D-SAIL – Installation procedure

Description

This document describes the steps to follow to deploy the D-SAIL application on Docker and in a Kubernetes cluster.

The application illustrated below (Fig. 1) is composed of four containers and requires access to a volume containing the data to be processed.

More information is available on the following repository https://github.com/XavierLessage/D-SAIL.

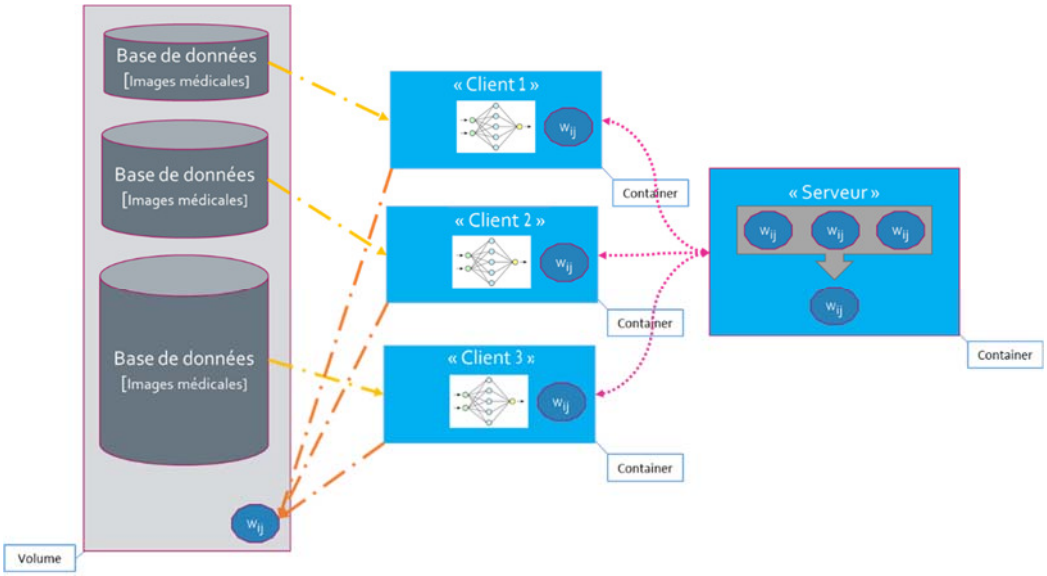

Fig. 1: Architecture of D-SAIL.

How to install D-SAIL on Docker?

Here are the command lines to run and some screenshots related to this installation. The following video [dsail-docker.mp4] will show you in detail the results of the execution and the start of the training when the server and the 3 clients are available.

<# Server | Starting #>
docker run -d -p 8080:8080 --name dsailserver dsail-server

```
PS C:\CETIC\ARIAC\FL-DSAIL> <# Server | Starting #>
docker run -d -p 8080:8080 --name dsailserver dsail-server
docker logs dsailserver
c62c73c4d4e80e375059811c41c2a5c43851651c5d34dacd4f7b18e70835e123
docker : INFO flower 2021-11-25 20:59:13,654 | app.py:73 | Flower server running (insecure, 3 rounds)
Au caractère Ligne:3 : 1
+ docker logs dsailserver
+ ~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (INFO flower 202...cure, 3 rounds):String) [], RemoteException
    + FullyQualifiedErrorId : NativeCommandError

INFO flower 2021-11-25 20:59:13,655 | server.py:118 | Getting initial parameters
```

<# Clients | Starting #>
docker run -d -p 8081 --name dsailclient1 -v /c/Users/pvdata/H1:/dataset --link dsailserver dsail-client

```
PS C:\CETIC\ARIAC\FL-DSAIL> <# Clients | Starting #>

PS C:\CETIC\ARIAC\FL-DSAIL> docker run -d -ti -p 8081 --name dsailclient1 -v /c/Users/pvdata/H1:/dataset --link dsailserver dsail-client
cd350b97d4eb86aef716dcaca1e80e0d0005263cb6241c1d51baf2d2a0cbafd8
```

docker logs dsailclient1

```
  97% 43.4M/44.7M [00:19<00:00, 3.19MB/s]
  98% 43.8M/44.7M [00:19<00:00, 3.29MB/s]
  99% 44.1M/44.7M [00:19<00:00, 3.26MB/s]
 100% 44.4M/44.7M [00:19<00:00, 3.28MB/s]
 100% 44.7M/44.7M [00:19<00:00, 2.38MB/s]
 [W NNPACK.cpp:79] Could not initialize NNPACK! Reason: Unsupported hardware.
 DEBUG flower 2021-11-25 21:00:16,316 | connection.py:36 | ChannelConnectivity.IDLE
 DEBUG flower 2021-11-25 21:00:16,317 | connection.py:36 | ChannelConnectivity.CONNECTING
 DEBUG flower 2021-11-25 21:00:16,318 | connection.py:36 | ChannelConnectivity.READY
 INFO flower 2021-11-25 21:00:16,319 | app.py:61 | Opened (insecure) gRPC connection
```

docker run -d -p 8082 --name dsailclient2 -v /c/Users/pvdata/H2:/dataset --link dsailserver dsail-client

```
PS C:\CETIC\ARIAC\FL-DSAIL> docker run -d -ti -p 8082 --name dsailclient2 -v /c/Users/pvdata/H2:/dataset --link dsailserver dsail-client
158bd12170d5d13163f8c4fad31c6d147c1dcbcbafe50cdb382fe3f65bc9fb78
```

docker logs dsailclient2

```
  99% 44.3M/44.7M [00:22<00:00, 1.93MB/s]
 100% 44.5M/44.7M [00:23<00:00, 1.43MB/s]
 100% 44.7M/44.7M [00:23<00:00, 2.02MB/s]
 [W NNPACK.cpp:79] Could not initialize NNPACK! Reason: Unsupported hardware.
 INFO flower 2021-11-25 21:01:14,427 | app.py:61 | Opened (insecure) gRPC connection
 DEBUG flower 2021-11-25 21:01:14,428 | connection.py:36 | ChannelConnectivity.IDLE
 DEBUG flower 2021-11-25 21:01:14,429 | connection.py:36 | ChannelConnectivity.READY
```

docker run -ti -p 8083 --name dsailclient3 -v /c/Users/pvdata/H3:/dataset --link dsailserver dsail-client

```
PS C:\CETIC\ARIAC\FL-DSAIL> docker run -ti -p 8083 --name dsailclient3 -v /c/Users/pvdata/H3:/dataset --link dsailserver dsail-client
DSAIL | Client Starting
Setting-up type transforms pipelines
Collecting items from /dataset
Found 1677 items
```

docker logs dsailclient3

```
  99% 44.4M/44.7M [00:23<00:00, 2.00MB/s]
 100% 44.7M/44.7M [00:23<00:00, 1.99MB/s]
 [W NNPACK.cpp:79] Could not initialize NNPACK! Reason: Unsupported hardware.
 INFO flower 2021-11-25 21:02:07,283 | app.py:61 | Opened (insecure) gRPC connection
 DEBUG flower 2021-11-25 21:02:07,287 | connection.py:36 | ChannelConnectivity.IDLE
 DEBUG flower 2021-11-25 21:02:07,287 | connection.py:36 | ChannelConnectivity.READY
 â–¯
 epoch     train_loss  valid_loss  accuracy  roc_auc_score  time
 â–¯

 Epoch 1/1 : || 0.00% [0/47 00:00<00:00]

 Epoch 1/1 : || 2.13% [1/47 01:04<49:25]

 Epoch 1/1 : || 4.26% [2/47 01:08<25:43 1.1712]

 Epoch 1/1 : || 6.38% [3/47 01:12<17:49 1.3386]

 Epoch 1/1 : || 8.51% [4/47 01:16<13:38 1.2359]
```

On this last operation, you can see that learning starts correctly when the third client is available.

How to install D-SAIL on Kubernetes?

Here are the command lines to run and some screenshots related to this installation. The following video [dsail-k8s.mp4] will show you in detail the results of the execution and the start of the training when the server and the 3 clients are available.

The first step is to download the following files from the D-SAIL repository:

- dsail-pv.yaml : To create the PersistantVolume

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: dsail-pv
  labels:
    type: local
spec:
  storageClassName: manual
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteMany
  hostPath:
    path: "/c/Users/pvdata"
```

- dsail-pvc.yaml : To create the PersistantVolumeClaim

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: dsail-pvc
spec:
  storageClassName: manual
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
```

- dsail-server-pod.yaml : To create the Pod “Server”

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: dsail-server
spec:
  containers:
  - name: dsail-server
    image: dsail/dsail-server
    ports:
      - containerPort: 8080
```

- dsail-client-1-pod.yaml : To create the Pod “Client 1”

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: dsail-client-1
spec:
  volumes:
  - name: dsail-dataset
    persistentVolumeClaim:
      claimName: dsail-pvc
  containers:
  - name: dsail-client-1
    image: dsail/dsail-clien
    ports:
      - containerPort: 8081
    env:
    - name: HOSPITAL_NAME
      value: "H1"
    - name: USE_NNPACK
      value: "0"
    volumeMounts:
    - mountPath: "/dataset"
      name: dsail-dataset
```

- dsail-client-2-pod.yaml : To create the Pod “Client 2”

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: dsail-client-2
spec:
  volumes:
  - name: dsail-dataset
    persistentVolumeClaim:
      claimName: dsail-pvc
  containers:
  - name: dsail-client-2
    image: dsail/dsail-client
    ports:
      - containerPort: 8082
    env:
    - name: HOSPITAL_NAME
      value: "H2"
    - name: USE_NNPACK
      value: "0"
    volumeMounts:
    - mountPath: "/dataset"
      name: dsail-dataset
```

- dsail-client-3-pod.yaml : To create the Pod “Client 3”

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: dsail-client-3
spec:
  volumes:
  - name: dsail-dataset
    persistentVolumeClaim:
      claimName: dsail-pvc
  containers:
  - name: dsail-client-3
    image: dsail/dsail-client
    ports:
      - containerPort: 8083
    env:
    - name: HOSPITAL_NAME
      value: "H3"
    - name: USE_NNPACK
      value: "0"
    volumeMounts:
    - mountPath: "/dataset"
      name: dsail-dataset
```

Next, simply enter the following instructions on the Kubernetes cluster:

kubectl get pod -o wide

```
C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl get pod -o wide
No resources found in default namespace.
```

kubectl apply -f xle-dsail-client-pv.yaml
kubectl get pv
kubectl describe pv dsail-pv

```
C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl apply -f xle-dsail-client-pv.yaml
persistentvolume/dsail-pv created

C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl get pv
NAME       CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM   STORAGECLASS   REASON   AGE
dsail-pv   10Gi       RWX            Retain           Available           manual                  10s

C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl describe pv dsail-pv
Name:            dsail-pv
Labels:          type=local
Annotations:     <none>
Finalizers:      [kubernetes.io/pv-protection]
StorageClass:    manual
Status:          Available
Claim:
Reclaim Policy:  Retain
Access Modes:    RWX
VolumeMode:      Filesystem
Capacity:        10Gi
Node Affinity:   <none>
Message:
Source:
    Type:          HostPath (bare host directory volume)
    Path:          /c/Users/pvdata
    HostPathType:
Events:            <none>
```

kubectl apply -f xle-dsail-client-pvc.yaml
kubectl apply -f xle-dsail-client-pvc.yaml
kubectl get pvc
kubectl describe pvc dsail-pvc

```
C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl apply -f xle-dsail-client-pvc.yaml
persistentvolumeclaim/dsail-pvc created

C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl get pvc
NAME        STATUS   VOLUME     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
dsail-pvc   Bound    dsail-pv   10Gi       RWX            manual         8s

C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl describe pvc dsail-pvc
Name:          dsail-pvc
Namespace:     default
StorageClass:  manual
Status:        Bound
Volume:        dsail-pv
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      10Gi
Access Modes:  RWX
VolumeMode:    Filesystem
Used By:       <none>
Events:        <none>
```

kubectl apply -f xle-dsail-server-pod.yaml
kubectl get pod

```
C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl apply -f xle-dsail-server-pod.yaml
pod/dsail-server created

C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl get pod
NAME          READY   STATUS    RESTARTS   AGE
dsail-server  1/1     Running   0          18s

C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl logs dsail-server
INFO flower 2021-11-25 15:21:52,889 | app.py:73 | Flower server running (insecure, 3 rounds)
INFO flower 2021-11-25 15:21:52,889 | server.py:118 | Getting initial parameters
```

kubectl logs dsail-server

kubectl apply -f xle-dsail-client-1-pod.yaml
kubectl get pod

```
C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl apply -f xle-dsail-client-1-pod.yaml
pod/dsail-client-1 created

C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl get pod
NAME            READY   STATUS    RESTARTS   AGE
dsail-client-1  1/1     Running   0          5s
dsail-server    1/1     Running   0          2m47s
```

kubectl logs dsail-client-1

```
C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl logs dsail-client-1
Downloading: "https://download.pytorch.org/models/resnet18-f37072fd.pth" to /root/.cache/torch/hub/checkpoints/resnet18-f37072fd.pth

DSAIL | Client Starting
Dataset:  /dataset/H1

Setting-up type transforms pipelines
Collecting items from /dataset/H1
Found 1583 items
2 datasets of sizes 664,449
Setting up Pipeline: PILBase.create
Setting up Pipeline: parent_label -> Categorize -- {'vocab': None, 'sort': True, 'add_na': False}

Building one sample
  Pipeline: PILBase.create
    starting from
      /dataset/H1/train/COVID/COVID-41.png
    applying PILBase.create gives
      PILImage mode=RGB size=299x299
  Pipeline: parent_label -> Categorize -- {'vocab': None, 'sort': True, 'add_na': False}
    starting from
      /dataset/H1/train/COVID/COVID-41.png
    applying parent_label gives
      COVID
    applying Categorize -- {'vocab': None, 'sort': True, 'add_na': False} gives
      TensorCategory(0)

Final sample: (PILImage mode=RGB size=299x299, TensorCategory(0))


Collecting items from /dataset/H1
Found 1583 items
2 datasets of sizes 664,449
Setting up Pipeline: PILBase.create
Setting up Pipeline: parent_label -> Categorize -- {'vocab': None, 'sort': True, 'add_na': False}
Setting up after_item: Pipeline: ToTensor
Setting up before_batch: Pipeline:
Setting up after_batch: Pipeline: IntToFloatTensor -- {'div': 255.0, 'div_mask': 1}
```

```
Building one batch
Applying item_tfms to the first sample:
  Pipeline: ToTensor
    starting from
      (PILImage mode=RGB size=299x299, TensorCategory(0))
    applying ToTensor gives
      (TensorImage of size 3x299x299, TensorCategory(0))

Adding the next 3 samples

No before_batch transform to apply

Collating items in a batch

Applying batch_tfms to the batch built
  Pipeline: IntToFloatTensor -- {'div': 255.0, 'div_mask': 1}
    starting from
      (TensorImage of size 4x3x299x299, TensorCategory([0, 0, 0, 0]))
    applying IntToFloatTensor -- {'div': 255.0, 'div_mask': 1} gives
      (TensorImage of size 4x3x299x299, TensorCategory([0, 0, 0, 0]))
100%|          | 44.7M/44.7M [00:31<00:00, 1.50MB/s][W NNPACK.cpp:79] Could not initialize NNPACK! Reason: Unsupported hardware.

DEBUG flower 2021-11-25 15:25:11,020 | connection.py:36 | ChannelConnectivity.IDLE
DEBUG flower 2021-11-25 15:25:11,020 | connection.py:36 | ChannelConnectivity.READY
INFO flower 2021-11-25 15:25:11,020 | app.py:61 | Opened (insecure) gRPC connection
```

kubectl apply -f xle-dsail-client-2-pod.yaml
kubectl get pod

```
C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl apply -f xle-dsail-client-2-pod.yaml
pod/dsail-client-2 created

C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl get pod
NAME             READY   STATUS    RESTARTS   AGE
dsail-client-1   1/1     Running   0          8m7s
dsail-client-2   1/1     Running   0          14s
dsail-server     1/1     Running   0          10m

C:\CETIC\ARIAC\FL-DSAIL\k8s>
```

kubectl logs dsail-client-2

```
C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl logs dsail-client-2
Downloading: "https://download.pytorch.org/models/resnet18-f37072fd.pth" to /root/.cache/torch/hub/checkpoints/resnet18-f37072fd.pth

DSAIL | Client Starting
Dataset:  /dataset/H2

Setting-up type transforms pipelines
Collecting items from /dataset/H2
Found 1652 items
2 datasets of sizes 733,449
Setting up Pipeline: PILBase.create
Setting up Pipeline: parent_label -> Categorize -- {'vocab': None, 'sort': True, 'add_na': False}

Building one sample
  Pipeline: PILBase.create
    starting from
      /dataset/H2/train/COVID/COVID-384.png
    applying PILBase.create gives
      PILImage mode=RGB size=299x299
  Pipeline: parent_label -> Categorize -- {'vocab': None, 'sort': True, 'add_na': False}
    starting from
      /dataset/H2/train/COVID/COVID-384.png
    applying parent_label gives
      COVID
    applying Categorize -- {'vocab': None, 'sort': True, 'add_na': False} gives
      TensorCategory(0)

Final sample: (PILImage mode=RGB size=299x299, TensorCategory(0))


Collecting items from /dataset/H2
Found 1652 items
2 datasets of sizes 733,449
Setting up Pipeline: PILBase.create
Setting up Pipeline: parent_label -> Categorize -- {'vocab': None, 'sort': True, 'add_na': False}
Setting up after_item: Pipeline: ToTensor
Setting up before_batch: Pipeline:
Setting up after_batch: Pipeline: IntToFloatTensor -- {'div': 255.0, 'div_mask': 1}

Building one batch
Applying item_tfms to the first sample:
  Pipeline: ToTensor
    starting from
      (PILImage mode=RGB size=299x299, TensorCategory(0))
    applying ToTensor gives
      (TensorImage of size 3x299x299, TensorCategory(0))
```

```
Adding the next 3 samples

No before_batch transform to apply

Collating items in a batch

Applying batch_tfms to the batch built
  Pipeline: IntToFloatTensor -- {'div': 255.0, 'div_mask': 1}
    starting from
      (TensorImage of size 4x3x299x299, TensorCategory([0, 0, 0, 0]))
    applying IntToFloatTensor -- {'div': 255.0, 'div_mask': 1} gives
      (TensorImage of size 4x3x299x299, TensorCategory([0, 0, 0, 0]))
100%|██████████| 44.7M/44.7M [00:18<00:00, 2.59MB/s][W NNPACK.cpp:79] Could not initialize NNPACK! Reason: Unsupported hardware.

DEBUG flower 2021-11-25 15:32:49,794 | connection.py:36 | ChannelConnectivity.IDLE
DEBUG flower 2021-11-25 15:32:49,794 | connection.py:36 | ChannelConnectivity.CONNECTING
INFO flower 2021-11-25 15:32:49,795 | app.py:61 | Opened (insecure) gRPC connection
DEBUG flower 2021-11-25 15:32:49,795 | connection.py:36 | ChannelConnectivity.READY
```

kubectl apply -f xle-dsail-client-3-pod.yaml
kubectl get pod

```
C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl apply -f xle-dsail-client-3-pod.yaml
pod/dsail-client-3 created

C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl get pod
NAME             READY   STATUS    RESTARTS   AGE
dsail-client-1   1/1     Running   0          55m
dsail-client-2   1/1     Running   0          47m
dsail-client-3   1/1     Running   0          8s
dsail-server     1/1     Running   0          57m
```

kubectl logs dsail-client-3

```
C:\CETIC\ARIAC\FL-DSAIL\k8s>kubectl logs dsail-client-3
Downloading: "https://download.pytorch.org/models/resnet18-f37072fd.pth" to /root/.cache/torch/hub/checkpoints/resnet18-f37072fd.pth

DSAIL | Client Starting
Dataset:  /dataset/H3

Setting-up type transforms pipelines
Collecting items from /dataset/H3
Found 1677 items
2 datasets of sizes 758,449
Setting up Pipeline: PILBase.create
Setting up Pipeline: parent_label -> Categorize -- {'vocab': None, 'sort': True, 'add_na': False}

Building one sample
  Pipeline: PILBase.create
    starting from
      /dataset/H3/train/COVID/COVID-382.png
    applying PILBase.create gives
      PILImage mode=RGB size=299x299
  Pipeline: parent_label -> Categorize -- {'vocab': None, 'sort': True, 'add_na': False}
    starting from
      /dataset/H3/train/COVID/COVID-382.png
    applying parent_label gives
      COVID
    applying Categorize -- {'vocab': None, 'sort': True, 'add_na': False} gives
      TensorCategory(0)

Final sample: (PILImage mode=RGB size=299x299, TensorCategory(0))
```

```
Collecting items from /dataset/H3
Found 1677 items
2 datasets of sizes 758,449
Setting up Pipeline: PILBase.create
Setting up Pipeline: parent_label -> Categorize -- {'vocab': None, 'sort': True, 'add_na': False}
Setting up after_item: Pipeline: ToTensor
Setting up before_batch: Pipeline:
Setting up after_batch: Pipeline: IntToFloatTensor -- {'div': 255.0, 'div_mask': 1}

Building one batch
Applying item_tfms to the first sample:
  Pipeline: ToTensor
    starting from
      (PILImage mode=RGB size=299x299, TensorCategory(0))
    applying ToTensor gives
      (TensorImage of size 3x299x299, TensorCategory(0))

Adding the next 3 samples

No before_batch transform to apply

Collating items in a batch

Applying batch_tfms to the batch built
  Pipeline: IntToFloatTensor -- {'div': 255.0, 'div_mask': 1}
    starting from
      (TensorImage of size 4x3x299x299, TensorCategory([0, 0, 0, 0]))
    applying IntToFloatTensor -- {'div': 255.0, 'div_mask': 1} gives
      (TensorImage of size 4x3x299x299, TensorCategory([0, 0, 0, 0]))
100%|          | 44.7M/44.7M [00:19<00:00, 2.41MB/s][W NNPACK.cpp:79] Could not initialize NNPACK! Reason: Unsupported hardware.

DEBUG flower 2021-11-25 16:19:58,601 | connection.py:36 | ChannelConnectivity.IDLE
INFO flower 2021-11-25 16:19:58,601 | app.py:61 | Opened (insecure) gRPC connection
DEBUG flower 2021-11-25 16:19:58,602 | connection.py:36 | ChannelConnectivity.READY
epoch     train_loss  valid_loss  accuracy  roc_auc_score  time
```

In this last step, as with the Docker installation, the training run on each client.