**1. 安装依赖项**

确保安装了所需的Python库。您可以使用pip来安装：

pip install datatrove

**2. 确保GPU驱动和CUDA安装正确**

确保您的VM上已经安装了适当的GPU驱动程序和CUDA工具包。您可以使用以下命令检查CUDA是否安装正确：

nvidia-smi

**3. 修改脚本以适应本地文件系统**

假设您将WARC文件存储在本地文件系统中，而不是S3存储桶。您需要修改脚本中的路径和相关配置。以下是修改后的脚本示例：

import sys

from datatrove.executor.local import LocalPipelineExecutor

from datatrove.pipeline.extractors import Trafilatura

from datatrove.pipeline.filters import (

GopherQualityFilter,

GopherRepetitionFilter,

LanguageFilter,

URLFilter,

)

from datatrove.pipeline.readers import WarcReader

from datatrove.pipeline.writers.jsonl import JsonlWriter

# DUMP should be given as an argument. Example: CC-MAIN-2023-23

if len(sys.argv) != 2:

print("Argument required: dump name")

sys.exit(-1)

DUMP = sys.argv[1]

MAIN\_OUTPUT\_PATH = "/path/to/local/output/directory"

executor = LocalPipelineExecutor(

job\_name=f"cc\_{DUMP}",

pipeline=[

WarcReader(

f"/path/to/local/warc/files/{DUMP}/segments/",

glob\_pattern="\*/warc/\*", # we want the warc files

default\_metadata={"dump": DUMP},

),

URLFilter(exclusion\_writer=JsonlWriter(f"{MAIN\_OUTPUT\_PATH}/removed/url/{DUMP}")),

Trafilatura(favour\_precision=True),

LanguageFilter(

exclusion\_writer=JsonlWriter(

f"{MAIN\_OUTPUT\_PATH}/non\_english/",

output\_filename="${language}/" + DUMP + "/${rank}.jsonl.gz", # folder structure: language/dump/file

)

),

GopherRepetitionFilter(exclusion\_writer=JsonlWriter(f"{MAIN\_OUTPUT\_PATH}/removed/repetitive/{DUMP}")),

GopherQualityFilter(exclusion\_writer=JsonlWriter(f"{MAIN\_OUTPUT\_PATH}/removed/quality/{DUMP}")),

JsonlWriter(f"{MAIN\_OUTPUT\_PATH}/output/{DUMP}"),

],

tasks=8000,

time="10:00:00",

logging\_dir=f"{MAIN\_OUTPUT\_PATH}/logs/base\_processing/{DUMP}",

slurm\_logs\_folder=f"logs/process\_dump/processing/base\_processing/{DUMP}/slurm\_logs",

randomize\_start\_duration=180,

mem\_per\_cpu\_gb=2,

)

executor.run()

**4. 运行脚本**

在终端中导航到脚本所在的目录，并运行脚本：

python process\_common\_crawl\_dump.py CC-MAIN-2023-23

**5. 确保使用GPU**

如果您的处理任务需要使用GPU，确保您的代码能够利用GPU资源。通常，深度学习框架（如TensorFlow或PyTorch）会自动检测并使用GPU。如果您使用的是这些框架，请确保在代码中正确配置GPU使用。例如，在PyTorch中，您可以这样做：

import torch

device = torch.device("cuda" if torch.cuda.is\_available() else "cpu")

model = MyModel().to(device)

**总结**

通过这些步骤，您可以在单个GPU VM上运行您的Python脚本，而不需要复杂的分区配置。确保安装必要的依赖项和驱动程序，并根据您的本地环境调整脚本中的路径和配置。这样，您可以充分利用GPU资源来加速数据处理任务。