e-ISSN 2764-2488

Documentos

190

Campinas, SP / Outubro, 2024

Guia de instalação do software AlphaFold para uso em Unidade de Processamento Gráfico

Embrapa
Agricultura Digital

Empresa Brasileira de Pesquisa Agropecuária
Embrapa Agricultura Digital
Ministério da Agricultura e Pecuária

e-ISSN 2764-2488

Documentos 190

Outubro, 2024

Guia de instalação do software AlphaFold para uso em Unidade de Processamento Gráfico

Jorge Luiz Correa
Ivan Mazoni

Editores técnicos

Embrapa Agricultura Digital
Campinas, SP
2024

Embrapa Agricultura Digital
Av. Dr. André Tosello, 209 - Cidade Universitária
Campinas, SP, Brasil
CEP. 13083-886
www.embrapa.br
www.embrapa.br/fale-conosco/sac

Projeto gráfico
Leandro Souza Fazio

Revisão de texto
Graziella Galinari

Diagramação
Lucas Campos Barros

Imagem de capa
Foto: João Costa Jr, com grafismos Lucas Barros

Publicação digital: PDF

Dados Internacionais de Catalogação na Publicação (CIP)
Embrapa Agricultura Digital

Correa, Jorge Luiz.

Guia de instalação do software AlphaFold para uso em Unidade de Processamento Gráfico / Jorge Luiz Correa, Ivan Mazoni. – Campinas : Embrapa Agricultura Digital, 2024.

PDF (27 p.) : il. color. – (Documentos / Embrapa Agricultura Digital, ISSN 2764-2488 ; 190).

1. AlphaFold. 2. Software. 3. Tutorial. I. Mazoni, Ivan. II. Título. III. Embrapa Agricultura Digital. IV. Série.

CDD (21. ed.) 005.15

Carla Cristiane Osawa (CRB-8/10421)

Autores

Jorge Luiz Correa
Cientista da Computação, mestre em Ciência da Computação, analista da Embrapa Agricultura Digital, Campinas, SP

Ivan Mazoni
Engenheiro eletricista, doutor em Genética e Biologia Molecular, analista da Embrapa Agricultura Digital, Campinas, SP

Apresentação

As superfícies das proteínas geralmente contêm os locais de ligação secundários (exosítios), que desempenham papéis-chave na regulação e modulação de diversas atividades, que vão desde a expressão gênica, estabilização conformacional, transporte, inibição e, por extensão, modulação e exibição de funções distintas ou multitarefas da mesma enzima em resposta a mudanças que nas condições físico-químicas são pouco compreendidas.

Uma das estratégias usadas para o estudo dos exosítios é fazer a previsão da estrutura de complexos proteína-proteína para identificar possíveis interações. Atualmente, a ferramenta que apresenta maior acurácia na predição dessas estruturas é o AlphaFold.

Este trabalho tem por objetivo servir como um tutorial de instalação do software AlphaFold em um ambiente corporativo, voltado para pesquisa científica. Além do software em si, mostra como utilizar processadores gráficos para aceleração de processamento, as GPUs (unidades de processamento gráfico, na tradução do inglês), assim como é realizada a instalação em um ambiente de cluster de GPUs. Para atender esse propósito, os autores procuraram demonstrar, passo a passo, como verificar as configurações do servidor, como instalar cada pacote de software necessário, como editar os arquivos de configuração e, finalmente, como testar a aplicação. Esperamos que esse documento oriente outros colegas na instalação e uso desse software tão importante para aqueles que trabalham com a modelagem de estruturas proteicas, bem como suas aplicações.

Stanley Robson de Medeiros Oliveira
Chefe-Geral da Embrapa Agricultura Digital

Sumário

Introdução

Prever a estrutura tridimensional de uma proteína baseado apenas em sua sequência de aminoácidos tem sido um importante problema de pesquisa, aberto há mais de 50 anos. Apesar do progresso recente, os métodos existentes ficam muito aquém da precisão atômica, especialmente quando nenhuma estrutura homóloga está disponível. O interesse dos pesquisadores do Grupo de Pesquisa em Computação Científica, Engenharia da Informação e Automação da Embrapa Agricultura Digital, que trabalham com Biologia Computacional, vai além da previsão das estruturas. Por exemplo, as superfícies das proteínas geralmente contêm os locais de ligação secundários (exosítios), que desempenham papéis-chave na regulação e modulação de diversas atividades, que vão desde a expressão gênica, estabilização conformacional, transporte, inibição e, por extensão, modulação e exibição de funções distintas ou multitarefas da mesma enzima em resposta a mudanças que nas condições físico-químicas são pouco compreendidas. Uma das estratégias usadas para o estudo dos exosítios é fazer a previsão da estrutura de complexos proteína-proteína para identificar possíveis interações. Atualmente, a ferramenta que apresenta maior acurácia na predição dessas estruturas é o AlphaFold (Jumper et al., 2021). AlphaFold é um sistema de inteligência artificial desenvolvido pela DeepMind[1] que prevê a estrutura tridimensional de uma proteína a partir da sua sequência de aminoácidos. O AlphaFold trabalha com aprendizado de máquina profundo (*Deep Learning*), incorporando conhecimento físico e biológico sobre a estrutura da proteína através do alinhamento de várias sequências. Os pesquisadores pretendem calcular os descritores físico-químicos e estruturais da plataforma STING (Neshich et al., 2003) para as estruturas previstas pelo AlphaFold. Esses dados contribuirão para a pesquisa e desenvolvimento de novos produtos, como fármacos e agroquímicos, que tem sua origem na biologia computacional estrutural.

(1) Para mais informações sobre a DeepMind, consulte https://www.deepmind.com/.

O projeto Fapesp 2020/08615-8, intitulado Exosítios de Proteínas, Sítios Crípticos e Moonlighting: identificação, mapeamento funcional e efeitos de alteração estrutural, prevê que sejam feitas análises de bioinformática e de bases de dados dos fragmentos de peptídeos, juntamente com as informações sobre as vias metabólicas afetadas, a fim de descobrir ou revelar as biomoléculas participantes/interatuantes. As proteínas que contenham segmentos lineares correspondentes a peptídeos identificados por Phage Display serão obtidas através de pesquisas utilizando o BLAST, um algoritmo para comparar informações de sequências biológicas primárias, tais como sequências de aminoácidos de diferentes proteínas ou nucleotídeos de sequências de DNA (Altschul et al., 1990). Para isso, uma análise estrutural integrando AlphaFold deve ser feita para modelar a estrutura 3D, quando necessário, e para predizer a estrutura de complexos proteína-proteína, a fim de identificar possíveis proteínas em interação.

O projeto tem liderança do professor Raghuvir Krishnaswamy Arni do Departamento de Física do Instituto de Biociências, Letras e Ciências Exatas, da Universidade de São Paulo (USP), campus de São José do Rio Preto, e participação de pesquisadores da Embrapa Agricultura Digital. Este trabalho apresenta passo a passo toda a metodologia para a instalação do software AlphaFold, desde a verificação e instalação dos pacotes de software necessários até o teste do seu funcionamento. Mostra também como utilizar processadores gráficos para aceleração de processamento (GPUs), além de como é realizada a instalação em um ambiente de Cluster de GPUs. Este documento não tem como foco apenas a ferramenta, o que, de fato, pode ser coberto pela documentação oficial. Seu público-alvo são aqueles que pretendem utilizar o Alphafold em um ambiente corporativo. Isso altera a maneira de se instalar o software coberta pela documentação oficial. Além disso, o documento mostra como fazer a instalação em um ambiente de cluster, onde o uso do recurso compartilhado de GPUs é totalmente otimizado. Este tutorial é baseado no sistema operacional Ubuntu Server, versão 22.04 LTS.

Apresenta-se a seguir todo o processo de instalação, começando pela verificação de quais drivers o sistema já possui instalado. Na sequência, será mostrado como instalar os drivers Cuda, que são necessários

para o processamento paralelo, o Docker CE, o NVIDIA Container Toolkit etc, como ajustar corretamente os arquivos de configuração do software e, finalmente, como testá-lo.

Instalação de pacotes de drivers do Cuda, Docker CE e NVIDIA Container Toolkit

A instalação de pacotes de drivers do Cuda, Docker CE e NVIDIA Container Toolkit é necessária para o funcionamento do software. Toda a parte de software deste guia é baseada no repositório do AlphaFold disponível no Github[2]. Antes de ser iniciada a instalação, é recomendável verificar a indicação dada pelo próprio sistema de drivers do Ubuntu (Figura 1).

```
# ubuntu-drivers devices
== /sys/devices/pci0000:00/0000:00:06.0 ==
modalias : pci:v000010DEd00002484sv000010DEsd0000146Bbc03sc00i00
vendor   : NVIDIA Corporation
model    : GA104 [GeForce RTX 3070]
driver   : nvidia-driver-470 - distro non-free
driver   : nvidia-driver-525-server - distro non-free
driver   : nvidia-driver-535 - distro non-free recommended
driver   : nvidia-driver-535-server - distro non-free
driver   : nvidia-driver-535-open - distro non-free
driver   : nvidia-driver-525 - distro non-free
driver   : nvidia-driver-470-server - distro non-free
driver   : nvidia-driver-535-server-open - distro non-free
driver   : nvidia-driver-525-open - distro non-free
driver   : xserver-xorg-video-nouveau - distro free builtin
```

Figura 1. Verificação dada pelo próprio sistema de drivers do Ubuntu.

(2) https://github.com/google-deepmind/alphafold

Pela saída anterior, foi detectada uma GPU GeForce RTX 3070 e recomendado o “nvidia-driver-535 - distro non-free” (Figura 2).

```
# apt install nvidia-driver-535
.
.
.
A modprobe blacklist file has been created at /etc/modprobe.d to prevent Nouveau
from loading. This can be reverted by deleting the following file:
/etc/modprobe.d/nvidia-graphics-drivers.conf

A new initrd image has also been created. To revert, please regenerate your
initrd by running the following command after deleting the modprobe.d file:
`/usr/sbin/initramfs -u`

*******************************************************************************
***Reboot your computer and verify that the NVIDIA graphics driver can   ***
*** be loaded.                                                            ***
*******************************************************************************
```

Figura 2. Instalação do driver nvidia-driver-535 - distro non-free.

Conforme as mensagens da instalação vistas na Figura 2, o módulo *nouveau* (módulo básico para placas NVIDIA) é desligado através do arquivo /etc/modprobe.d/nvidia-graphics-drivers.conf. Caso a parte de vídeo apresente algum problema, a remoção desse arquivo retornará ao estado anterior, de uso do driver básico. Para ambientes de servidores, que não possuem interfaces gráficas, as mensagens podem ser diferentes.

Para verificarmos quais módulos estão em uso, precisamos reiniciar o sistema (Figura 3).

```
# reboot
```

Figura 3. Reiniciando o Ubuntu.

Após reinicialização, é possível verificar os módulos em uso por meio dos comandos *lsmod* e *grep*, conforme mostrado na Figura 4.

```
# lsmod | grep nvidia
nvidia_uvm           1032192  0
nvidia_drm             61440  3
nvidia_modeset       1196032  6 nvidia_drm
nvidia              35266560  295 nvidia_uvm,nvidia_modeset
drm_kms_helper        237568  1 nvidia_drm
drm                   548864  7 drm_kms_helper,nvidia,nvidia_drm
i2c_nvidia_gpu         16384  0
```

Figura 4. Uso dos comandos *lsmod* e *grep* para verificar os módulos do NVIDIA estão em uso.

Cuda drivers

Em GPUs que possuem processadores CUDA, para utilizá-los em determinadas aplicações, deve-se ter instalado o Toolkit CUDA compatível com o driver da GPU instalado. Existem diversas versões de Toolkit CUDA, cada uma compatível com certas versões de drivers. No site da NVIDIA existe uma tabela que mostra quais Toolkits são compatíveis com quais versões de drivers (NVIDIA, 2024).

O pacote cuda-toolkit instalado a seguir é um metapacote compatível com a versão dos drivers instalados. Deve-se atentar para a instalação do cuda-toolki, pois ela pode remover o pacote de drivers anteriormente instalado. Não é comum que isso ocorra, mas, dependendo do estado do sistema de pacotes, pode acontecer. Outra possibilidade, essa mais provável, é que, ao declarar o novo repositório da NVIDIA para instalação do *toolkit*, o driver anteriormente instalado para a GPU seja atualizado, necessitando a reinicialização do sistema para que ele entre em funcionamento (Figura 5).

```
# apt install linux-headers-$(uname -r)
# distro=ubuntu2204
# arch=x86_64
#    wget    https://developer.download.nvidia.com/compute/cuda/repos/$distro/$arch/cuda-
keyring_1.1-1_all.deb
# dpkg -i cuda-keyring_1.1-1_all.deb
# apt update
# apt install cuda-toolkit
```

Figura 5. Instalação do cuda-toolkit.

Instalação do Docker CE

O Docker CE é o ambiente de containers utilizado pelo AlphaFold. A instalação do Docker CE está apresentada na Figura 6.

```
# apt-get update
# apt-get install ca-certificates curl gnupg
# install -m 0755 -d /etc/apt/keyrings
# curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/
keyrings/docker.gpg
# chmod a+r /etc/apt/keyrings/docker.gpg
# echo \
  “deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://
download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo “$VERSION_CODENAME”) stable” | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
# apt-get update
#   apt-get   install   docker-ce   docker-ce-cli   containerd.io   docker-buildx-plugin
docker-compose-plugin
```

Figura 6. Instalação do Docker CE.

Para realizar configurações de proxy, é preciso alterar o arquivo /etc/default/docker para o arquivo de configuração apresentado na Figura 7.

```
export http_proxy=”http://proxy.cnptia.embrapa.br:3128/”
export https_proxy=”http://proxy.cnptia.embrapa.br:3128”
export       no_proxy=”127.0.0.1,10.129.0.0/16,200.0.70.0/24,2801:80:1400::/48,localhost,.
cnptia.embrapa.br,.sbiagro.org.br,.agritempo.gov.br,.agrolivre.gov.br”
```

Figura 7. Arquivo de configuração /etc/default/Docker.

O próximo passo é criar o diretório do arquivo e o arquivo /etc/systemd/system/docker.service.d/http-proxy.conf com o conteúdo apresentado na Figura 8.

```
# mkdir -p /etc/systemd/system/docker.service.d
```

Figura 8. Criação do diretório docker.service.d.

Os comandos usados na criação do arquivo de configuração /etc/systemd/system/docker.service.d/http-proxy.conf estão apresentadas na Figura 9.

```
[Service]
Environment="HTTP_PROXY=http://proxy.cnptia.embrapa.br:3128"
Environment="HTTPS_PROXY=http://proxy.cnptia.embrapa.br:3128"
Environment="NO_PROXY=127.0.0.1,10.129.0.0/16,200.0.70.0/24,2801:80:1400::/48,localhost,.
cnptia.embrapa.br,.sbiagro.org.br,.agritempo.gov.br,.agrolivre.gov.br"
```

Figura 9. Conteúdo do arquivo /etc/systemd/system/docker.service.d/http--proxy.conf.

Em seguida, é preciso reiniciar o *systemd* usando os comandos apresentados na Figura 10.

```
# systemctl daemon-reload
# systemctl restart docker
```

Figura 10. Comandos usados para reiniciar o systemd.

Para checar se o serviço está funcionando, foi usado o comando apresentado na Figura 11.

```
# systemctl show --property=Environment Docker
```

Figura 11. Comando usado para verificar se o serviço está no ar.

Para a utilização de chamadas "*docker build*", deve-se alterar o arquivo Dockerfile inserindo a declaração de variáveis e também utilizar o parâmetro *--build-arg*. A Figura 12 apresenta um exemplo de um arquivo Dockerfile alterado.

```
FROM ubuntu
ENV http_proxy "http://proxy.cnptia.embrapa.br:3128/"
ENV https_proxy "http://proxy.cnptia.embrapa.br:3128"
ENV  no_proxy  "127.0.0.1,10.129.0.0/16,200.0.70.0/24,2801:80:1400::/48,localhost,.cnptia.
embrapa.br,.sbiagro.org.br,.agritempo.gov.br,.agrolivre.gov.br"

RUN apt-get update && apt-get install -y curl
```

Figura 12. Exemplo de arquivo Dockerfile alterado.

As chamadas com o parâmetro *--build-arg* de proxy devem estar no formato apresentado na Figura 13.

```
# docker-compose build --build-arg HTTP_PROXY=http://proxy.cnptia.embrapa.br:3128 --build-
arg HTTPS_PROXY=http://proxy.cnptia.embrapa.br:3128
```

Figura 13. Chamada para o Docker-compose usando o parâmetro *--build-arg*.

Caso algum container utilize o sistema APT[3] para a instalação de pacotes, deve-se criar um arquivo apt.conf (Figura 14) no diretório e inserir uma diretiva COPY no Dockerfile para que ele seja copiado para dentro do container.

```
Acquire::http::proxy::mirror.cnptia.embrapa.br DIRECT;
Acquire::http::proxy “http://proxy.cnptia.embrapa.br:3128/”;
Acquire::ftp::proxy “ftp://proxy.cnptia.embrapa.br:3128/”;
Acquire::https::proxy “http://proxy.cnptia.embrapa.br:3128/”;
```

Figura 14. Arquivo apt.conf

O Dockerfile deve ser alterado inserindo a diretiva apresentada na Figura 15.

```
COPY apt.conf /etc/apt/apt.conf
```

Figura 15. Diretiva para alteração do apt.conf.

NVIDIA Container Toolkit

O NVIDIA Container Toolkit permite que os usuários criem e executem containers acelerados por GPU. O kit de ferramentas inclui uma biblioteca de tempo de execução de contêiner e utilitários para configurar containers automaticamente para aproveitar as GPUs NVIDIA. Para instalar o pacote “nvidia-containter-toolkit”, procede-se como mostrado na Figura 16.

(3) O APT é sistema de gerenciamento de pacotes de programas que possui resolução automática de dependências entre pacotes, método fácil de instalação de pacotes, facilidade de operação e permite atualizar facilmente sua distribuição etc.

```
# curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --dearmor -o /
usr/share/keyrings/nvidia-container-toolkit-keyring.gpg \
  && curl -s -L https://nvidia.github.io/libnvidia-container/stable/deb/nvidia-container-
toolkit.list | \
       sed ‘s#deb  https://#deb  [signed-by=/usr/share/keyrings/nvidia-container-toolkit-
keyring.gpg] https://#g’ | \
    sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list
# apt-get update
# apt-get install -y nvidia-container-toolkit
# nvidia-ctk runtime configure --runtime=docker
# systemctl restart docker
```

Figura 16. Comandos para instalação do pacote nvidia-containter-toolkit.

Para testar se o container foi devidamente instalado, utilizar o comando apresentado na Figura 17.

```
# docker run --rm --runtime=nvidia --gpus all ubuntu nvidia-smi
Unable to find image ‘ubuntu:latest’ locally
latest: Pulling from library/ubuntu
29202e855b20: Pull complete
Digest: sha256:e6173d4dc55e76b87c4af8db8821b1feae4146dd47341e4d431118c7dd060a74
Status: Downloaded newer image for ubuntu:latest
Mon Jan 22 12:28:22 2024
+---------------------------------------------------------------------------------------+
| NVIDIA-SMI 535.154.05             Driver Version: 535.154.05   CUDA Version: 12.2     |
|-----------------------------------------+----------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M. |
|                                         |                      |               MIG M. |
|=========================================+======================+======================|
|   0  NVIDIA GeForce RTX 3070        On  | 00000000:00:06.0 Off |                  N/A |
|  0%   33C    P8              7W / 220W |      1MiB /  8192MiB |      0%      Default |
|                                         |                      |                  N/A |
+-----------------------------------------+----------------------+----------------------+

+---------------------------------------------------------------------------------------+
| Processes:                                                                            |
|  GPU   GI   CI        PID   Type   Process name                            GPU Memory |
|        ID   ID                                                             Usage      |
|=======================================================================================|
|  No running processes found                                                           |
+---------------------------------------------------------------------------------------+
```

Figura 17. Comando para testar o container.

A Figura 17 mostra que o container foi executado e pode encontrar a GPU NVIDIA GeForce RTX 3070, além de informações sobre as versões de drivers utilizadas.

Instalação do AlphaFold

Para instalação do AlphaFold, deve-se clonar o repositório Git, fazer a customização de alguns arquivos, executar os scripts de download de conjunto de dados e, por fim, tendo o ambiente Docker e a GPU funcionais, disparar o container. Para este guia, será utilizado o caminho /ALPHAFOLD como destino dos arquivos necessários para a instalação (Figura 18).

```
# cd /ALPHAFOLD
```

Figura 18. Caminho para acessar o diretório AlphaFold.

Para melhor organização da instalação do AlphaFold, é preciso criar subdiretórios, como os subdiretórios apresentados na Figura 19.

```
# mkdir /ALPHAFOLD/data /ALPHAFOLD/results
```

Figura 19. Subdiretórios dentro do AlphaFold.

Para obter o software, é preciso clonar o repositório utilizando os comandos apresentados na Figura 20.

```
# cd /ALPHAFOLD
# git clone https://github.com/deepmind/alphafold.git
```

Figura 20. Comandos usados para clonar o repositório do AlphaFold.

A seguir, precisamos instalar o aria2, que é utilitário usado para o download de arquivos, e também a linguagem de programação Python. Essas dependências são instaladas usando o comando mostrado na Figura 21.

```
# apt install aria2 python3-pip
```

Figura 21. Comando usado para instalar dependências aria2 e Python.

Para a execução do AlphaFold, foi preciso baixar os bancos de dados genéticos e de parâmetros dos modelos. Para fazer download de bancos de dados genéticos e parâmetros dos modelos, utiliza-se o script de download com seus parâmetros. A quantidade de dados ao final do download é relativamente grande e pode demorar. Não colocar o DOWNLOAD_DIR

dentro do diretório “alphafold” clonado do repositório, caso contrário, a imagem Docker construída carregará toda a base de dados, resultado em um tamanho inviável, além de um processo de geração demasiadamente lento. A Figura 22 apresenta os comandos para download dos bancos de dados genéticos e dos parâmetros dos modelos.

```
# cd /ALPHAFOLD/alphafold
# scripts/download_all_data.sh /ALPHAFOLD/data > download.log 2> download_all.log &
```

Figura 22. Comandos para download dos bancos de dados genéticos e dos parâmetros dos modelos.

O próximo passo é alterar o arquivo docker/Dockerfile adicionando as variáveis de proxy e a diretiva de cópia do arquivo apt.conf, conforme as instruções da seção anterior de instalação do Docker. A Figura 23 apresenta o conteúdo do arquivo docker/Dockerfile.

```
ARG CUDA=11.1.1
FROM nvidia/cuda:${CUDA}-cudnn8-runtime-ubuntu18.04
# FROM directive resets ARGS, so we specify again (the value is retained if
# previously set).
ENV http_proxy “http://proxy.cnptia.embrapa.br:3128/”
ENV https_proxy “http://proxy.cnptia.embrapa.br:3128”
ENV  no_proxy  “127.0.0.1,10.129.0.0/16,200.0.70.0/24,2801:80:1400::/48,localhost,.cnptia.
embrapa.br,.sbiagro.org.br,.agritempo.gov.br,.agrolivre.gov.br”
ARG CUDA
COPY apt.conf /etc/apt/apt.conf
```

Figura 23. Conteúdo do arquivo docker/Dockerfile.

Em seguida, é preciso criar o arquivo apt.conf em /ALPHAFOLD/alphafold com o conteúdo mostrado anteriormente, e instalar as dependências Python, usando o comando da Figura 24.

```
# pip3 install -r docker/requirements.txt
```

Figura 24. Comando usado para instalar as dependências do Python.

Para a criação do container do AlphaFold, é preciso utilizar o comando apresentado na Figura 25.

```
#  docker  build  --build-arg  HTTP_PROXY=http://proxy.cnptia.embrapa.br:3128  --build-arg
HTTPS_PROXY=http://proxy.cnptia.embrapa.br:3128 -f docker/Dockerfile -t alphafold .
```

Figura 25. Comando para criação do container do AlphaFold.

Por padrão, ao disparar o *docker build*, comando utilizado para geração da imagem alphafold do container, todos os arquivos que estiverem dentro do diretório “alphafold” são copiados para a imagem. Logo, é essencial que o diretório de dados esteja fora do “alphafold” ou que se utilize a funcionalidade de ignorar arquivos do Docker. Caso contrário, o container será gerado com mais de 2.2 TB de tamanho.

Para o teste inicial, foi criado um arquivo “fasta” aleatório, apenas para executar a chamada que dispara o container. O arquivo no formato FASTA (Figura 26) foi criado em /ALPHAFOLD/alphafold/fasta.fasta.

```
>P41131 Alpha-amylase precursor (1,4-alpha-D-glucan glucanohydrolase).
MHNTLFRTALLAAALGSFSHTASAEGVMVHLFEWKFNDIANECETVLGPKGFGGVQVSPP
AEHKQGSQVWWTVYQPVSYKNFNSFGGCEAELRSMIARCNAAGVKVYADAVFNHMASGSG
TATGGGSYNSGQYQYPQFGYNDFHHSGDITNYGDSNNVWNGALYGLPDLNTGSSYVQDQI
ATYMKTLLGWGVAGFRIDAAKHMAPADVKAILDKAGSPRAYLEVIGAGGESPDIQPGRYT
YIDTVTEFKYGTDLAANFNGQIKNLKTLGESWGLLPSNKAFVFVDNHDPERAHGGGGMLT
FMQGARYDLANTFMLAWPYGWKQVMSAYRFENMGTYETDKGAPGSTPCTDSQWNCEQRRP
TIMNMVLFRNRTEGQPVSNWWDNGNNQIAFSRGTRASSPSTTRAARWWPRCRRSRPASTA
TSWGAMTTAAAVMSPSTAAARPA
```

Figura 26. Arquivo no formato FASTA.

A chamada para o teste é no formato mostrado na Figura 27.

```
# cd /ALPHAFOLD/alphafold/
#   python3   docker/run_docker.py   --fasta_paths=fasta.fasta   --data_dir=/ALPHAFOLD/data/
--output_dir=/ALPHAFOLD/results/ --max_template_date=2020-05-14
```

Figura 27. Teste do AlphaFold.

Essa chamada inicia um container que analisa o arquivo passado como parâmetro. Após executá-lo, em um outro terminal, é possível checar a saída do comando de informações da NVIDIA (Figura 28) e,

consequentemente, verificar que existe um processo *python* utilizando a GPU, sendo este processo o container disparado.

```
# nvidia-smi
Tue Jan 23 22:26:00 2024
+---------------------------------------------------------------------------------------+
| NVIDIA-SMI 535.154.05             Driver Version: 535.154.05   CUDA Version: 12.2     |
|-----------------------------------------+----------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M. |
|                                         |                      |               MIG M. |
|=========================================+======================+======================|
|   0  NVIDIA GeForce RTX 3070        On  | 00000000:00:0A.0 Off |                  N/A |
|  0%   38C    P8              17W / 220W |   1952MiB /  8192MiB |      0%      Default |
|                                         |                      |                  N/A |
+-----------------------------------------+----------------------+----------------------+

+---------------------------------------------------------------------------------------+
| Processes:                                                                            |
|  GPU   GI   CI        PID   Type   Process name                            GPU Memory |
|        ID   ID                                                             Usage      |
|=======================================================================================|
|    0   N/A  N/A      2862      C   python                                      154MiB |
+---------------------------------------------------------------------------------------+
```

Figura 28. Saída do comando nvidia-smi.

O comando *docker ps* também exibe o container em execução (Figura 29).

```
# docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED         STATUS       PORTS     NAMES
2d5fdfb29d37     alphafold     “/app/run_alphafold.…”     3  minutes  ago     Up  3  minutes
busy_blackwell
```

Figura 29. Comando *docker ps*.

A partir desse momento, o software encontra-se instalado e pronto para ser utilizado para as análises. Considerando que um grande conjunto de dados será analisado, é interessante configurar o cluster de GPUs, possibilitando que elas sejam executadas em paralelo.

Configuração do cluster de GPUs

As instruções mostradas até aqui referem-se a uma instalação do AlphaFold de forma *standalone*, em um único equipamento equipado com uma GPU. No ambiente da Embrapa Agricultura Digital existe a possibilidade de uso de um cluster de GPUs, capaz de realizar

processamentos que delas façam uso, em diversos nós componentes do cluster. A seguir são mostrados os passos para utilização do AlphaFold neste ambiente.

Inicialmente, é preciso preparar a imagem no próprio cluster e clonar o repositório novamente seguindo os comandos apresentados na Figura 30.

```
mkdir /home/alphafold
mkdir /home/alphafold/data
cd /home/alphafold
git clone https://github.com/deepmind/alphafold.git
cd /home/alphafold/alphafold
```

Figura 30. Comandos para clonar o repositório do AlphaFold.

O próximo passo é alterar o docker/Dockerfile inserindo as variáveis de proxy e a diretiva de cópia do arquivo apt.conf, que deve ser colocado dentro do diretório “alphafold” clonado.

Em seguida, é preciso gerar a nova imagem em todos os nós para que todas as camadas que compõem a imagem final sejam baixadas. Apenas exportar uma imagem pronta de um ambiente Docker e importar nos outros não funciona. A Figura 31 apresenta o comando para gerar uma nova imagem do ambiente Docker.

```
docker build -f docker/Dockerfile -t alphafold
```

Figura 31. Comando para geração de uma nova imagem do ambiente Docker.

Para que seja possível a utilização das imagens por qualquer usuário, é preciso copiar os diretórios de dados em uma estrutura de arquivos no /home. A estrutura desses diretórios está em /home/alphafold e está apresentada na Figura 32.

```
/home/alphafold
        alphafold (clone do repositório)
        data
                  bfd
                  mgnify
                  params
                  pdb70
                  pdb_mmcif
                  uniclust30
                  uniref30
                  uniref90
```

Figura 32. Estrutura dos diretórios de dados.

Para que qualquer usuário consiga ler o diretório de dados e do “alphafold”, é necessário alterar as permissões conforme os comandos apresentados na Figura 33.

```
chmod -R g+r /home/alphafold/
chmod -R o+r /home/alphafold/
```

Figura 33. Comandos utilizados para mudar as permissões.

Dentro da área de cada usuário, é preciso criar uma estrutura de arquivos para permitir a execução, com os diretórios de entrada de dados e de saída de resultados. A estrutura de arquivos sugerida está apresentada na Figura 34.

```
mkdir -p $HOME/ALPHAFOLD/results
mkdir -p $HOME/ALPHAFOLD/fasta
```

Figura 34. Comandos usados na criação da estrutura de diretórios.

No diretório “fasta” deve ficar o arquivo a ser analisado. O diretório “results” conterá os resultados. Ambos deverão ser passados como parâmetro na chamada de execução.

O próximo passo é executar a instalação dos requisitos do Python para cada usuário, usando o comando da Figura 35.

```
pip3 install -r /home/alphafold/alphafold/docker/requirements.txt
```

Figura 35. Comando para executar o Python.

Para exportação do novo *PATH* que inclui o $HOME/local/bin, é necessário fechar o *bash* atual e abri-lo novamente.

Antes de executarmos o AlphaFold, precisamos copiar um arquivo de exemplo chamado fasta.fasta para dentro do diretório de entrada do usuário, usando o comando mostrado na Figura 36.

```
cp /home/alphafold/alphafold/fasta.fasta ~/ALPHAFOLD/fasta
```

Figura 36. Comando usado para copiar o arquivo fasta.fasta.

Para disparar a execução da seguinte forma, é necessário usar o comando apresentado na Figura 37.

```
echo 'python3 /home/alphafold/alphafold/docker/run_docker.py --output_dir=$HOME/ALPHAFOLD/
results  --fasta_paths=$HOME/ALPHAFOLD/fasta/fasta.fasta  --data_dir=/home/alphafold/data
--gpu_devices=$CUDA_VISIBLE_DEVICES --max_template_date=2020-05-14' | qsub
```

Figura 37. Comando para executar o AlphaFold.

Esta chamada instruirá o PBS a disparar um *job* em algum dos nós do cluster. Para acompanhamento dos *jobs* deve-se utilizar os comandos do PBS (Figura 38).

```
qstat (lista os jobs)
qdel NNNN.cgpusub (deleta o job NNNN)
```

Figura 38. Comandos para acompanhar os *jobs*.

Considerações finais

No escopo do projeto Fapesp "Exosítios de Proteínas, Sítios Crípticos e Moonlighting: identificação, mapeamento funcional e efeitos de alteração estrutural" (2020/08615-8), considerou-se que o AlphaFold poderá ser usado para modelação da estrutura 3D, na previsão da estrutura de complexos proteína-proteína para identificar possíveis interações, e que a sua instalação demonstrou-se complexa pois, conforme apresentado, foi necessária a instalação de pacotes de drivers do Cuda, Docker CE e NVIDIA Container Toolkit, além da configuração de vários arquivos. Este trabalho demonstrou como fazer tal instalação detalhadamente, passo a passo, na forma de um tutorial para que outros grupos de pesquisa possam se beneficiar das informações aqui apresentadas. Em especial, focou-se naqueles que pretendem utilizar o Alphafold em um ambiente corporativo. Isso altera a maneira de se instalar o software coberta pela documentação oficial. Além disso, o documento mostra como fazer a instalação em um ambiente de cluster, onde o uso do recurso compartilhado de GPUs é totalmente otimizado.

Referências

ALTSCHUL, S. F.; GISH, W.; MILLER, W.; MYERS, E. W.; LIPMAN, D. J. Basic local alignment search tool. **Journal of Molecular Biology**, v. 215, n. 3, p. 403-410, 1990. DOI: https://doi.org/10.1016/S0022-2836(05)80360-2.

JUMPER, J.; EVANS, R.; PRITZEL, A.; GREEN, T.; FIGURNOV, M.; RONNEBERGER, O.; TUNYASUVUNAKOOL, K.; BATES, R.; ŽIDEK, A.; POTAPENKO, A.; BRIDGLAND, A.; MEYER, C.; KOHL, S. A. A.; BALLARD, A. J.; COWIE, A.; ROMERA-PAREDES, B.; NIKOLOV, S.; JAIN, R.; ADLER, J.; BACK, T.; PETERSEN, S.; REIMAN, D.; CLANCY, E.; ZIELINSKI, M.; STEINEGGER, M.; PACHOLSKA, M.; BERGHAMMER, T.; BODENSTEIN, S.; SILVER, D.; VINYALS, O.; SENIOR, A. W.; KAVUKCUOGLU, K.; KOHLI, P.; HASSABIS, D. Highly accurate protein structure prediction with AlphaFold. **Nature**, 596, n. 7873, p. 583-589, Aug. 2021. DOI: https://doi.org/10.1038/s41586-021-03819-2.

NESHICH, G.; TOGAWA, R. C.; MANCINI, A. L.; KUSER, P. R.; YAMAGISHI, M. E. B.; PAPPAS JUNIOR, G.; TORRES, W. V.; CAMPOS, T. F. e; FERREIRA, L. L.; LUNA, F. M.; OLIVEIRA, A. G.; MIURA, R. T.; INOUE, M. K.; HORITA, L. G.; SOUZA, D. F. de; DOMINIQUINI, F.; ÁLVARO, A.; LIMA, C. S.; OGAWA, F. O.; GOMES, G. B.; PALANDRANI, J. F.; SANTOS, G. F. dos; FREITAS, E. M. de; MATTIUZ, A. R.; COSTA, I. C.; ALMEIDA, C. L. de; SOUZA, S.; BAUDET, C.; HIGA, R. H. STING Millennium: a web-based suite of programs for comprehensive and simultaneous analysis of protein structure and sequence. **Nucleic Acids Research**, v. 31, n. 13, p. 3386-3392, July 2003. DOI: https://doi.org/10.1093/nar/gkg578.

NVIDIA. **CUDA compability**. Santa Clara, [2024]. Disponível em: https://docs.nvidia.com/deploy/cuda-compatibility/index.html#deployment-consideration-forward. Acesso em: 22 fev. 2024.

CGPE 018730