UNSUPERVISED DOMAIN ADAPTATION BY BACKPROPAGATION

SMAI Team 37
-Krati Agrawal (2022201063)
-Anuja Gulhane (2022201070)
-Hrishikesh Deshpande (2022201065)

CONTENT

PROBLEM STATEMENT

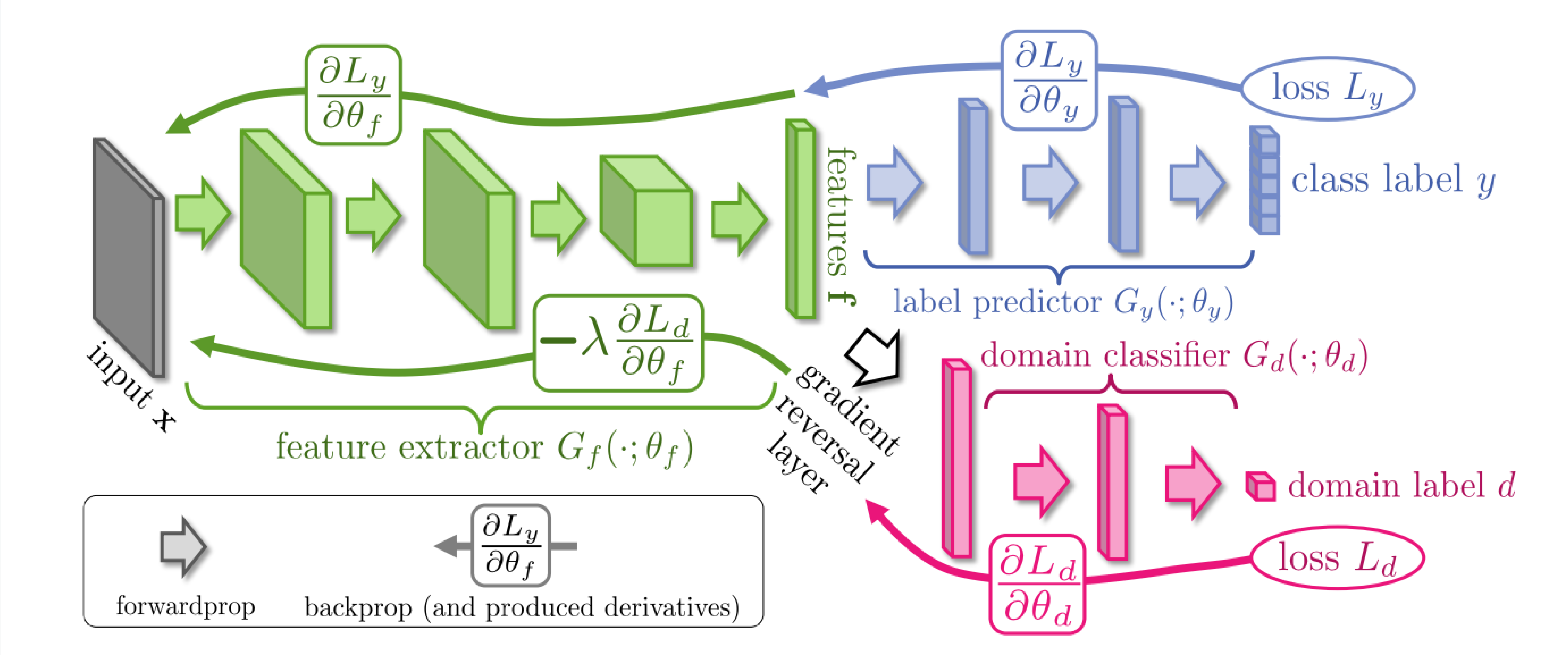
Top-performing deep architectures are trained on massive amounts of labeled data.
In the absence of labeled data for a certain task, domain adaptation often provides an attractive option given that labeled data of similar nature but from a different domain are available.

PROPOSED APPROACH

- Train a deep neural network with a domain classifier and a gradient reversal layer.
- Encourage the network to learn domain-invariant features that can be applied to the target domain.
- The method is unsupervised and does not require labelled data from the target domain.
- Uses only labelled data from the source domain and unlabelled data from the target domain for training

Reference paper: https://arxiv.org/pdf/1409.7495.pdf

MODEL ARCHITECTURE



Experimental Setup

DATA PREPARATION

Experiment is conducted over three different datsets- MNIST, SVHN, MNISTM (MNIST with background patches), ASL and ASL (with patches)

MODEL ARCHITECTURE

Model consists of CNN, label classification and domain classification layers over different source and target domains

TESTING METRICS

Accuracies for baseline and proposed approach considering training on data without adaptation and with adaptation

FEATURE VISUALISATION

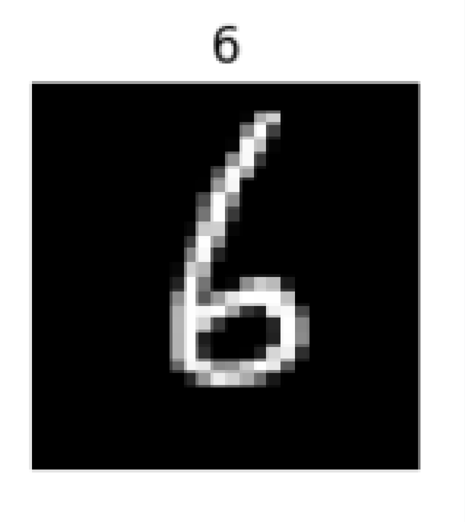
Visualization of features in case of both domain adaptation and without domain adaptation using tsne plot

EXPERIMENT - 1

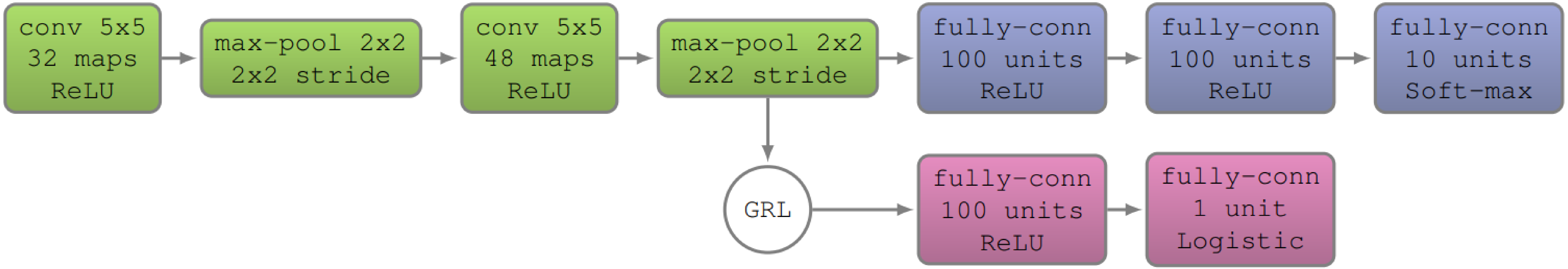
Dataset:

Source Domain: MNIST

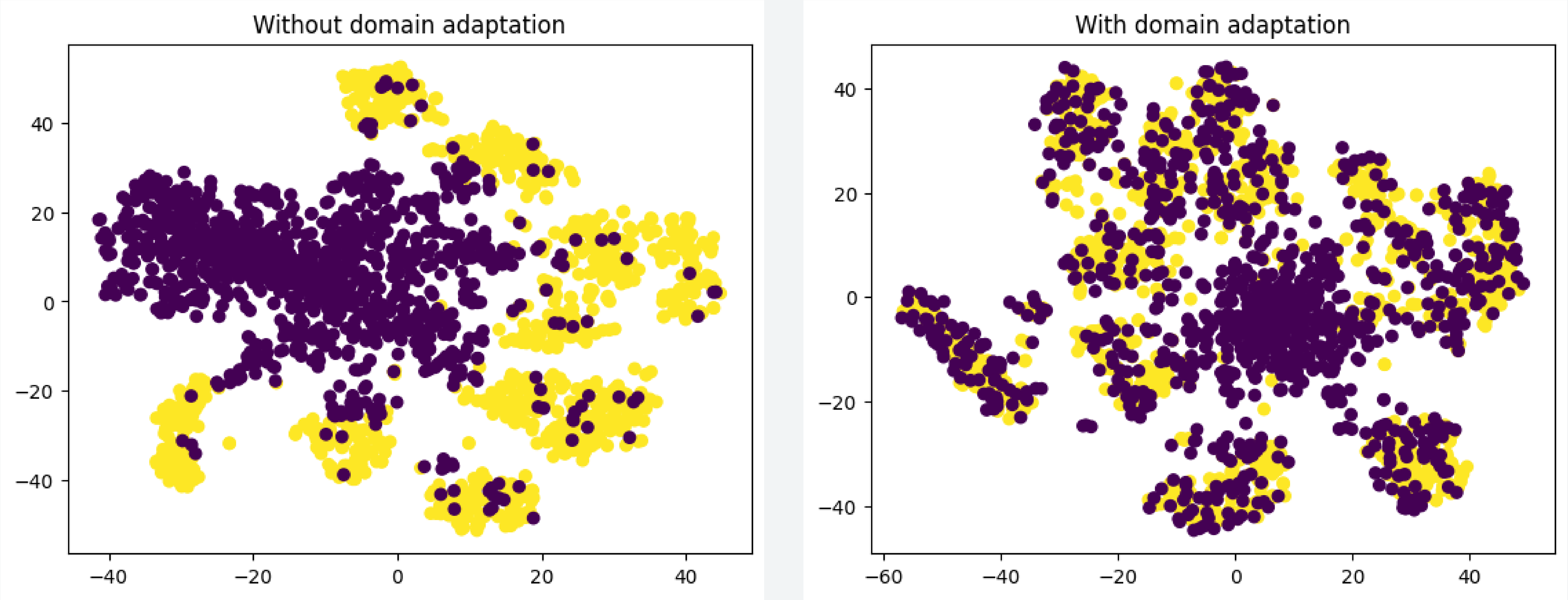
Target Domain: MNISTM



Architecture:



(a) MNIST architecture

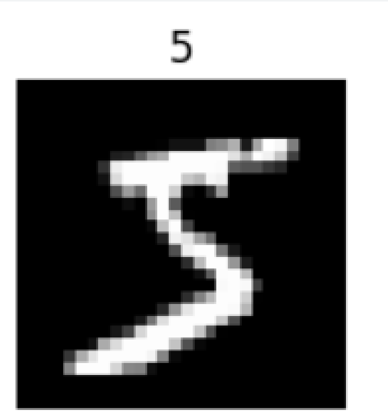

Without domain adaptation
With domain adaptation
40
20
0
−20
−40
−60

EXPERIMENT 2

Dataset:

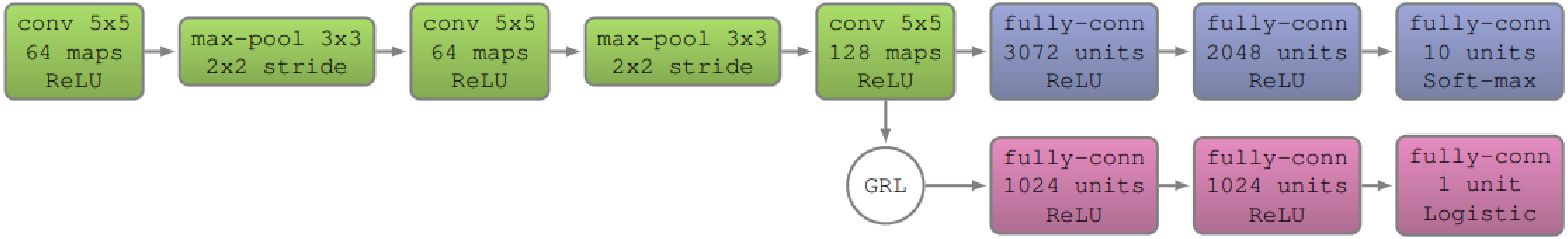
Source Domain: SVHN

Target Domain: MNIST

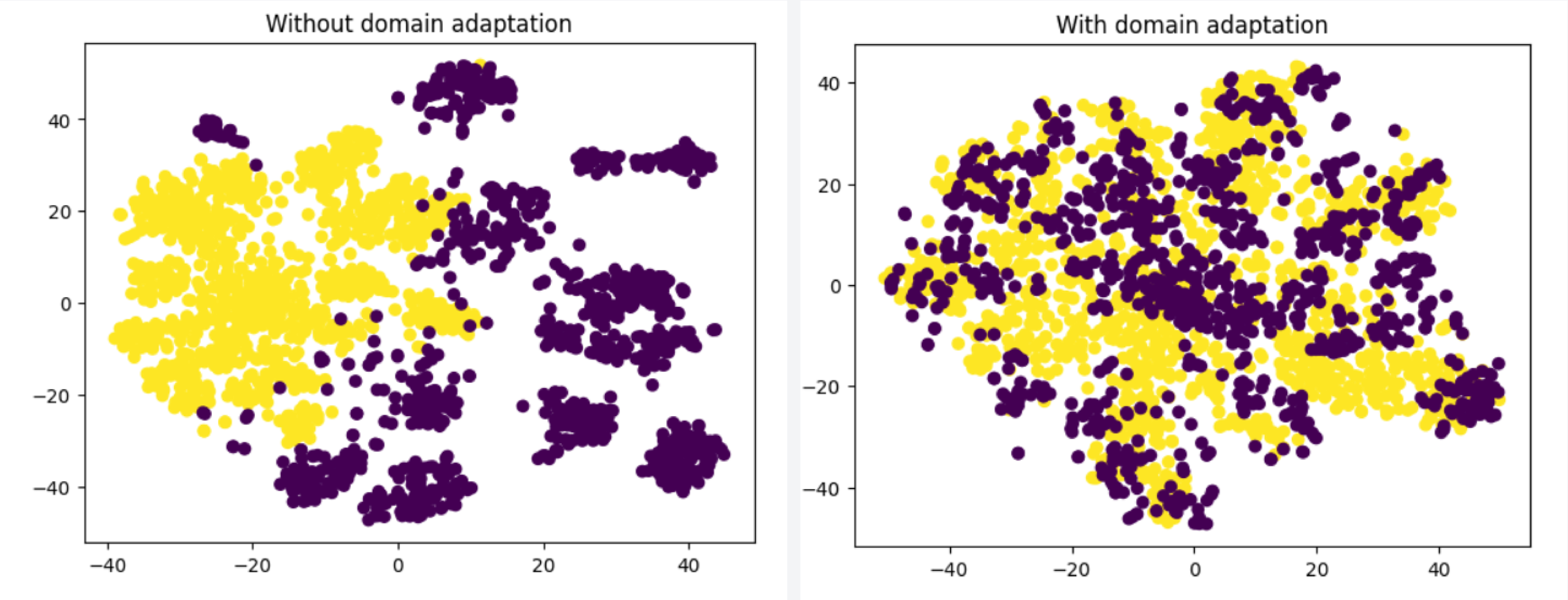


MNIST

SVHN



(b) SVHN architecture

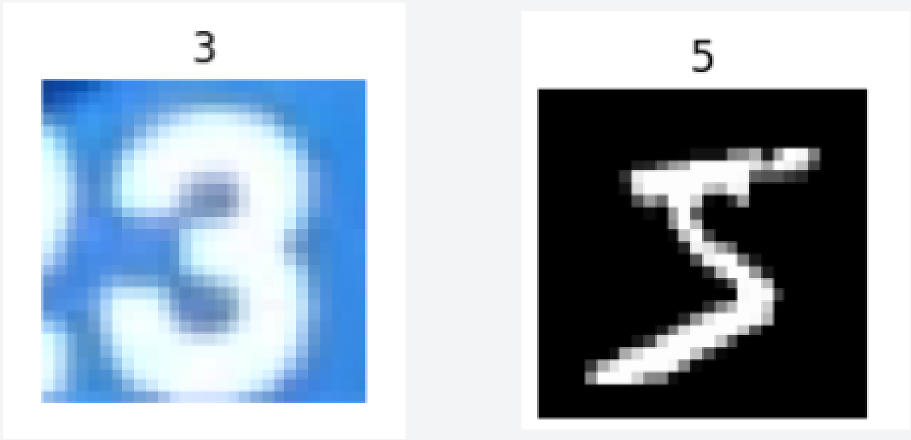

Without domain adaptation
40
20
0
−20
−40
−40
−20
0
20
40
With domain adaptation
40
20
0
−20
−40
−40
−20
0
20
40

EXPERIMENT 3

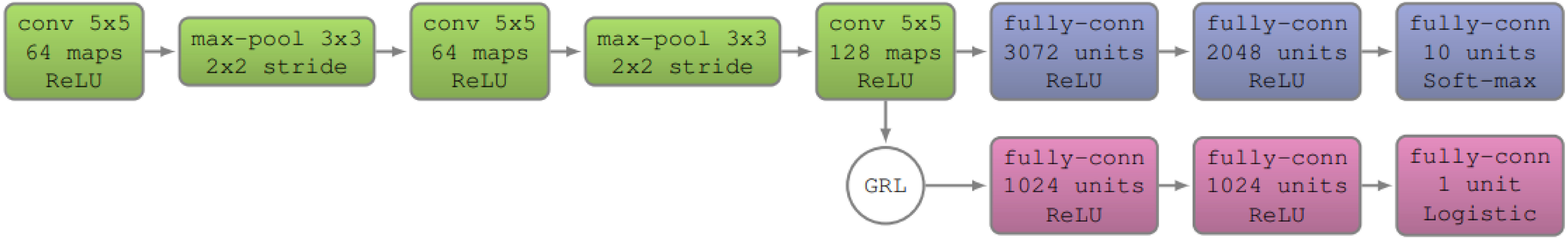
Dataset:

Source Domain: MNIST

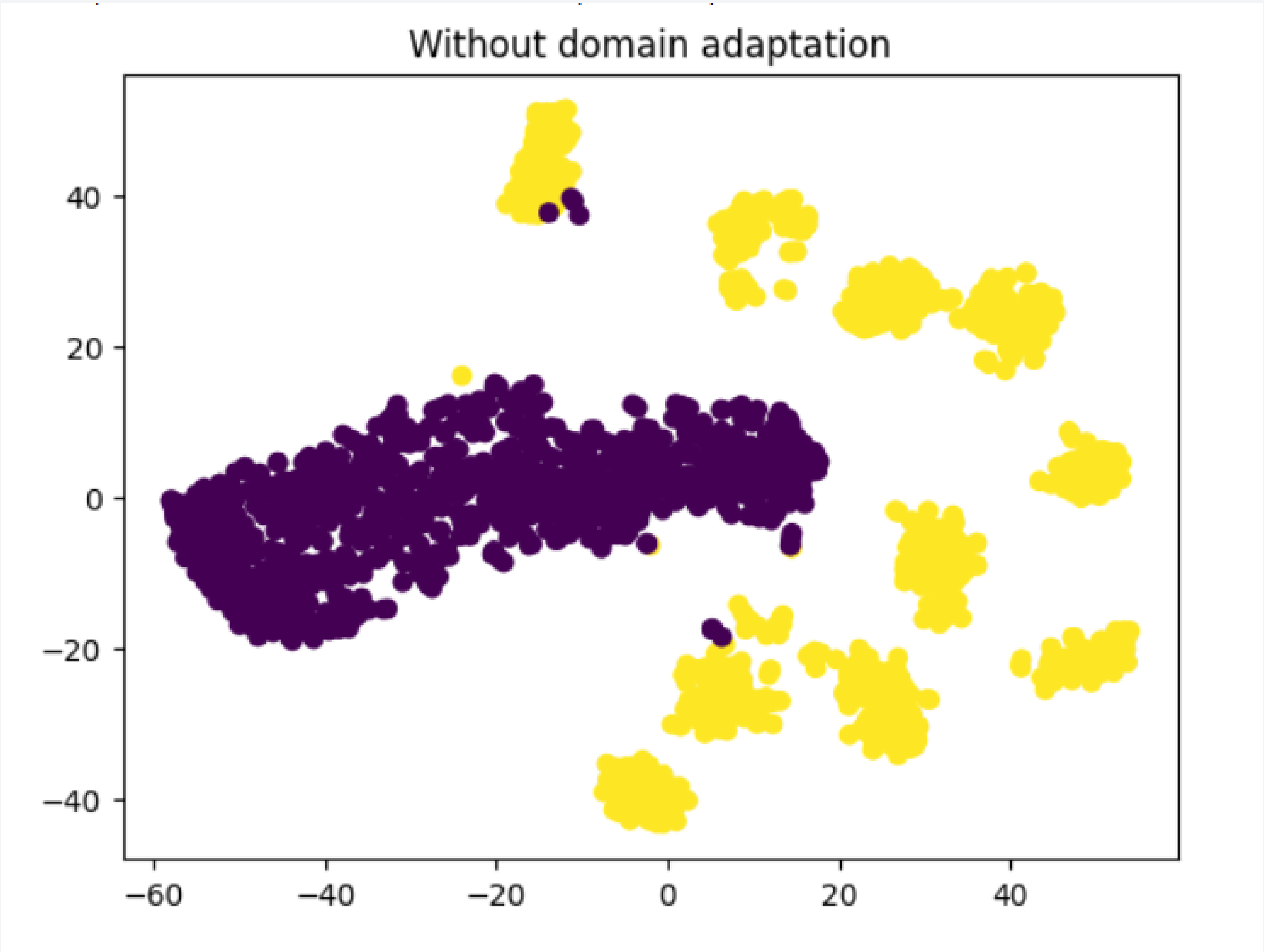
Target Domain: SVHN

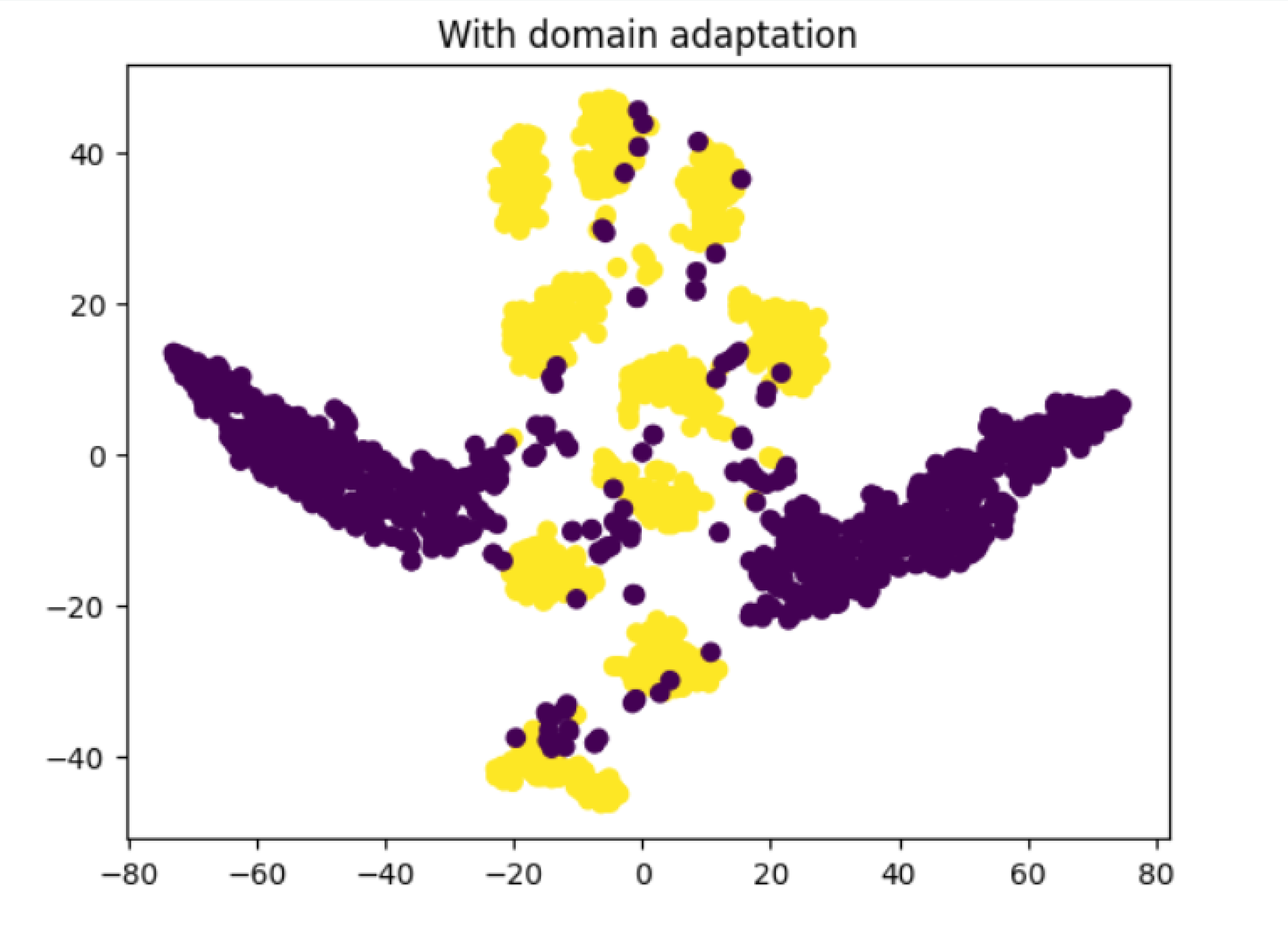


SVHN MNIST



(b) SVHN architecture

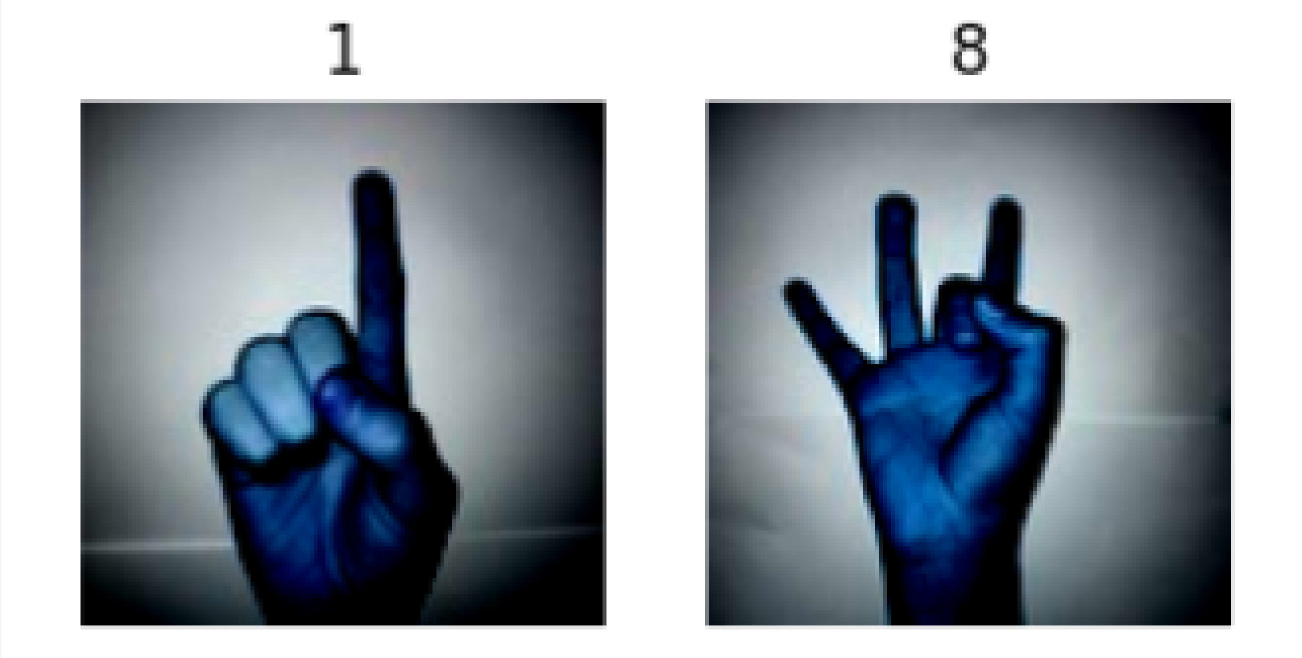

Without domain adaptation
40
20
0
−20
−40
−60
−40
−20
0
20
40


With domain adaptation
40
20
0
−20
−40
−80
−60
−40
−20
0
20
40
60
80

EXPERIMENT 4

Dataset:
Source Domain: ASL (American Sign Language)
Target Domain: ASL-M

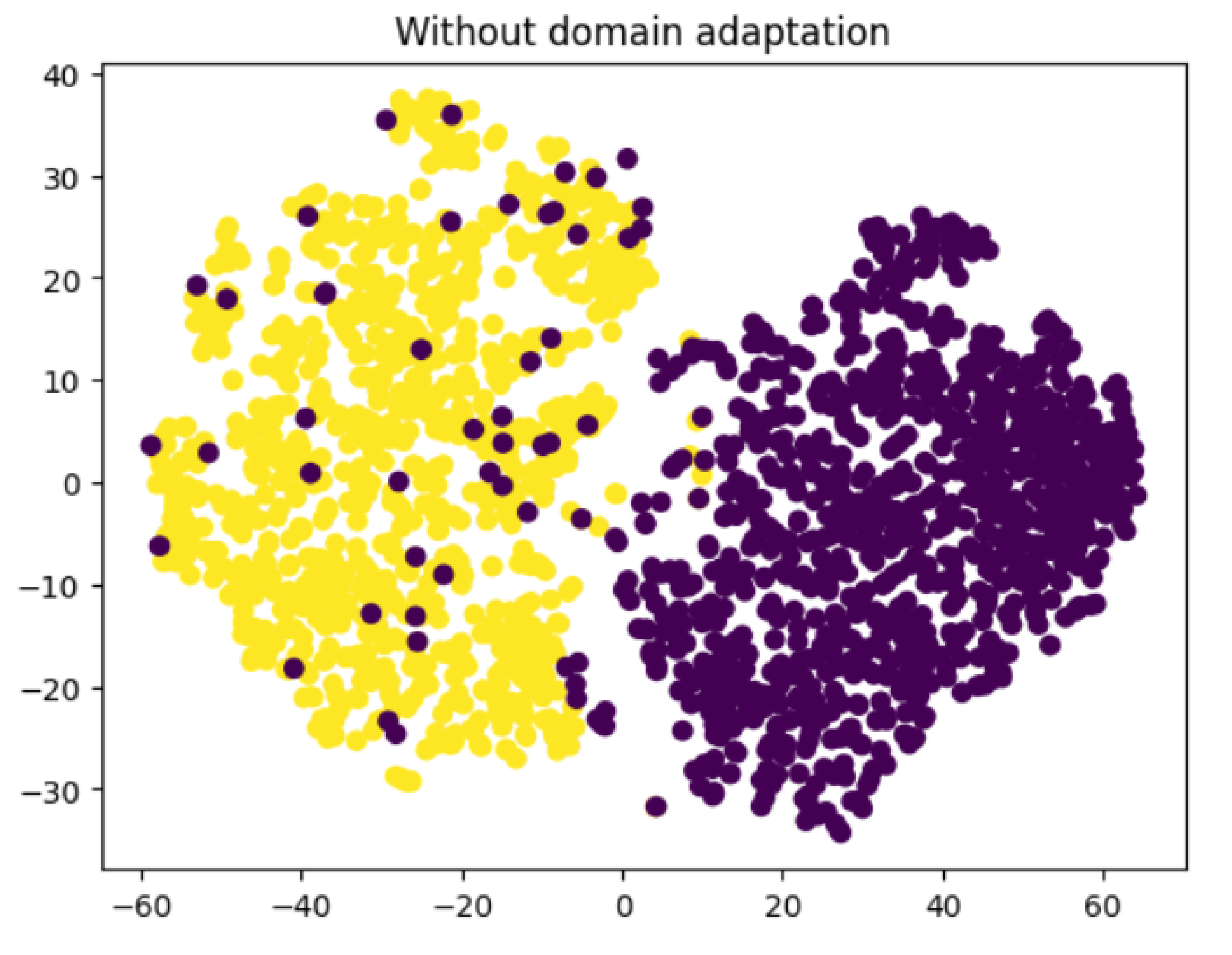
ASL-M



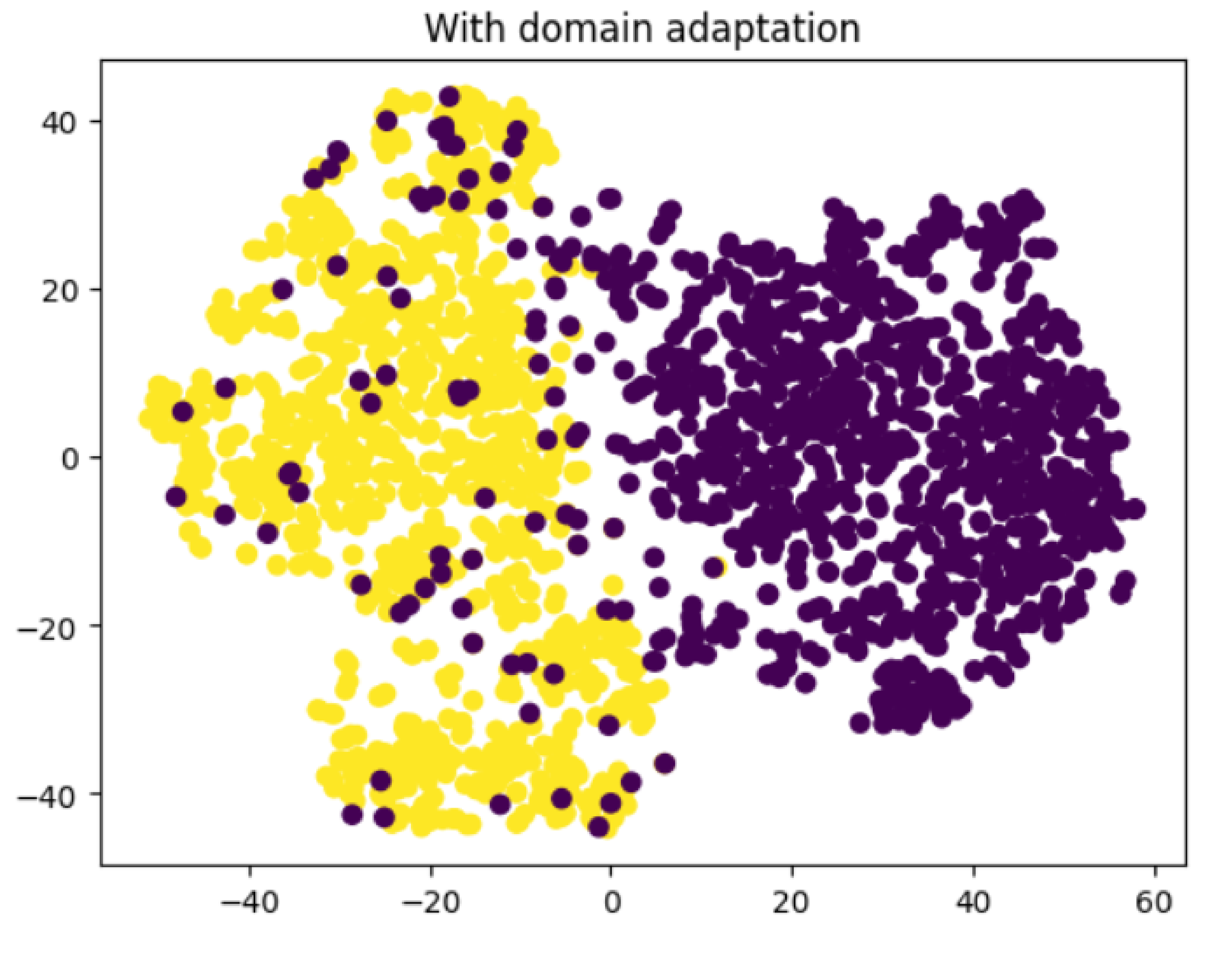
ASL

```python
from torchsummary import summary
model = DANN()
model
```

```
DANN(
  (cnn): Sequential(
    (0): Conv2d(3, 64, kernel_size=(5, 5), stride=(1, 1))
    (1): ReLU(inplace=True)
    (2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=Fal
    (3): Conv2d(64, 64, kernel_size=(5, 5), stride=(1, 1))
    (4): ReLU(inplace=True)
    (5): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=Fal
    (6): Conv2d(64, 128, kernel_size=(5, 5), stride=(1, 1))
  )
  (classif): Sequential(
    (0): Linear(in_features=41472, out_features=100, bias=True)
    (1): ReLU(inplace=True)
    (2): Linear(in_features=100, out_features=100, bias=True)
    (3): ReLU(inplace=True)
    (4): Linear(in_features=100, out_features=10, bias=True)
  )
  (domain): Sequential(
    (0): Linear(in_features=41472, out_features=1024, bias=True)
    (1): ReLU(inplace=True)
    (2): Linear(in_features=1024, out_features=1024, bias=True)
    (3): ReLU(inplace=True)
    (4): Linear(in_features=1024, out_features=1, bias=True)
  )
)
```



Without domain adaptation
40
30
20
10
0
−10
−20
−30
−60
−40
−20
0
20
40
60



With domain adaptation
40
20
0
−20
−40
−40
−20
0
20
40
60

ACCURACY COMPARISION

METHOD	MNIST TO MNIST-M	SVHN TO MNIST	MNIST TO SVHN	ASL TO ASL-M
SOURCE ONLY	58.14%	58.01%	28.78%	58%
PROPOSED METHOD	71.54%	69.22%	16.58%	63.20%
TRAIN ON TARGET	95.99%	99.39%	91.06%	72.60%

Table 1: Classification Accuracies for digit image Classification for different sources and target domains.The last row corresponds to training on the target domain data with known class labels

KEY OBSERVATIONS:

- GRL factor:

GRL=0 : Setting lambda to 0 means the GRL layer has no effect, leading to poor performance on the target domain.

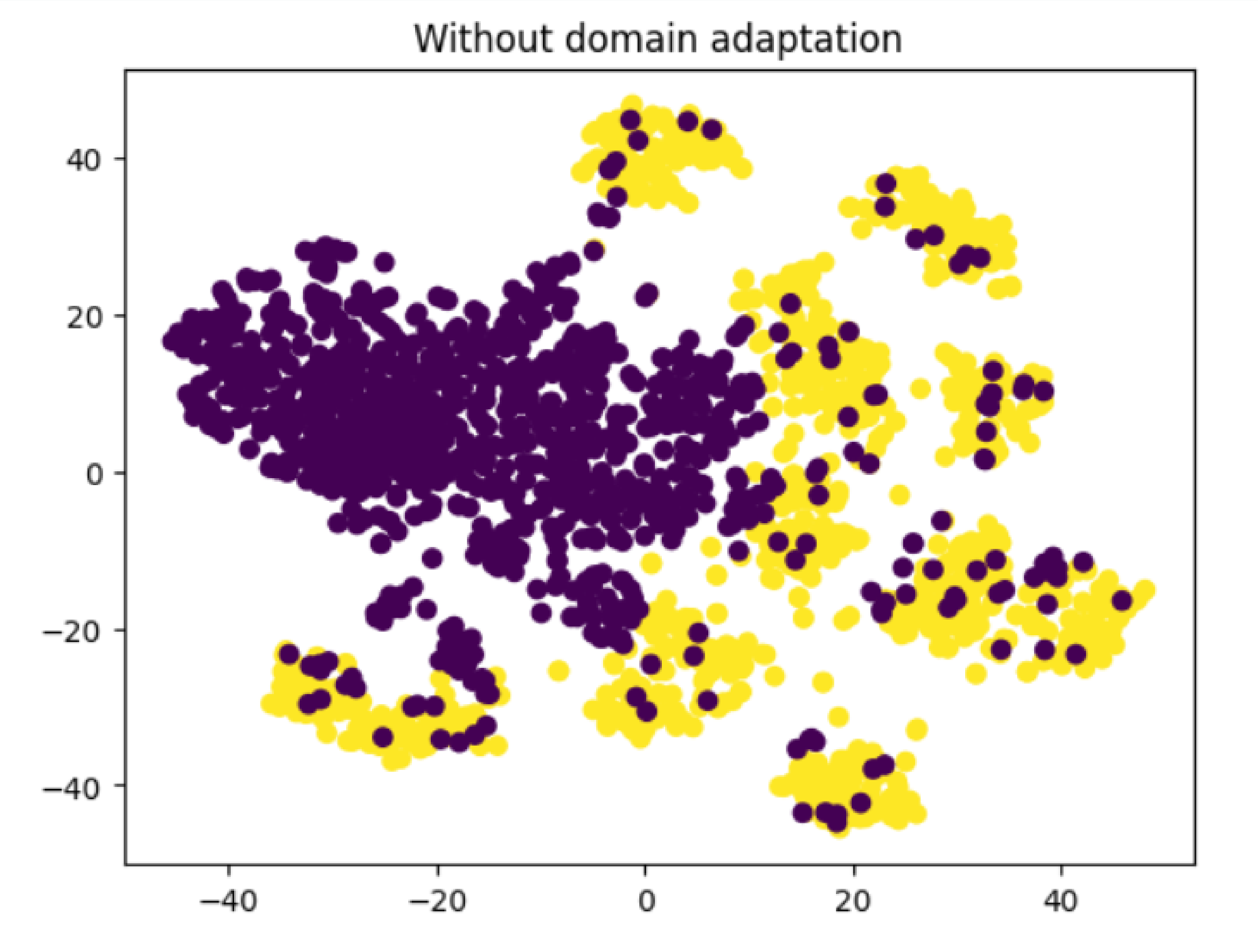
GRL=+1 : Setting lambda to 1 reverses the sign of the gradients, encouraging the network to learn domain-invariant features.

GRL=-1 : Setting lambda to -1 keeps the sign of the gradients unchanged, leading to domain-specific features and poor performance on the target domain.

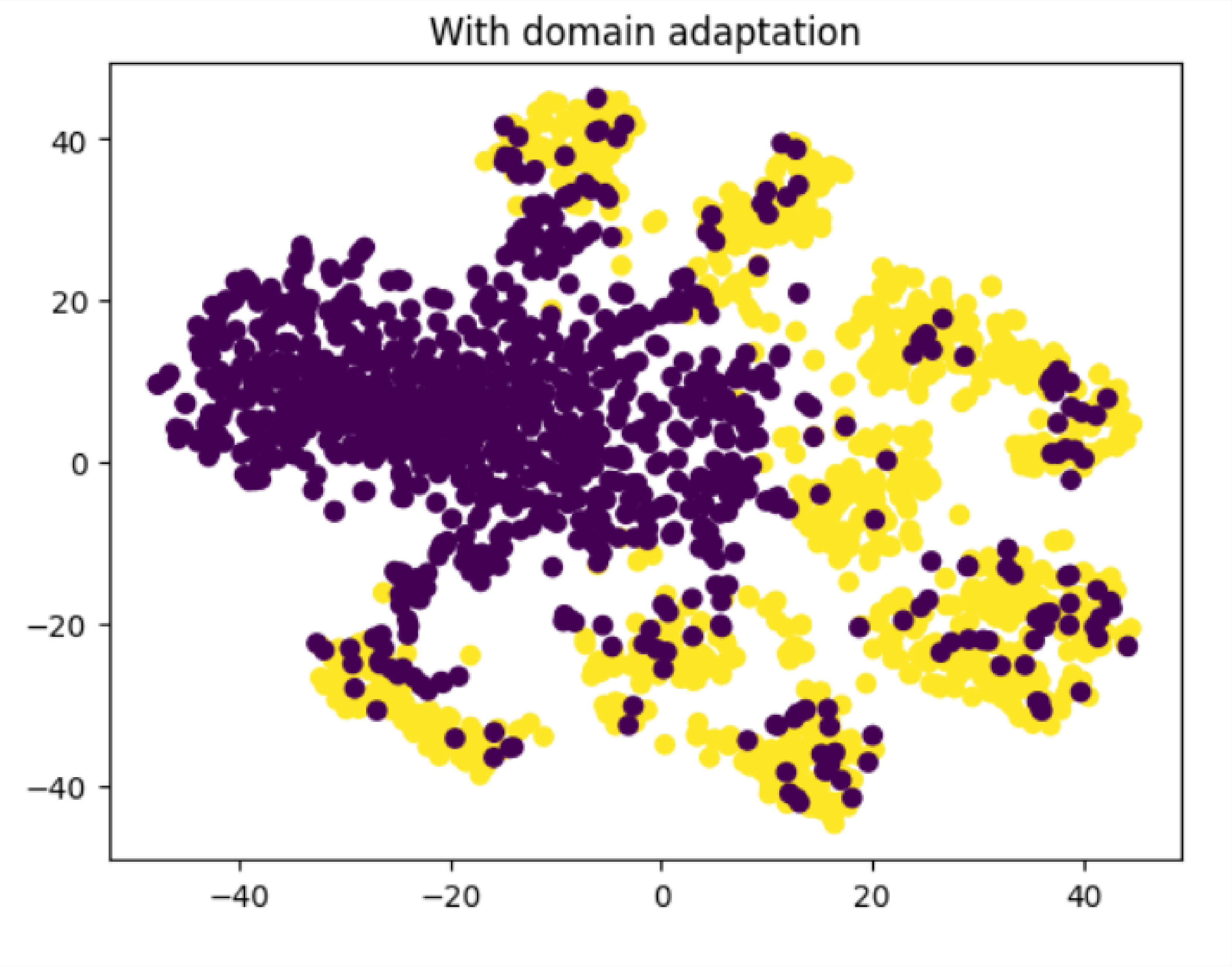
- MNISt <->SVHN

Strong separation between domains which explains why our method failed in improving performance by adaptation in MNIST to SVHN

GRL = 0

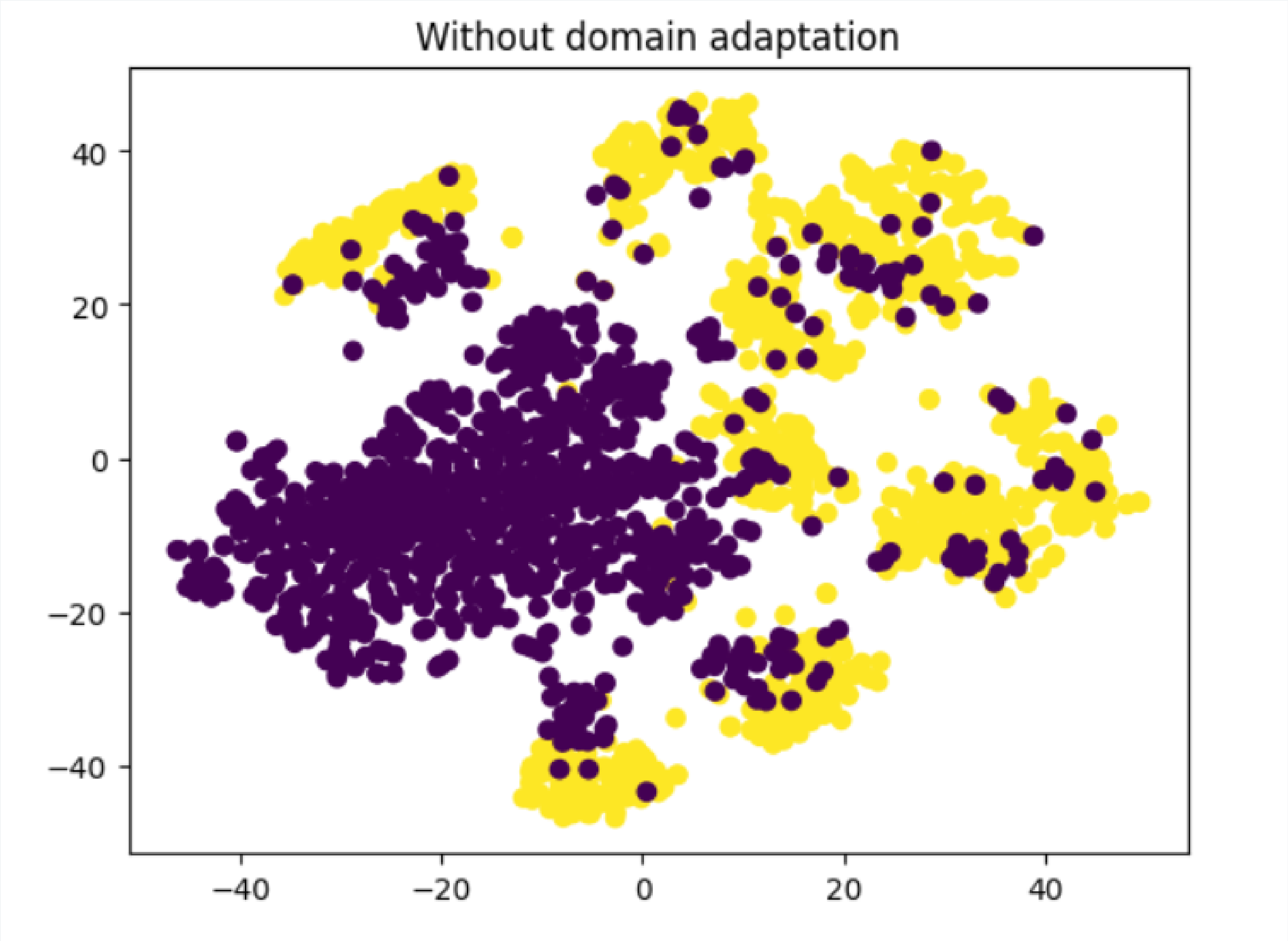


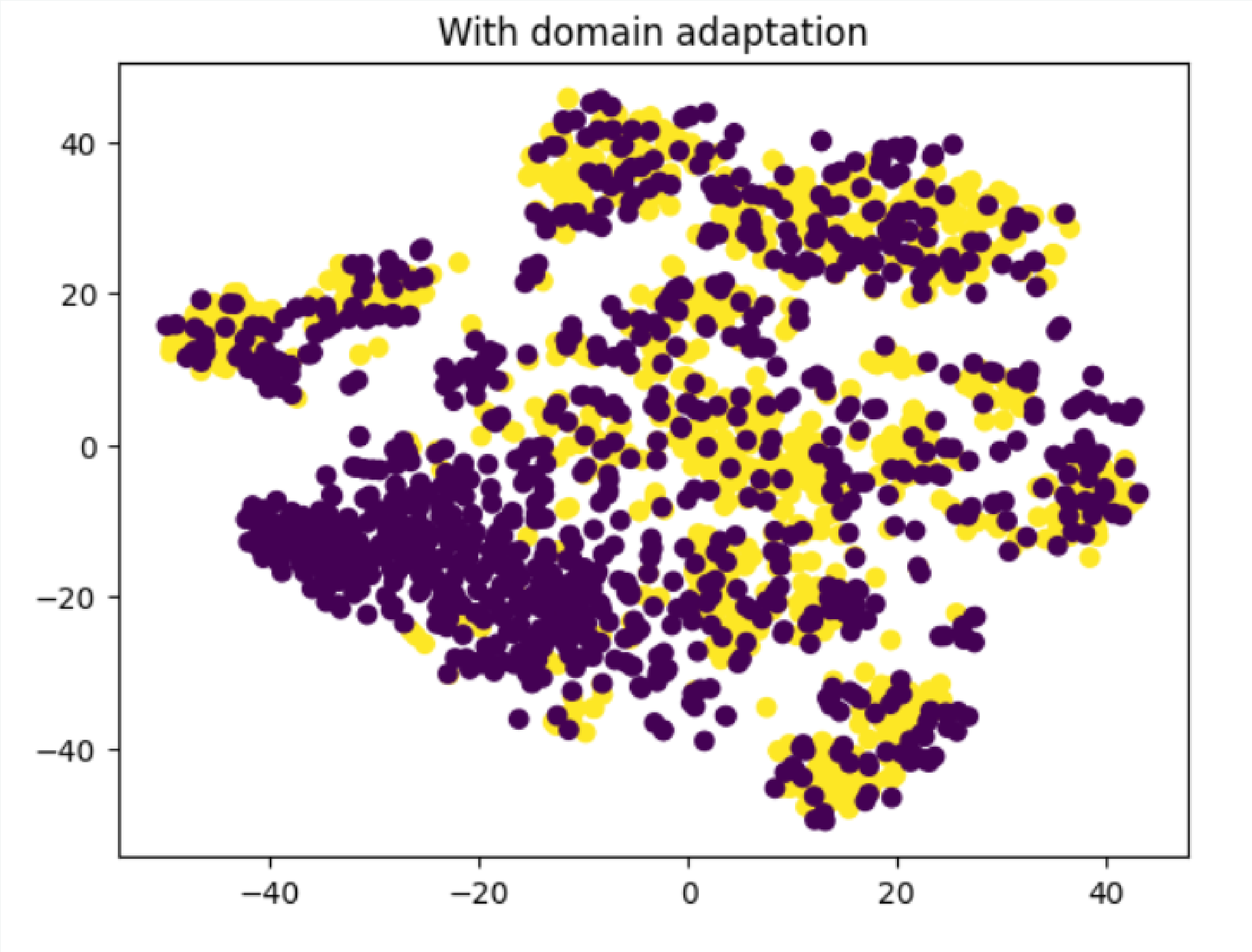
Without domain adaptation
40
20
0
−20
−40
−40
−20
0
20
40



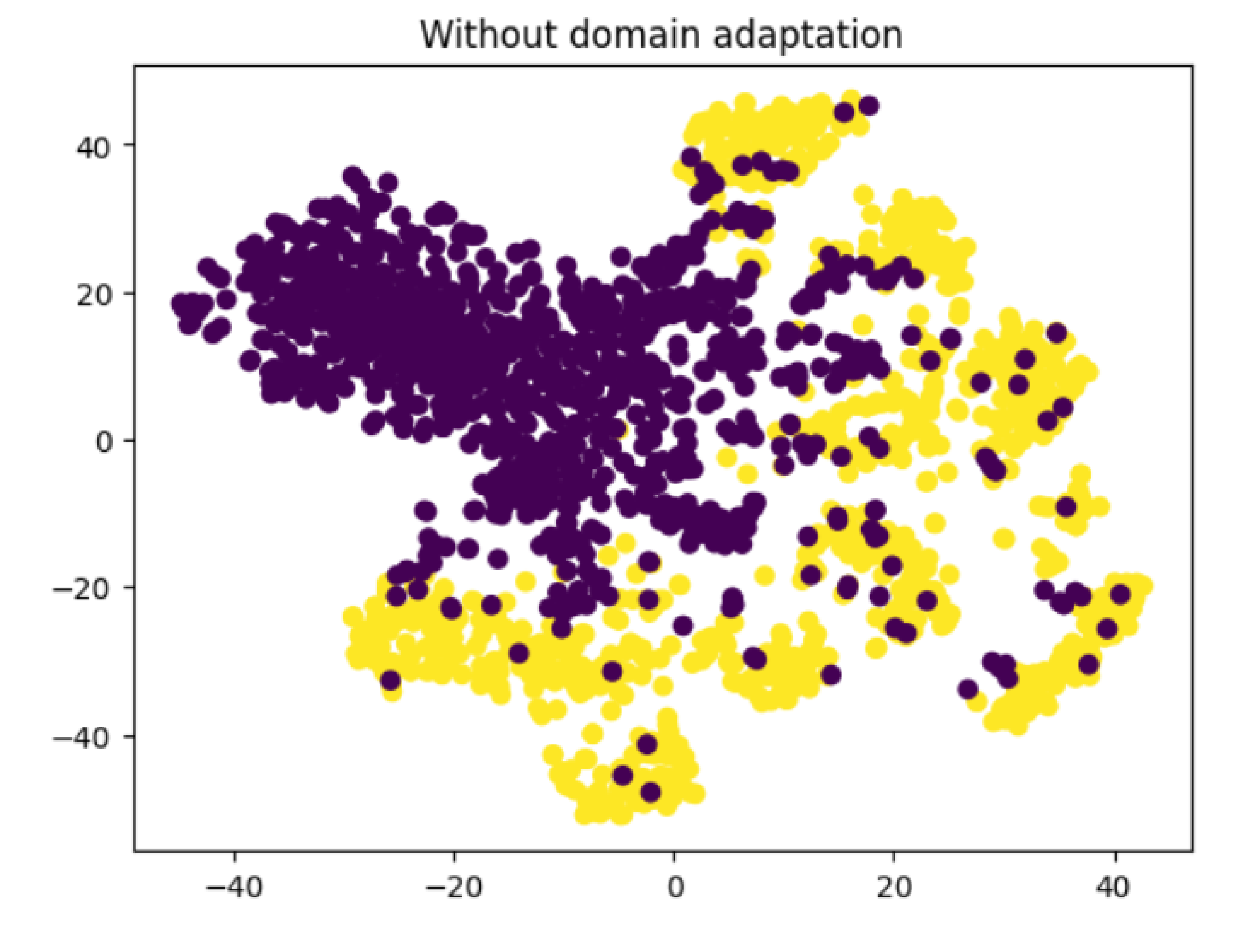
With domain adaptation
40
20
0
−20
−40
−40
−20
0
20
40

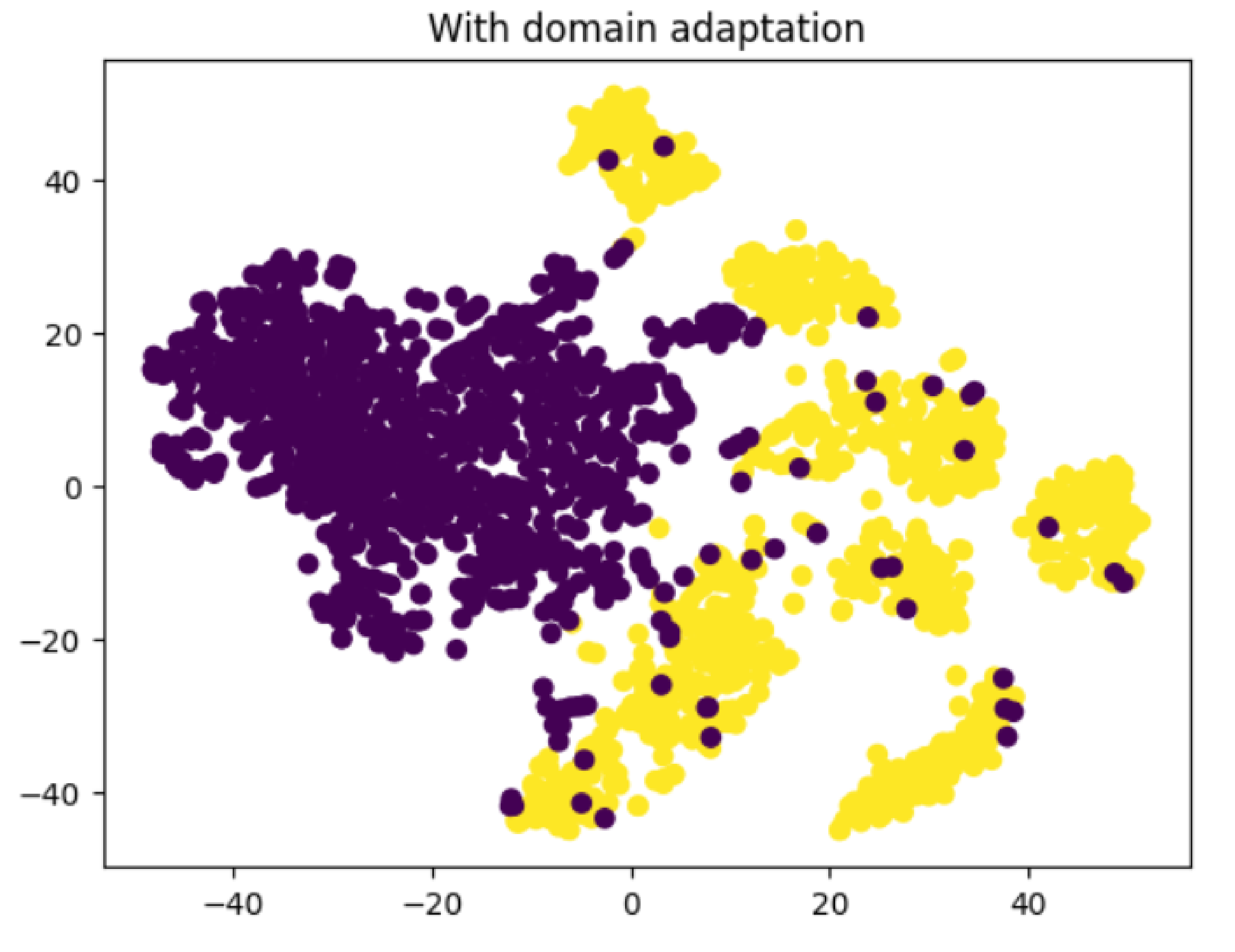
GRL = 1


Without domain adaptation
40
20
0
−20
−40
−40
−20
0
20
40


With domain adaptation
40
20
0
−20
−40
−40
−20
0
20
40

GRL = -1


Without domain adaptation
40
20
0
−20
−40
−40
−20
0
20
40


With domain adaptation
40
20
0
−20
−40
−40
−20
0
20
40

CONTRIBUTION

- **MNIST-MNISTM** : Hrishikesh
- **SVHN-MNIST** : Anuja
- **MNIST-SVHN** : Krati
- **ASL-ASLM** : Krati
- **Dataset Loading/Finding**: Hrishikesh, Anuja
- **Model architecture**: Krati, Anuja
- **Background Patching** : Hrishikesh, Anuja, Krati
- **Feature Extraction**: Hrishikesh, Krati
- **GRL Factor Analysis**: Anuja
- **Visualization**: Anuja, Krati
- **Results**: Anuja, Hrishikesh
- **Analysis**: Hrishikesh, Anuja
- **Presentation**: Krati, Anuja
- **Readme**: Krati
- **Code Structuring**: Hrishikesh, Anuja, Krati

THANKYOU!!!!