringsproblem konstrueras [6]. För att ta fram det används den så kallade Lagrangianen och tillhörande Lagrangemultiplikatorer $\alpha_i \geq 0$ enligt

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^M (\alpha_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) y_i - 1), \quad (\text{A.2})$$

där

$$\boldsymbol{\alpha} = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_M \end{pmatrix}. \quad (\text{A.3})$$

Lagrangianen är oberoende av variabler från det primala optimeringsproblemet, varpå dessa derivator försvinner

$$\frac{\partial L}{\partial b} = 0, \quad \frac{\partial L}{\partial \mathbf{w}} = 0. \quad (\text{A.4})$$

Detta innebär att villkoren för att maximera marginalen och optimera hyperplanet för fallet då $(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) y_i = 1$ ges av

$$\sum_{i=1}^M \alpha_i y_i = 0, \quad \mathbf{w} = \sum_{i=1}^M \alpha_i y_i \mathbf{x}_i. \quad (\text{A.5})$$

Genom att använda villkoren (A.5) tillsammans med (A.1), fås följande olikhet

$$\alpha_i ((\langle \mathbf{w}, \mathbf{x}_i \rangle + b) y_i - 1) \geq 0, \quad \forall i = 1, \dots, M. \quad (\text{A.6})$$

Det finns två möjliga fall som kan tillfredsställa (A.6). Det ena fallet är då $(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) y_i - 1 = 0$. Detta innebär att punkten $\mathbf{x}_i$ befinner sig på marginalen, dessa punkter kallas för stödvektorer. Det är stödvektorerna som används för att konstruera hyperplanet, vid detta fall sätts Lagrangemultiplikatorn $\alpha_i > 0$. Det andra fallet $(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) y_i - 1 \neq 0$ innebär att punkten $\mathbf{x}_i$ inte befinner sig på marginalen. I detta fall är punkten inte av intresse då den inte är en stödvektor och Lagrangemultiplikatorn sätts till $\alpha_i = 0$.

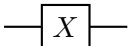
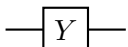
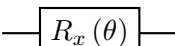
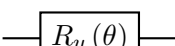
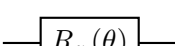
Genom att stoppa in uttrycken från (A.5) i Lagrangianen (A.2), fås dualproblemet som är en undre gräns för lösningen av det primala optimeringsproblemet [6]. Dualproblemet ges då av

$$\begin{aligned} \max_{\boldsymbol{\alpha} \in \mathbb{R}^M} L_d(\boldsymbol{\alpha}) &= \sum_{i=1}^M \alpha_i - \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ \text{då } \sum_i \alpha_i y_i &= 0, \quad \alpha_i \geq 0, \quad \forall i = 1, \dots, M, \end{aligned} \quad (\text{A.7})$$

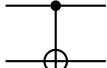
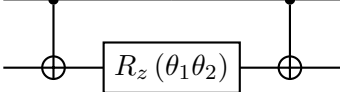
där målet nu är att finna α som maximerar L_d då summan av alla $\alpha_i y_i$ -termer är noll och alla α_i är icke-negativa [6]. Detta är samma uttryck som presenteras i (2.6).

B Kvantgrindar

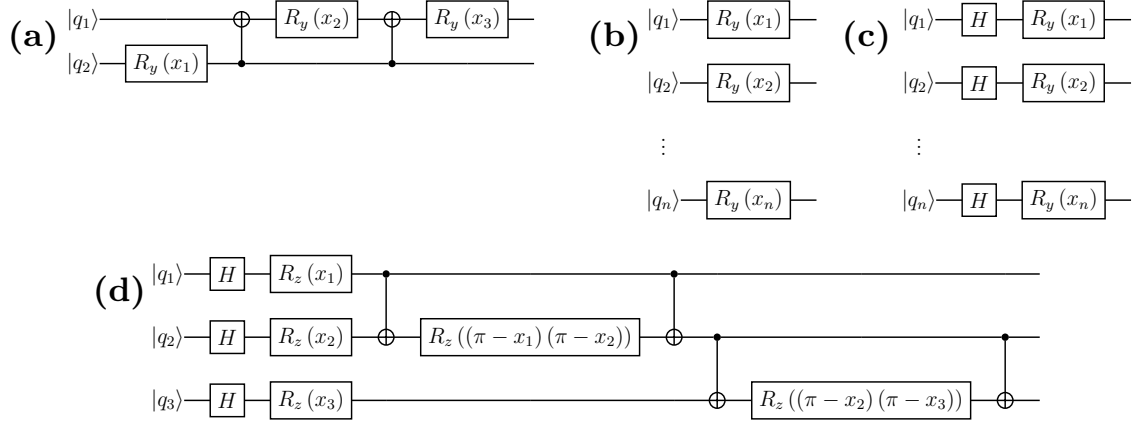
Tabell B.1: Tabell över enkvantbitsgrindar med kretsrepresentation och matrisrepresentation.

Namn	Kretsrepresentation	Matrisrepresentation
Hadamard		$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$
"Pauli X"		$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
"Pauli Y"		$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$
"Pauli Z"		$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$
X-rotation		$\begin{pmatrix} \cos \frac{\theta}{2} & -i \sin \frac{\theta}{2} \\ -i \sin \frac{\theta}{2} & \cos \frac{\theta}{2} \end{pmatrix}$
Y-rotation		$\begin{pmatrix} \cos \frac{\theta}{2} & -\sin \frac{\theta}{2} \\ \sin \frac{\theta}{2} & \cos \frac{\theta}{2} \end{pmatrix}$
Z-rotation		$\begin{pmatrix} \exp -i\frac{\theta}{2} & 0 \\ 0 & \exp i\frac{\theta}{2} \end{pmatrix}$

Tabell B.2: Tabell över tvåkvantbitsgrindar med kretsrepresentation och matrisrepresentation.

Namn	Kretsrepresentation	Matrisrepresentation
CNOT		$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$
ZZ		$\begin{pmatrix} e^{-i\frac{\theta_1\theta_2}{2}} & 0 & 0 & 0 \\ 0 & e^{i\frac{\theta_1\theta_2}{2}} & 0 & 0 \\ 0 & 0 & e^{i\frac{\theta_1\theta_2}{2}} & 1 \\ 0 & 0 & 0 & e^{-i\frac{\theta_1\theta_2}{2}} \end{pmatrix}$

C Kvantkretsar för tillståndsförberedelse



Figur C.1: Kretsar för tillståndsförberedelse. I (a) ses en amplitudtillståndsförberedande krets med två kvantbitar $|q_1\rangle$ och $|q_2\rangle$ tillsammans med en klassisk datapunkt med $N = 3$ egenskaper x_1, x_2 och x_3 . I (b) och (c) ses en vinkel- respektive Z-tillståndsförberedande krets med n kvantbitar $|q_1\rangle, |q_2\rangle, \dots, |q_n\rangle$ och en klassisk datapunkt med $N = n$ egenskaper $x_1, x_2, \dots, x_n$. I (d) ses en ZZ-tillståndsförberedande krets med tre kvantbitar $|q_1\rangle, |q_2\rangle$ och $|q_3\rangle$ och en klassisk datapunkt med $N = 3$ egenskaper x_1, x_2 och x_3 .

D Parametrar för klassisk stödvektormaskin

Tabell D.1: Parametrar som gav bäst resultat för den SVM:en, resultaten visas i tabell 4.1.

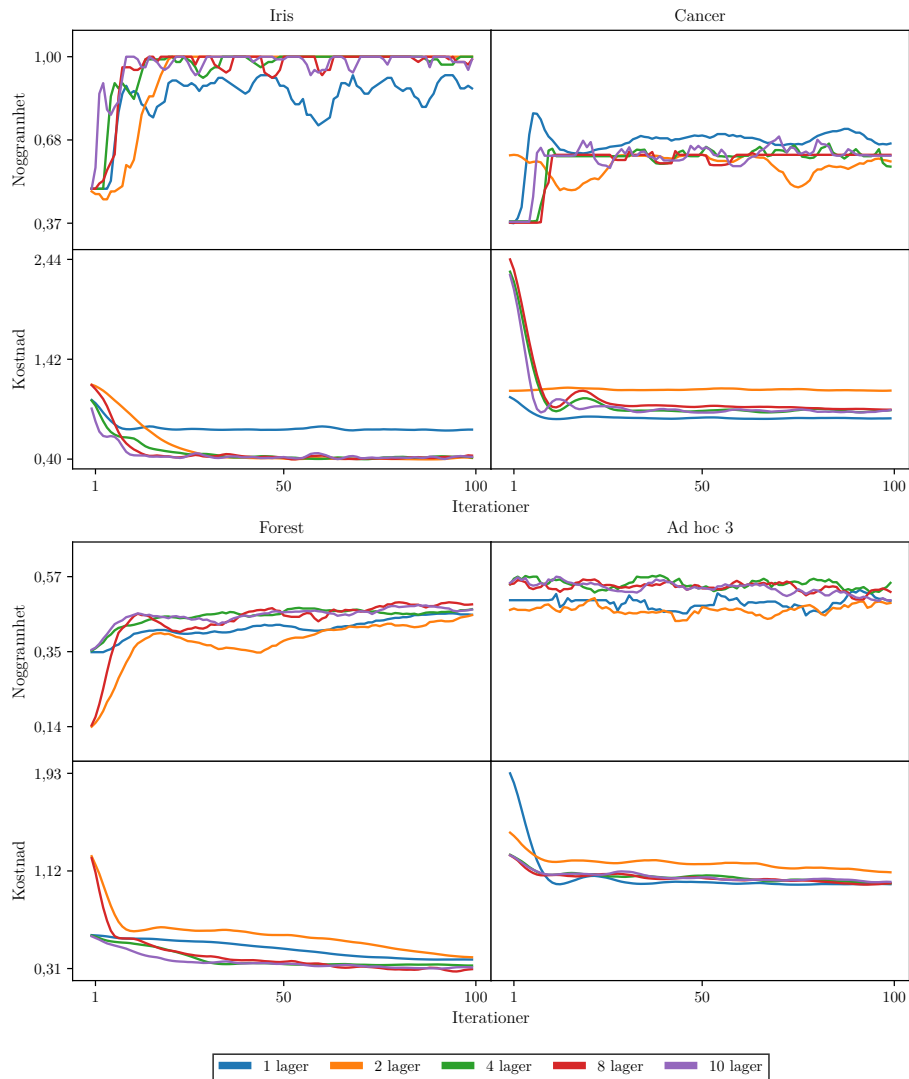
Datamängd	Linjär	Polynomiell	Gaussisk	Sigmoid
Iris	C=20	degree = 5	gamma='scale', C = 20	coef0 = -5
Bröstcancer	C=20	degree = 3	gamma= 10^{-4} , C = 10	coef0 = 2
Forest	C=1	degree = 7	gamma='scale', C = 200	coef0 = 10
Adhoc	C=1	degree = 7	gamma='scale', C = 200000	coef0 = -2

E Kod

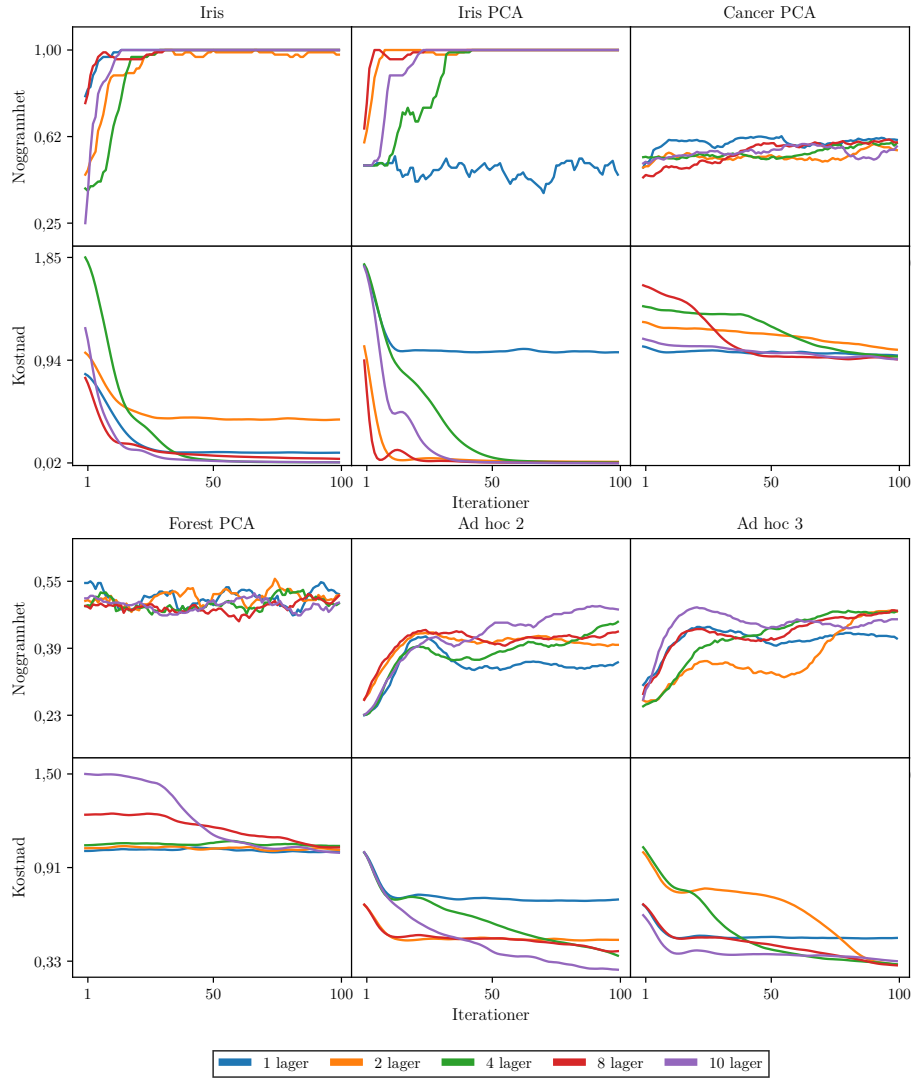
Kod som producerats under arbetet finns tillgänglig via

https://github.com/osclil/bachelor_thesis_qml/releases/tag/1.0.0

F Resultat för kvantmekanisk variationell klassificerare och kvantmekanisk kärnfunktionsuppskattare



Figur F.1: Noggrannhet och kostnad för VQC:n med amplitudtillståndsförberedelse i varje iteration under hundra iterationer. Antalet lager varierade vilket visas med de olika linjerna. Notera att kostnaden inte beräknats med avseende på bara träningsmängden utan med avseende på hela datamängden.



Figur F.2: Noggrannhet och kostnad för VQC:n med vinkeltillståndsförberedelse. Samma kommentarer gäller som i figur F.1.

Tabell F.1: Resultat för QKE:n med korsvalidering då $k = 10$. Noggrannhet beskrivs som medelvärde $\pm$ standardavvikelsen.

Datamängd	Tillståndsförberedelse	Antal kvantbiter	Noggrannhet
Iris	Z	4	$1,000 \pm 0,000$
Iris PCA	Vinkel	2	$1,000 \pm 0,000$
Cancer	Amplitud	5	$0,623 \pm 0,042$
Cancer PCA	Vinkel	5	$0,583 \pm 0,051$
Cancer PCA*	Z	5	$0,929 \pm 0,045$
Forest PCA	Vinkel	7	$0,714 \pm 0,054$
	Z	6	$0,466 \pm 0,080$
Ad hoc 2	ZZ	2	$0,906 \pm 0,022$
	Z		$0,789 \pm 0,049$
	Vinkel		$0,644 \pm 0,042$
Ad hoc 3	ZZ	3	$0,646 \pm 0,055$
	Z		$0,637 \pm 0,062$
	Amplitud		$0,540 \pm 0,067$

* Datamängden skalas från $(-1, 1)$ och normaliseras.

Institutionen för mikroteknologi och nanovetenskap

Chalmers tekniska högskola

Göteborg, Sverige

www.chalmers.se



CHALMERS