Introdução a Redes Neurais

Marlon Sproesser Mathias

Aula 3

Conda

- Gerenciamento de ambientes

Docker

- Conteinerização

Conda

- Gerenciamento de ambientes virtuais
- Por que criar diferentes ambientes?
  - Compatibilidade
  - Repetibilidade

Conda

Distribuições

- Miniconda
- Anaconda

Conda

Conteúdo

- Python
- Bibliotecas
- Outras linguagens

Conda vs Docker

- Gerenciador de dependências ao nível de pacotes
- Similar ao pipenv
- Armazena versões de pacotes

- Gerenciador de dependências ao nível de sistema operacional
- Similar a uma máquina virtual

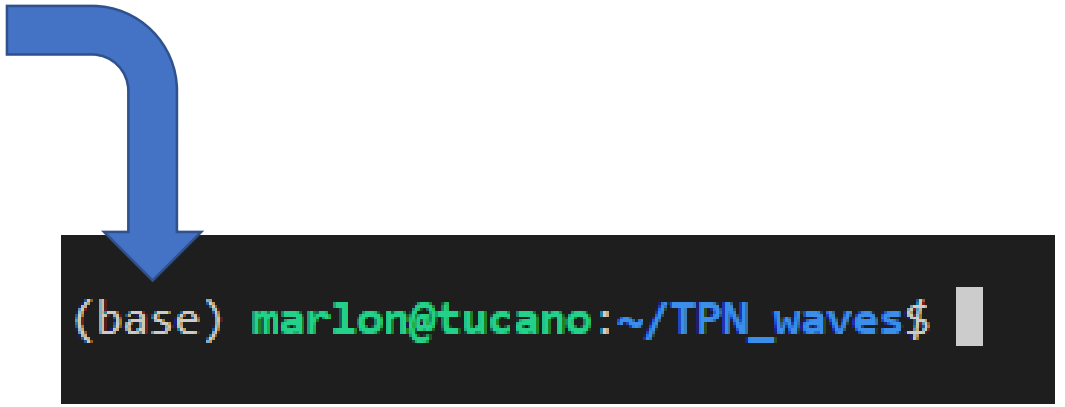
Como instalar?

- No Linux → Instalação por script
- No Windows → Arquivo executável

https://docs.conda.io/projects/conda/en/latest/user-guide/install/index.html

E depois de instalado?

- conda init
- Indica o ambiente em uso



Como criar e ativar um ambiente?

- conda create --name meu_ambiente python=3.9
- conda activate meu_ambiente

- Usar commandos
  - which python
  - conda list

Como exportar um ambiente

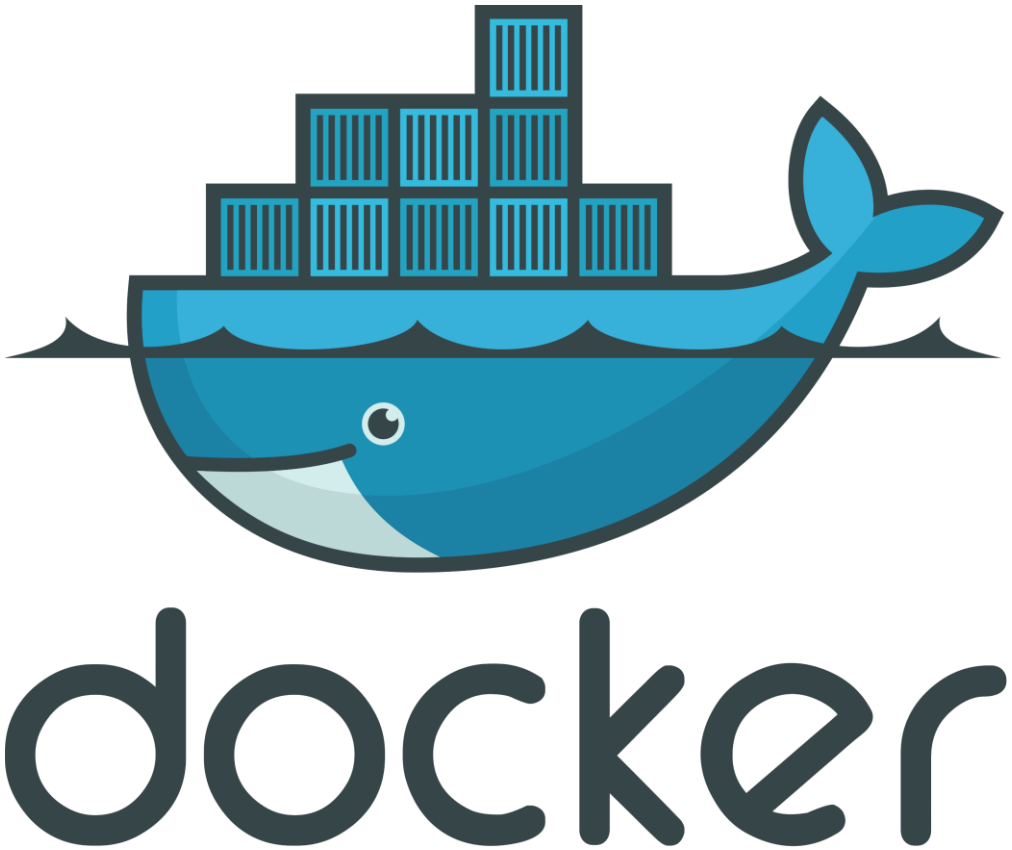
- conda list --explicit > env.yml
- conda env create --file env.yml
- Note que a versão do python não é listada no arquivo

```yaml
name: torch39
channels:
  - pytorch
  - defaults
dependencies:
  - _libgcc_mutex=0.1=main
  - _openmp_mutex=5.1=1_gnu
  - asttokens=2.0.5=pyhd3eb1b0_0
  - backcall=0.2.0=pyhd3eb1b0_0
  - blas=1.0=mkl
  - bottleneck=1.3.4=py39hce1f21e_0
  - brotli=1.0.9=he6710b0_2
  - brotlipy=0.7.0=py39h27cfd23_1003
  - bzip2=1.0.8=h7b6447c_0
  - ca-certificates=2022.4.26=h06a4308_0
  - certifi=2022.5.18.1=py39h06a4308_0
```

Onde buscar mais recursos?

- https://docs.conda.io/projects/conda/en/latest/user-guide/getting-started.html

- Cheat sheet



Docker

- Sistema de computação em contêiners
- Carrega tudo que o código precisa para ser executado
- Evita problemas com dependências e versões
- Semelhante a máquinas virtuais

Docker

Glossário

- Contêiner
  - Ambiente de execução com tudo que o código precisa para rodar
  - Isolado da máquina anfitriã
- Imagem
  - Informações de como criar um contêiner
  - Cada imagem pode gerar vários contêiners
- Volume
  - Interface entre arquivos do contêiner e da máquina anfitriã

Como executar o docker

- docker run -it --rm nvcr.io/nvidia/pytorch:22.11-py3

- -it → Modo interativo
- --rm → Apagar contêiner após execução
- nvcr.io/nvidia/pytorch:22.11-py3 → Imagem da nvidia com tudo que é necessário para rodar um modelo pytorch em Cuda

Usando um volume

- -v /home/$USER/data:/workspace/data

  Pasta na máquina | Pasta no contêiner

- O volume vincula uma pasta externa a uma pasta interna

- docker run -it --rm -v /home/$USER/data :/workspace/data nvcr.io/nvidia/pytorch:22.11-py3

Criando uma imagem

- O comando docker build cria uma nova imagem
- As instruções para criar a imagem ficam no arquivo dockerfile
- A imagem é criada e armazenada em uma pasta específica do Docker

dockerfile

```
FROM nvcr.io/nvidia/pytorch:22.11-py3
COPY . .
ENTRYPOINT ["python3", "helloworld.py"]
```

helloworld.py

```
print(“Hello world”)
```

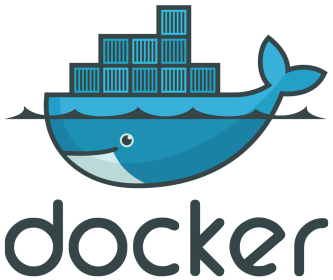
```
docker build . -t hello
```

Uma imagem mais completa

https://github.com/marlonsmathias/GATT_DNS

```
# Get base image
FROM mathworks/matlab:r2022b

RUN ulimit -c unlimited

# Copy 2decomp files
COPY etc/2decomp_fft /usr/local/2decomp_fft

# Install dependencies
RUN sudo apt -y update
RUN sudo apt -y install make gfortran-9-multilib openmpi-bin openmpi-common openmpi-doc libopenmpi-dev

# Compile 2decomp
WORKDIR /usr/local/2decomp_fft
RUN sudo make all

# Go back to the main directory
WORKDIR /home/dns
ENTRYPOINT ["matlab"]
```

Em resumo

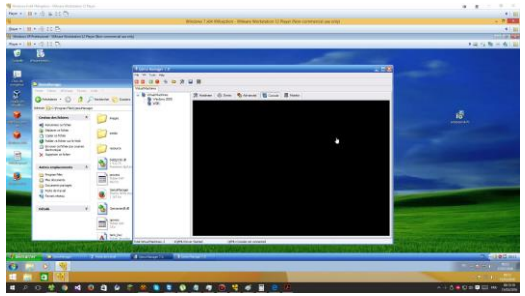


Docker

- Código executado em contêiners
- Compartilhamento de arquivos por volumes
- Impacto mínimo na performance
- Ambiente controlado, com todas as dependências e versões
- Construção das imagens em camadas reduz tamanho total em disco

Conda

- Código executado na máquina anfitriã
- Acesso direto aos arquivos
- Acesso direto ao hardware
- Pode haver conflito de dependências e versões
- Não ocupa espaço extra em disco



Máquina virtual

- Código executado numa máquina virtual
- Compartilhamento de arquivos por volumes
- Certo overhead no processamento
- Ambiente controlado, com todas as dependências e versões
- Cada máquina virtual carrega seu sistema operacional