Universidade Estadual de Londrina

MATEUS KOMARCHESQUI

DETECÇÃO DE ANOMALIAS EM REDES DE COMPUTADORES UTILIZANDO COMPUTAÇÃO QUÂNTICA

LONDRINA
2023

MATEUS KOMARCHESQUI

DETECÇÃO DE ANOMALIAS EM REDES DE COMPUTADORES UTILIZANDO COMPUTAÇÃO QUÂNTICA

Dissertação apresentada ao Programa de Mestrado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Mario Lemes Proença Jr.

LONDRINA

2023

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema de Bibliotecas da UEL

Sobrenome, Nome.
Título do Trabalho : Subtitulo do Trabalho / Nome Sobrenome. - Londrina, 2017.
100 f. : il.

Orientador: Nome do Orientador Sobrenome do Orientador.
Coorientador: Nome Coorientador Sobrenome Coorientador.
Dissertação (Mestrado em Ciência da Computação) - Universidade Estadual de Londrina, Centro de Ciências Exatas, Programa de Pós-Graduação em Ciência da Computação, 2017.
Inclui bibliografia.

1. Assunto 1 - Tese. 2. Assunto 2 - Tese. 3. Assunto 3 - Tese. 4. Assunto 4 - Tese. I. Sobrenome do Orientador, Nome do Orientador. II. Sobrenome Coorientador, Nome Coorientador. III. Universidade Estadual de Londrina. Centro de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação. IV. Título.

MATEUS KOMARCHESQUI

DETECÇÃO DE ANOMALIAS EM REDES DE COMPUTADORES UTILIZANDO COMPUTAÇÃO QUÂNTICA

Dissertação apresentada ao Programa de Mestrado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Mestre em Ciência da Computação.

BANCA EXAMINADORA

Orientador: Prof. Dr. Mario Lemes Proença Jr.
Universidade Estadual de Londrina

Prof. Dr. Segundo Membro da Banca
Universidade/Instituição do Segundo Membro da Banca – Sigla instituição

---

Prof. Dr. Terceiro Membro da Banca
Universidade/Instituição do Terceiro Membro da Banca – Sigla instituição

---

Prof. Ms. Quarto Membro da Banca
Universidade/Instituição do Quarto Membro da Banca – Sigla instituição

Londrina, 24 de novembro de 2023.

Este trabalho é dedicado às crianças adultas
que, quando pequenas, sonharam em se
tornar cientistas.

AGRADECIMENTOS

"Não vos amoldeis às estruturas deste mundo, mas transformai-vos pela renovação da mente, a fim de distinguir qual é a vontade de Deus: o que é bom, o que Lhe é agradável, o que é perfeito.
(Bíblia Sagrada, Romanos 12, 2)"

KOMARCHESQUI, M.. **Detecção de anomalias em redes de computadores utilizando Computação Quântica**. 2023. 45f. Dissertação (Mestrado em Ciência da Computação) – Universidade Estadual de Londrina, Londrina, 2023.

RESUMO

Dada a ampla aplicação das redes de computadores e o potencial de exposição de dispositivos e dados, o que pode servir como catalisador para ataques, o investimento na pesquisa e desenvolvimento de métodos para assegurar a segurança nesse meio faz-se primordial. Com esse objetivo, os Sistemas de Detecção de Intrusão, como proposto neste trabalho, monitoram o tráfego da rede e determinam em tempo satisfatoriamente próximo ao real se dado fluxo de dados é considerado normal ou anômalo, detectando possíveis ameaças. Ao longo dos anos, diversas implementações de Sistemas de Detecção de Intrusão de Rede utilizaram Aprendizado de Máquina no âmbito da computação clássica. Mesmo com modelos complexos de Aprendizado Profundo, a discussão acerca de intrusão de rede continua em aberto. Tendo isso em vista, almeja-se com este trabalho estudar e implementar um modelo de Aprendizado de Máquina Híbrido clássico-quântico, mostrando as particularidades desse novo paradigma e explorando sua viabilidade no contexto proposto.

Incluir resultados

Palavras-chave: Segurança de Redes. Detecção de Anomalias. Detecção de Intrusão. Computação Quântica.

KOMARCHESQUI, M.. **Anomaly detection in computer networks using Quantum Computing**. 2023. 45p. Master's Thesis (Master in Science in Computer Science) – State University of Londrina, Londrina, 2023.

ABSTRACT

Given the widespread use of computer networks and the potential exposure of devices and data, which can serve as catalysts for attacks, investment in research and development of methods to ensure security in this environment is crucial. With this goal in mind, Intrusion Detection Systems, as proposed in this work, monitor network traffic and determine in a satisfactorily close-to-real-time whether a given data flow is considered normal or anomalous, detecting possible threats. Over the years, various implementations of Network Intrusion Detection Systems have utilized Machine Learning within the scope of classical computing. Even with complex Deep Learning models, the discussion regarding network intrusion remains open. Keeping this in mind, the aim of this work is to study and implement a classic-quantum Hybrid Machine Learning model, showcasing the peculiarities of this new paradigm and exploring its feasibility in the proposed context.

Keywords: Network security. Anomaly Detection. Intrusion detection. Quantum Computing.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

NIDS	Network Intrusion Detection System
SDN	Software Defined Network

SUMÁRIO

1 INTRODUÇÃO

Com a democratização do acesso às redes e aos computadores pessoais, sejam *Desktops*, telefones celulares, *gadgets* inteligentes, entre outros, praticamente todos os serviços do cotidiano traçaram uma relação inextricável com a rede. Seu uso é evidenciado desde a navegação na internet, envio de mensagens, acesso a entretenimento até ensino a distância, reuniões virtuais e *e-commerce* [1], [2].

A fim de estabelecer a conexão desses diversos dispositivos que acessam a rede de computadores, uma infraestrutura complexa formada por ativos de rede deve ser gerenciada de maneira eficiente [3], [4]. Entretanto, esses equipamentos possuem interfaces e fabricantes distintos, o que dificulta sua integração. Dessa maneira, utiliza-se a chamada Rede Definida por Software, do inglês *Software Defined Network* (SDN), onde definem-se os planos de dados e de controle. Centraliza-se, assim, todo o controle da rede num único dispositivo chamado controlador, programado em alto nível visando definir as políticas e lógica de controle da rede [4]. Sendo assim, os dispositivos físicos do plano de dados (roteadores e *switches*) apenas encaminham os pacotes segundo o protocolo *Openflow* e a lógica do controlador [4], [5].

Visto essa alta adesão ao uso da rede e, consequentemente, a exposição de dispositivos e usuários, ataques com diversas motivações ocorrem [6]. Ataques causam ou aproveitam-se de falhas, comprometendo a integridade, disponibilidade e confidencialidade do que trafega ou é armazenado em rede, podendo causar perdas substanciais e irreparáveis [7]. Essas falhas podem surgir fisicamente, por software ou por intervenção humana [6].

O comportamento da rede é alterado uma vez que uma falha ocorre ou é explorada por um ataque, gerando uma anormalidade no fluxo de pacotes. Essa anormalidade é denominada anomalia. Ataques, por sua vez, são atividades anômalas promovidas por agentes maliciosos mal-intencionados [8].

Uma abordagem para a detecção desses ataques amplamente difundida no meio científico é o chamado Sistema de Detecção de Intrusão de Rede, do inglês *Network Intrusion Detection System* (NIDS) [9], [8], [10]. Um NIDS observa o tráfego da rede a procura de indícios de comportamento anômalo (todo comportamento que destoa do perfil considerado normal tendo em vista dados históricos), alertando os administradores da rede, os quais tomam as medidas cabíveis para mitigar o problema. Esses sistemas são implementados utilizando modelos de Aprendizado de Máquina, podendo variar desde Aprendizado Profundo [11], [12], até algoritmos mais simples denominados de Aprendizado Raso [13], [14].

Antes até da invenção o computador pessoal, Peter Shor propôs um algoritmo quântico capaz de fatorar números grandes de maneira exponencialmente mais veloz que os computadores tradicionais [15]. Apesar da proposta ser interessante e inovadora, dela falou-se pouco, uma vez que não havia sequer projeto de um computador quântico, muito menos a possibilidade de implementar esse algoritmo [16]. Nos últimos anos, no entanto, a cena da computação quântica mudou. Empresas como *Google* e *IBM* têm investido em larga escala no desenvolvimento desses dispositivos.

Tendo em vista a grande capacidade computacional e potencial supremacia quântica sobre o paradigma tradicional, a computação quântica tem despertado um crescente interesse na comunidade científica [17], [18]. O tópico foi inundado de questionamentos, desde a sobrevivência dos computadores clássicos como conhecemos até a segurança da criptografia contemporânea. Dessa maneira, a fim de tirar proveito dos benefícios dessa nova abordagem, modelos de Aprendizado de Máquina com aperfeiçoamento quântico, o *Quantum Enhanced Machine Learning*, têm sido empregados para diversas tarefas [19], [20], [21], inclusive a detecção de intrusão [22], [23], [24].

Até mesmo com o uso de abordagens clássicas mais complexas de Aprendizado Profundo como *Generative Adversarial Network* (GAN) [25], *Long Short-Term Memory* (LSTM) [26] e *Convolutional Neural Network* (CNN) [27], o problema de detecção de intrusão continua uma discussão em aberto [8]. Motivado por esse fato, este trabalho propõe-se a estudar um modelo de aprendizado de máquina híbrido clássico-quântico no contexto de NIDS.

O restante deste documento está organizado da seguinte maneira: são expostos os conceitos teóricos e os métodos necessários para o desenvolvimento do estudo no capítulo 2. No capítulo 3 é apresentada a teoria sobre Computação Quântica. Em seguida, no capítulo 4, faz-se uma revisão do estado da arte por meio dos trabalhos relacionados a este. No capítulo 5 um estudo de caso é descrito. Finalmente, no capítulo 6, são discutidas as conclusões.

2 FUNDAMENTAÇÃO TEÓRICA-METODOLÓGICA

2.1 Redes Definidas por Software

A rede é a base para a comunicação entre dispositivos. Essa comunicação, porém, só pode ocorrer com a presença de uma infraestrutura capaz de direcionar os dados de um nó X para um nó Y ou um conjunto de nós Z. Essa infraestrutura é genericamente organizada em três planos de funcionalidade, verticalmente integrados [28].

Os planos de funcionalidade são [28], [29], [30]:

- Plano de Encaminhamento: também chamado de plano de dados, nesse plano ficam localizados os dispositivos de rede responsáveis por encaminhar o tráfego de rede, como roteadores e *switches*. Esses equipamentos seguem regras definidas pelo plano de controle.
- Plano de Controle: essa subdivisão define como o tráfego de rede deve ser manipulado. As regras de encaminhamento e lógicas de controle são definidas por esse plano.
- Plano de Gerenciamento: também denominado de plano de aplicação, é o plano onde os gerentes de rede definem as regras que serão empregadas no plano de controle.

A arquitetura convencional das redes de computadores implementa os planos de funcionalidade de maneira colapsada, ou seja, o plano de encaminhamento e de controle ficam dispostos em uma única interface em cada equipamento de maneira individual. Dessa maneira, cada dispositivo possui sua interface de programação a qual o gerente de rede irá configurar as lógicas de encaminhamento e políticas [31]. Fabricantes diferentes possuem interfaces de programação distintas, o que aumenta significativamente o custo de administração da rede e inviabiliza um gerenciamento centralizado [30].

Uma *Software Defined Network* (SDN) por sua vez, centraliza o controle da rede em um único ponto, o controlador, separando-o do plano de encaminhamento, delimitado pelos dispositivos que realizam a tarefa de direcionar os pacotes de dados (roteadores e *switches*) [29]. Sendo assim, o controlador é responsável por implantar as políticas da rede por meio do protocolo aberto *OpenFlow* pela interface *southbound* [30].

O controlador é delimitado por duas interfaces de programação: *northbound* e *southbound* [29], [30]. A *southbound* é a responsável por providenciar a interface de comunicação entre o plano de controle e o plano de encaminhamento, configurando os dispositivos deste plano. Essa é chamada, por esse motivo, de cérebro da rede ou sistema

operacional da rede. A *northbound*, por sua vez, oferece a interface de programação para o desenvolvimento de aplicações como monitoramento da rede, gerenciamento de políticas, engenharia de tráfego, aplicação de detecção e mitigação de intrusão, entre outros [29].

A Rede Definida por Software é de crucial importância para a gerência de rede uma vez que permite centralizar a lógica de encaminhamento, oferecendo uma abstração da rede para os aplicativos que nela serão implantados, promovendo maior flexibilidade e adaptabilidade da rede às possíveis demandas [29], [31]. Os administradores de rede podem definir políticas de serviço que direcionam o tráfego para funções específicas, como análise de tráfego, otimização de largura de banda e prevenção de ameaças, tudo isso de forma coordenada e automatizada [28], [30]. Para este trabalho, o uso de *SDN* permite o desenvolvimento de um Sistema de Detecção de Intrusão centralizado que analisa o tráfego da rede e gera alertas aos administradores sobre possíveis ameaças.

2.2 Anomalias de Rede

Tendo em vista que o fluxo de dados de uma rede é reflexo da interação dos dispositivos sob seu domínio, os pacotes que nela trafegam tendem a seguir um padrão [32], chamado de *baseline* [33]. Esse padrão pode ser observado ao coletar-se dados da rede durante um período de tempo [32], [33], [34]. Aqueles pacotes de dados que destoam do padrão são considerados anômalos [32], [34], [35].

Existem dois grupos abrangentes os quais as anomalias de rede podem ser categorizadas [32]:

- Problemas de Falhas e Performance da rede: esse grupo compreende falhas de servidores de arquivos, paginação pela rede, tempestades de *broadcast*, "nós tagarelas", congestão transitória e erros em protocolos.
- Problemas de Segurança: ataques como *Denial of Service* ou intrusões de rede são exemplos dessa categoria. Com esses, partes cruciais da rede podem acabar desativadas ou usuários legítimos deixados em *starvation*. De toda forma, um enorme volume de tráfego ocorre nessa categoria de anomalias.

É de suma importância que o gerente de rede possa identificar e mitigar problemas de segurança a fim de garantir a integridade, disponibilidade e confidencialidade do que trafega ou é armazenado em rede. Dessa maneira, Sistemas de Detecção de Intrusão são amplamente implementados para Redes Definidas por *Software*.

2.3 Detecção de Anomalias de Rede

Os Sistemas de Detecção de Intrusão de Rede, *Network Intrusion Detection System* (NIDS), são uma das soluções mais aceitas para a detecção de anomalias de rede no meio científico. Sua função é reportar qualquer tráfego com traços anormais ao administrador de rede [36], funcionando como um complemento para o *firewall*.

Os ataques estão em constante evolução e concepção [37], dificultando sua detecção. Dessa maneira, além da agilidade na detecção de uma anomalia [3], o NIDS deve ser capaz de identificar anomalias nunca antes vistas. Para isso, existem duas principais implementações de NIDS, categorizadas de acordo com a estratégia utilizada para a detecção [36], [38], [39]. Essas são:

- Baseado em Assinatura: Armazena em um banco de dados os padrões de anomalias já conhecidos e tenta sempre identificar algum desses padrões no tráfego de rede [37]. Devido ao fato de gerar alertas apenas quando uma anomalia for detectada, esse método gera menos falsos positivos [8] e é mais eficiente em identificar anomalias conhecidas. No entanto, anomalias não conhecidas não são detectadas [39].

- Baseado em Anomalia: Constrói um modelo que representa o tráfego normal da rede utilizando dados históricos [8]. O NIDS compara de forma constante o comportamento atual do tráfego da rede com o comportamento considerado normal o qual foi treinado. Isso leva à detecção de anomalias sempre que o comportamento atual difere do esperado como padrão. Essa abordagem possibilita a identificação de anomalias que ainda não foram observadas [8], [39].

A abordagem mais utilizada na literatura é a Baseada em Anomalia e é a que será implementada neste trabalho, visto que é mais flexível na detecção de novas ameaças.

Desde 2004, o grupo de pesquisa ORION da Universidade Estadual de Londrina tem contribuído para o estudo e implementação de Sistemas de Detecção de Intrusão eficientes e flexíveis. Diversos trabalhos foram publicados pelo grupo ao longo dos anos e citados por outros autores acerca da segurança em redes, desde uma revisão detalhada da literatura e da fundamentação teórica [8] até variadas abordagens para detecção de anomalias na abordagem computacional clássica [40], [41], [42], [43], [27], [26], [25], [44], [33], [45]. Além disso, trabalhos como [46] e [47] contribuem de maneira direta para o gerenciamento da rede, sendo cruciais para a área do conhecimento. De maneira geral, a comunidade científica nos últimos anos tem amplamente utilizado modelos de aprendizado de máquina para implementar esses sistemas.

2.4 Aprendizado de Máquina

O Aprendizado de Máquina, *Machine Learning* (ML), é uma subárea da Inteligência Artificial, *Artificial Inteligence* (AI). Essa solução computacional utiliza um conjunto de dados base para treinar um modelo que tem por objetivo ser capaz de fazer previsões sobre os dados que o alimentam [48], distintos dos utilizados no treinamento. O *Machine Learning* vem para suprir a necessidade computacional de automatizar a análise e interpretação de grandes volumes de dados complexos, tornando possível identificar padrões e relações que seriam difíceis ou impossíveis de serem percebidos por meio de métodos tradicionais [49].

Pode-se exemplificar a superioridade do ML por meio da aplicação apresentada no artigo [50]. Nele, é exposto o problema: identificar e classificar vida animal capturada por armadilhas fotográficas. Utilizando um modelo treinado com imagens de vida animal, foi possível automatizar o processo de identificação e classificação de novas fotografias com uma precisão notável, tarefa que seria impossível de ser realizada com algoritmos tradicionais devido à complexidade e variabilidade das imagens.

O aprendizado pode ser classificado em três categorias principais, cujas aplicações estão diretamente relacionadas ao conjunto de dados de treinamento [51]: supervisionado 2.4.1, semi-supervisionado 2.4.2 e não-supervisionado 2.4.3 [49], [52], [53].

2.4.1 Aprendizado Supervisionado

O aprendizado supervisionado utiliza dados onde cada observação X possui um rótulo Y relacionado capaz de descrevê-lo [54]. Dessa maneira, o objetivo do modelo que utiliza o aprendizado supervisionado é mapear uma observação de entrada X para um Y' satisfatoriamente próximo de Y [49].

Essa abordagem pode ser implementada utilizando diversos algoritmo tanto de classificação quanto de regressão [55]. Neste trabalho um modelo híbrido clássico-quântico supervisionado será estudado e implementado para classificar o tráfego de uma rede de computadores como normal ou anômalo.

2.4.2 Aprendizado Semi-supervisionado

Visto que a etapa de rotulação dos dados é custosa e grande parte das observações não carregam consigo rótulos [49], a aplicação do aprendizado semi-supervisionado pode ser interessante, uma vez que usa dados que contenham rótulos juntamente aos dados que não os contém no processo de treinamento [54].

Uma analogia para a melhor compreensão dessa abordagem é utilizada no livro [49], comparando o aprendizado do ser humano ao aprendizado semi-supervisionado:

> The way we learn is similar to the process of semi-supervised learning. A child is supplied with
>
> 1. Unlabeled data provided by the environment. The surroundings of a child are full of unlabeled data in the beginning.
> 2. Labeled data from the supervisor. For example, a father teaches his children about the names (labels) of objects by pointing toward them and uttering their names.

2.4.3 Aprendizado Não-supervisionado

O aprendizado não-supervisionado faz uso de conjuntos de dados sem rotulação. O objetivo dessa abordagem é encontrar estruturas escondidas nos dados. Uma variedade de motivos podem levar os dados a não possuírem rotulação, desde o alto custo e trabalho agregado à rotulação manual até a natureza intrínseca não-rotulada dos dados [49].

Essa abordagem é amplamente utilizada em sistemas de recomendação [56], por exemplo, que permeiam a internet seja em propagandas baseadas em interesses pessoais, sugestões de produtos em plataformas de comércio eletrônico, ou recomendações de conteúdo em serviços de *streaming*.

2.4.4 Aprendizado Raso

Um sistema de aprendizado é um programa de computador capaz de tomar decisões com base no conhecimento que tem acumulado de casos resolvidos no passado. Sendo assim, o Aprendizado Raso é todo sistema que toma decisões com base no conhecimento acumulado de maneira que não busca imitar o raciocínio humano [57]. Na sua forma primordial, o aprendizado de máquina é raso [58]. Diversas técnicas podem ser empregadas para fazer com que uma máquina aprenda, desde métodos matemáticos complexos até busca sistemática de um grande número de possibilidades [57].

Mesmo nos dias atuais, com a existência de modelos de aprendizado de máquina que buscam a cognição humana por meio de neurônios artificiais, o uso de modelos de aprendizado raso se faz presente [58]. Modelos como *Support Vector Machines*, *AdaBoost*, *Random Forest*, entre diversos outros são amplamente utilizados em sistemas comerciais [58] e comparados com modelos *mainstream* complexos e recentes [59], [60].

Este trabalho tem o intuito de estudar e implementar um modelo de *Support Vector Machine* (SVM) híbrido de computação clássica e quântica. Assim como em [60], estudo recente que otimiza um modelo de SVM para a implementação de Sistema de Detecção de Intrusão, este trabalho busca aproveitar o potencial do aprendizado raso aliado à computação quântica para implementar um NIDS.

2.4.4.1 Support Vector Machine

O modelo *Support Vector Machine* (SVM) é um método de aprendizado de máquina raso baseado na teoria de aprendizado estatístico proposto por Vapnik *et al.* na década de 1990 [61], [62], [63]. Esse possui uma base teórica sólida respaldada em princípios matemáticos bem estabelecidos, dessa maneira seu comportamento é compreensível e de certa maneira previsível. Pela forma como funciona, oferece intrinsecamente uma solução esparsa, onde pontos mais importantes são elencados para definir os limites de decisão do modelo, e otimização global, buscando uma solução globalmente ótima [61], [62].

A forma mais básica do SVM é aplicável apenas ao aprendizado supervisionado, no entanto, existem extensões do modelo para trabalhar com os outros tipos de aprendizado [61]. Apesar de ser uma abordagem antiga e mais simples, esse modelo é amplamente utilizado em áreas como categorização de texto, reconhecimento de escrita manual, detecção de faces, análise de dados de expressão genética e várias outras [63], inclusive detecção de anomalias e NIDS [59], [60].

O princípio básico do SVM é encontrar um hiperplano H capaz de separar espacialmente as observações de dados segundo suas classes [61]. Outros dois hiperplanos H_1 e H_2, paralelos a H, conterão os pontos de amostra, pontos esses localizados à "beira"do intervalo de cada uma das classes. Esses pontos são chamados de *support vectors*. Dessa maneira, a distância entre os hiperplanos H e H_1 ou H e H_2 é chamada de margem. O objetivo do modelo quanto à margem pode ser diferente conforme a separabilidade dos dados e/ou a escolha da função de manipulação da dimensionalidade dos dados [61], [63], [64], como exposto a seguir.

Os dados de entrada do modelo podem ser lineares ou não-lineares. No caso de dados lineares, a dimensionalidade das observações é mantida, uma vez que um hiperplano linear é capaz de realizar a separação dos mesmos [64]. A figura 1 exemplifica o caso de dados lineares e separáveis em um plano bi-dimensional. As classes são representadas pelas cores distintas e os pontos contidos nas retas H_1 e H_2 são os *support vectors* de suas respectivas classes [63], [64].

É possível notar que os pontos azuis e vermelhos estão agrupados de maneira consistente, sem os chamados *outliers* (dados significantemente diferentes dos outros dados da mesma classe). Dessa maneira, o plano de separação H é capaz de dividir os dados de forma satisfatória e os dados de teste serão, de maneira geral, bem classificados. Nesse caso é feito o uso da margem rígida, *Hard Margin*, onde os vetores de suporte são os pontos absolutamente mais à margem. Essa abordagem é chamada também de *maximal margin*, uma vez que a distância entre H e H_1/H_2 é a máxima possível [63].

No entanto, casos como da figura 2, onde existem *outliers*, tornam o uso da *Hard*

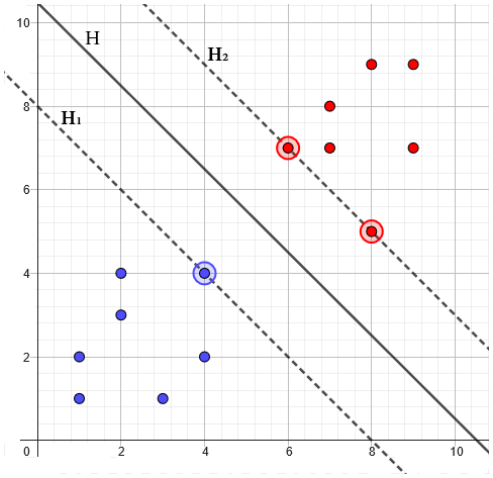

Figura 1 – Dados Lineares e Separáveis usando Hard Margin

Margin inviável, uma vez que esses dados muito fora do padrão influenciam no posicionamento do plano de separação H de maneira que novos dados são classificados de forma inconsistente. Esse caso é denominado de linear e não-separável [63]. Isso pode ser observado no ponto rosa, que é similar aos pontos vermelhos, mas acaba ficando ao lado dos pontos azuis pela separação.

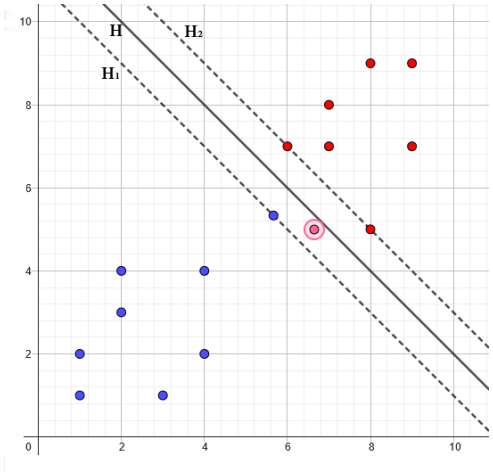

Figura 2 – Dados Lineares e Não-Separáveis usando Hard Margin

Sendo assim, o conceito de *Soft-Margin* faz-se necessário [63]. Essa versão do algoritmo é mais robusta uma vez que tolera algumas classificações incorretas, geradas por ruídos, visando uma melhora global dos resultados do modelo. Ela funciona relaxando a restrição de margem quando necessário [62], [63]. É possível observar uma aplicação de *soft-margin* na figura 3. Essa figura segue a mesma distribuição de dados da figura 2, onde o ponto rosa deve ser classificado juntamente aos pontos vermelhos e um *outlier* azul, destacado na figura, deve ser classificado com os demais pontos azuis. Uma vez que espacialmente esse ponto azul aproxima-se dos pontos vermelhos, considerá-lo rigidamente como *support vector* acarretaria numa piora global dos resultados de classificação, como

destacado anteriormente. Sendo assim, tomá-lo como ruído e tolerar a sua classificação incorreta permite a classificação do ponto rosa e demais futuros pontos juntamente aos vermelhos, como deve ser.

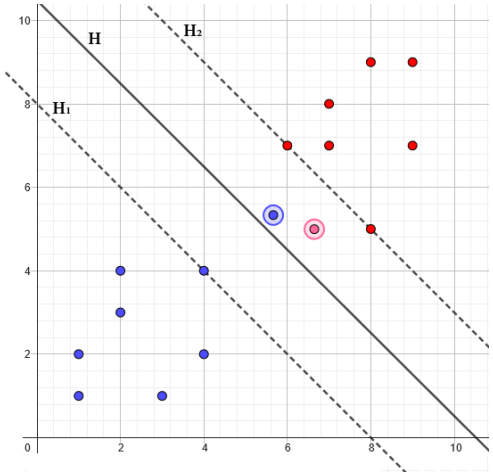

Figura 3 – Dados Lineares e Não-Separáveis usando Soft-Margin

Por fim, os dados podem ainda ser não-lineares, como mostrado na figura 4. Dessa forma, não é possível traçar uma reta que separe os pontos azuis dos vermelhos no plano bi-dimensional nativo dos dados. O modelo de SVM não-linear é responsável, portanto, em transformar os dados de entrada em dados de maior dimensão num espaço chamado *feature space*, onde é capaz de encontrar uma separação linear dos dados. Esse processo é chamado de *kernel trick* [65], onde é aplicada uma função de transformação em todos os pontos dos dados, mapeando-os como desejado [63], [64], [66]. No exemplo da figura 5, os dados nativamente apresentados no plano 2D são mapeados segundo uma função ϕ e ficam dispostos no espaço 3D, sendo separáveis linearmente nesse novo espaço.

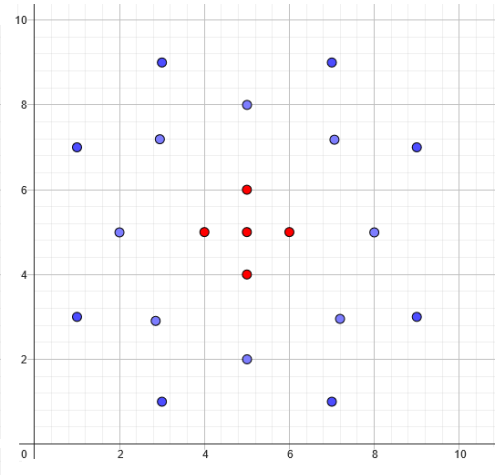

Figura 4 – Dados Não-lineares 2D

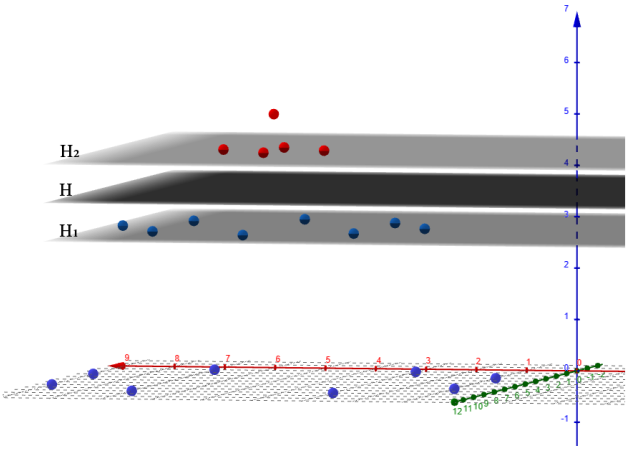

Figura 5 – Dados Mapeados em 3D

A função ϕ, responsável por mapear os dados do espaço original para o *feature space*, é chamada de *kernel function* e é definida de acordo com o problema a ser resolvido

[63]. De maneira geral, uma função de *kernel* pode ser representada como uma matriz quadrada $K \in \mathbb{R}^{l \times l}$ tal que $K_{ij} = k(X_i, X_j)$ para um conjunto de vetores $\{X_0, \ldots, X_l\} \subseteq X$ e uma dada função *kernel* k [62], [63], [66].

2.4.4.2 Treinamento e Hiperparâmetros

Como destacado em 2.4.4, um modelo de aprendizado de máquina é capaz de tomar decisões baseando-se em conhecimento acumulado. A fim de proporcionar esse conhecimento à máquina, o modelo deve passar pela fase de treinamento. O treinamento envolve oferecer uma parcela do conjunto de dados ao modelo de maneira que ele possa aprender e, posteriormente, possa ser testado com a parcela restante dos dados. É importante separar corretamente os dados em conjunto de treino e teste, uma vez que testar o modelo com dados que ele já conhece acarreta em respostas "viciadas"e não reflete corretamente o desempenho global do mesmo [64], [66].

Mesmo com a separação correta dos dados, um treino que gere resultados não satisfatório pode ocorrer. Nesse caso, o ajuste dos hiperparâmetros pode ser o detalhe necessário para obter um modelo melhor. De forma geral, os hiperparâmetros são todos aqueles parâmetros que não são aprendidos pelo modelo durante seu treinamento. Esses são definidos previamente a execução do programa e podem ser ajustados pelo programador por meio de diversas técnicas, como o *Grid Search*, que é uma busca exaustiva pela melhor combinação de hiperparâmetros [65].

O modelo de SVM possui diversos hiperparâmetros, no entanto, grande parte deles é referente à funções de *kernel* clássicas, previamente implementadas, e quanto à margem do hiperplano [65]. Sendo assim, os parametros importantes para este trabalho são:

- Parâmetro de Regularização: Conhecido como parâmetro C, é responsável por controlar o *trade-off* entre maximizar a margem do hiperplano de classificação e minimizar a classificação incorreta. Em linhas gerais, controla o quanto se quer evitar de classificação incorreta.

- Kernel: Define o *kernel* a ser usando pelo modelo. No caso deste trabalho, um *kernel* computado pelo módulo quântico da implementação.

A maneira mais simples e direta de treinar um modelo SVM é com o uso de uma matriz quadrada que relaciona os pontos de dados par a par segundo uma função ϕ [63], [64], como explicado na seção anterior. Apesar dessa maneira de computar a função de *kernel* demandar alto uso de memória, ela resulta numa implementação mais explicita do processo matemático e com menos abertura para erros de aproximação ou decomposição [64], [66]. Uma vez que este trabalho mira em implementar um modelo híbrido clássico-

quântico, onde o processo de cálculo da função de *kernel* é designada à parcela quântica do modelo, a computação total da matriz de *kernel* é mais adequada e assim será feita.

2.4.5 Aprendizado Profundo

Uma especialização do Aprendizado de Máquina amplamente utilizada é a chamada Rede Neural, *Neural Network* (NN), inspirada no cérebro, utilizando elementos chamados de neurônios e suas conexões para efetuar cálculos complexos [49]. Os neurônios são organizados em camadas sequenciais e adjacentes, o que permite a conexão somente entre neurônios de camadas adjacentes. A primeira e última camada são denominadas respectivamente de camada de entrada e de saída. A primeira é responsável por receber uma entrada X proveniente do conjunto de dados. A última representa a saída Y calculada para a entrada X, seja classificação ou regressão. Entre essas camadas existe pelo menos uma camada oculta. A combinação de número de neurônios por camada e número de camadas ocultas constitui a arquitetura da NN, permitindo inúmeras variações de implementações [55].

Por sua vez, o Aprendizado Profundo, *Deep Learning* (DL), é uma especialização mais potente da Rede Neural, sendo denominada Rede Neural Profunda, *Deep Neural Network* (DNN), pois implementa múltiplas camadas ocultas entre a camada de entrada e a de saída [36], [51], [37]. Por esse motivo, a DNN é capaz de performar modelagens mais complexas que a NN [36], [67], solucionando problemas mais elaborados em detrimento de maior uso do poder computacional. O Aprendizado Profundo requer um número maior de observações para ser treinado em comparação ao Aprendizado de Máquina, utilizando as características dos dados para se auto-ajustar [37].

Uma Rede Neural Profunda pode ser classificada como [36]:

Discriminativa	Generativa	Híbrida
Aprendizado Supervisionado	Aprendizado Não-supervisionado	Combinação de Ambos

Tabela 1 – Tipos de DNN.

Um modelo de aprendizado profundo chamado de Rede Generativa Adversária, *Generative Advesarial Network* (GAN), proposto em 2014 por *Goodfellow et al.* [68] têm sido amplamente empregado em diversas áreas, sendo considerado o estado da arte do aprendizado de máquina no paradigma da computação clássica. Suas aplicações vão desde a visão computacional, seja convertendo fotografias tiradas durante a noite para simular dia [69], seja gerando imagens realistas de alta qualidade a partir de descrição textual [70], até a detecção de anomalias [71], [72], e, consequentemente, a implementação se Sistemas de Detecção de Intrusão.

A Rede Adversária Generativa, *Generative Adversarial Network* (GAN), é uma abordagem de Aprendizado Profundo híbrido, como exposto na tabela 1, baseada na Teoria dos Jogos. Essa é composta por duas Redes Neurais internas que competem entre si, onde uma NN gera dados sintéticos e a outra classifica os seus dados de entrada como sintéticos (gerados) ou orgânicos (advindos do conjunto de dados) [73], [74], [75].

A Rede Neural responsável por sintetizar dados é chamada de gerador e a responsável por distinguir dados reais de dados gerados é chamada de Discriminador. O objetivo do gerador é enganar o Discriminador e do Discriminador discernir o que lhe é alimentado.

Visto que a GAN é uma abordagem generativa, uma de suas principais aplicações é a geração de dados. Dessa maneira, o treinamento desse modelo tem como objetivo criar um gerador capaz de enganar completamente o discriminador, gerando dados muito similares aos reais [74].

3 COMPUTAÇÃO QUÂNTICA

Muita incerteza tem permeado o tema "Computação Quântica". Alguns dizem ser o fim da computação como conhecemos, o fim da criptografia, mas ainda estamos muito distantes de substituir os computadores clássicos [17], [16].

Desde 1965, os computadores tiveram sua capacidade de processamento descrita segundo a Lei de Moore [76], a qual afirma que a cada dois anos o poder de processamento de um dispositivo dobra [77]. Essa preditiva que se mostrou válida desde *chips* integrados com 100 transistores até os dias atuais, onde bilhões de transistores compõem um único *chip* [16]. Dessa maneira, autores como [78] afirmaram com plena convicção que a evolução tecnológica segue um caminho razoavelmente previsível. E que, apesar das previsões do eventual fim da Lei de Moore por alguma barreira tecnológica ou científica, os engenheiros e cientistas têm encontrado formas de contornar essas barreiras, prolongando indeterminadamente a vigência dessa Lei.

Essas afirmações, no entanto, contradizem a física moderna, que garante um limite absoluto que a ciência e engenharia consegue atingir. O princípio da Incerteza de Heisenberg postula que há uma limitação da precisão com que podemos medir certas propriedades de partículas subatômicas [79]. Ainda, o físico Stephen Hawking em 2005, durante sua visita à Intel, afirmou que a velocidade da luz e a natureza atômica da matéria seriam os limites fundamentais para a microeletrônica.

De maneira geral, o fim ou não da Lei de Moore é uma discussão em aberto, mas é inegável que a incerteza da sua continuidade foi crucial para direcionar os holofotes em direção à computação quântica. Anos depois, em 1993, Berstein et al. mostrou que os computadores quânticos poderiam violar a Tese Estendida de Church-Turing [80], que afirma que qualquer "modelo razoável"de computação pode ser eficientemente simulado em um modelo padrão, como uma Máquina de Turing, uma Máquina de Acesso Aleatório ou um autômato celular.

Apenas um ano depois, em 1994, Peter Shor mostrou um algoritmo capaz de resolver um problema exponencialmente mais rápido que um computador clássico. O chamado Algoritmo de Shor, é um exemplo prático de fatoração de números grandes onde é confirmada a tese apresentada em 1993 por Berstein et al. [15].

Por mais que esse algoritmo fosse interessante para o período, nenhum cientista tinha sequer ideia de como construir um computador quântico. Nos dias de hoje, no entanto, não apenas é possível implementar o Algoritmo de Shor[0] , mas também executá-

[0] O Algoritmo de Shor é um algoritmo de tempo polinomial que usa, como mostrado em [81], $2n + 3$ *qubits* para fatorar um número inteiro de n *bits*.

lo em simulações ou computadores quânticos reais. É importante levar em conta que a implementação genérica desse está limitada pela disponibilidade de *qubits*.

3.1 O Paradigma Quântico

Os computadores clássicos possuem circuitos integrados contendo bilhões de transistores que operam sobre os dígitos binários (*bits*), que por sua vez podem assumir valores determinísticos 0 ou 1. Através da manipulação desses bits os dispositivos são capazes de representar informações, operações e, em larga escala, aplicativos e serviços [16].

Um computador quântico também representa informação em uma espécie de *bit*, o *qubit*. O *qubit*, abreviação para *Quantum Bit*, também pode assumir o valor de 0 ou 1, mas não fica limitado a isso. Essa unidade de dados pode ficar em um estado de superposição, onde é 0 e 1 ao mesmo tempo. Dessa maneira, um sistema com múltiplos *qubits* pode estar em todos os estados binários possíveis, levando em conta a quantidade de dígitos binários, ao mesmo tempo [82], [83], [84], [16]. Ao longo de um algoritmo quântico, manipulam-se as distribuições probabilísticas desses estados de superposição. Devido à natureza probabilística desses algoritmos, esses são repetidos um número de vezes, a fim de manter um resultado relativamente consistente [21]. Esse fenômeno é conhecido como interferência quântica e é fundamental para o poder de processamento dos computadores quânticos [85], [16].

Os *qubits*, no entanto, são sensíveis a interferências do ambiente, o que pode levar a erros no processamento. Para mitigar esse problema, os computadores quânticos utilizam um processo chamado correção de erros quânticos. Nele, os *qubits* são redundantes para tentar alcançar uma precisão maior nos cálculos [86]. É natural que um *qubit* decaia ao longo de sua manipulação, principalmente ao ser aplicado em circuitos maiores e mais complexos, perdendo o estado de superposição em que estava, invalidando seu uso [87].

Além disso, uma propriedade importante dos *qubits* é a chamada emaranhamento, do inglês *entanglement*. Quando *qubits* estão emaranhados, o estado de um *qubit* não pode ser descrito independentemente do estado de outros *qubits* emaranhados com ele [87], [88]. Essa característica é explorada em algoritmos quânticos num geral, assim como no Algoritmo de Shor [15].

É importante destacar que nem todos os problemas podem ser resolvidos de forma mais eficiente com computadores quânticos. Eles se destacam na resolução de problemas inerentemente quânticos, inteligência artificial e criptografia. Para as demais tarefas esses computadores podem não ser atraentes [16].

Visto que a computação quântica é uma ciência muito recente, até mesmo os últimos *surveys* sobre a tecnologia apresentam informações desatualizadas de seus ambientes de execução e ferramentas. Como destacado em [89], artigo intitulado "QUANTUM

COMPUTING: SURVEY AND ANALYSIS"de 2019, a construção de dispositivos reais de computação quântica é um problema fundamental da física moderna e, até então, existiam apenas implementações limitadas desses aparelhos. Essa limitação persiste até a atualidade, no entanto dispositivos com maior capacidade de *qubits* e correção de erros têm sido construídos.

Além dos recursos de simulação de computadores quânticos, existem dois tipos principais de implementações físicas desses aparelhos: a universal e a não-universal. O computador universal é aquele onde a implementação de algoritmos é livre, possibilitando a execução de operações e resolução de problemas arbitrários. Um dispositivo não-universal, por sua vez, é aquele voltado para a resolução duma classe específica de algoritmos, como a otimização de certos modelos de aprendizado de máquina ou otimização combinatória. Um exemplar desse computador pode ser selecionado da coleção de dispositivos desenvolvidos pela empresa canadense D-Wave, como o computador de 16 *qubits* capaz de resolver sudokus. Quando se trata de computadores universais, três grandes empresas se destacam: IBM, Google e Intel.

A IBM conta com *chips* quânticos de acesso gratuito e pago. O acesso gratuito tem *runtime* mensal limitado a 10 minutos e oferece disponibilidade de 7 e 127 *qubits*. Os planos pagos, por sua vez, não limitam o tempo de uso e contam com sistemas de 27 e 127 *qubits*. A empresa tem seu próprio *framework* de simulação, cujo uso não é limitado temporalmente. O Qiskit é baseado em Python e traz a mesma programação de alto nível dos algoritmos executados nos *chips* quânticos. A simulação disponibiliza acesso irrestrito aos *qubits* em detrimento de um tempo de execução mais alto [90].

O Google, no entanto, tem uma abordagem mais reservada sobre o acesso aos seus computadores quânticos. A fim de utilizá-los, é necessário escrever uma proposta de pesquisa de 2 a 3 páginas, providenciando informações dos pesquisadores envolvidos, sumário da pesquisa, visão geral do projeto, recursos necessários para execução (tempo de máquina e acesso a *qubits*), experiências passadas com hardware quântico e colaborações para o meio científico e, por fim, o que se espera desse projeto em termos de resultados. Após a submissão da proposta, há uma fase de análise e, caso seja aprovado, o projeto pode ter acesso à *chips* de fila aberta (cuja execução ocorre juntamente a demais projetos aprovados). Conforme a pesquisa avança, acesso dedicado a *chips* pode ser pleiteado [91]. A empresa também aposta na simulação com o *framework* Cirq, de código aberto. O Cirq é uma biblioteca para Python que permite escrever, manipular e otimizar circuitos quânticos [92].

A Intel não disponibiliza acesso aos seus computadores quânticos para o público. A empresa incentiva a pesquisa na área, investindo milhões de dólares em universidades pela Fundação Nacional da Ciência dos Estados Unidos, e oferece vagas de emprego em diversas localidades do mundo [93].

Outros dois grandes *players* que têm apostado na computação quântica são Microsoft e Amazon. Seus *hardwares* quânticos são acessáveis via assinatura, onde o cliente paga conforme os utiliza. Ambos possuem abordagens de simulação, o Q# e o *Braket Local Simulator* respectivamente.

Ainda, a empresa canadense Xanadu, financiada por diversas outras empresas incluindo a montadora automobilística Porsche, tem por missão construir computadores quânticos acessíveis ao público. Desenvolveram o primeiro computador quântico nanofotônico, que manipula e mede fótons ao longo de seus circuitos, acessível por nuvem do mundo. Para ter acesso a esse *hardware* é necessário entrar em contato com a empresa via *e-mail*. Soluções de *software* também são do escopo da Xanadu, proporcionando a biblioteca PennyLane para Python. Essa biblioteca permite a integração de ferramentas de aprendizado de máquina e oferece suporte a um conjunto abrangente de recursos, incluindo as plataformas citadas previamente.

Por fim, a empresa estadunidense Rigetti, premiada como uma das três empresas líderes em computação quântica juntamente ao Google e IBM em 2016, não apenas desenvolve os próprios *chips* (que compõem as máquinas de outras plataformas como as da Microsoft e Amazon) mas também os integra numa arquitetura de controle e, por meio de software proprietário, permite que programadores possam construir algoritmos para rodar nesses dispositivos. Apesar do website da empresa afirmar o acesso completamente gratuito a *hardware* quântico através da plataforma parceira Strangeworks QC, o plano gratuito não contempla tempo de máquina em dispositivos quânticos reais, sendo sujeito a custos não especificados.

3.2 Passado, Presente e Futuro

Como visto anteriormente, o estudo da computação quântica é recente, tendo suas primeiras especulações na década de 90. A implementação de *hardware* quântico veio surgir por volta de 2001 com a empresa IBM. Nesse ano, cientistas da IBM anunciaram teste bem sucedido de um computador quântico de 7 *qubits*. Esse dispositivo fora implementado com base no fenômeno de ressonância magnética nuclear e foi capaz de fatorar o número 15 pelo algoritmo de Shor [89].

De acordo com [89], a próxima implementação quântica veio apenas em 2007, pela empresa D-Wave, com um dispositivo de 16 *qubits*. Esse computador resolvia sudokus e outras tarefas de determinação de padrão. A empresa anunciou ainda o processador One, em maio de 2011, alegando ser o primeiro computador quântico comercial, operando com 128 *qubits*. No ano seguinte, um dispositivo com 512 *qubits* fora anunciado e, em agosto de 2015, um dispositivo com 1152 *qubits*. Em janeiro de 2017, a empresa anunciou a venda do seu futuro dispositivo D-Wave 2000Q, que operava com 2000 *qubits* e custava 15 milhões

de dólares, para a empresa de segurança cibernética TDS. Em 2020, a D-Wave anunciou seu computador *Advantage*, com 5000 *qubits* [94].

A maioria dos pesquisadores, no entanto, não consideram os dispositivos desenvolvidos pela D-Wave como dispositivos quânticos de fato [89]. Visto que sua implementação é adiabática, onde os *qubits* não são manipulados por portas lógicas mas sim por transformações termodinâmicas, alguns algoritmos quânticos podem ser implementados nesse modelo, mas nem sempre são viáveis, favorecendo algoritmos de otimização combinatória [89], [94]. A própria empresa D-Wave enfatizou, após o lançamento do seu modelo 2000Q, que o algoritmo de Shor não poderia ser implementado em seus dispositivos [89]. Em 2015 pesquisadores do Google alegaram que os dispositivos da D-Wave, apesar de apresentarem os fenômenos quânticos esperados, agrupavam os *qubits* em clusteres menores. Ainda, em [95], as características desses computadores foram analisadas e foi concluído que esses não seriam capazes de prover nenhuma vantagem computacional sobre os computadores clássicos.

Considerando dispositivos quânticos baseados em portas lógicas, no ano de 2007, um grupo de cientistas da Universidade de Queensland reportaram a fatoração bem sucedida do número 15 usando 7 *qubits*, num computador cujas portas lógicas eram baseadas em polarização de fótons. No mesmo ano, cientistas da Universidade de Ciência e Tecnologias da China também reportou a fatoração bem sucedida do número 15 usando apenas 6 *qubits*, numa implementação do algoritmo de Shor com portas lógicas baseadas em fótons [89].

Em 2009, a fatoração do número 15 por meio do algoritmo de Shor pôde ser implementada em um chip de silício usando 4 *qubits*. No ano de 2012, pesquisadores da Universidade da Califórnia reportaram a demonstração do algoritmo de Shor implementado com 4 *qubits* de fase (tipo específico de *qubit*) e ressonadores de ondas supercondutoras para criar e manipular os circuitos, fatorando o número 15. Nesse mesmo ano, E. Martin-Lopez et al. fatorou o número 21 usando uma implementação do mesmo algoritmo com apenas 2 *qubits* baseados em fótons [89].

Transformando o algoritmo de Shor em um algoritmo de otimização combinatória, em 2012, foi possível fatorar o número 143 usando 4 *qubits* na implementação adiabática. Em 2015, um grupo de pesquisadores liderados por T. Montz reportou uma nova demonstração experimental do algoritmo de Shor usando armadilha de íons para decompor o número 15. Nesse experimento, 5 íons $^{40}Ca^{+}$ numa armadilha linear de Paul foram usados, sendo cada íon traduzido para um *qubit* [89].

Em 2016, a IBM anunciou a criação de um computador quântico de 5 *qubits*, sendo um deles usado para correção de erros. Esse dispositivo é baseado num *chip* supercondutor com geometria estrela e implementação completa da álgebra de Clifford, sendo programável e permitindo criar portas lógicas, bem como modelar a operação das mesmas. Em

maio de 2017, a empresa anunciou um novo dispositivo de 50 *qubits*, onde 20 *qubits* são usáveis e o restante é destinado à correção de erros. Nesse computador, um *qubit* fica em estado coerente por até 90 microssegundos, tornando-se incoerente após esse período [89]. De acordo com o site da empresa [96], foram lançados em 2020 o *chip* Hummingbird de 65 *qubits*, em 2021 o Eagle de 127 *qubits*, 2022 o Osprey com 433 *qubits* e, por fim, o chip Condor com 1121 *qubits* em 2023.

Em janeiro de 2018, a Intel também se juntou à corrida dos computadores quânticos e declarou a criação de um *chip* quântico supercondutor de 49 *qubits*, chamado de Tangle Lake [89]. De acordo com o site da empresa [93], o *chip* Tunnel Falls é seu último lançamento. O Tunnel Falls é um *chip* de pesquisa e é baseado em *qubits* de *spin* de silício. Sua criação é, de acordo com a Intel, um progresso significante na criação de um *chip* quântico confiável e produzível em escala, uma vez que utiliza a tecnologia de transistores CMOS de silício. Esse chip possui 12 *qubits*.

Em março de 2018, o Google apresentou o *chip* Bristlecone com capacidade para 72 *qubits*, que aproveitava a construção de um *chip* de 9 *qubits* da própria empresa. O diferencial em relação ao modelo de 9 *qubits* é o uso de estruturas bi-dimensionais de $6x6$ *qubits* empilhadas verticalmente. De acordo com a empresa, essa construção permite que o dispositivo localize e corrija erros em tempo de execução [89]. De acordo com o site do Google [97], seu *chip* mais recente é o Sycamore, com 54 *qubits*. A empresa alega que esse *chip* pode executar uma computação de teste em 200 segundos, que no supercomputador clássico mais rápido do mundo demoraria 10000 anos.

Atualmente estamos na denominada *Noisy Intermediate-Scale Quantum Era* (NISQ Era), termo usado para expressar a situação atual dos computadores quânticos, de Escala Intermediária e com Ruídos [18], [17], [85]. A dita "vantagem quântica"só pode ser atingida em sua completude com alta disponibilidade de *qubits* e menor taxa de erros devido à decaimentos [17], sendo assim, a tecnologia que temos até então não é matura o suficiente.

A pesquisa e estudo na área é promissora e o futuro aguarda implementações mais estáveis e poderosas, como destaca o autor John Preskill, em 2018, sobre a então futura computação quântica [18]:

> NISQ devices will be useful tools for exploring many-body quantum physics, and may have other useful applications, but the 100-qubit quantum computer will not change the world right away — we should regard it as a significant step toward the more powerful quantum technologies of the future. Quantum technologists should continue to strive for more accurate quantum gates and, eventually, fully fault-tolerant quantum computing.

De maneira geral, as empresas de tecnologia têm buscado a criação de *chips* quân-

ticos proprietários e universais, devido à promissora perspectiva da computação quântica. A atualidade, no entanto, está limitada à poucos *qubits* utilizáveis. Por mais que diversos *chips* possuam uma porcentagem grande dos seus *qubits* designados para a correção de erros, ainda é cedo para declarar em qualquer instância que a vantagem quântica já foi alcançada [89].

3.3 Arquitetura do Computador Quântico

Como destacado em 3.1, diversas implementações de computadores quânticos têm sido executadas por diferentes empresas. Diferentemente dos computadores clássicos, onde uma única abordagem é utilizada para a manufatura desses dispositivos, os computadores quânticos têm diferentes modelos teóricos que embasam suas implementações. Os modelos teóricos podem ser resumidos em *gate model*, *measurement model*, *adiabatic model*, *quantum annealing* e *topological quantum computing*, sendo que o último ainda não conta com exemplares físicos.

O chamado *gate model*, também conhecido como *circuit model*, é um modelo que se assemelha ao máximo com a computação clássica como conhecemos. Como o próprio nome diz, esse modelo teórico de computador quântico é baseado em circuitos e portas lógicas. Um circuito aplica portas lógicas em seus *qubits* a fim de manipular seus estados de superposição. Um algoritmo, por sua vez, é um circuito de X *qubits* manipulados por operações arbitrárias e medidos ao final desse circuito. Vale destacar que uma extensão desse modelo permite a medição dos *qubits* em qualquer ponto de um circuito, não apenas ao final, sendo nomeada de "circuitos quânticos com medições intermediárias"[98].

O modelo *measurement model* é, ainda, um caso especial dos circuitos com medições intermediárias. A principal diferença e conceito central desse modelo é que os *qubits* do circuito são inicializados com um estado emaranhado (*entangled*, como explicado em 3.1) e são medidos individualmente ao longo desse circuito. Esse modelo é matematicamente equivalente ao modelo *gate model*, e é capaz de simular diversos circuitos subsequentes em um grande circuito contínuo [98].

É no modelo adiabático que os conceitos da computação clássica começam a dar lugar aos conceitos da física pura e simples. No *adiabatic model* não existem circuitos nem portas lógicas, ao invés disso, esse modelo tira proveito do principio geral da física que postula que todo sistema sempre se move em direção ao estado de menor energia. Dessa forma, o problema que se deseja resolver por um computador quântico adiabático deve ser construído de tal maneira que o estado de menor energia represente a resposta para esse problema. De maneira menos abstrata, um sistema quântico adiabático é iniciado com uma paisagem energética plana e é manipulada de forma que gradualmente são introduzidos os relevos energéticos do problema. Sendo assim, cada ponto na paisagem energética é

um potencial *output* para o computador quântico e o ponto de menor energia representa a resposta ao problema. A chave para esse modelo funcionar é a lenta introdução da paisagem energética do problema no sistema [99].

Quantum annealing funciona da mesma maneira que o modelo adiabático, no entanto é direcionado para a solução de problemas específicos e não oferece o mesmo grau de liberdade para a implementação de um algoritmo arbitrário, sendo considerado um esquema não-universal [100].

Os modelos baseados em circuitos e o modelo adiabático são modelos universais de computação quântica, sendo assim são capazes de implementar qualquer sistema quântico. Por esse motivo, são considerados equivalentes em termos de escopo de solução de problemas, visto que problemas podem ser modelados para ambos os esquemas [101].

Por fim, o modelo topológico de computação quântica é o mais teórico dos modelos existentes. Seus *qubits* são construídos a partir de uma entidade física chamada *majorana zero-mode quasi-particle*, um tipo de ânyon não-abeliano. Os ânyons são um tipo especial de partícula, denominada quasipartícula, não fundamental e bidimensional. Ou seja, não são partículas como átomos, elétrons ou fótons, são criadas através do comportamento de diversas partículas juntas e acabam tendo propriedades de partícula, apesar de não serem reais. O exemplo mais simples de quasipartícula é o buraco de elétron, onde uma grade de elétrons têm um buraco e esse buraco acaba se comportando como uma partícula, mesmo sendo apenas a ausência de um elétron. A quasipartícula *majorana zero-mode quasi-particle*, por sua vez, foi teoricamente prevista e observada. Seu uso na computação quântica é promissor uma vez que é muito mais estável que os demais modelos de *qubits*, já que é formada por partes fisicamente separadas por um vão energético, o que as torna muito mais resistentes à ruído [102].

Devido à maior facilidade em discernir problemas para o modelo de circuitos, além do maior acesso à computadores e softwares de simulação do tipo *circuit model*, esse trabalho utilizará esse modelo para a implementação e futura revisão bibliográfica.

3.3.1 Qubit

Dito isso, podemos definir formalmente a unidade básica do computador quântico, o *Quantum Bit*. Até este ponto, o *qubit* foi apenas apresentado como uma unidade similar ao *bit* do computador clássico, porém capaz de estar num estado de superposição onde não é nem deterministicamente 0 nem 1, mas sim duas probabilidades distintas de estar em cada estado base. É apenas quando observamos o *qubit* que ele assume um valor determinístico de 0 ou 1.

Na mecânica quântica, uma partícula em superposição é representada por um estado quântico. Esse estado é a combinação linear de todos os seus possíveis resultados

e suas respectivas probabilidades de ocorrência. As partículas podem assumir infinitos possíveis resultados e, dessa maneira, seu estado quântico resultante seria uma combinação linear de infinitos termos. Outras partículas, como o átomo de hidrogênio, têm energias discretas, sendo possível listar todas as possíveis medições.

A fim de garantir universalidade, o estudo da física quântica deve contemplar tanto partículas com infinitos possíveis estados, como as de estados enumeráveis. Trabalhar com infinidade é um tanto complicado, uma vez que o infinito não é uma grandeza ou um número de fato, mas sim um conceito matemático. Dessa maneira, é natural que uma sequência convergente de infinitos termos seja resolvida usando limite. Usar limite, no entanto, pode gerar um vetor resultante que não pertence ao espaço vetorial da sequência original. Por esse motivo, usa-se o espaço vetorial especial, chamado espaço de Hilbert ($\mathcal{H}$) para trabalhar com estados quânticos, refletindo no nosso estudo de computação quântica.

Um espaço de Hilbert é todo e qualquer espaço vetorial com a propriedade de produto escalar Cauchy completo, ou seja, toda sequência convergente de vetores converge para um elemento dentro desse mesmo espaço vetorial. Isso garante que, mesmo com N termos que se aproximam de infinito, o limite nunca será atingido e o vetor resultante dessa combinação linear permaneça no mesmo espaço vetorial.

Sendo assim, o estado quântico de um *qubit* é representado por um vetor do espaço vetorial de Hilbert, onde $\mathcal{H} = \mathbb{C}^2$. O espaço é complexo bi-dimensional uma vez que o estado de um *qubit* é medido segundo o spin da partícula que o implementa fisicamente. Caso fosse optado por medir a posição ou o momento dessa partícula, o espaço $\mathcal{H}$ seria de dimensão infinita, dificultando a manipulação de informações.

Ω spins de particulas

Nesse estudo a norma do vetor não importa, ou seja, dois vetores $\mathbf{a}$ e $\mathbf{b}$ diferenciados apenas pelo seu escalar complexo são estados quânticos $\boldsymbol{\alpha}$ e $\boldsymbol{\beta}$ equivalentes.

$$\alpha\mathbf{a} = \mathbf{b} \;\; \Rightarrow \;\; \boldsymbol{\alpha} \sim \boldsymbol{\beta}$$

Para isso, os estados quânticos são tratados como "raios"que passam pela origem de $\mathcal{H}$. Uma representação mais amigável desses estados quânticos é a *complex projective sphere*, onde toda a coleção de raios é representada por raios equivalentes de comprimento unitário, formando uma superfície esférica em torno da origem de $\mathcal{H}$. Os pontos nessa superfície representam também os estados quânticos do *qubit*. A *complex projective sphere*, portanto, não é um espaço vetorial, é apenas uma abstração do espaço de Hilbert complexo bi-dimensional.

Ω modelo de representação esférica a fundo

Ω time independent quantum mechanics

Ω decoherence

Ω quantum error correction

Ω escalabilidade do sistema (emaranhar, manipular e medir)

3.3.2 Portas Lógicas

Ω operadores unitários

Ω definir as operações e efeitos de cada porta lógica

3.3.3 Resultado Não-determinístico

Ω medição é feita por operadores Hermitianos

3.4 Solução de Problemas

σ Problemas quânticos ou não? é possível diferenciar ou todos serão resolvidos? o que diz até agora a teoria?

3.4.1 Problemas Quânticos

Ω Problemas inerentemente quânticos...

3.4.2 Problemas Clássicos

Ω Usa-se feature map para mapear um dado clássico num qubit, colocando-o num estado quântico arbitrário equivalente

3.5 Aprendizado de Máquina Híbrido

3.5.1 Quantum Enhanced Support Vector Machine

4 TRABALHOS RELACIONADOS

5 ESTUDO DE CASO

σ Desenvolver um problema no mundo tradicional, neste caso detecção de anomalias com SVM com dataset Orion, somente um exemplo

σ Desenvolver o mesmo problema com o que tem hoje para quântico, simulado e/ou real?

6 CONCLUSÕES

- Comparar resultados?
- Ficou melhor ou pior? por que dá para saber?
- Será possível você convencer no modelo quântico o que ficou diferente do modelo tradicional?

REFERÊNCIAS

[1] MIORANDI, D. et al. Internet of things: Vision, applications and research challenges. *Ad Hoc Networks*, v. 10, n. 7, p. 1497–1516, 2012. ISSN 1570-8705. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1570870512000674>.

[2] WHITMORE, A.; AGARWAL, A.; XU, L. D. The internet of things—a survey of topics and trends. *Information systems frontiers*, Springer, v. 17, p. 261–274, 2015.

[3] ASSIS, M. V. D. et al. Fast defense system against attacks in software defined networks. *IEEE Access*, IEEE, v. 6, p. 69620–69639, 2018.

[4] MASOUDI, R.; GHAFFARI, A. Software defined networks: A survey. *Journal of Network and computer Applications*, Elsevier, v. 67, p. 1–25, 2016.

[5] CARVALHO, L. F. et al. An ecosystem for anomaly detection and mitigation in software-defined networking. *Expert Systems with Applications*, Elsevier, v. 104, p. 121–133, 2018.

[6] LATHA, S.; PRAKASH, S. J. A survey on network attacks and intrusion detection systems. In: *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*. [S.l.: s.n.], 2017. p. 1–7.

[7] DASTRES, R.; SOORI, M. A review in recent development of network threats and security measures. *International Journal of Information Sciences and Computer Engineering*, 2021.

[8] FERNANDES, G. et al. A comprehensive survey on network anomaly detection. *Telecommunication Systems*, Springer, v. 70, p. 447–489, 2019.

[9] PENA, E. H. et al. Anomaly detection using the correlational paraconsistent machine with digital signatures of network segment. *Information Sciences*, Elsevier, v. 420, p. 313–328, 2017.

[10] KHAN, F. A. et al. A novel two-stage deep learning model for efficient network intrusion detection. *IEEE Access*, IEEE, v. 7, p. 30373–30385, 2019.

[11] DONG, B.; WANG, X. Comparison deep learning method to traditional methods using for network intrusion detection. In: *2016 8th IEEE International Conference on Communication Software and Networks (ICCSN)*. [S.l.: s.n.], 2016. p. 581–585.

[12] SHONE, N. et al. A deep learning approach to network intrusion detection. *IEEE Transactions on Emerging Topics in Computational Intelligence*, v. 2, n. 1, p. 41–50, 2018.

[13] WANG, H.; GU, J.; WANG, S. An effective intrusion detection framework based on svm with feature augmentation. *Knowledge-Based Systems*, Elsevier, v. 136, p. 130–139, 2017.

[14] KIM, D. E.; GOFMAN, M. Comparison of shallow and deep neural networks for network intrusion detection. In: IEEE. *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*. [S.l.], 2018. p. 204–208.

[15] EKERT, A.; JOZSA, R. Quantum computation and shor's factoring algorithm. *Reviews of Modern Physics*, APS, v. 68, n. 3, p. 733, 1996.

[16] SCIENCES, E. National Academies of; MEDICINE et al. Quantum computing: progress and prospects. National Academies Press, 2019.

[17] GHEORGHE-POP, I.-D. et al. Quantum devops: Towards reliable and applicable nisq quantum computing. In: *2020 IEEE Globecom Workshops (GC Wkshps*. [S.l.: s.n.], 2020. p. 1–6.

[18] PRESKILL, J. Quantum computing in the nisq era and beyond. *Quantum*, Verein zur Förderung des Open Access Publizierens in den Quantenwissenschaften, v. 2, p. 79, 2018.

[19] GARCÍA, D. P.; CRUZ-BENITO, J.; GARCÍA-PEÑALVO, F. J. Systematic literature review: Quantum machine learning and its applications. *arXiv preprint arXiv:2201.04093*, 2022.

[20] BIAMONTE, J. et al. Quantum machine learning. *Nature*, Nature Publishing Group UK London, v. 549, n. 7671, p. 195–202, 2017.

[21] SCHULD, M.; SINAYSKIY, I.; PETRUCCIONE, F. An introduction to quantum machine learning. *Contemporary Physics*, Taylor & Francis, v. 56, n. 2, p. 172–185, 2015.

[22] PAYARES, E.; MARTÍNEZ-SANTOS, J. C. Quantum machine learning for intrusion detection of distributed denial of service attacks: a comparative overview. *Quantum Computing, Communication, and Simulation*, SPIE, v. 11699, p. 35–43, 2021.

[23] KALININ, M.; KRUNDYSHEV, V. Security intrusion detection using quantum machine learning techniques. *Journal of Computer Virology and Hacking Techniques*, Springer, v. 19, n. 1, p. 125–136, 2023.

[24] LIANG, J.-M. et al. Quantum anomaly detection with density estimation and multivariate gaussian distribution. *Physical Review A*, APS, v. 99, n. 5, p. 052310, 2019.

[25] NOVAES, M. P. et al. Adversarial deep learning approach detection and defense against ddos attacks in sdn environments. *Future Generation Computer Systems*, Elsevier, v. 125, p. 156–167, 2021.

[26] NOVAES, M. P. et al. Long short-term memory and fuzzy logic for anomaly detection and mitigation in software-defined network environment. *IEEE Access*, IEEE, v. 8, p. 83765–83781, 2020.

[27] ASSIS, M. V. de et al. Near real-time security system applied to sdn environments in iot networks using convolutional neural network. *Computers & Electrical Engineering*, Elsevier, v. 86, p. 106738, 2020.

[28] KREUTZ, D. et al. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, Ieee, v. 103, n. 1, p. 14–76, 2014.

[29] SAHAY, R.; MENG, W.; JENSEN, C. D. The application of software defined networking on securing computer networks: A survey. *Journal of Network and Computer Applications*, Elsevier, v. 131, p. 89–108, 2019.

[30] LI, W.; MENG, W.; KWOK, L. F. A survey on openflow-based software defined networks: Security challenges and countermeasures. *Journal of Network and Computer Applications*, Elsevier, v. 68, p. 126–139, 2016.

[31] TANK, G. et al. Software defined networks: The new norm for networks. *International Journal of Science and Research (IJSR)*, 2017.

[32] THOTTAN, M.; JI, C. Anomaly detection in ip networks. *IEEE Transactions on signal processing*, IEEE, v. 51, n. 8, p. 2191–2204, 2003.

[33] LENT, D. M. B. et al. A gated recurrent unit deep learning model to detect and mitigate distributed denial of service and portscan attacks. *IEEE Access*, IEEE, v. 10, p. 73229–73242, 2022.

[34] ZHANG, T. et al. A survey of network anomaly visualization. *Science China Information Sciences*, Springer, v. 60, p. 1–17, 2017.

[35] CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, ACM New York, NY, USA, v. 41, n. 3, p. 1–58, 2009.

[36] ALDWEESH, A.; DERHAB, A.; EMAM, A. Z. Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. *Knowledge-Based Systems*, Elsevier, v. 189, p. 105124, 2020.

[37] LEE, S.-W. et al. Towards secure intrusion detection systems using deep learning techniques: Comprehensive analysis and review. *Journal of Network and Computer Applications*, Elsevier, v. 187, p. 103111, 2021.

[38] YANG, Z. et al. A systematic literature review of methods and datasets for anomaly-based network intrusion detection. *Computers & Security*, Elsevier, v. 116, p. 102675, 2022.

[39] OTOUM, Y.; NAYAK, A. As-ids: Anomaly and signature based ids for the internet of things. *Journal of Network and Systems Management*, Springer, v. 29, p. 1–26, 2021.

[40] PROENÇA, M. L.; ZARPELÃO, B. B.; MENDES, L. S. Anomaly detection for network servers using digital signature of network segment. In: IEEE. *Advanced Industrial Conference on Telecommunications/Service Assurance with Partial and Intermittent Resources Conference/E-Learning on Telecommunications Workshop (AICT/SAPIR/ELETE'05)*. [S.l.], 2005. p. 290–295.

[41] HAMAMOTO, A. H. et al. Network anomaly detection system using genetic algorithm and fuzzy logic. *Expert Systems with Applications*, Elsevier, v. 92, p. 390–402, 2018.

[42] SCARANTI, G. F. et al. Unsupervised online anomaly detection in software defined network environments. *Expert Systems with Applications*, Elsevier, v. 191, p. 116225, 2022.

[43] CARVALHO, L. F. et al. A novel anomaly detection system to assist network management in sdn environment. In: IEEE. *2017 IEEE international conference on communications (ICC)*. [S.l.], 2017. p. 1–6.

[44] SCARANTI, G. F. et al. Artificial immune systems and fuzzy logic to detect flooding attacks in software-defined networks. *IEEE Access*, IEEE, v. 8, p. 100172–100184, 2020.

[45] ASSIS, M. V. et al. A gru deep learning system against attacks in software defined networks. *Journal of Network and Computer Applications*, Elsevier, v. 177, p. 102942, 2021.

[46] JR, M. L. P. et al. Digital signature to help network management using flow analysis. *International Journal of Network Management*, Wiley Online Library, v. 26, n. 2, p. 76–94, 2016.

[47] PROENÇA, M. L. et al. The hurst parameter for digital signature of network segment. In: SPRINGER. *Telecommunications and Networking-ICT 2004: 11th International Conference on Telecommunications, Fortaleza, Brazil, August 1-6, 2004. Proceedings 11*. [S.l.], 2004. p. 772–781.

[48] ALPAYDIN, E. *Introduction to machine learning*. [S.l.]: MIT press, 2020.

[49] MOHAMMED, M.; KHAN, M. B.; BASHIER, E. B. M. *Machine learning: algorithms and applications*. [S.l.]: Crc Press, 2016.

[50] TABAK, M. A. et al. Machine learning to classify animal species in camera trap images: Applications in ecology. *Methods in Ecology and Evolution*, Wiley Online Library, v. 10, n. 4, p. 585–590, 2019.

[51] HATCHER, W. G.; YU, W. A survey of deep learning: Platforms, applications and emerging research trends. *IEEE Access*, IEEE, v. 6, p. 24411–24432, 2018.

[52] XU, L.; WHITE, M.; SCHUURMANS, D. Optimal reverse prediction: A unified perspective on supervised, unsupervised and semi-supervised learning. In: *Proceedings of the 26th Annual International Conference on Machine Learning*. New York, NY, USA: Association for Computing Machinery, 2009. (ICML '09), p. 1137–1144. ISBN 9781605585161. Disponível em: <https://doi.org/10.1145/1553374.1553519>.

[53] SULTANA, N. et al. Survey on sdn based network intrusion detection system using machine learning approaches. *Peer-to-Peer Networking and Applications*, Springer, v. 12, p. 493–501, 2019.

[54] XIN, Y. et al. Machine learning and deep learning methods for cybersecurity. *Ieee access*, IEEE, v. 6, p. 35365–35381, 2018.

[55] XIE, J. et al. A survey of machine learning techniques applied to software defined networking (sdn): Research issues and challenges. *IEEE Communications Surveys & Tutorials*, IEEE, v. 21, n. 1, p. 393–430, 2018.

[56] BAKSHI, S. et al. Enhancing scalability and accuracy of recommendation systems using unsupervised learning and particle swarm optimization. *Applied Soft Computing*, Elsevier, v. 15, p. 21–29, 2014.

[57] OSUNA, E. E. *Support vector machines: Training and applications*. Tese (Doutorado) — Massachusetts Institute of Technology, 1998.

[58] YIN, X.-C. et al. Shallow classification or deep learning: An experimental study. In: IEEE. *2014 22nd International Conference on Pattern Recognition*. [S.l.], 2014. p. 1904–1909.

[59] JANABI, A. H.; KANAKIS, T.; JOHNSON, M. Convolutional neural network based algorithm for early warning proactive system security in software defined networks. *IEEE Access*, v. 10, p. 14301–14310, 2022.

[60] AHMAD, I.; WAN, Z.; AHMAD, A. A big data analytics for ddos attack detection using optimized ensemble framework in internet of things. *Internet of Things*, v. 23, p. 100825, 2023. ISSN 2542-6605. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2542660523001488>.

[61] DING, S.; ZHU, Z.; ZHANG, X. An overview on semi-supervised support vector machine. *Neural Computing and Applications*, Springer, v. 28, p. 969–978, 2017.

[62] HUANG, C.; DAVIS, L.; TOWNSHEND, J. An assessment of support vector machines for land cover classification. *International Journal of remote sensing*, Taylor & Francis, v. 23, n. 4, p. 725–749, 2002.

[63] MAMMONE, A.; TURCHI, M.; CRISTIANINI, N. Support vector machines. *Wiley Interdisciplinary Reviews: Computational Statistics*, Wiley Online Library, v. 1, n. 3, p. 283–289, 2009.

[64] OSUNA, E.; FREUND, R.; GIROSIT, F. Training support vector machines: an application to face detection. In: IEEE. *Proceedings of IEEE computer society conference on computer vision and pattern recognition*. [S.l.], 1997. p. 130–136.

[65] KALITA, D. J.; SINGH, V. P.; KUMAR, V. A survey on svm hyper-parameters optimization techniques. In: SPRINGER. *Social Networking and Computational Intelligence: Proceedings of SCI-2018*. [S.l.], 2020. p. 243–256.

[66] BURGES, C. J. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, Springer, v. 2, n. 2, p. 121–167, 1998.

[67] SHRESTHA, A.; MAHMOOD, A. Review of deep learning algorithms and architectures. *IEEE access*, IEEE, v. 7, p. 53040–53065, 2019.

[68] GOODFELLOW, I. et al. Generative adversarial nets. *Advances in neural information processing systems*, v. 27, 2014.

[69] YU, B.; WEI, H.; WANG, W. Gan-based day and night image cross-domain conversion research and application. In: *2022 11th International Conference of Information and Communication Technology (ICTech))*. [S.l.: s.n.], 2022. p. 230–235.

[70] ZHANG, H. et al. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In: *Proceedings of the IEEE international conference on computer vision*. [S.l.: s.n.], 2017. p. 5907–5915.

[71] LEE, C.-K.; CHEON, Y.-J.; HWANG, W.-Y. Studies on the gan-based anomaly detection methods for the time series data. *IEEE Access*, IEEE, v. 9, p. 73201–73215, 2021.

[72] SCHLEGL, T. et al. f-anogan: Fast unsupervised anomaly detection with generative adversarial networks. *Medical image analysis*, Elsevier, v. 54, p. 30–44, 2019.

[73] LI, Y. et al. The theoretical research of generative adversarial networks: an overview. *Neurocomputing*, Elsevier, v. 435, p. 26–41, 2021.

[74] NAVIDAN, H. et al. Generative adversarial networks (gans) in networking: A comprehensive survey & evaluation. *Computer Networks*, Elsevier, v. 194, p. 108149, 2021.

[75] SAJEEDA, A.; HOSSAIN, B. M. Exploring generative adversarial networks and adversarial training. *International Journal of Cognitive Computing in Engineering*, Elsevier, v. 3, p. 78–89, 2022.

[76] MOORE, G. E. Cramming more components onto integrated circuits. *Proceedings of the IEEE*, Ieee, v. 86, n. 1, p. 82–85, 1998.

[77] MOORE, G. Moore's law. *Electronics Magazine*, v. 38, n. 8, p. 114, 1965.

[78] YODER, M.; ORSAK, G. Engineering: our digital future. 2004.

[79] BUSCH, P.; HEINONEN, T.; LAHTI, P. Heisenberg's uncertainty principle. *Physics reports*, Elsevier, v. 452, n. 6, p. 155–176, 2007.

[80] BERNSTEIN, E.; VAZIRANI, U. Quantum complexity theory. In: *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*. [S.l.: s.n.], 1993. p. 11–20.

[81] BEAUREGARD, S. Circuit for shor's algorithm using 2n+ 3 qubits. *arXiv preprint quant-ph/0205095*, 2002.

[82] HEY, T. Quantum computing: an introduction. *Computing & Control Engineering Journal*, IET, v. 10, n. 3, p. 105–112, 1999.

[83] AMIRI, P. K. Quantum computers. *IEEE Potentials*, IEEE, v. 21, n. 5, p. 6–9, 2003.

[84] DEUTSCH, D. Quantum theory, the church–turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, The Royal Society London, v. 400, n. 1818, p. 97–117, 1985.

[85] BHARTI, K. et al. Noisy intermediate-scale quantum algorithms. *Reviews of Modern Physics*, APS, v. 94, n. 1, p. 015004, 2022.

[86] NIELSEN, M. A.; CHUANG, I. L. Quantum computation and quantum information. *Phys. Today*, v. 54, n. 2, p. 60, 2001.

[87] TERHAL, B. M. Quantum error correction for quantum memories. *Reviews of Modern Physics*, APS, v. 87, n. 2, p. 307, 2015.

[88] LALOë, F. Quantum entanglement. In: _____. *Do We Really Understand Quantum Mechanics?* 2. ed. [S.l.]: Cambridge University Press, 2019. p. 189–222.

[89] SAVCHUK, M.; FESENKO, A. Quantum computing: Survey and analysis. *Cybernetics and Systems Analysis*, Springer, v. 55, p. 10–21, 2019.

[90] IBM quantum computing. 2015. Disponível em: <https://www.ibm.com/quantum>.

[91] QCS Access Program : google quantum ai. Disponível em: <https://quantumai.google/quantum-computing-service/access>.

[92] CIRQ : google quantum ai. Disponível em: <https://quantumai.google/cirq>.

[93] QUANTUM computing and systems with Intel Labs: Intel®. Disponível em: <https://www.intel.com/content/www/us/en/research/quantum-computing.html>.

[94] SOUZA, P. J. et al. Computação quântica adiabática: Do teorema adiabático ao computador da d-wave. *Revista Brasileira de Ensino de Física*, SciELO Brasil, v. 43, 2021.

[95] CHO, A. *Quantum or not, controversial computer yields no speedup*. [S.l.]: American Association for the Advancement of Science, 2014.

[96] CHOW, J.; DIAL, O.; GAMBETTA, J. *IBM quantum breaks the 100-qubit processor barrier*. IBM, 2022. Disponível em: <https://research.ibm.com/blog/127-qubit-quantum-processor-eagle>.

[97] GOOGLE : Quantum computing hardware. Disponível em: <https://quantumai.google/hardware>.

[98] RAUSSENDORF, R. Measurement-based quantum computation with cluster states. *International Journal of Quantum Information*, World Scientific, v. 7, n. 06, p. 1053–1203, 2009.

[99] AMBAINIS, A.; REGEV, O. An elementary proof of the quantum adiabatic theorem. *arXiv preprint quant-ph/0411152*, 2004.

[100] DAS, A.; CHAKRABARTI, B. K. Colloquium: Quantum annealing and analog quantum computation. *Reviews of Modern Physics*, APS, v. 80, n. 3, p. 1061, 2008.

[101] AHARONOV, D. et al. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM review*, SIAM, v. 50, n. 4, p. 755–787, 2008.

[102] NAYAK, C. et al. Non-abelian anyons and topological quantum computation. *Reviews of Modern Physics*, APS, v. 80, n. 3, p. 1083, 2008.