Capítulo 4

Aprendizado de Máquina e Computação de Alto Desempenho

Manuel Binelo, Edson Luiz Padoin
Universidade Regional do Noroeste do Estado do Rio Grande do Sul

Resumo

A Inteligência Artificial é um campo que vem evoluindo radicalmente nos últimos anos. Ela tem se mostrado uma ferramenta com alto poder de capilaridade, sendo incorporada nas mais diversas áreas econômicas e sociais. Sua aplicação em áreas como automação, robótica, classificação de dados, reconhecimento de imagens e em sistemas especialistas já é bastante sólida, mas tem chamado a atenção sua aplicação em áreas relacionadas à subjetividade humana, como a criação de ilustrações, textos e música. A principal metodologia para a criação de agentes inteligentes é baseada no aprendizado máquina, que consiste em apresentar uma série de exemplos a um sistema computacional. Esses exemplos frequentemente compreendem grandes volumes de dados, e o processo de aprendizagem é um método iterativo de otimização que precisa ser repetido muitas vezes sobre esse conjunto de dados. Essas características fazem com que o desenvolvimento de modelos de inteligência artificial dependam de computação de alto desempenho. Modelos de inteligência artificial mais complexos, baseados em redes neurais artificiais com muitas camadas ocultas, empregam modelos de aprendizagem profunda (deep learning). A aprendizagem profunda é um problema de otimização de alta ordem, que exige uma capacidade de processamento ainda maior. Muitos modelos de inteligência artificial são baseados em redes neurais artificiais, que são modelos matemáticos/computacionais inspirados em neurônios biológicos. Assim como sua fonte de inspiração, sua arquitetura é massivamente paralela, o que torna possível a aplicação de diversas técnicas de computação paralela para acelerar o processo de aprendizagem. Neste minicurso serão abordados os fundamentos do aprendizado de máquina, suas implicações quanto à computação de alto desempenho, e as principais técnicas empregadas nesse contexto. Serão explorados modelos computacionais, frameworks mais utilizados, e diversos trabalhos científicos do estado da arte.

4.1. Introdução

A Inteligência Artificial (IA), e em especial o Aprendizado de Máquina (ML do Inglês *Machine Learning*), têm tido um grande crescimento, não apenas quanto aos resultados de pesquisa, mas quanto à sua aplicação na base estrutural de grandes sistemas de informação, como redes sociais e *e-commerce*, e em aplicações de interesse do usuário final, como nos sistemas geradores de conteúdo.

Como será mostrado, o ML depende de computação de alto desempenho (HPC do Inglês *High Performance Computing*), mas por outro lado, pode também contribuir para a evolução da HPC. O presente capítulo tem como objetivo explorar os conceitos relacionados a ML e HPC, as interações entre esses dois campos, e traçar algumas perspectivas.

4.2. Aprendizado de Máquina (ML)

A aprendizagem é um fenômeno complexo que envolve diferentes processos, como adquirir conhecimentos, desenvolver habilidades motoras e cognitivas por meio de instrução ou prática, organizar novos conhecimentos em representações eficazes e descobrir novos fatos e teorias por meio de observação e experimentação. Desde a era dos computadores, os pesquisadores têm buscado replicar essas capacidades em máquinas, um objetivo desafiador e fascinante na área de IA. O estudo e a modelagem computacional dos processos de aprendizagem são o foco do ML (Carbonell et al. 1983).

O ML é um subcampo da IA que envolve treinar algoritmos para realizar tarefas aprendendo padrões a partir de dados, em vez de usar programação explícita. Métodos de ML analisam automaticamente entradas e derivam respostas, ao contrário de técnicas convencionais de IA que usam critérios baseados em manuais. Exemplos de algoritmos de ML incluem k-vizinhos mais próximos, árvores de decisão e algoritmo Naive Bayes. Aprendizado de representação (RL do Iglês *Representative Learning*) é um subcampo de ML que se concentra em aprender representações de dados sem depender de pipelines de pré-processamento. Redes neurais artificiais (RNAs) são um exemplo comum de RL, e o aprendizado profundo (DL do Inglês *deep learning*) se refere a RNAs com muitas camadas, também conhecidas como redes neurais profundas (RNAPs) (Wataya et al. 2020).

De cordo com (Wang et al. 2020), os principais avanços em DL estão relacionados com quatro principais categorias de sistemas. A primeira categoria são RNAPs, que são extensões de redes neurais padrão com múltiplas camadas ocultas, permitindo representações mais complexas dos dados de entrada. As Resed Neurais Convolucionais (RNCs), inspiradas no córtex visual de animais, consistem em camadas de convolução, *pooling* e camadas completamente conectadas. As camadas de convolução e *pooling* reduzem a complexidade e diminuem o tamanho dos dados de entrada, enquanto as camadas completamente conectadas aprendem representações abstratas. Modelos baseados em RNCs têm alcançado resultados impressionantes no processamento de imagens e visão computacional.

A terceira categoria são as redes neurais recorrentes (RNRs), que são algoritmos de aprendizado profundo capazes de mapear dados de entrada sequenciais para sua saída. As RNRs possuem autoconexões entre os nós em cada camada, permitindo que memorizem informações ao longo do tempo a partir de uma sequência de dados. Apesar de

desafios quanto à sua aplicação, especialmente em relações a modelos de memória curta e longa, os modelos baseados em RNNs têm sido amplamente utilizados em problemas de aprendizado sequencial.

A quarta categoria são os modelos generativos, que têm como objetivo gerar novas amostras com variações através do aprendizado da distribuição das amostras de treinamento. Os autoencoders variacionais (VAE) e as redes adversariais generativas (GAN) são dois membros proeminentes dos modelos generativos. Modelos de aprendizado profundo frequentemente requerem uma grande quantidade de amostras rotuladas para o treinamento, o que pode ser difícil e computacionalmente caro de obter em aplicações práticas. Modelos generativos podem ajudar a aliviar esse problema e podem ser usados em várias tarefas, como reconhecimento, aprendizado semi-supervisionado, aprendizado de características não supervisionado e *denoising*.

A partir do trabalho de (Vaswani et al. 2017), modelos transformadores generativos pré-treinados (GPT) ganharam muito impulso. Seu sucesso decorre principalmente do mecanismo de auto-atenção, que permite o treinamento semi-supervisionado com grandes volumes de dados não rotulados. A auto-atenção, é um mecanismo de atenção que relaciona diferentes posições de uma única sequência para calcular uma representação da sequência. A auto-atenção tem sido usada com sucesso em uma variedade de tarefas, incluindo compreensão de leitura, sumarização abstrativa, implicação textual e aprendizado de representações de frases independentes de tarefas.

Os modelos de ML discutidos baseiam-se essencialmente em grande redes neurais artificiais, que para seu treinamento dependem de grandes volumes de dados. As seções seguintes irão discutir a HPC no contexto de ML e os desafios e oportunidades que se apresentam.

4.3. Contribuições da Computação de Alto Desempenho para o Aprendizado de Máquina

Os métodos utilizados para ML, de forma geral, são métodos iterativos, em que a cada iteração são modificados os pesos das conexões neurais (ou parâmetros do modelo), buscando minimizar o erro do aprendizado. Em resumo, o ML é um problema de otimização. Entre os métodos mais utilizados estão aqueles baseados em descida por gradiente (Pang et al. 2020, Netrapalli 2019).

A cada iteração do método, todo o volume de dados usados para o treinamento é acessado e interfaceado com a RNA, portanto existem dois aspectos fundamentais do custo computacional relacionado a ML, o primeiro é em relação à memória, seu tamanho e tempo de acesso aos dados de treinamento; já o segundo aspecto é de processamento matemático, para a solução do problema de otimização. Os algoritmos de aprendizagem podem ser considerados em seus dois aspectos principais, um sequencial, que são as iterações, e outro paralelo, que são as atualizações dos valores dos parâmetros.

4.3.1. Paralelismo de modelo

Para otimizar redes neurais profundas, é necessário analisar os dados de forma pontual, com baixa latência de atualização, devido ao aumento do tamanho dos modelos e dos

dados de treinamento. Uma abordagem escalável é distribuir e paralelizar o processo de treinamento. O paralelismo de modelo é uma ideia que consiste em dividir o modelo entre diferentes unidades de processamento, permitindo treinar modelos muito grandes que não caberiam na memória de um único dispositivo. No entanto, a eficiência do paralelismo de modelo depende da arquitetura e da forma como o modelo é dividido. Redes neurais totalmente conectadas são extremamente difíceis de ter seu modelo paralelizado já que cada camada depende do resultado da camada anterior para computar seus parâmetros, já redes mais esparsas, como redes neurais convolucionais, têm maior facilidade de paralelização do modelo (Jäger et al. 2018).

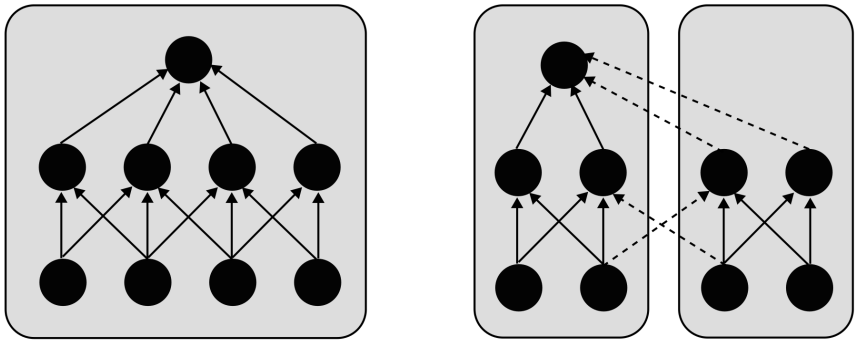

Figura 4.1. Esquema do paralelismo de modelo. Adaptado de (Jäger et al. 2018)

Como ilustrado na Figura 4.1, a rede neural é dividida verticalmente e cada unidade de processamento atualiza um conjunto de parâmetros. Para os nós da rede que possuem conexão com nós que estão em outra unidade de processamento, é necessário que o parâmetro correspondente seja enviado, o que explica o fato de que quanto mais esparsa for a rede, melhor o paralelismo de modelo funciona.

4.3.2. Paralelismo de dados

Algoritmos de paralelismo de dados utilizam vários trabalhadores que processam um subconjunto do conjunto de dados completo para escalonar a computação. Ou seja, a rede neural é replicada em cada nó de processamento que executa os mesmos passos de treinamento com dados diferentes em paralelo. Para que essa abordagem funcione, é necessário que a rede neural caiba na memória, e há duas vantagens interessantes em relação ao paralelismo de modelo. Esse método é independente da arquitetura da rede, podendo ser aplicado com sucesso em redes totalmente conectadas, e tem a possibilidade de ocultar os custos de comunicação quando um modelo assíncrono é adotado (Jäger et al. 2018).

Como mostra a Figura 4.2, o paralelismo de dados funciona da seguinte forma: um ou mais nós, chamados de servidor de parâmetros (PS), são responsáveis por calcular as atualizações de parâmetros e, se solicitado, redistribuí-las. O passo de atualização pode ser feito basicamente de duas maneiras: síncrona e assíncrona, o que tem um alto impacto no desempenho do treinamento. Quando é utilizado um modelo de atualizações síncronas, o PS espera por todas as mensagens dos trabalhadores antes de calcular as atualizações de

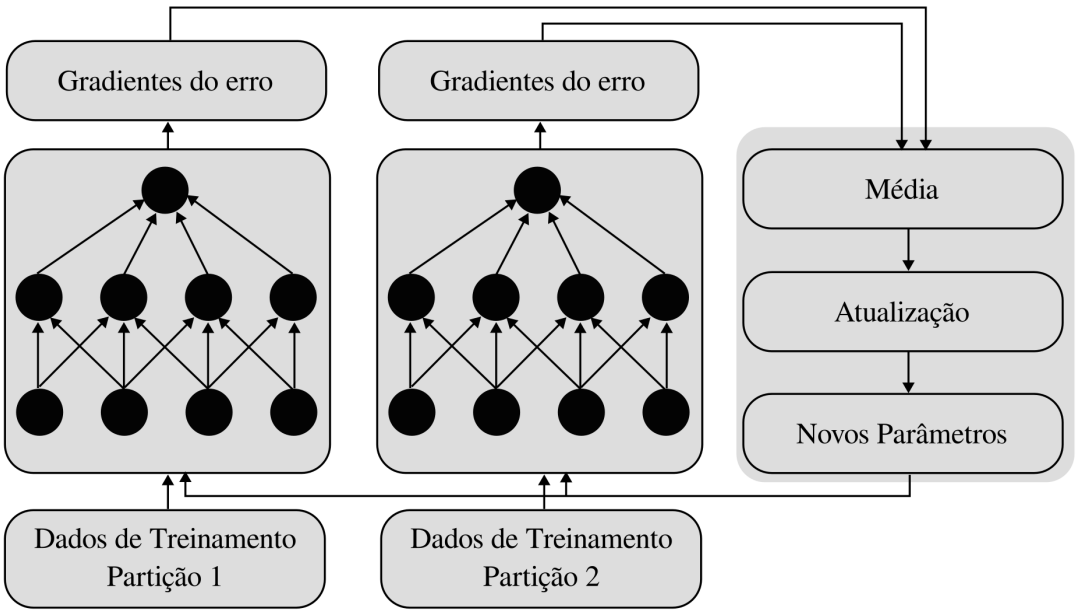

Figura 4.2. Esquema do paralelismo de dados. Adaptado de (Jäger et al. 2018)

parâmetros. Já para o modelo de atualizações assíncronas, para cada mensagem recebida, o PS calcula as atualizações de parâmetros (Jäger et al. 2018, Li et al. 2014).

4.3.3. Frameworks para ML

Diferentes frameworks estão disponíveis para computação de alto desempenho, e cada framework pode definir diferentes estratégias de paralelismo. Existem frameworks estáticos como é o caso do TensorFlow, que são baseados em duas fases. Na fase de cosntrução é definido o grafo de execução, e na fase de computação esse grafo é calculado. Possui como vantagem otimizar mais facilmente a execução, mas como desvantagem possui maior dificuldade de lidar com dados de entrada dinâmicos. Também quanto à estrutura adotada para o caso de modelos de paralelismo de dados existem diferentes abordagens. Como exemplo, o TensorFlow utiliza uma abordagem centralizada, enquanto o MXNet utiliza uma abordagem descentralizada (Jäger et al. 2018). Este capítulo não tem como objetivo trazer uma lista completa dos frameworks utilizados, mas abordará três dos mais utilizados.

4.3.4. TensorFlow

O TensorFlow é uma biblioteca de software flexível e escalável para cálculos numéricos usando grafos de fluxo de dados. Ele permite aos usuários programar e treinar eficientemente modelos de redes neurais e outras técnicas de aprendizado de máquina, e implantá-los em produção. O TensorFlow é escrito em C++ e CUDA (Compute Unified Device Architecture), uma plataforma de computação paralela criada pela NVIDIA. Ele possui APIs disponíveis em várias linguagens, sendo a API Python a mais completa e estável. Outras linguagens oficialmente suportadas incluem JavaScript, C++, Java, Go e Swift. Pacotes de terceiros estão disponíveis para mais linguagens, como C# e Ruby

(Pang et al. 2020, Kunas et al. 2023).

A arquitetura das redes neurais profundas tem tido um progresso significativo, e os framworks Keras e o TensorFlow tê tido bastante sucesso pois possuem diferentes modelos pré-treinados já incluídos em suas bibliotecas, incluindo VGG16, VGG1, ResNet50, Inception V3, Xception e MobileNet. As redes VGG e AlexNet seguem um padrão típico das redes convolucionais clássicas. Existem muitos fatores que explicam a revolução do aprendizado profundo, destacando-se a disponibilidade de conjuntos de dados enormes e de alta qualidade, o uso de computação paralela (GPU) que permite uma ativação eficiente para a retropropagação, novas arquiteturas, novas técnicas de regularização que permitem treinar redes mais extensas com menos risco de overshooting, otimizadores robustos e plataformas de software com grandes comunidades, como é o caso do TensorFlow (Sanchez et al. 2020).

Os modelos de redes neurais e outros modelos de aprendizado de máquina frequentemente envolvem multiplicação de matrizes. A arquitetura de GPU é ideal para esse tipo de computação, pois pode ser várias vezes mais rápida do que a CPU no treinamento de modelos de redes neurais. O código do TensorFlow é otimizado para ser executado em GPU, utilizando CUDA e cuDNN (biblioteca de redes neurais profundas baseada em CUDA), inclusive com o uso de várias GPUs em paralelo. A tf.distribute.Strategy é uma API do TensorFlow para distribuir o treinamento. Com mudanças mínimas no código, os pesquisadores podem distribuir modelos existentes com diferentes estratégias, como a tf.distribute.MirroredStrategy, que cria uma réplica em cada GPU e replica todas as variáveis em todas as réplicas. A tf.distribute.experimental.CentralStorageStrategy coloca todas as variáveis na CPU e replica as operações em todas as GPUs. A tf.distribute. experimental.ParameterServerStrategy designa algumas máquinas como trabalhadores e outras como servidores de parâmetros, replicando a computação em todos os trabalhadores, enquanto cada variável é colocada em um servidor de parâmetros (Pang et al. 2020).

4.3.5. Pytorch

O PyTorch é uma biblioteca em Python baseada no framework Torch, projetada para utilizar tanto GPUs quanto CPUs. Foi lançado em outubro de 2016 pelo grupo de pesquisa em inteligência artificial do Facebook. O PyTorch é amplamente utilizado em aplicações de processamento de linguagem natural e é conhecido por seu software Pyro, uma linguagem de programação probabilística desenvolvida pela Uber Research Labs. Em março de 2018, o PyTorch foi aprimorado com a incorporação do Caffe2, um framework de aprendizado profundo originário da UC Berkeley. Ao contrário de simples wrappers para outras linguagens populares, o PyTorch foi reescrito para ser rápido e ter uma aparência nativa, muitas vezes sendo usado como uma alternativa ao Numpy. O PyTorch oferece duas principais capacidades de alto nível: a computação de tensores com aceleração GPU poderosa e suporte a redes neurais profundas por meio do pacote torch.autograd (Flexa et al. 2019).

O PyTorch é um framework de aprendizado profundo que é amplamente recomendado devido à sua facilidade de uso, extensibilidade, desenvolvimento e depuração. Sua natureza Pythonica o torna popular tanto na comunidade de engenharia de software quanto entre pesquisadores e desenvolvedores. O PyTorch também é valorizado por sua capacidade de colocar modelos em produção, com um runtime C++ de alto desempenho.

É uma escolha dominante em conferências de pesquisa de aprendizado profundo. Em resumo, o PyTorch é uma plataforma avançada e fácil de usar para resolver problemas de aprendizado profundo, permitindo experimentação, depuração e implantação de forma eficiente (Ketkar et al. 2021).

4.3.6. Scikit-learn

O Scikit-learn é um pacote de aprendizado de máquina de código aberto e abrangente para Python, que é amplamente utilizado na comunidade de ciência de dados. Uma das vantagens do Scikit-learn é sua ampla cobertura de métodos de aprendizado de máquina, o que o torna uma escolha versátil para uma variedade de tarefas de aprendizado de máquina. Além disso, o Scikit-learn é conhecido por suas implementações otimizadas para eficiência computacional, uma vez que muitos dos métodos de aprendizado de máquina são baseados em bibliotecas binárias compiladas em Fortran, C ou C++, o que melhora significativamente o desempenho computacional. O Scikit-learn impõe convenções padronizadas de entrada/saída de dados e possui um procedimento de ajuste de modelo fixo, o que facilita a transição entre diferentes métodos de aprendizado de máquina dentro do pacote. Isso permite que os usuários experimentem diferentes modelos e técnicas de forma eficiente e sem problemas de integração (Hao and Ho 2019).

4.3.7. Desenvolvimentos recentes

De acordo com (Gan et al. 2020), devido à flexibilidade de seus recursos de configuração em chip, as plataformas de computação reconfiguráveis, como as FPGAs, são capazes de obter tempos de solução e eficiência energética superiores em comparação aos processadores de propósito geral. Essas plataformas têm sido amplamente adotadas em diversas aplicações importantes, desde o processamento numérico tradicional até os sistemas emergentes de aprendizado profundo. Considerando que as FPGAs têm se mostrado promissoras para a computação de alto desempenho, os autores resumem e analisam os esforços recentes relacionados a FPGAs, incluindo as abordagens industriais mais recentes, as soluções reconfiguráveis de ponta e várias questões relevantes, como o uso dos recursos em chip e a produtividade no desenvolvimento.

O trabalho de (Pyzer-Knapp et al. 2022) aborda como as novas ferramentas possibilitam novas formas de trabalho na área da ciência dos materiais. Tradicionalmente, a descoberta de materiais envolve um trabalho manual, serial e intensivo em recursos humanos, porém, essa abordagem está sendo complementada por processos automatizados, paralelos e iterativos impulsionados pela Inteligência Artificial (IA), simulação, automação experimental e computação de alto desempenho. O artigo descreve como essas novas capacidades permitem acelerar e enriquecer cada estágio do ciclo de descoberta de materiais. Utilizando o exemplo do desenvolvimento de um novo fotoresistor quimicamente amplificado, os autores demonstram como o impacto dessas tecnologias é amplificado quando utilizadas em conjunto como fluxos de trabalho heterogêneos e mais eficientes.

O trabalho de (Liu et al. 2022) aborda o tema de aprendizado federado, uma abordagem eficiente para explorar recursos de dados e computação distribuídos em dispositivos de usuários finais, em regiões ou organizações diferentes, garantindo a conformidade com leis e regulamentos, bem como a segurança e privacidade dos dados. Os autores for-

necem uma revisão abrangente das pesquisas existentes em aprendizado federado. Eles propõem uma arquitetura funcional de sistemas de aprendizado federado e uma taxonomia de técnicas relacionadas. Além disso, eles explicam os sistemas de aprendizado federado em quatro aspectos: tipos diversos de paralelismo, algoritmos de agregação, comunicação de dados e segurança dos sistemas de aprendizado federado. Os autores também apresentam quatro sistemas federados amplamente utilizados com base na arquitetura funcional proposta. Por fim, eles resumem as limitações e propõem direções para futuras pesquisas nessa área.

O artigo de (Kim et al. 2022) discute a questão do deslocamento de dados entre os dispositivos de memória e os elementos de processamento como um gargalo nos sistemas de computação tradicionais de arquitetura von-Neumann, que possuem elementos de processamento e dispositivos de memória separados, o que impacta modelos de aprendizado de máquina. Para tratar desse problema, o artigo apresenta a abordagem de computação centrada em memória, conhecida como Processing-In-Memory (PIM), que consiste em integrar os dispositivos de memória com os elementos de processamento, permitindo que os cálculos sejam realizados no mesmo local sem mover dados. Isso tem o potencial de melhorar substancialmente a eficiência energética dos sistemas de computação centrados em memória, minimizando o movimento de dados. O artigo revisa os trabalhos de pesquisa mais recentes sobre PIM, considerando diferentes tipos de dispositivos de memória, como SRAM, DRAM e ReRAM. Ele apresenta uma visão geral dos designs de PIM em cada tipo de memória, abrangendo desde células de bits até circuitos e arquiteturas. Em seguida, discute os desafios e limitações da integração de PIM com a arquitetura de computação convencional, incluindo a necessidade de novos padrões de pilha de software e os desafios associados a essa integração.

O trabalho de (Wang et al. 2021) aborda os desafios de treinar modelos de ML de grande tamanho em ambientes de Internet das Coisas (IoT) que possuem recursos limitados. Apesar de o ML ser a base para diversas aplicações de inteligência artificial, modelos grandes exigem recursos computacionais significativos para terem um desempenho adequado. A necessidade de treiná-los em ambientes de IoT com recursos limitados implica em dificuldades para desenvolver e aplicar técnicas de IA no futuro. O artigo apresenta uma revisão abrangente dos avanços recentes na redução do custo computacional durante a etapa de treinamento, mantendo a acurácia do modelo comparável. O foco é em algoritmos de otimização que visam melhorar a taxa de convergência, em arquiteturas de aprendizado distribuído que exploram recursos de computação ubíquos e em aceleração de hardware computacional e otimização de comunicação para treinamento colaborativo entre várias entidades de aprendizado.

O artigo de (Wang et al. 2022) aborda o campo em rápido crescimento da aprendizagem de máquina aprimorada por técnicas quânticas, conhecido como *quantum-enhanced machine learning*.As limitações de hardware dos computadores clássicos e o aumento do tamanho dos conjuntos de dados têm levado a comunidade a explorar novas técnicas para aprimorar a aprendizagem de máquina. A aprendizagem de máquina aprimorada por técnicas quânticas refere-se a algoritmos quânticos implementados em computadores quânticos, que podem melhorar a velocidade computacional de tarefas de aprendizagem de máquina clássicas e prometer um aumento exponencial na velocidade de computação. Nos últimos anos, o desenvolvimento de tecnologias quânticas experimentais tem levado

a muitas demonstrações experimentais de aprendizagem de máquina aprimorada por técnicas quânticas em diversos sistemas físicos. O artigo revisa o progresso experimental recente nesse campo e discute as demonstrações experimentais realizadas nessas plataformas e os desafios em andamento associados à implementação de técnicas quânticas na aprendizagem de máquina.

4.4. Contribuições do Aprendizado para Máquina para a Computação de Alto Desempenho

É evidente a contribuição da HPC para o desenvolvimento da IA, em especial do campo de ML. No entanto, algo não tão evidente à primeira vista é a contribuição de ML para a HPC. Nessa seção, algumas dessas contribuições mais recentes serão evidenciadas.

Uma questão fundamental para HPC é o projeto de novos chips. Esse tipo projeto tem inúmeras etapas de alta complexidade, sendo o projeto da máscara de litografia uma das mais demoradas e complicadas. As máscaras fotolitográficas são geralmente finas camadas de metal cromado com padrões, sobre vidro, formando efetivamente uma metassuperfície. Descobrir qual o padrão de litografia que resultará no padrão correto de luz projetada no substrato do chip é um problema complexo de otimização. Esse problema tem o nome de Design Eletromagnético Inverso, mas também é chamado de Tecnologia de Litografia Inversa (ILT) em fotomáscaras (Cecil et al. 2022). Métodos tradicionais levam semanas para calcular a máscara de chip, mas a Nvidia anunciou um novo método baseado em ML que pode acelerar em quarenta vezes esse processo, o que contribuiria para um avanço mais rápido no desenvolvimento de novos chips fundamentais para HPC (IEEESpectrum 2023).

Outra contribuição importante do ML para a HPC é utilização de modelos que aprendem o comportamento de sistemas mais complexos e que exigem um grande volume de cálculo ou tempo de simulação, aumentando a capacidade computacional por meio da substituição do modelo computacional original por outro modelo análogo baseado em ML. Em (Binelo et al. 2021), um método baseado em uma RNA foi criado para fazer a estimação e adaptação de parâmetros de modelos elétricos de descarga de baterias. Inicialmente, um algoritmo genético foi criado para estimar os parâmetros de um modelo de descarga de bateria, mas como esse método exige a execução de um grande número de simulações do modelo, sua execução em um dispositivo móvel é inviável. Ao criar uma RNA que aprende o comportamento do estimador de parâmetros, esse processo passou a ter um custo computacional muito mais baixo, abrindo a possibilidade de ser embarcado em dispositivos móveis, tendo um tempo de resposta muito mais rápido.

Dinâmica dos fluídos é um problema numérico computacional que depende de um grande volume de processamento. O trabalho de (Wang et al. 2018) apresenta um método inovador de redução de modelo baseado em métodos de aprendizado profundo. A abordagem de aprendizado profundo tem a capacidade de aprender um sistema não linear com vários níveis de representação e prever dados. No trabalho, os dados de treinamento são obtidos a partir de soluções de modelos de alta fidelidade em níveis de tempo selecionados. A RNA de memória de curto e longo prazo é usada para construir um conjunto de hipersuperfícies representando o sistema dinâmico de fluidos reduzido. O desempenho do novo modelo de ordem reduzida foi avaliado usando dois exemplos numéricos: um giro

do oceano e um fluxo passado um cilindro. Esses resultados ilustram que o custo da CPU é reduzido por várias ordens de magnitude.

Outro problema interessante de HCP é a crescente demanda por gráficos de alta qualidade em jogos, que vem aumentando o custo computacional e apresentando desafios para o hardware de processamento gráfico. Para superar essa limitação, os cientistas da computação precisam desenvolver novas maneiras criativas de melhorar o desempenho do hardware de processamento gráfico. Uma das soluções de maior sucesso é o uso de técnicas de ML para super-resolução de vídeo, que podem permitir que jogos de vídeo tenham gráficos de alta qualidade sem aumentar tanto o custo computacional. Essas tecnologias emergentes têm o potencial de melhorar o desempenho e a satisfação dos consumidores de jogos e caminham para se tornar padrão na indústria de desenvolvimento de jogos. A técnica consiste na renderização da imagem do jogo em uma resolução baixa, e então uma RNA generativa que recebe essa imagem, gera outra em resolução maior, diminuindo consideravelmente o custo computacional e gerando uma imagem bastante próxima da que seria gerada pela renderização convencional (Watson 2020).

4.5. Conclusões e Perspectivas

Os sistemas de inteligência artificial, especialmente no campo do aprendizado de máquina, vêm se desenvolvendo muito nos últimos anos, não apenas em ambientes de pesquisa ou corporativos, mas chegando até os usuários finais com taxas de adoção inéditas. Esse avanço só é possível por causa da computação de alto desempenho, que permite que sistemas com um gigantesco número de parâmetros possam ser treinados em grande volumes de dados obtidos por meio da Internet.

Outro aspecto muito interessante desse desenvolvimento é a contribuição que o aprendizado de máquina tem dado para a computação de alto desempenho. A contribuição se manifesta em métodos mais inteligentes para gerenciar os recursos computacionais, sistemas inteligentes para projetar novos chips e sistemas de computação, e também sistemas baseados em aprendizado de máquina que resolvem problemas de muita exigência computacional por meio de modelos análogos. O que tem se demonstrado é um ciclo no qual a computação de alto desempenho auxilia no avanço da inteligência artificial, que por sua vez impulsiona o avanço da computação de alto desempenho.

A possibilidade de treinar sistemas cada vez maiores com base de dados cada vez maiores, faz com que capacidades emergentes se manifestem, mostrando habilidades não previstas para esses sistemas. O ritmo com que essa evolução tem acontecido impõe uma série de desafios, não apenas técnicos, mas especialmente sociais e econômicos, que precisam ser abordados pela sociedade de forma ampla. Essa discussão e ação só pode ter lugar em uma sociedade que compreende o que é a inteligência artificial e seus impactos.

Referências

Binelo et al. 2021 Binelo, M. F., Sausen, A. T., Sausen, P. S., Binelo, M. O., and de Campos, M. (2021). Multi-phase method of estimation and adaptation of parameters of electrical battery models. *International Journal of Energy Research*, 45(1):1023–1037.

Carbonell et al. 1983 Carbonell, J. G., Michalski, R. S., and Mitchell, T. M. (1983). An

overview of machine learning. *Machine learning*, pages 3–23.

Cecil et al. 2022 Cecil, T., Peng, D., Abrams, D., Osher, S. J., and Yablonovitch, E. (2022). Advances in inverse lithography. *ACS Photonics*.

Flexa et al. 2019 Flexa, C., Gomes, W., and Viademonte, S. (2019). An exploratory study on machine learning frameworks. In *Anais do XVIII Workshop em Desempenho de Sistemas Computacionais e de Comunicação*. SBC.

Gan et al. 2020 Gan, L., Yuan, M., Yang, J., Zhao, W., Luk, W., and Yang, G. (2020). High performance reconfigurable computing for numerical simulation and deep learning. *CCF Transactions on High Performance Computing*, 2:196–208.

Hao and Ho 2019 Hao, J. and Ho, T. K. (2019). Machine learning made easy: a review of scikit-learn package in python programming language. *Journal of Educational and Behavioral Statistics*, 44(3):348–361.

IEEESpectrum 2023 IEEESpectrum (2023). Nvidia speeds key chipmaking computation by 40x. https://spectrum.ieee.org/inverse-lithography.

Jäger et al. 2018 Jäger, S., Zorn, H.-P., Igel, S., and Zirpins, C. (2018). Parallelized training of deep nn: comparison of current concepts and frameworks. In *Proceedings of the Second Workshop on Distributed Infrastructures for Deep Learning*, pages 15–20.

Ketkar et al. 2021 Ketkar, N., Moolayil, J., Ketkar, N., and Moolayil, J. (2021). Recent advances in deep learning. *Deep Learning with Python: Learn Best Practices of Deep Learning Models with PyTorch*, pages 287–300.

Kim et al. 2022 Kim, D., Yu, C., Xie, S., Chen, Y., Kim, J.-Y., Kim, B., Kulkarni, J. P., and Kim, T. T.-H. (2022). An overview of processing-in-memory circuits for artificial intelligence and machine learning. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 12(2):338–353.

Kunas et al. 2023 Kunas, C. A., Padoin, E. L., and Navaux, P. O. A. (2023). Accelerating deep learning model training on cloud tensor processing unit. In *International Conference on Cloud Computing and Services Science (CLOSER)*, pages 1–8, Prague, Czech Republic. Springer.

Li et al. 2014 Li, M., Andersen, D. G., Park, J. W., Smola, A. J., Ahmed, A., Josifovski, V., Long, J., Shekita, E. J., and Su, B.-Y. (2014). Scaling distributed machine learning with the parameter server. In *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, pages 583–598.

Liu et al. 2022 Liu, J., Huang, J., Zhou, Y., Li, X., Ji, S., Xiong, H., and Dou, D. (2022). From distributed machine learning to federated learning: A survey. *Knowledge and Information Systems*, 64(4):885–917.

Netrapalli 2019 Netrapalli, P. (2019). Stochastic gradient descent and its variants in machine learning. *Journal of the Indian Institute of Science*, 99(2):201–213.

Pang et al. 2020 Pang, B., Nijkamp, E., and Wu, Y. N. (2020). Deep learning with tensorflow: A review. *Journal of Educational and Behavioral Statistics*, 45(2):227–248.

Pyzer-Knapp et al. 2022 Pyzer-Knapp, E. O., Pitera, J. W., Staar, P. W., Takeda, S., Laino, T., Sanders, D. P., Sexton, J., Smith, J. R., and Curioni, A. (2022). Accelerating materials discovery using artificial intelligence, high performance computing and robotics. *npj Computational Materials*, 8(1):84.

Sanchez et al. 2020 Sanchez, S., Romero, H., and Morales, A. (2020). A review: Comparison of performance metrics of pretrained models for object detection using the tensorflow framework. In *IOP Conference Series: Materials Science and Engineering*, volume 844, page 012024. IOP Publishing.

Vaswani et al. 2017 Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Wang et al. 2021 Wang, H., Qu, Z., Zhou, Q., Zhang, H., Luo, B., Xu, W., Guo, S., and Li, R. (2021). A comprehensive survey on training acceleration for large machine learning models in iot. *IEEE Internet of Things Journal*, 9(2):939–963.

Wang et al. 2022 Wang, X., Lin, Z., Che, L., Chen, H., and Lu, D. (2022). Experimental quantum-enhanced machine learning in spin-based systems. *Advanced Quantum Technologies*, 5(8):2200005.

Wang et al. 2020 Wang, X., Zhao, Y., and Pourpanah, F. (2020). Recent advances in deep learning. *International Journal of Machine Learning and Cybernetics*, 11:747–750.

Wang et al. 2018 Wang, Z., Xiao, D., Fang, F., Govindan, R., Pain, C. C., and Guo, Y. (2018). Model identification of reduced order fluid dynamics systems using deep learning. *International Journal for Numerical Methods in Fluids*, 86(4):255–268.

Wataya et al. 2020 Wataya, T., Nakanishi, K., Suzuki, Y., Kido, S., and Tomiyama, N. (2020). Introduction to deep learning: minimum essence required to launch a research. *Japanese journal of radiology*, 38:907–921.

Watson 2020 Watson, A. (2020). Deep learning techniques for super-resolution in video games. *arXiv preprint arXiv:2012.09810*.