Đề tài: Xử lý dữ liệu lớn với kỹ thuật Đối sánh mẫu
Nhóm 3 – 64HTTT1

Phần I. Chạy MapReduce với Image Encoder

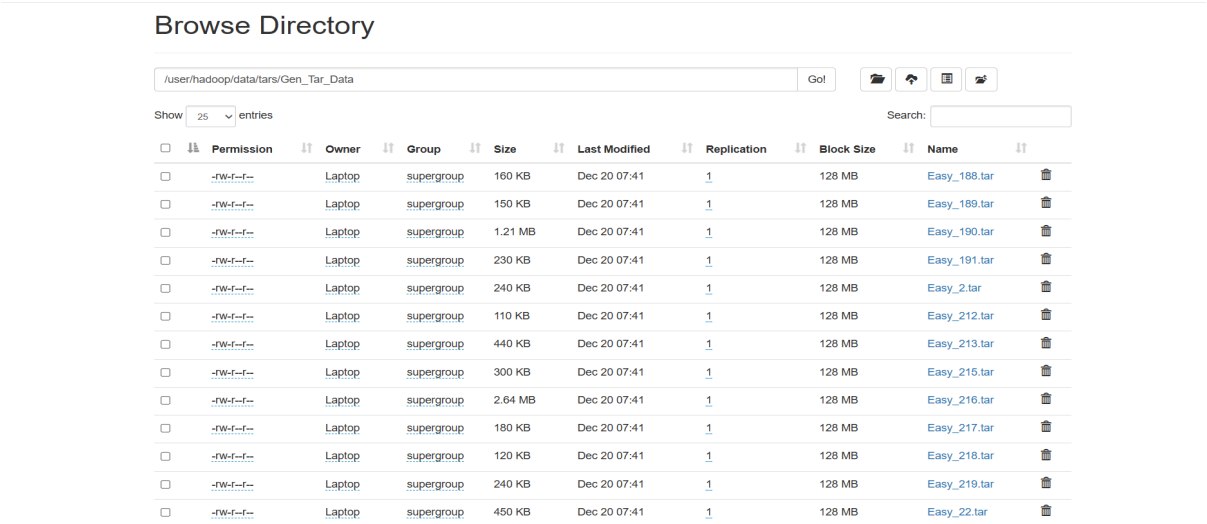
Bước 1: Nén ảnh thành file .tar để upload lên HDFS

Easy_2.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	240 KB
Easy_5.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	150 KB
Easy_6.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	70 KB
Easy_7.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	290 KB
Easy_22.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	450 KB
Easy_23.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	580 KB
Easy_24.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	130 KB
Easy_25.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	310 KB
Easy_26.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	120 KB
Easy_27.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	1,040 KB
Easy_28.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	50 KB
Easy_29.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	1,040 KB
Easy_30.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	530 KB
Easy_31.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	920 KB
Easy_32.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	1,560 KB
Easy_33.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	7,730 KB
Easy_34.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	26,720 KB
Easy_35.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	200 KB
Easy_36.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	280 KB
Easy_37.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	70 KB
Easy_38.tar	12/18/2025 6:16 PM	Tập tin nén WinRAR	100 KB

Bước 2: tạo thư mục chứa ảnh: hdfs dfs -mkdir /user/hadoop/data/tars/Gen_Tar_Data

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	Laptop	supergroup	0 B	Dec 20 07:43	0	0 B	Gen_Tar_Data

Bước 3: đẩy toàn bộ các file ảnh lên thư mục vừa tạo bằng lệnh: hdfs dfs -put D:\BigData\CustomsTMR\Gen_Tar_Data*.tar /user/hadoop/data/tars/Gen_Tar_Data



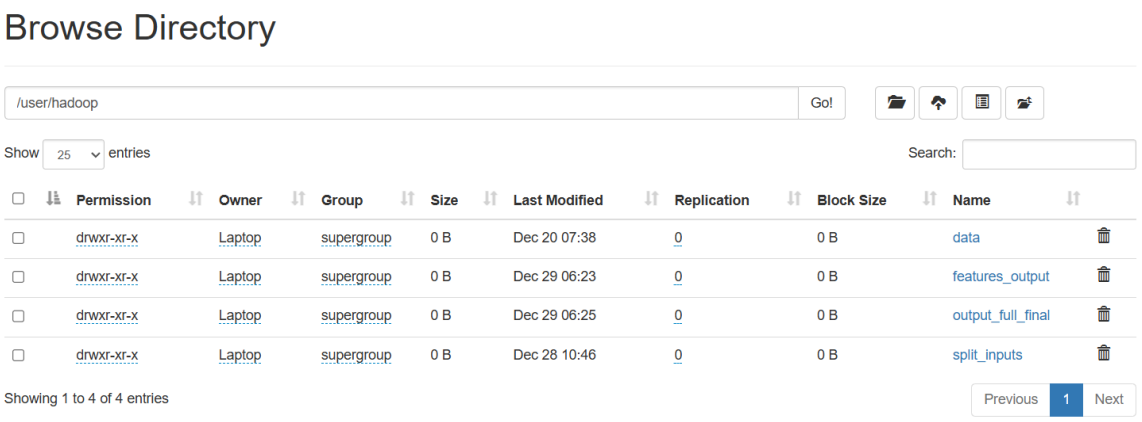
Browse Directory

/user/hadoop/data/tars/Gen_Tar_Data Go!

Show 25 entries Search:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	Laptop	supergroup	160 KB	Dec 20 07:41	1	128 MB	Easy_188.tar
-rw-r--r--	Laptop	supergroup	150 KB	Dec 20 07:41	1	128 MB	Easy_189.tar
-rw-r--r--	Laptop	supergroup	1.21 MB	Dec 20 07:41	1	128 MB	Easy_190.tar
-rw-r--r--	Laptop	supergroup	230 KB	Dec 20 07:41	1	128 MB	Easy_191.tar
-rw-r--r--	Laptop	supergroup	240 KB	Dec 20 07:41	1	128 MB	Easy_2.tar
-rw-r--r--	Laptop	supergroup	110 KB	Dec 20 07:41	1	128 MB	Easy_212.tar
-rw-r--r--	Laptop	supergroup	440 KB	Dec 20 07:41	1	128 MB	Easy_213.tar
-rw-r--r--	Laptop	supergroup	300 KB	Dec 20 07:41	1	128 MB	Easy_215.tar
-rw-r--r--	Laptop	supergroup	2.64 MB	Dec 20 07:41	1	128 MB	Easy_216.tar
-rw-r--r--	Laptop	supergroup	180 KB	Dec 20 07:41	1	128 MB	Easy_217.tar
-rw-r--r--	Laptop	supergroup	120 KB	Dec 20 07:41	1	128 MB	Easy_218.tar
-rw-r--r--	Laptop	supergroup	240 KB	Dec 20 07:41	1	128 MB	Easy_219.tar
-rw-r--r--	Laptop	supergroup	450 KB	Dec 20 07:41	1	128 MB	Easy_22.tar

Bước 4: Tạo thư mục chứa file đường dẫn file ảnh bằng lệnh: hdfs dfs -mkdir /user/hadoop/split_inputs/



Browse Directory

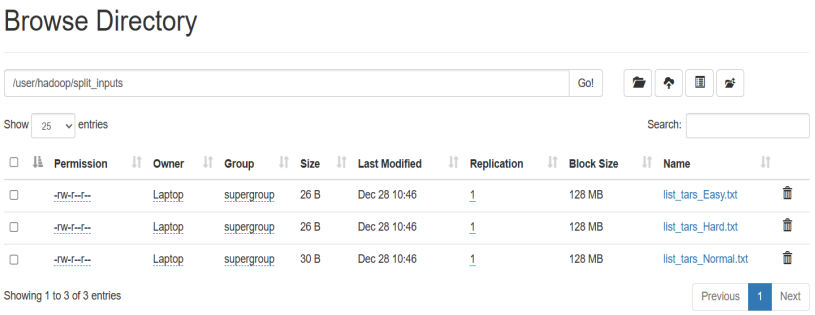
/user/hadoop Go!

Show 25 entries Search:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	Laptop	supergroup	0 B	Dec 20 07:38	0	0 B	data
drwxr-xr-x	Laptop	supergroup	0 B	Dec 29 06:23	0	0 B	features_output
drwxr-xr-x	Laptop	supergroup	0 B	Dec 29 06:25	0	0 B	output_full_final
drwxr-xr-x	Laptop	supergroup	0 B	Dec 28 10:46	0	0 B	split_inputs

Showing 1 to 4 of 4 entries Previous 1 Next

Bước 5: Đẩy file .txt chứa đường dẫn các file ảnh vào thư mục mới tạo bằng lệnh: hdfs dfs -put D:\BigData\CustomsTMR\Template-Matching-and-Regression\list_tars_Easy.txt /user/hadoop/split_inputs/ (Tương tự với 2 file Normal và Hard)



Bước 6: Chuẩn bị file model ViT đã được tối ưu dưới dạng ONNX

Bước 7: Tạo 2 file code python mapper.py:

```
import sys
import os
import tarfile
import numpy as np
import onnxruntime as ort
import subprocess
import shutil
from PIL import Image

HADOOP_CMD = "D:/hadoop-3.3.0/bin/hadoop.cmd"
HDFS_OUTPUT_DIR = "/user/hadoop/features_output"

def get_category(folder_name):
    """Xác định nhãn Easy/Normal/Hard từ tên thư mục"""
    if folder_name.startswith("Easy_"): return "Easy"
    if folder_name.startswith("Normal_"): return "Normal"
    if folder_name.startswith("Hard_"): return "Hard"
    return "Unknown"

def preprocess_image(img_path, input_shape=(1024, 1024)):
    """Đọc ảnh, resize và chuẩn hóa đầu vào cho Model"""
    try:
```

```
        img = Image.open(img_path).convert('RGB')
        img = img.resize(input_shape)
        img_array = np.array(img).astype(np.float32) / 255.0
        img_array = np.transpose(img_array, (2, 0, 1))
        img_array = np.expand_dims(img_array, axis=0)
        return img_array
    except Exception as e:
        return None

def main():
    if not os.path.exists("model.onnx"):
        sys.stderr.write("FATAL: model.onnx not found!\n")
        return

    try:
        sess_options = ort.SessionOptions()
        sess_options.log_severity_level = 3
        ort_session = ort.InferenceSession("model.onnx", sess_options)
        input_name = ort_session.get_inputs()[0].name
    except Exception as e:
        sys.stderr.write(f"FATAL: Model load failed: {e}\n")
        return

    for line in sys.stdin:
        tar_filename = line.strip()
        if not tar_filename: continue

        folder_name = tar_filename.replace(".tar", "")
        category = get_category(folder_name)

        tar_sum_mean = 0.0
        tar_sum_std  = 0.0
        tar_sum_max  = 0.0
        tar_sum_spar = 0.0
        tar_image_count = 0

        try:
            if os.path.exists(folder_name): shutil.rmtree(folder_name,
ignore_errors=True)

            hdfs_tar_path =
f"/user/hadoop/data/tars/Gen_Tar_Data/{tar_filename}"
```

```
            subprocess.check_call([HADOOP_CMD, "fs", "-get",
hdfs_tar_path, "."], stderr=subprocess.DEVNULL)

            with tarfile.open(tar_filename, "r") as tar:
                tar.extractall(path=folder_name)

            os.remove(tar_filename)
        except Exception as e:
            sys.stderr.write(f"Error Setup {tar_filename}: {e}\n")
            continue
        output_local_folder = f"out_{folder_name}"
        os.makedirs(output_local_folder, exist_ok=True)

        for root, dirs, files in os.walk(folder_name):
            for file in files:
                if file.lower().endswith(('.png', '.jpg', '.jpeg')):
                    img_path = os.path.join(root, file)

                    img_tensor = preprocess_image(img_path)
                    if img_tensor is None: continue

                    try:
                        feature = ort_session.run(None, {input_name:
img_tensor})[0]
                        val_mean = float(np.mean(feature))
                        val_std  = float(np.std(feature))
                        val_max  = float(np.max(feature))
                        val_spar = float(np.mean(feature <= 0))

                        tar_sum_mean += val_mean
                        tar_sum_std  += val_std
                        tar_sum_max  += val_max
                        tar_sum_spar += val_spar
                        tar_image_count += 1
                        save_name = f"{os.path.splitext(file)[0]}.npy"
                        np.save(os.path.join(output_local_folder,
save_name), feature)

                    except Exception as e:
                        pass

        if tar_image_count > 0:
```

```
            final_hdfs_path =
f"{HDFS_OUTPUT_DIR}/{category}/{folder_name}"
            try:
                subprocess.call([HADOOP_CMD, "fs", "-rm", "-r",
final_hdfs_path], stderr=subprocess.DEVNULL)
                subprocess.call([HADOOP_CMD, "fs", "-mkdir", "-p",
f"{HDFS_OUTPUT_DIR}/{category}"], stderr=subprocess.DEVNULL)
                subprocess.check_call([HADOOP_CMD, "fs", "-put",
output_local_folder, final_hdfs_path], stderr=subprocess.DEVNULL)
                sys.stderr.write(f"Processed {tar_filename}:
{tar_image_count} images\n")
            except Exception as e:
                sys.stderr.write(f"Upload Failed {tar_filename}:
{e}\n")


print(f"{category}\t{tar_sum_mean},{tar_sum_std},{tar_sum_max},{tar_sum
_spar},{tar_image_count}")

        shutil.rmtree(folder_name, ignore_errors=True)
        shutil.rmtree(output_local_folder, ignore_errors=True)

if __name__ == "__main__":
    main()
```

File [reducer.py]():

```
import sys
import numpy as np

def process_batch_and_print(category, stats_list):
    """Tính toán thống kê từ dữ liệu đã tổng hợp"""
    if not stats_list:
        sys.stderr.write(f"[WARNING] No stats for category:
{category}\n")
        return

    try:
        total_images = sum(s['count'] for s in stats_list)
        total_mean = sum(s['sum_mean'] for s in stats_list)
        total_std = sum(s['sum_std'] for s in stats_list)
        total_max = sum(s['sum_max'] for s in stats_list)
        total_spar = sum(s['sum_spar'] for s in stats_list)
```

```
        avg_mean = total_mean / total_images
        avg_std = total_std / total_images
        avg_max = total_max / total_images
        avg_spar = total_spar / total_images

        print(f"{category:<12} | {total_images:>6} | "
            f"{avg_mean:>8.4f} | {avg_std:>8.4f} | "
            f"{avg_max:>8.4f} | {avg_spar:>7.2%}")

        sys.stderr.write(f"[INFO] Completed {category}: {total_images}
images from {len(stats_list)} TARs\n")

    except Exception as e:
        sys.stderr.write(f"[ERROR] Failed to calculate stats for
{category}: {e}\n")

def main():
    current_category = None
    batch_stats = []

    print(f"{'CATEGORY':<12} | {'IMAGES':>6} | "
        f"{'AVG_MEAN':>8} | {'AVG_STD':>8} | "
        f"{'AVG_MAX':>8} | {'SPARSITY':>9}")
    print("-" * 70)

    sys.stderr.write("[INFO] Reducer started\n")
    line_count = 0

    for line in sys.stdin:
        line = line.strip()
        if not line: continue

        line_count += 1
        parts = line.split('\t')
        if len(parts) != 2:
            sys.stderr.write(f"[WARNING] Invalid line format:
{line}\n")
            continue

        category = parts[0]
        stats_str = parts[1]
```

```
        try:
            values = stats_str.split(',')
            if len(values) != 5:
                sys.stderr.write(f"[WARNING] Invalid stats format:
{stats_str}\n")
                continue

            stats = {
                'sum_mean': float(values[0]),
                'sum_std': float(values[1]),
                'sum_max': float(values[2]),
                'sum_spar': float(values[3]),
                'count': int(values[4])
            }
        except Exception as e:
            sys.stderr.write(f"[ERROR] Failed to parse stats:
{stats_str} - {e}\n")
            continue

        if current_category and category != current_category:
            process_batch_and_print(current_category, batch_stats)
            batch_stats = []

        current_category = category
        batch_stats.append(stats)

        if line_count % 100 == 0:
            sys.stderr.write(f"[PROGRESS] Processed {line_count}
lines\n")

    if current_category and batch_stats:
        process_batch_and_print(current_category, batch_stats)

    sys.stderr.write(f"[INFO] Reducer finished. Total lines:
{line_count}\n")

if __name__ == "__main__":
    main()
```

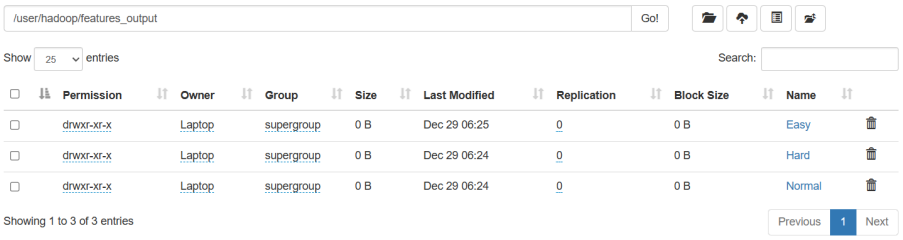
Bước 8: Di chuyển đến thư mục chứa các file code:

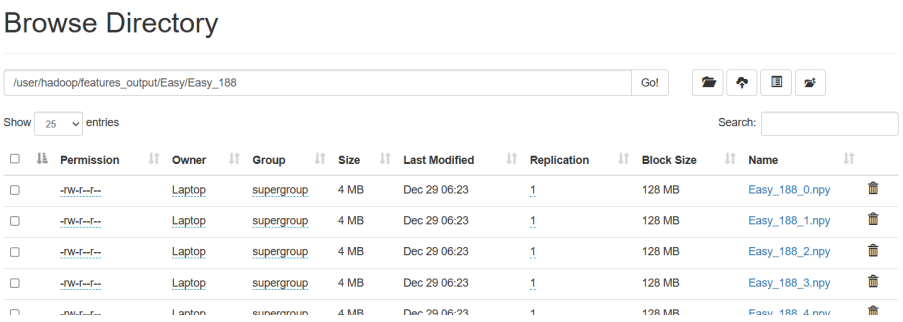
```
C:\Windows\System32>cd /d D:\BigData\CustomsTMR\Template-Matching-and-Regression

D:\BigData\CustomsTMR\Template-Matching-and-Regression>
```

Chạy lệnh sau:
hadoop jar "D:\hadoop-3.3.0\share\hadoop\tools\lib\hadoop-streaming-3.3.0.jar" -D mapreduce.job.name="Full_Job_Multi_Splits" -D mapreduce.map.memory.mb=3072 -D mapreduce.reduce.memory.mb=3072 -D mapreduce.task.timeout=0 -files "file:///D:/BigData/CustomsTMR/Template-Matching-and-Regression/mapper.py,file:///D:/BigData/CustomsTMR/Template-Matching-and-Regression/reducer.py,file:///D:/BigData/CustomsTMR/Template-Matching-and-Regression/model.onnx" -input /user/hadoop/split_inputs -output /user/hadoop/output_full_final -mapper "C:\Users\Laptop\AppData\Local\Programs\Python\Python311\python.exe mapper.py" -reducer "C:\Users\Laptop\AppData\Local\Programs\Python\Python311\python.exe reducer.py"

Chờ đợi lệnh chạy xong và kiểm tra kết quả thu được ở thư mục chứa Output trên HDFS:





Kết quả thu được những file đặc trưng .npy và file output đánh giá: