Statistical Machine Learning

Phase2 :

Privacy and Interpretability

Team member:

Zahra Dehghanian

Armin Behnamnia

Mahdi Dehshiri

Summer 2023

In this phase first we add differential privacy to each part of our proposed structure and report the results of it. After that we try to interpret each part and show the output.

First we implement the epsilon-differential privacy for first block. We do this job by using opacus library. To do it, we first use module validator to fix the layer that are not compatible with privacy. After that we initialize a privacy engine and pass data_loader, optimizer and model into it. We set the epsilon parameter in desired range and the accuracy of first part of model is as follow:

```
Epsilon = 0.05
------------------------→
TRAIN done: loss = 1.6188, accuracy = 43.23
VAL done: loss = 1.4619, accuracy = 45.45
----------------
Openset recognition accuracy =  0.7228260869565217
Best Separator Accuracy =  78.26086956521739


Epsilon = 0.5
------------------------→
TRAIN done: loss = 1.5645, accuracy = 51.85
VAL done: loss = 1.6472, accuracy = 47.27
-----------------
Openset recognition accuracy =  0.7445652173913043
Best Separator Accuracy =  78.26086956521739


Epsilon = 1
------------------------→
TRAIN done: loss = 1.4842, accuracy = 54.67
VAL done: loss = 1.4859, accuracy = 48.74
----------------
Openset recognition accuracy =  0.7936956521739131
Best Separator Accuracy =  0.7936956521739131
```

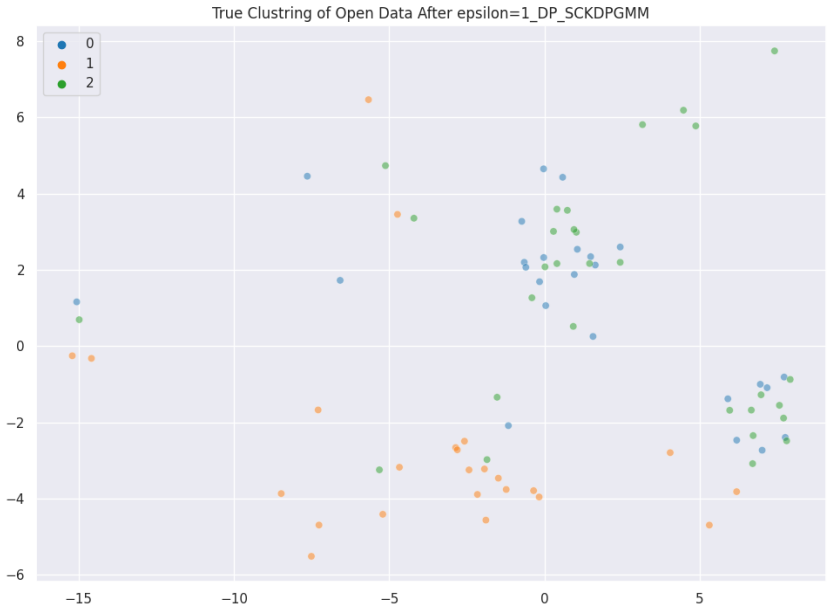
The result is exactly as we expected, how much epsilon get smaller, privacy gets better but the accuracy of model scales down.

To make the second part differential private we add laplace noise with different powers (0.05,0.5,1) to the gradients of the parameters in fully connected layers in CNN Model during training. It should be noted that due to the sampling a random Laplacian noise in each epoch, there is no exact convergence as the one we saw in phase1, but it can be seen that the convergence will occur approximately after some epochs while if we do not stop training in that state, because the outputs of supervised contrastive model is used as input in Dirichlet Process model, eventually Dirichlet model would be collapsed and puts

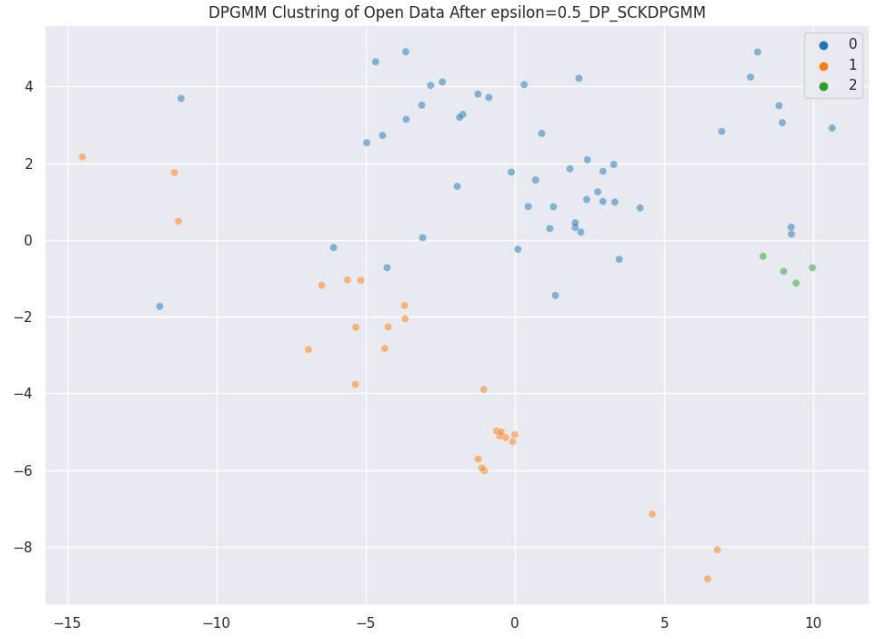
all images in one single cluster, due to that we need to stop training after observing some kind of stability in clustering.

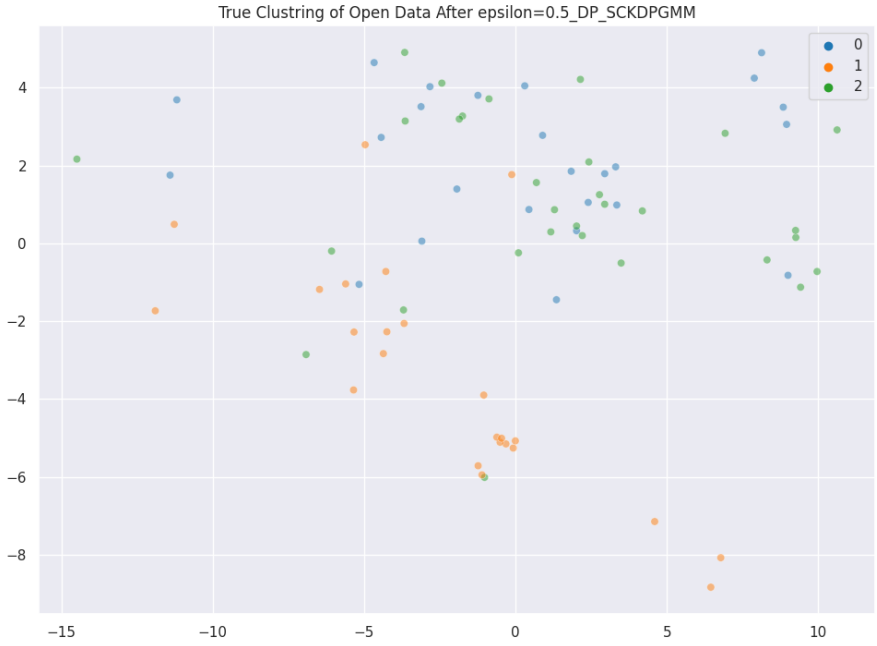
To see this results we have :

For epsilon = 1 , accuracy would be 0.58 in clustering :

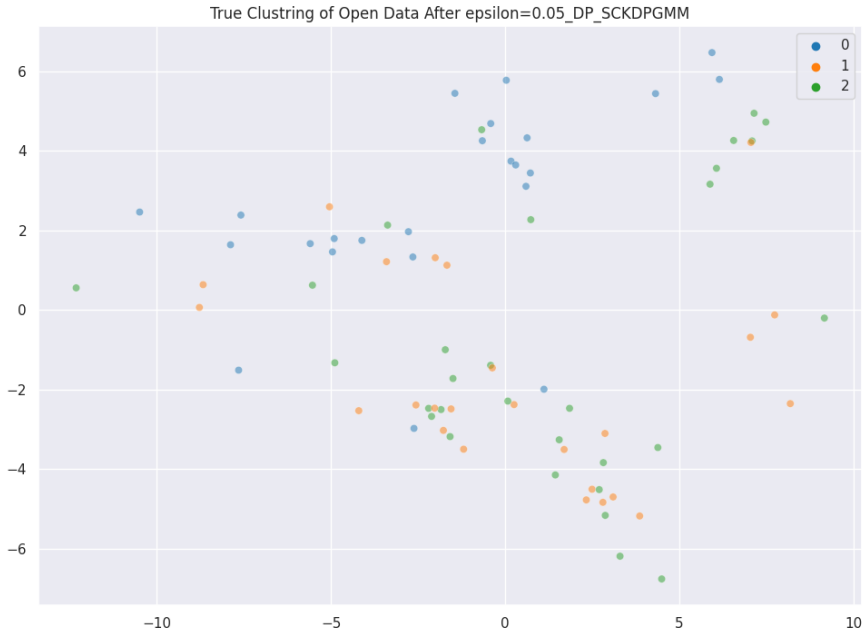


for epsilon=0.5 , accuracy would be 0.56 as :





And for epsilon=0.05 , accuracy would be 0.48 as :

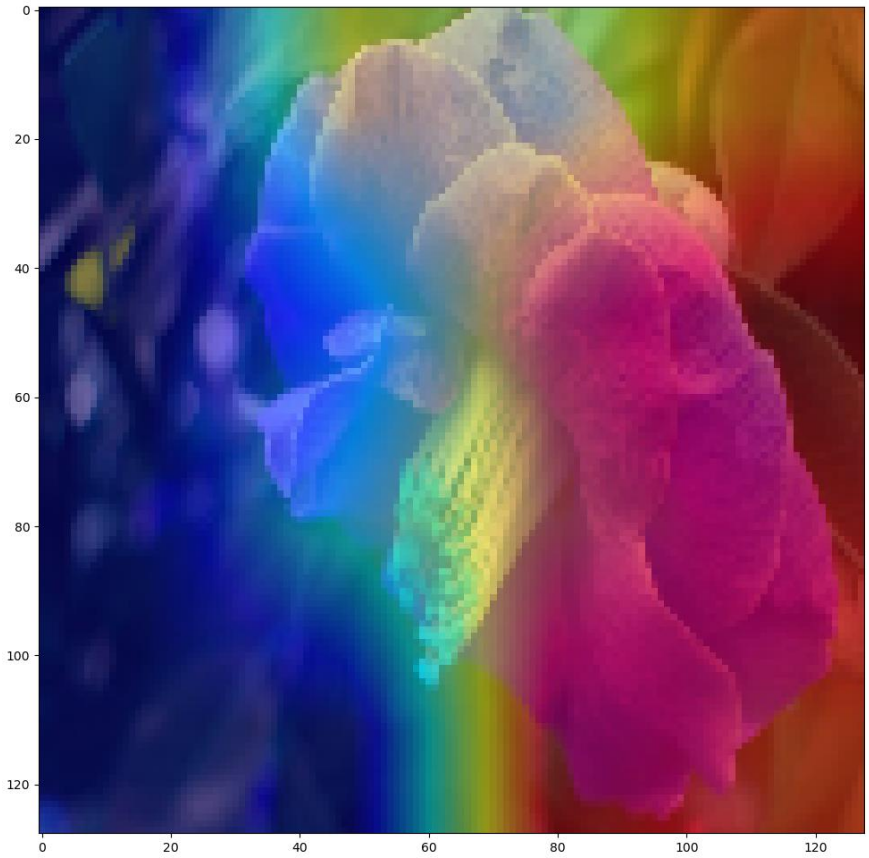


As you see increasing the power of noise would results in decreasing the accuracy and obtaining stronger privacy.

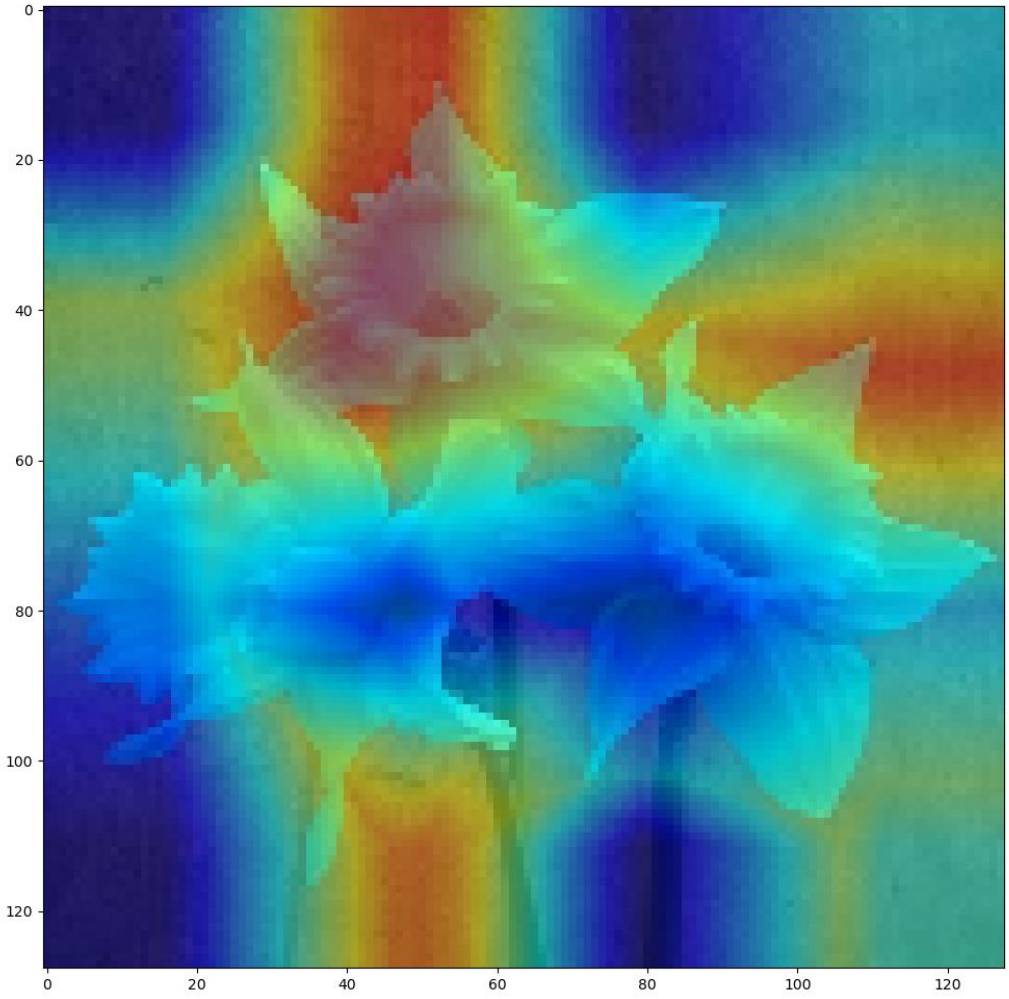
For the interpretability, we create interpretations of the open set recognizer, that is the part that detects closed set classes and whether the data is in the opens et or not. The clustering part is intrinsically interpretable because it works directly based on the extracted features of the deep model. So, we apply post-hoc interpretation on the deep model to visualize important parts of the image that makes the model decide.

We use CAM-based interpretation here. Model first detects if the sample is in the open set or not, if it's not, interpretation is applied based on the predicted class. If the sample is in the open set, the area in the image that is not important in the detection of any of the closed set classes is shown. It may contain meaningless areas in the image. But because from model's perspective, open set is ANYTHING outside the closed set domain, meaningless areas are also parts of the image that are seen as open set.

Here is a visualization of model output based on GradCAM that is in the closed set:



Here is a sample from the open set.



The interpreter is able to work with the following CAM-based algorithms:

```
GradCAM
GradCAMPlusPlus
AblationCAM
XGradCAM
EigenCAM
FullGrad
```

The sample runnable script is test_interpretability.py that has also a notebook version alongside.