Multiclass Novelty Detection with Domain Adaptation

Implementation Track, 4 Team Members

Abstract

The novelty detection models improve the performance of classification by detecting novel classes encountered during testing. However, the distribution of testing datasets can be different from training datasets in real-world cases. It is proved that the distribution shift can be mitigated by combining novelty detection with data adaptation technologies. Thus, the purpose of our project is to re-implement the proposed method for novelty detection presented in "Multiple Class Novelty Detection Under Data Distribution Shift" (Oza et al. [2020]), evaluate the model with different datasets, and compare the results with other off-the-shelf methods. After verifying our implementation by comparing it with the reported results in the paper, we further implement a new real-world dataset that is not used in the paper. The output of our project agrees with the reported results in relative order of performance for different methods on novelty detection. However, the results of our implementation reveal some deviations from the original model. We analyzed and discussed these issues in this paper.

1 Introduction

Advances in the robustness study of Convolutional Neural Networks (CNN) have provided solutions to various practical problems such as adversarial attacks (Liu et al. [2017]), out-of-distribution samples (Hendrycks and Gimpel [2016]), and novelty detection (Sabokrou et al. [2018]). Among them, novelty detection has received increasing attention for its application in inferences of the dataset from critical systems, e.g., tracking suspicious activity patterns for bank fraud detection (Cabanes et al. [2013]). Basically, novelty detection preprocesses the testing dataset from real-world cases and make decisions about whether the given sample belongs to unknown categories of training model. These outliers are filtered out before passing through CNN classification, and the robustness of models is improved.

Conventional novelty detection methods assume that the test data has a similar distribution as training data. However, in real-world cases, the testing datasets are highly likely to have different styles/domains of training datasets. When distribution shifts happen and conventional novelty methods are applied as usual, the risk of false detection as a novel category will increase. One potential solution is exploiting the multi-class structure of the testing dataset to help novelty detection, which is costly and time-consuming (Oza et al. [2020]). Another solution, which has been well studied as unsupervised domain adaption (Sun and Saenko [2016]) but is novel to novelty detection under distribution shift, is applying the gained knowledge from a labeled dataset to a different style/domain to the interested dataset.

The method introduced in Oza et al. [2020], for the first time combines novelty detection with domain adaptation techniques and learns the shared feature space through cross-domain mapping, which further proved that data distribution shift in novelty detection is alleviated and the detection performance is enhanced. Our project aims to reproduce this proposed algorithm, evaluate it according to digit recognition datasets and compare its performance with several baseline methods.

To summarize, this project makes the following contributions: (1) We reproduce the proposed algorithm and several baseline algorithms including Softmax, ALOCC, GRL, and ALOCC+GRL

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

Table 1: Notations used in this report.

Notations	Descriptions
$\boldsymbol{X}_s$	Feature vectors of samples from source domain
$\boldsymbol{y}_s$	Labels of samples from source domain
$\boldsymbol{X}_t$	Feature vectors of samples from target domain
$\boldsymbol{y}_t$	Labels of samples from target domain
$f(\cdot)$	Model function and its output
ℓ	Loss function of a certain model
$\boldsymbol{\theta}$	Parameters of a certain model
$\mathcal{N}(\cdot)$	Corresponding novelty detector based on the given model
ξ_s	Novelty score of samples from source domain
ξ_t	Novelty score of samples from target domain

according to Oza et al. [2020]. (2) We reimplement the experiment in Oza et al. [2020] and compare our results with the paper's. (3) We apply the reproduced algorithms to a novel dataset and further analyze the performance of proposed method.

2 Problem Statement

The goal is to train a novelty detector based on labeled source domain training data and unlabeled target domain training data to detect samples in novel, unseen categories in the target domain data. Notations are given in Table 1. Note that some extensions, including "k" for "known" and "n" for "novel", are appended to the superscripts and subscripts to denote samples of known/novel classes from training/test set.

In this project, we are focusing on the digit recognition problem. Labels 0-4 are deemed as known classes and will be used to train the models, and labels 5-9 are to be novel classes.

$$\boldsymbol{X}_s^k = \{\boldsymbol{X}_s\}_{0\leq y_s\leq 4}, \quad \boldsymbol{X}_s^n = \{\boldsymbol{X}_s\}_{5\leq y_s\leq 9},$$

$$\boldsymbol{X}_t^k = \{\boldsymbol{X}_t\}_{0\leq y_s\leq 4}, \quad \boldsymbol{X}_t^n = \{\boldsymbol{X}_t\}_{5\leq y_s\leq 9},$$

The first phase is to train the base model. It will be trained on only training samples from known classes. Depending on models, the corresponding loss function should be minimized. The loss function can be explicitly dependent on the feature vectors, if the model contains a GAN and the loss has the generator loss. Although the three datasets we used in our project provide labels, we do not feed them during training if they are served as the target domain dataset.

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \ell(f(\boldsymbol{X}_{s,train}^k, \boldsymbol{y}_{s,train}^k, \boldsymbol{X}_{t,train}^k; \boldsymbol{\theta}), \mathbf{X}_{s,train}^k, \boldsymbol{y}_{s,train}^k, \boldsymbol{y}_{t,train}^k)$$

After training the base model, we use the novelty detector to output a novelty score for a test sample only from the target domain. The input comes from the output of the base model, such as maximum softmax probability if it contains a classifier.

$$\xi_t^k = \mathcal{N}(f(\boldsymbol{X}_{t,test}^k)), \quad \xi_t^n = \mathcal{N}(f(\boldsymbol{X}_{t,test}^n))$$

Ideally, if the base model is well designed and trained, the novelty score for novel samples should be higher than that for known samples.

3 Related Work

Many methods have been proposed to address novelty detection in recent decades. Support Vector Machine (SVM) is adopted to obtain the decision boundary to distinguish the outliers (Schölkopf et al. [1999]). The decision boundary could be expressed by the kernel expansion in terms of a small subset of the training data. As a supervised learning method, SVM needs labels for training; thus, the limitation to known labels in some scenarios may restrict its application. Kernel Principal

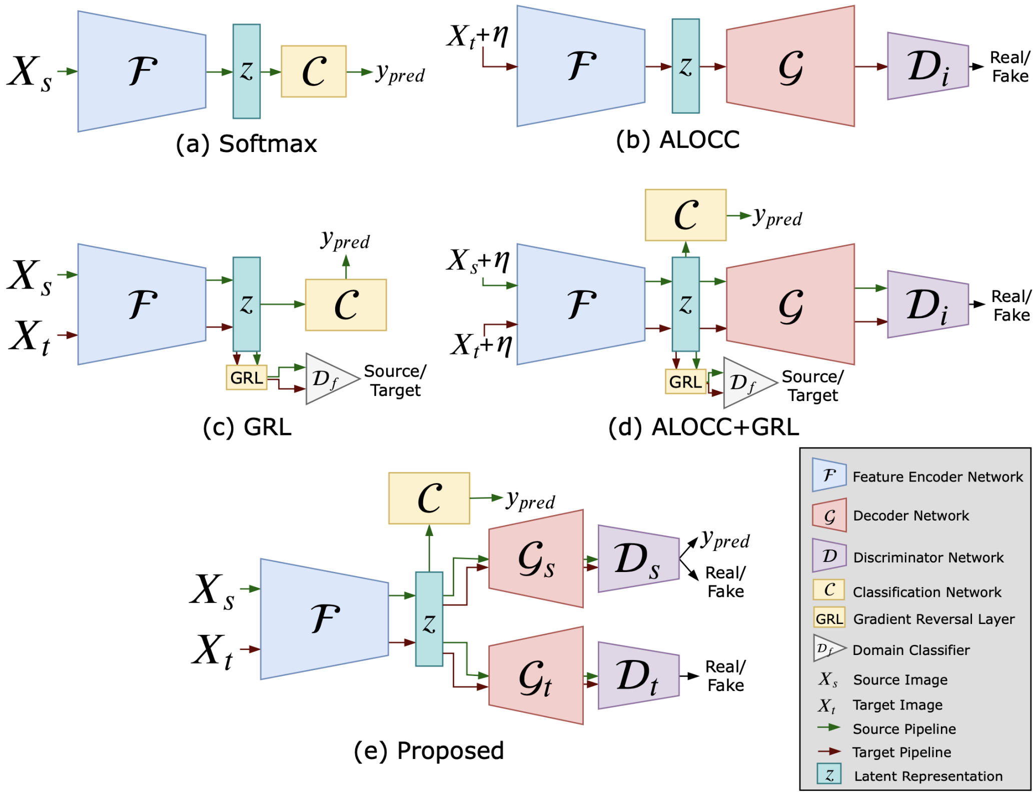
(a) Softmax
(b) ALOCC
(c) GRL
(d) ALOCC+GRL
(e) Proposed

Figure 1: Illustration of multiple potential solutions to address the distribution shift problem for novelty detection (Fig. 3 from Oza et al. [2020])

Component Analysis (KPCA) (Hoffmann [2007]), an unsupervised learning method, is adopted for novelty detection. The KPCA could provide the decision boundary based on the input training data without knowing the label. However, these kernel methods assume the training and testing data have the same distribution, so the distribution shifts in testing data will lead to degraded performance. For novelty detection involving image data, deep neural networks based method has been adopted. Research (Abati et al. [2019]) used auto-encoders to extract the feature of the image to a latent vector and then use a decoder to reconstruct the image with the latent vector as input. By minimizing the reconstruction error, the latent vector could represent the feature of the data. The latent data is then performed as the input of the estimation network for novelty detection. To better learn the data distribution, a deep convolutional GAN is adopted in (Schlegl et al. [2017]) to provide an anomaly scoring scheme based on the mapping from image to latent vector. The game between the generator network and discriminator network can lead to better latent feature extraction.

4 Methodology

4.1 Simple methods

According to the problem setting and related work discussed in previous section, several potential methods are considered as baselines in (Oza et al. [2020]) for tackling the problem. We describe these approaches as follows.

Softmax. The baseline softmax model only consists of a feature extractor and a classification network. The model strucutre is shown in Figure 1(a). We set the feature extractor based on LeNet architecture according to the supplementary materials of Oza et al.'s paper (Oza et al. [2020]). The classifier consists of single linear layer to predict the class probabilities for multi-class classification. In this model, only the labeled source data would be fed to train the networks. The training loop uses a categorical entropy loss to penalize the misclassification and updates parameters of the feature extractor and classification network.

ALOCC. The ALOCC model is composed of a feature extractor and a generative adversarial network that includes a decoder network and a discriminator network. The model structure is visualized in Figure 1(b), and the architectures of the decoder (generator) network and the discriminator network follow the structure presented in Table S1 in the Appendix. The ALOCC model is trained on unlabeled target data with an injection of random noise, represented by η in Figure 1(b) to distort the input samples and make the model more robust against noise. The training follows Sabokrou et al.'s paper (Sabokrou et al. [2018]): in the training loop, the extractor and decoder reconstruct the input and try to fool the discriminator that it is the original data while the discriminator has access to the original samples and learns to reject the reconstructed ones. The loss function takes a combination of the reconstruction loss to update the generator networks and the loss of identifying real and fake images to update the discriminator network.

GRL. Gradient reversal layer (Ganin and Lempitsky [2015]) is introduced to solve the domain gap issue between labeled data from the source domain and unlabeled data from the target domain. As an extension of Softmax baseline, a domain classifier is added and connected to the feature extractor via a gradient reversal layer that multiplies the gradient by a negative constant during backpropagation training. The training proceeds in a conventional way that minimize the label prediction loss for source data and the domain classification loss for both source and target data. The approach promotes the deep architecture by making it discriminative for classification task on the source domain and at the same time invariant with the shift between the domains.

ALOCC+GRL. ALOCC+GRL, a combination of novelty detector ALOCC and domain invariant feature learning method GRL is introduced to compare with the proposed method. Based on GRL, a GAN is implemented to guide the encoder to extract latent feature from both source and target data set. To train the ALOCC+GRL, we first minimize the label prediction loss for training data and domain prediction loss for both source and target data. This step is the same as in GRL. Then we train the generator by maximizing the prediction loss of discriminator, and minimizing the difference between the reconstructed and original image. Finally, the discriminator is trained by minimizing the prediction loss for fake and real labels.

4.2 Proposed method

The proposed model composes of a feature extractor, a classification network, a conditional GAN for the domain dataset (decoder + discriminator), and an ordinary GAN for the target dataset (decoder + discriminator). The model structure is visualized in Figure 1(e), and the configuration of each component is presented in Table S1 in the Appendix. Both training and test samples will be fed to the feature extractor to produce latent embeddings. Only the source embeddings will flow to the classification network and generate predictions of labels, since only the source dataset is supposed to have available labels. Embeddings from both domains will be fed to both GANs, and each discriminator is responsible for identifying true images from the corresponding domain. The design of two GANs for different domains is supposed to address the distribution shift problem, and the classification network as well as the choice of a conditional GAN as the source domain discriminator are supposed to utilize the multiclass information.

The custom training loop follows the original paper, as shown in Algorithm 1 in the appendix. First of all, both discriminators will be trained to identify whether inputs are real images or fake images. The binary cross-entropy loss function thus penalizes the wrong identification of real or fake images. For the source domain discriminator, it is also trained to output a label for images generated from source sample representations. Correspondingly, an additional categorical entropy loss is added to the loss function to penalize the misclassification. After the update of discriminators, the rest of the network will be trained. The generators should produce images so as to trick the discriminators, so the binary cross-entropy loss function penalizes failures to do so. In addition, to improve the image quality, the generated images will be compared to the original images, and a reconstruction loss based on L_1-loss will be applied to both generators. The classification network is trained also based on categorical cross-entropy loss to produce correct labels for the source domain images. Finally, the feature extractor's loss is a weighted sum of the three downstreaming components, controlled by two hyperparameters λ_1 and λ_2. Details of the implementation, including step-by-step mathematical formulation, can be found in Section A.1 in the appendix and in the original paper (Oza et al. [2020]).

The novelty detector based on trained proposed model can thus be constructed. New samples will be fed to the network, and we will need the outputs of class probabilities from the classification network, generated images from the target domain generator, and discriminator score from the target domain discriminator to distinguish whether they are from known or novel categories. Explanations of each score are discussed below:

1. Maximum softmax probability comes from the classification network. The smaller the value, the more uncertain the model is. For novel samples, the model should be much more uncertain than the known samples. They would have smaller softmax probability scores. (a minimized variable)
2. Generator loss comes from the target domain decoder. The larger the value, the larger difference the reconstructed image has compared to the original image. For novel samples, this value should be much larger. (a maximized variable)
3. Discriminator score comes from the target domain discriminator. For novel samples, they are more likely to be classified as fake images. So, they would have much smaller discriminator scores. (a minimized variable)

We followed the original paper and used the addition of these three scores as the final novelty score from the proposed model. Further discussion can be found in Section A.5 in the appendix.

4.3 Datasets

We used MNIST, USPS, and SVHN datasets to test the method similar to the procedure in the original paper. They are briefly introduced below:

- Modified National Institute of Standards and Technology database (MNIST; Lecun et al. [1998]) is one of the most famous datasets for digit recognition tasks. It is a size-normalized and presplit dataset with 60,000 handwritten digits in the training set and 10,000 samples in the test set. The image dimension is 28×28. The dataset can be accessed from `http://yann.lecun.com/exdb/mnist/`.
- US Post Office Zip Code Data (USPS; Hull [1994]) dataset consists of handwritten digits automatically scanned from envelopes by the U.S. Postal Service. The training set and the test set are also predetermined. There are 7,291 samples in the training set and 2,007 samples in the test set. Samples have been normalized to 16×16 grayscale images. The dataset can be obtained from `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html#usps`.
- The Street View House Numbers (SVHN) Dataset (SVHN; Netzer et al. [2011]) is a dataset for digit recognition tasks extracted from the real-world photos. It contains 73,257 digits for training, 26,032 digits for testing, and 531,131 additional training data. Here, for the sake of limited time and computational resources, we only used the provided training and test set. Samples are 32×32 true-color images with RGB channels. The images have already been cropped by the authors, but some distracting digits are introduced to the sides of the digit of interest. Thus, the SVHN is the most difficult one to train among these three datasets. The data can be downloaded from `http://ufldl.stanford.edu/housenumbers`.

To evaluate the method beyond the original paper, we also incorporate one additional dataset called Semeion.

- Semeion Handwritten Digit Dataset (Buscema [1998]) is produced from 1,593 scanned handwritten digits from different persons. They are scaled to 16×16 and processed to become boolean values (1/0). That is to say, they are black-and-white images. The dataset can be obtained from `https://archive.ics.uci.edu/ml/datasets/semeion+handwritten+digit`.

4.4 Data Preprocessing

We constructed the same four domain adaptation scenarios in our evaluation as the original paper: (1) MNIST->USPS; (2) SVHN->MNIST; (3) USPS->MNIST; (4) SVHN->USPS. That is to say, the

Table 2: Area Under ROC Curve (AUROC) score of four baseline models and the proposed model in four adaptation scenarios. The first number is the result from our own implementation, while the second number with brackets is the result from the original paper (Oza et al. [2020]). Notations: M - MNIST, U - USPS, S - SVHN. As for the proposed model, we used the discriminator score rather than the overall novelty score to perform the evaluation. See Section A.2 and A.5 in the appendix for more discussion.

	M->U	**U->M**	**S->M**	**S->U**	**Average**
Softmax	0.696 (0.602)	0.516 (0.651)	0.574 (0.642)	0.594 (0.587)	0.595 (0.620)
ALOCC	0.692 (0.633)	0.592 (0.702)	0.592 (0.702)	0.691 (0.633)	0.642 (0.667)
GRL	0.710 (0.863)	0.802 (0.859)	0.696 (0.718)	0.620 (0.667)	0.707 (0.776)
ALOCC+GRL	0.746 (0.903)	0.814 (0.895)	0.739 (0.851)	0.696 (0.845)	0.749 (0.873)
Proposed	0.754 (0.945)	0.874 (0.928)	0.741 (0.919)	0.715 (0.895)	0.771 (0.921)

Table 3: Area Under ROC Curve (AUROC) score of four baseline models and the proposed model when applied to our new dataset Semeion.

	MNIST->Semeion
Softmax	0.462
ALOCC	0.532
GRL	0.631
ALOCC+GRL	0.650
Proposed	0.662

left-hand side dataset will be regarded as the source domain dataset, while the right-hand side dataset will be regarded as the target domain dataset. We suspect that the reason why SVHN is not deemed as the target dataset is the difficulty of transferring knowledge from gray-scale images to true-color images. We also added an extra scenario to incorporate the Semeion dataset: (5) MNIST->Semeion.

As for the data preprocessing, we chose digits 0 to 4 as known categories and digits 5 to 9 as novel categories. We used the predetermined training and test set provided by each dataset. We split the Semeion dataset by stratified sampling. 80% of the data are randomly chosen as the training set, and the remaining 20% are put into the test set. All datasets are resized to 32×32 using bilinear interpolation method. Since SVHN contains true-color images and MNIST, USPS, and Semeion are gray-scale or black-and-white, the latter three are broadcast to 3 RGB channels. MNIST and SVHN data are rescaled to [0,1]. The original paper did not mention the data normalization, so we did not apply it in the first place. In addition, we formatted the class label of SVHN to be consistent of the other four datasets.

5 Evaluation

To evaluate and compare the model's effectiveness on novelty detection, we followed the original paper to introduce criteria for different models to quantify the novelty scores in the form of Area Under ROC Curve (AUROC) scores. We then calculated the novelty scores of each model in the four domain adaptation scenarios as well as the scenario with our implementation on the new Semeion dataset.

Like the original paper, we utilize the maximum softmax probability which can be obtained by TensorFlow's embedded function `tf.nn.softmax()` as the novelty score to evaluate the effectiveness of the Softmax model. The comparisons between results of our implemented model with the results of original paper in the first four adaptation scenarios is shown in the first row of Table 2. The scores of our model have 1.19% to 20.7% deviations from the paper's results. The score is higher when MNIST is used as the source data. Our implemented Softmax model gets an average AUROC score of 0.595 in the four adaptation scenarios which is 96.0% of the paper's result. In the scenario where we use the new Semeion data set that is not in the original paper as the target data, the AUROC score we get for the Softmax model is 0.462. The results of the implementation of Softmax model is as expected since this model only applies the conventional CNN training for recognition. The classification networks are prone to novel classes even in the source domain, hence would not translate well for the target domain novelty detection (Oza et al. [2020]) and cause the Softmax to get a lower score comparing to other more complicated models.

Since we do not have a classification network but have a GAN network in the ALOCC model, we applied a different criterion to use the score from the discriminator of the reconstructed image as the novelty score. The results on the same four adaptation scenarios are shown in the second row of Table 2. The scores of our model have 8.43% to 15.7% deviations from the paper's results. Our implemented ALOCC model gets an average AUROC score of 0.642, i.e. 96.3% of the paper's result. In the two scenarios where USPS is used as the target data, our model scores higher than the paper's reported results. In the other two cases where MNIST is used as the target data, scores of our model are lower than the output of the original paper. Same as the reported output of the paper, the introduction of GAN network in place of the classification network has improved the performance of the Softmax model on novelty detection by around 10%. When we implemented the ALOCC model on the new scenario with Semeion data set as the target set, the AUROC score is increased to 0.532.

We also extended the Softmax model to GRL model by adding gradient reversal layer and domain classifier. The model basically performs better comparing to Softmax and ALOCC, as domain adaption is considered and both labeled source and unlabeled target data are used during the training. The AUROC scores of the four cases are lower than that from the original paper and the deviations are smaller when SVHN is used as source data. We also applied GRL model to the novel dataset Semeion. The model achieves an AUROC score 0.631, which is much smaller than the similar case MNIST -> USPS. Details of the model and the possible reasons that might cause this deviation could be found in Section A.7.

ALOCC+GRL is a combination of ALOCC and GRL model. The Gaussian noise corrupted source and target data are used for training. The maximum softmax probability of classification net, image re-construction loss by GAN discriminator and the output of discriminator are adopted to detect the novelty score. To be consistent with GRL model we implemented, the score by domain classifier are not used for novelty detection. As is suggested in Table 2, ALOCC+GRL model outperforms the GRL and ALOCC in the dataset in paper Oza et al. [2020]. When applied to novel dataset Semeion, the AUROC score is 0.650.

As stated in Section 4.2, the proposed model utilized three criteria to produce the final novelty score. Nevertheless, we output all individual novelty score together with the overall novelty score, and plotted the ROC curve for each of them. In Table 2, the AUROC score of the proposed model is presented. Compared to other baseline models, the proposed model as a relatively better AUROC score in all adaptation scenarios. The estimated performance is the best in USPS->MNIST experiment, and the AUROC score can reach 0.874. The model performs worse in generalizing the SVHN dataset, with the AUROC score below 0.75. These results are not consistent with what the original paper's presentation, and the differences are larger than 0.10 in most cases. When applied to the Semeion dataset, the proposed model ends up with an AUROC score of 0.662.

In summary, although our results show that the models' relative order of performance in detecting novel samples is consistent with the original model (Proposed > ALOCC+GRL > GRL > ALOCC > Softmax), our results do not agree well with the original model. Instead of continuing the evaluation on other datasets or other adaptation scenarios, we performed some diagnostic analysis on each component of the proposed model. Given the limited space here, we summarized some key points here, and details could be found in Section A.2 to Section A.6 in the appendix.

1. Randomization matters. When we trained the models, the final results are not stable. Even with the same configuration, the trained model can have a very large fluctuation in the performance. This occurs even in the training of our simplest softmax model. We hypothesize that the random initialization of parameters might be a key factor, and the Adam optimizer might not be the optimal choice here.

2. Both discriminators in the proposed model suffer from overfitting issues. If we kept the original algorithm stated in the paper, we would have worse results. After using the regular GAN training loop and applying a Gaussian noise to the input images, the overfitting of the discriminators is partially relieved, and the performance is slightly improved.

3. The discriminator score makes a greater contribution to the overall novelty score than the maximum softmax probability and the generator loss.

4. If we tune the weights of the three novel scores rather than simply adding them together, we could potentially achieve a better AUROC score for the proposed model.

6 Conclusion

Our project discussed solutions of novelty detection under dataset distribution shift. We re-implemented the proposed method as well as four baseline models in Oza et al. [2020]. The output of our project agrees with the reported results in a relative order of performance for different methods on novelty detection. However, results of our implementation reveal some deviations from the original model. We also implemented our models with a novel dataset Semeion and further proved that the proposed model outperforms baseline models in solving distribution shift problem in novelty detection. Diagnostic analysis suggested some issues of the proposed model that the original paper did not mention, including the randomness of training results and the limited improvement when true-color images are used as source data.

The main takeways of our project include:

1. Instead of reporting the simple numbers of the final results, doing some diagnostic tests and presenting them can promote the transparency and interpretability of the model and help us understand the real performance;
2. Randomization matters and could result in different results from the same configuration.

The results of the 5 methods also inspired us to think about what causes the distinct performances. Comparing Softmax and ALOCC with the other 3 methods, Softmax and ALOCC only uses target or source data for novelty detection. Thus, is expected that the other 3 methods, which uses both target and source data for training, outperform Softmax and ALOCC. As additional GAN is combined with ALOCC, ALOCC+GRL is expected to outperform the GRL. The proposed method uses another GAN in the place of gradient reversal layer presented in ALOCC+GRL. The improved performance suggests that GAN is more efficient in guiding the information extraction than the gradient reversal layer. The distinction of performances between Softmax and ALOCC also verify that the implementation of GAN is a great plus in performance.

Despite the advantages, the complexity of the proposed methods greatly increased the difficulty in the implementation. As more criterion are adopted to compute the novelty score, hyper-parameters, e.g. the weights between different criterion, are hard to tune. This problem may result in the gap between our implementation and the one by the original paper. On the contrary, our implementation of Softmax and ALOCC are closer to the original paper, which we believe is attributed to their simpler structures and less hyper-parameters. The trade-off between the difficulties in tuning and performance should be well addressed in real-world application.

In conclusion, our project succeed in revealing more details and insights into the models solving distribution shift during novelty detection. However, due to limited time and limited epochs, we did not get explicit convergence criteria. For future work, we have contacted the author and they will release the source code by the end of December. And we will find out what's really happening here.

References

D. Abati, A. Porrello, S. Calderara, and R. Cucchiara. Latent space autoregression for novelty detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 481–490, 2019.

M. Buscema. Metanet*: The theory of independent judges. *Substance Use & Misuse*, 33 (2):439–461, 1998. doi: 10.3109/10826089809115875. URL `https://doi.org/10.3109/10826089809115875`.

G. Cabanes, Y. Bennani, and N. Grozavu. Unsupervised learning for analyzing the dynamic behavior of online banking fraud. In *2013 IEEE 13th International Conference on Data Mining Workshops*, pages 513–520, 2013. doi: 10.1109/ICDMW.2013.109.

Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 1180–1189. JMLR, 2015.

D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *CoRR*, abs/1610.02136, 2016. URL `http://arxiv.org/abs/1610.02136`.

H. Hoffmann. Kernel PCA for novelty detection. *Pattern Recognition*, 40(3):863–874, 2007. ISSN 00313203. doi: 10.1016/j.patcog.2006.07.009. URL `https://linkinghub.elsevier.com/retrieve/pii/S0031320306003414`.

J. J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994. doi: 10.1109/34.291440.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. ISSN 1558-2256. doi: 10.1109/5.726791.

Y. Liu, Y. Xie, and A. Srivastava. Neural trojans, 2017. URL `http://arxiv.org/abs/1710.00942`.

Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading Digits in Natural Images with Unsupervised Feature Learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

P. Oza, H. V. Nguyen, and V. M. Patel. Multiple Class Novelty Detection Under Data Distribution Shift. In *Proceedings of the European Conference of Computer Vision (ECCV)*, pages 1–17, Glasgow, Scotland, 2020. URL `https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123520426.pdf`.

M. Sabokrou, M. Khalooei, M. Fathy, and E. Adeli. Adversarially Learned One-Class Classifier for Novelty Detection. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3379–3388. IEEE, 2018. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00356. URL `https://ieeexplore.ieee.org/document/8578454/`.

T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging*, pages 146–157. Springer, 2017.

B. Schölkopf, R. C. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt. Support vector method for novelty detection. *Advances in neural information processing systems*, 12:582–588, 1999.

B. Sun and K. Saenko. Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9915 LNCS, pages 443–450. 2016. ISBN 9783319494081. doi: 10.1007/978-3-319-49409-8_35. URL `http://link.springer.com/10.1007/978-3-319-49409-8_35`.

Appendix

A.1 Inference and Training of Proposed Model

We implemented the proposed model according to the instructions provided by the original paper. Denote $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$ as feature vectors of samples from the source domain and the target domain, $\boldsymbol{y}_s$ and $\boldsymbol{y}_t$ as label of samples from source domain and the target domain. We use the same notations in Figure 1(e) in the main text to refer to the individual components of the model.

Both source feature vectors $\boldsymbol{X}_s$ and target feature vectors $\boldsymbol{X}_t$ are fed into the feature extractor to produce the corresponding feature embeddings:

$$\boldsymbol{z}_s = \mathcal{F}(X_s), \qquad \boldsymbol{z}_t = \mathcal{F}(\boldsymbol{X}_t)$$

The classification network only takes the source embeddings as the input, and outputs a label.

$$\widehat{\boldsymbol{y}}_s = \mathcal{C}(\boldsymbol{z}_s)$$

The source domain generator and target domain generator take both source embeddings and target embeddings as the input, and they eventually reconstruct four types of images:

$$\begin{aligned} \widehat{\boldsymbol{X}}_{s2s} &= \mathcal{G}_s(\boldsymbol{X}_s), \qquad \widehat{\boldsymbol{X}}_{t2s} = \mathcal{G}_s(\boldsymbol{X}_t) \\ \widehat{\boldsymbol{X}}_{s2t} &= \mathcal{G}_t(\boldsymbol{X}_s), \qquad \widehat{\boldsymbol{X}}_{t2t} = \mathcal{G}_t(\boldsymbol{X}_t) \end{aligned}$$

The source domain discriminator will identify whether the images are fake or real by calculating the discriminator score. Plus, it outputs an extra variable to classify the generated images to one of the five training categories.

$$\widehat{\boldsymbol{d}}_{s2s}, \widehat{\boldsymbol{y}}_{s2s} = \mathcal{D}_s\left(\widehat{\boldsymbol{X}}_{s2s}\right), \qquad \widehat{\boldsymbol{d}}_{t2s}, \widehat{\boldsymbol{y}}_{t2s} = \mathcal{D}_s\left(\widehat{\boldsymbol{X}}_{t2s}\right)$$

The target domain discriminator will only identify the real or fake images but not classify the reconstructed images, because the label is not available for target dataset.

$$\widehat{\boldsymbol{d}}_{s2t} = \mathcal{D}_t\left(\widehat{\boldsymbol{X}}_{s2t}\right), \qquad \widehat{\boldsymbol{d}}_{t2t} = \mathcal{D}_t\left(\widehat{\boldsymbol{X}}_{t2t}\right)$$

During the training phase, the discriminators are first trained based on binary cross-entropy loss. For the source domain discriminator, the categorical cross-entropy loss is added to penalize the classification error.

$$\begin{aligned} \ell_{\mathcal{D}_s} &= -\mathbb{E}\left[\log\left(\mathcal{D}_s(\boldsymbol{X}_s)\right)\right] - \mathbb{E}\left[\log\left(1-\widehat{\boldsymbol{d}}_{s2s}\right)\right] - \mathbb{E}\left[\log\left(1-\widehat{\boldsymbol{d}}_{t2s}\right)\right] + \mathbb{E}[\ell_{ce}(\widehat{\boldsymbol{y}}_{s2s}, \boldsymbol{y}_s)] \\ \ell_{\mathcal{D}_t} &= -\mathbb{E}\left[\log\left(\mathcal{D}_t(\boldsymbol{X}_t)\right)\right] - \mathbb{E}\left[\log\left(1-\widehat{\boldsymbol{d}}_{s2t}\right)\right] - \mathbb{E}\left[\log\left(1-\widehat{\boldsymbol{d}}_{t2t}\right)\right] \end{aligned}$$

where ℓ_{ce} is the categorical cross-entropy loss (here *j* represents class index):

$$\ell_{ce}(\widehat{\boldsymbol{y}}, \boldsymbol{y}) = -\sum_j \boldsymbol{y}_j \log \widehat{\boldsymbol{y}}_j$$

Then the rest of the network is trained. The generator loss is defined as:

$$\begin{aligned} \ell_{\mathcal{G}_s} &= -\mathbb{E}\left[\log\left(\widehat{\boldsymbol{d}}_{s2s}\right)\right] - \mathbb{E}\left[\log\left(\widehat{\boldsymbol{d}}_{t2s}\right)\right] + \mathbb{E}\left[\left\|\widehat{\boldsymbol{X}}_{s2s} - \boldsymbol{X}_s\right\|\right] \\ \ell_{\mathcal{G}_t} &= -\mathbb{E}\left[\log\left(\widehat{\boldsymbol{d}}_{s2t}\right)\right] - \mathbb{E}\left[\log\left(\widehat{\boldsymbol{d}}_{t2t}\right)\right] + \mathbb{E}\left[\left\|\widehat{\boldsymbol{X}}_{t2t} - \boldsymbol{X}_t\right\|\right] \end{aligned}$$

This is to train the generators to generate images and fool the discriminators. Plus, the last term is a ℓ_1 reconstruction loss to improve the image quality. The classification loss in the classification network is defined as:

$$\ell_{\mathcal{C}} = \mathbb{E}[\ell_{ce}(\widehat{\boldsymbol{y}}_s, \boldsymbol{y}_s)]$$

Finally, the feature extractor's loss function is a weighted sum of classification loss and generator loss.

$$\ell_{\mathcal{F}} = \ell_{\mathcal{C}} + \lambda_1 \ell_{\mathcal{G}_s} + \lambda_2 \ell_{\mathcal{G}_t}$$

where λ_1 and λ_2 are hyperparameters. The pseudo-code is presented in Algorithm 1 below. In our implementation, we closely followed the original paper and used the same training configuration. We used Adam optimizer with a learning rate of 0.0001 and batch size of 64. The hyperparameters λ_1 and λ_2 are both set to 0.03. These hyperparameters are provided by the original paper, and we do not need to tune them in our training.

Algorithm 1 Pseudocode for Training Proposed Method

```
Require: Network models $\mathcal{F}, \mathcal{C}, \mathcal{G}_s, \mathcal{D}_s, \mathcal{G}_t, \mathcal{D}_t$
Require: Initial parameters $\boldsymbol{\theta}_{\mathcal{F}}, \boldsymbol{\theta}_{\mathcal{C}}, \boldsymbol{\theta}_{\mathcal{G}_s}, \boldsymbol{\theta}_{\mathcal{D}_s}, \boldsymbol{\theta}_{\mathcal{G}_t}, \boldsymbol{\theta}_{\mathcal{D}_t}$
Require: Source data $X^k_{s,train}, y^k_{s,train}$; Target data $X^k_{t,train}$
Require: Hyperparameters $N, \gamma, \lambda_1, \lambda_2$
while not done do
    for each batch with size N do
        for i=1 to N do
            Feedfoward through the network
        end for
        Calculate losses $\ell_{\mathcal{D}_s}$, $\ell_{\mathcal{D}_t}$, $\ell_{\mathcal{G}_s}$, $\ell_{\mathcal{G}_t}$, $\ell_{\mathcal{C}}$, $\ell_{\mathcal{F}}$
        Update $\boldsymbol{\theta}_{\mathcal{D}_s} \leftarrow \boldsymbol{\theta}_{\mathcal{D}_s} - \gamma \cdot \nabla_{\boldsymbol{\theta}_{\mathcal{D}_s}} \ell_{\mathcal{D}_s}$
        Update $\boldsymbol{\theta}_{\mathcal{D}_t} \leftarrow \boldsymbol{\theta}_{\mathcal{D}_t} - \gamma \cdot \nabla_{\boldsymbol{\theta}_{\mathcal{D}_t}} \ell_{\mathcal{D}_t}$
        Update $\boldsymbol{\theta}_{\mathcal{G}_s} \leftarrow \boldsymbol{\theta}_{\mathcal{G}_s} - \gamma \cdot \nabla_{\boldsymbol{\theta}_{\mathcal{G}_s}} \ell_{\mathcal{G}_s}$
        Update $\boldsymbol{\theta}_{\mathcal{G}_t} \leftarrow \boldsymbol{\theta}_{\mathcal{G}_t} - \gamma \cdot \nabla_{\boldsymbol{\theta}_{\mathcal{G}_t}} \ell_{\mathcal{G}_t}$
        Update $\boldsymbol{\theta}_{\mathcal{C}} \leftarrow \boldsymbol{\theta}_{\mathcal{C}} - \gamma \cdot \nabla_{\boldsymbol{\theta}_{\mathcal{C}}} \ell_{\mathcal{C}}$
        Update $\boldsymbol{\theta}_{\mathcal{F}} \leftarrow \boldsymbol{\theta}_{\mathcal{F}} - \gamma \cdot \nabla_{\boldsymbol{\theta}_{\mathcal{F}}} \ell_{\mathcal{F}}$
    end for
end while
Output: Learned parameters $\widehat{\boldsymbol{\theta}}_{\mathcal{F}}, \widehat{\boldsymbol{\theta}}_{\mathcal{C}}, \widehat{\boldsymbol{\theta}}_{\mathcal{G}_s}, \widehat{\boldsymbol{\theta}}_{\mathcal{D}_s}, \widehat{\boldsymbol{\theta}}_{\mathcal{G}_t}, \widehat{\boldsymbol{\theta}}_{\mathcal{D}_t}$
```

A.2 Improvements on the Proposed Model

After we have trained the model strictly following the instructions given by the original paper, we found that the performance is unsatisfactory at all. The discriminator loss drops to almost zero very quickly, and the generator loss increases all the way up. As a result, the AUROC score based on this original model is slightly above 0.5 (random guess). The performance is even worse when SVHN is deemed as the source dataset, probability due to the complexity of the dataset. Therefore, we have applied the following improvements to the proposed model and tested whether they will induce a performance increase.

First, we modified the training loop according to the general GAN training procedure. In Algorithm 1 above, the discriminator update and the generator update are performed together within one forward pass. On the other hand, the regular GAN update trains the discriminator first with the generator frozen, and then trains the generator freezing the discriminator. We adopted this method and trained this set of models. The results showed that the generator loss is significantly reduced, and the AUROC score improves by approximately 5% in all scenarios. Remarkably, when it comes to the three novelty scores by each component (maximum probability score, reconstruction loss, and discriminator score), the discriminator score stands out and makes a great contribution to the overall novelty score. In the USPS->MNIST case, the discriminator reaches a test AUROC score of 0.952, boosting the overall AUROC score to 0.78.

Second, we tried centering the data set. Both generators end up with a convolutional transpose layer with the hyperbolic tangent function. The hyperbolic tangent function is bounded between (-1,1). Meanwhile, our input images are scaled to [0,1] in our preprocessing procedure. It is likely that the discriminator will become overfitted by focusing on the scale of some certain pixels, leading to poor results. The test results do not show a performance boost after we centered the datasets, though.

Third, we added some Gaussian noise to the input images for the discriminator, as can be seen in Figure S1. Practical experience shows that the generator might be much harder to train than the discriminator. The technique of adding Gaussian noise is to overcome the overfitting problem of the discriminator, as the discriminator dropped to 0 in the original method. We applied the noise to all datasets with the standard deviation of 0.05. Also, we only added the noise to the images fed directly into the discriminator when training the discriminators, not to the rest of the network. The purpose is to increase the difficulty for the discriminators but not the generators. We combined all the techniques above and evaluated the models, and we found some improvements over the original model. Thus, the results presented below and in the main text all come from this series of models.

A.3 Discussion of Training Loss

Figure S2, Figure S6, Figure S10, and Figure S14 shows the training loss of the proposed model in four different adaptation scenarios. Take the MNIST->USPS case as an example. We can see that all losses drop very quickly after a few updates at the beginning, and the losses become steady afterwards. After applying the three additional techniques introduced above, we still see an overfitting issue for the discriminators, especially for the target domain discriminator. The target domain discriminator loss approaches close to 0 in a few epochs. The generators experience some extent of instabilities during the training, as can be observed from the generator loss.

Training results are different among various adaptation scenarios. For example, when the SVHN dataset is used as a source domain dataset, the source discriminator loss and the classification loss are much higher than the other two cases. One of the reasons might be the complicity of the SVHN dataset, since it contains true-color information instead of gray-scale information.

The training results are also sensitive to the random initialization of the model. Even though the model is trained with the same configuration, the training loss and the final evaluation scores could be totally different. This issue even happened on the training of our softmax model, the simplest baseline model. It will absolutely be worse for the complicated proposed model with millions of parameters. The original paper did not demonstrate the details of how they trained the model, how did the loss functions look like, and how each component made a contribution to the final results presented in the paper.

A.4 Diagnostic Analysis of Generators

Figure S3, Figure S7, Figure S11, and Figure S15 show the actual generated images from the source domain generator in each scenario. These sample images are generated by feeding them the source embeddings. Figure S4, Figure S8, Figure S12, and Figure S16 are the corresponding generated images from the target domain generator. That is to say, the source domain generator utilized the source embeddings to generate reconstructed images, while the target domain generator utilized the target embeddings to generate new images. The first row of each group images is the original images of known categories, and the second row is the reconstructed version of them. The third row is the original images of novel categories, and the fourth row is the reconstructed version of them.

We can see that the generated images for the known classes and novel classes are very different. We can somehow recognize the known digits from the reconstructed images, and they correspond well to the true label. When fed with a novel digit's image, the generators cannot produce an image of the certain novel digit. Intuitively, we can see that the generators are trained fairly to make it easier for the ultimate novelty detection task.

Note that the goal is not to recover the original images as much as possible. If so, we only need an autoencoder to achieve the goal and we do not need the discriminators. An additional interesting experiment has been done to remove the discriminators and retrain the models. It turns out that the generators together with the previous components can act as an autoencoder and recover the images very well (results not shown in this paper). When the input is either a source embedding or a target embedding, the generator can reproduce the images very clear.

A.5 Discussion of Novelty Scores

In the original paper, the author mentioned to use the addition of maximum softmax probability, generator loss, and discriminator score to be the final novelty score. First, we would like to explain each score in detail.

1. Maximum softmax probability comes from the classification network. The smaller the value, the more uncertain the model is. For novel samples, the model should be much more uncertain than the known samples. They would have smaller softmax probability scores. (a minimized variable)
2. Generator loss comes from the target domain decoder. The larger the value, the larger difference the reconstructed image has compared to the original image. For novel samples, this value should be much larger. (a maximized variable)
3. Discriminator score comes from the target domain discriminator. For novel samples, they are more likely to be classified as fake images. So, they would have much smaller discriminator scores. (a minimized variable)

Given the explanation above, we constructed the final novelty score using the following formula:

$$n_{total} = -c_{sm} \cdot n_{sm} + c_g \cdot n_g - c_d \cdot n_d$$

where n_{sm}, n_g, n_d are the maximum softmax probability, generator loss, and discriminator score, correspondingly. The coefficients c_{sm}, c_g, c_d are arbitrarily determined. Considering that the original paper used "the addition of" the three scores, we kept those coefficients to be equal to 1. However, it should be noted that these coefficients could be tuned to improve the performance score of the novelty detector.

Figure S5, Figure S9, Figure S13, and Figure S17 show the ROC curve and the AUROC score in the legend. We've also plotted individual scores as a diagnostic evaluation. We can see that the generator loss and the discriminator loss can be useful in the task of novelty detection. The maximum softmax probability seems poor, especially when the SVHN dataset is used as the source domain dataset. As a result, in the main text we used the discriminator score to represent the final novelty score instead of the overall score here. This can be achieved by setting the coefficients:

$$c_{sm} = 0, c_g = 0, c_d = 1$$

A.6 Diagnostic Analysis of Latent Representations

Given the limited time we are given for the final project, we did not perform analysis on the latent representations extracted by the feature extractor. The supplementary materials of the original paper showed a figure of the t-SNE visualization of the latent representations from each model. The figure is also reproduced as Figure S18. The author argued that softmax, ALOCC and GRL models have known and novel category features overlapped. When looking at the figure, our understanding is the opposite: the overlapping issue is only apparent and severe in ALOCC. In fact, the softmax model generalizes well. As for the ALOCC+GRL and proposed model, the author stated that the features are well separated, while we argued that the severity of overlapping is not reduced too much compared to softmax and GRL.

A.7 Additional Information of GRL Model

Some components that are different from the proposed model are introduced below.

Flip Gradient Builder: The gradient reversal layer has no parameters as-sociated with it (apart from the meta-parameter λ, which is not updated by backpropagation). During the forward propagation, GRL acts as an identity transform. During the backpropagation though, GRL takes the gradient from the subsequent level, multiplies it by $-\lambda$ and passes it to the preceding layer.

Domain Classifier: It has similar architecture with classification network. The difference is that the output shape of domain prediction is 2 because there are only two categories: source and target.

The loss of the GRL model is written as:

$$\ell_{total} = \ell_{\mathcal{C}} + \ell_{domain}$$

where $\ell_{\mathcal{C}}$ is the categorical cross-entropy loss used to penalize source data classification error, and ℓ_{domain} is the total binary cross-entropy loss of both source and target data and is used to penalize domain classification error.

The original paper did not reveal details about the definition of novelty scores using GRL method. In this case, we did not consider the contribution of domain classifier and the novelty score is simply defined as the maximum softmax probability score, which is the same with softmax method.

Figure S19 shows the ROC curves and AUC scores. The scores of the four cases are smaller than the scores provided in the paper, which may cause by the differences defining novelty scores. The performance of the model on these four cases basically follow the same pattern of the paper.

We also trained the model using new dataset Semeion. The ROC curve and AUC score are shown in Figure S20. Here we use MNIST dataset as the source data, and Semeion dataset as the target data. Comparing with the similar case ‘MNIST to USPS’ in Figure S19, the performance of GRL model doing novelty detection dropped when using novel dataset. One possible reason is that the Semeion dataset only contains 1274 training samples while USPS has 7291, which is much larger than Semeion. Moreover, the model itself has some randomness and the result of each experiment could vary.

A.8 Supplementary Tables

Table S1. Configuration of layers in each component. BN refers to Batch Normalization, and LReLU refers to Leaky ReLU with the parameter equal to 0.2.

Component	Layer	Configuration
Feature Extractor	I0	Input 32 × 32 × 3
	D1	Dropout (0.2)
	C2	Conv2D, 5 × 5, 64, stride 1, pad 0, ReLU
	M3	MaxPool2D, 2 × 2, stride 2
	D4	Dropout (0.2)
	C5	Conv2D, 5 × 5, 64, stride 1, pad 0, ReLU
	M6	MaxPool2D, 2 × 2, stride 2
	D7	Dropout (0.2)
	C8	Conv2D, 5 × 5, 128, stride 1, pad 0, ReLU
	F9	Flatten Features
Classification Network	I0	Input 128
	D1	Dense, 128, ReLU
	D2	Dense, 5, ReLU
Generator	I0	Input 128
	CT1	Conv2DTranspose, 2 × 2, 512, stride 1, pad 0, BN, ReLU
	CT2	Conv2DTranspose, 4 × 4, 256, stride 2, pad 1, BN, ReLU
	CT3	Conv2DTranspose, 4 × 4, 128, stride 2, pad 1, BN, ReLU
	CT4	Conv2DTranspose, 4 × 4, 64, stride 2, pad 1, BN, ReLU
	CT5	Conv2DTranspose, 4 × 4, 3, stride 2, pad 1, Tanh
Discriminator	I0	Input 32 × 32 × 3
	C1	Conv2D, 3 × 3, 64, stride 1, pad 1, BN, LReLU
	M2	MaxPool2D, 2 × 2, stride 2
	C3	Conv2D, 3 × 3, 128, stride 1, pad 1, BN, LReLU
	M4	MaxPool2D, 2 × 2, stride 2
	C5	Conv2D, 3 × 3, 256, stride 1, pad 1, BN, LReLU
	M6	MaxPool2D, 4 × 4, stride 4
	F7	Flatten Features
	D8	(Binary classifier) Dense, 1, Sigmoid (Multiclass classifier) Dense, 5, ReLU

A.9 Supplementary Figures

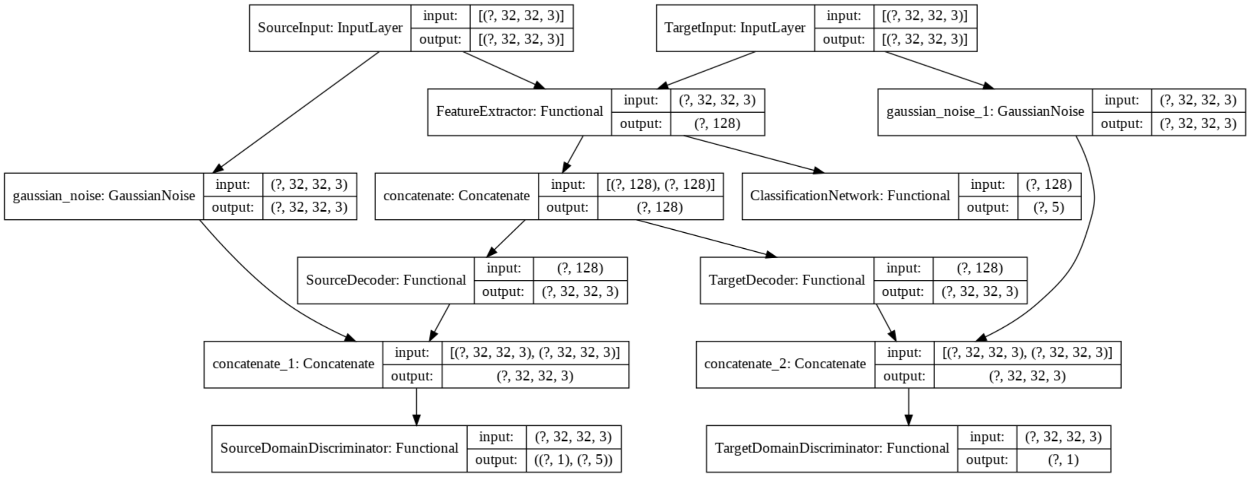

Figure S1. Our implementation of the proposed model. The Gaussian Noise layer is added to try to reduce the overfitting of the discriminators, as discussed in Section A.2. Concatenate layers are used to combine the source domain data and target domain data to form a single batch vector, which is to assure the correct functioning of batch normalization layers.

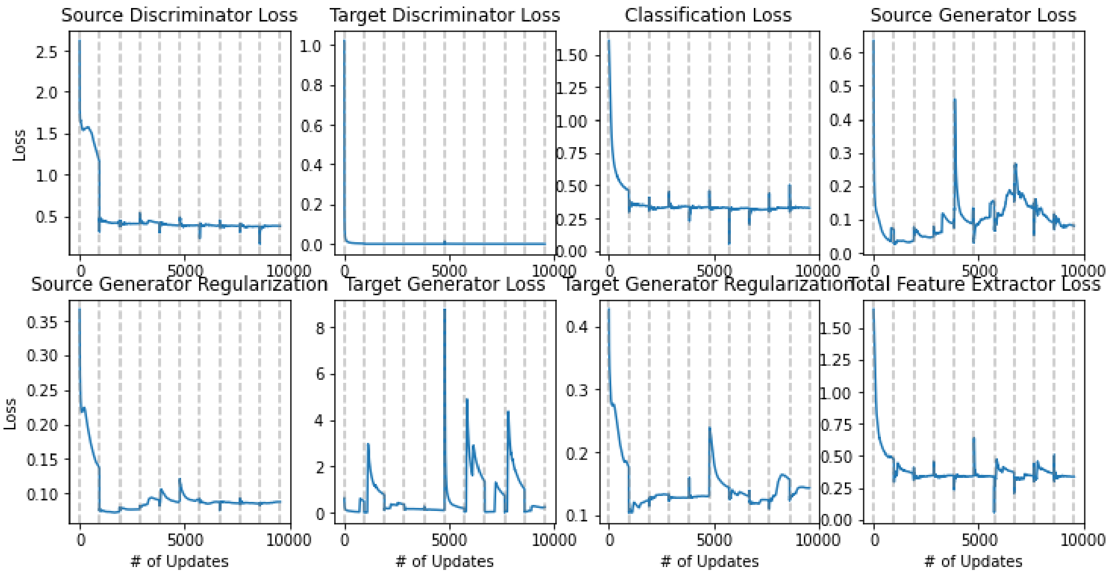

Figure S2. Training loss of the proposed model in the MNIST->USPS experiment. The gray vertical lines mark the epoch start.

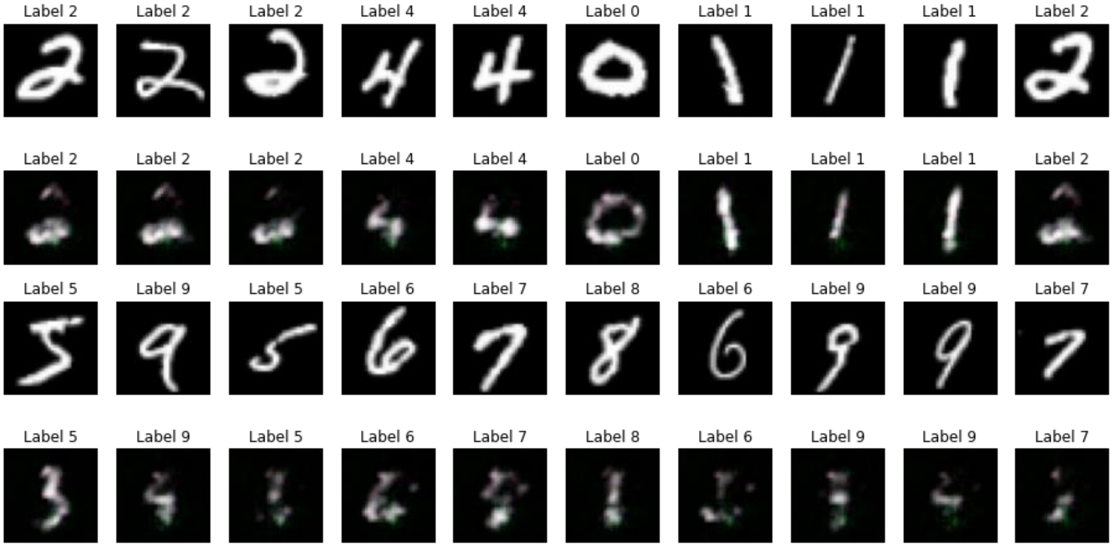

Figure S3. Samples of reconstructed images by the source domain generator when fed with source embeddings. The first and the third row are the original image, while the second and the fourth row are the reconstructed one. They are based on the proposed model trained on MNIST->USPS experiment.

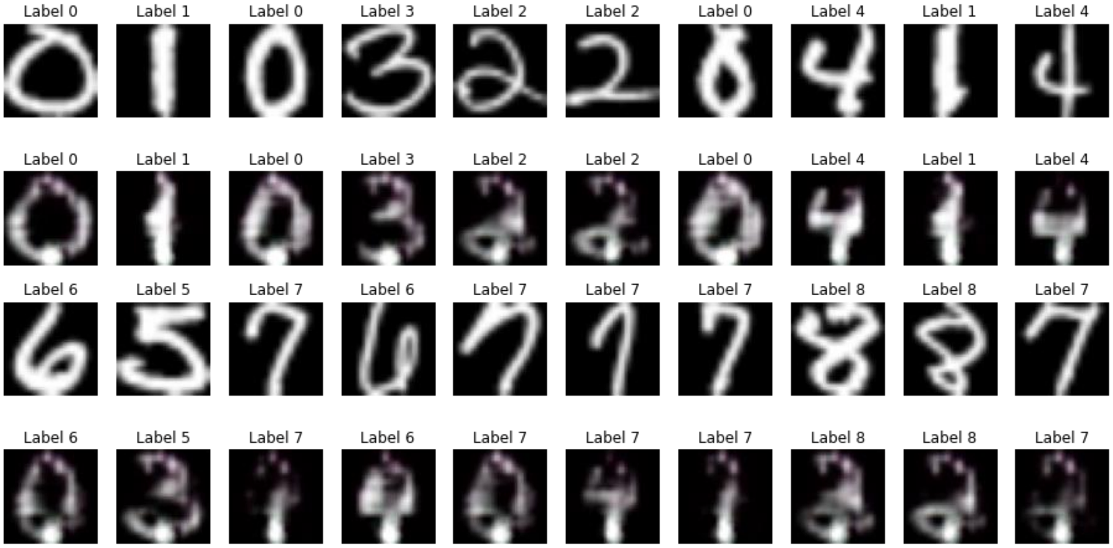

Figure S4. Same as Figure S3, but generated by the target domain generator when fed with target embeddings.

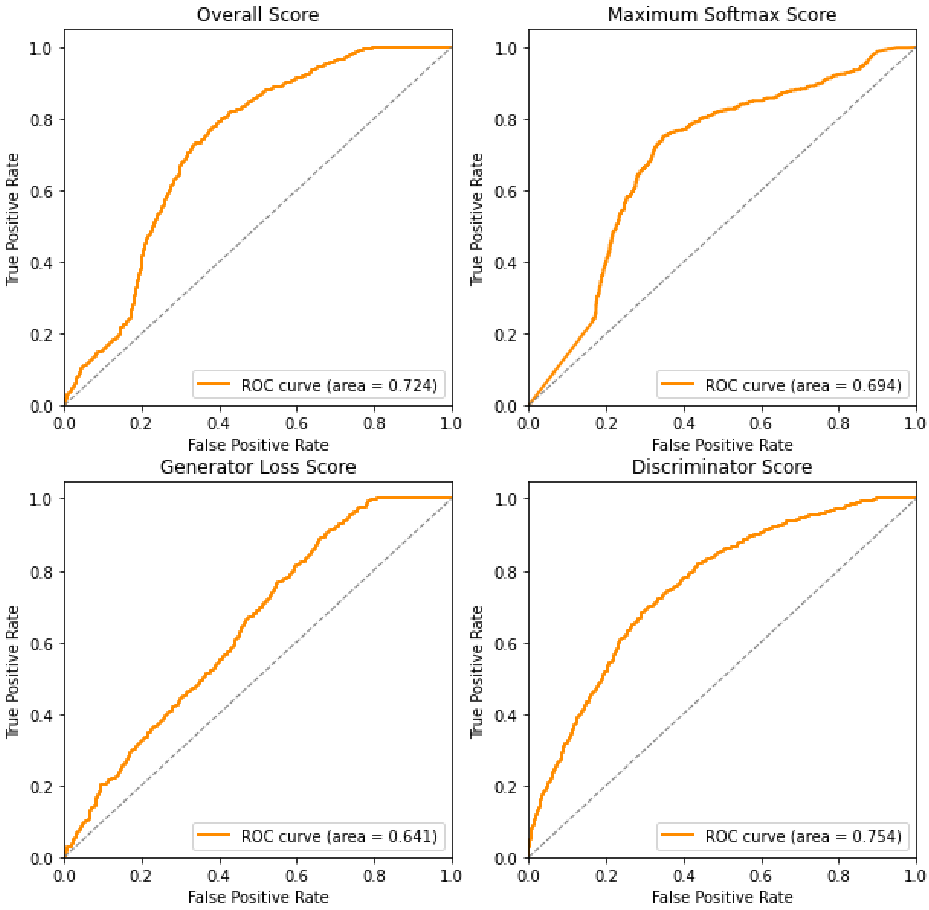

Figure S5. ROC curve based on the proposed model trained on MNIST->USPS experiment. The evaluation is on the test set from USPS dataset (target domain). The AUROC score is shown in the legend. Overall score is calculated by simply adding the three individual scores, as suggested by the original paper.

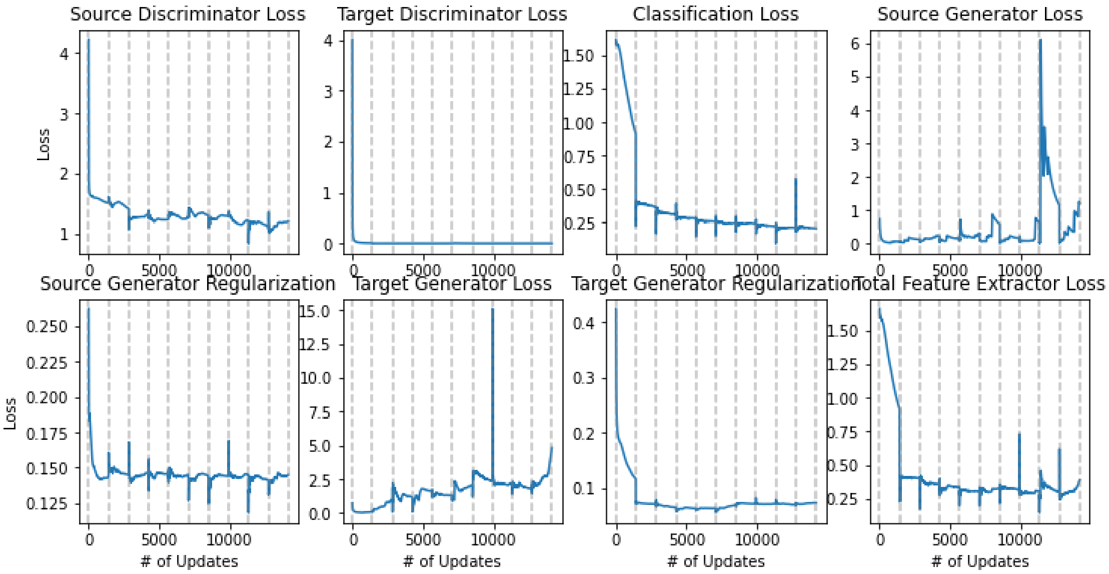

Figure S6. Same as **Figure S2**, but for the SVHN->MNIST case.

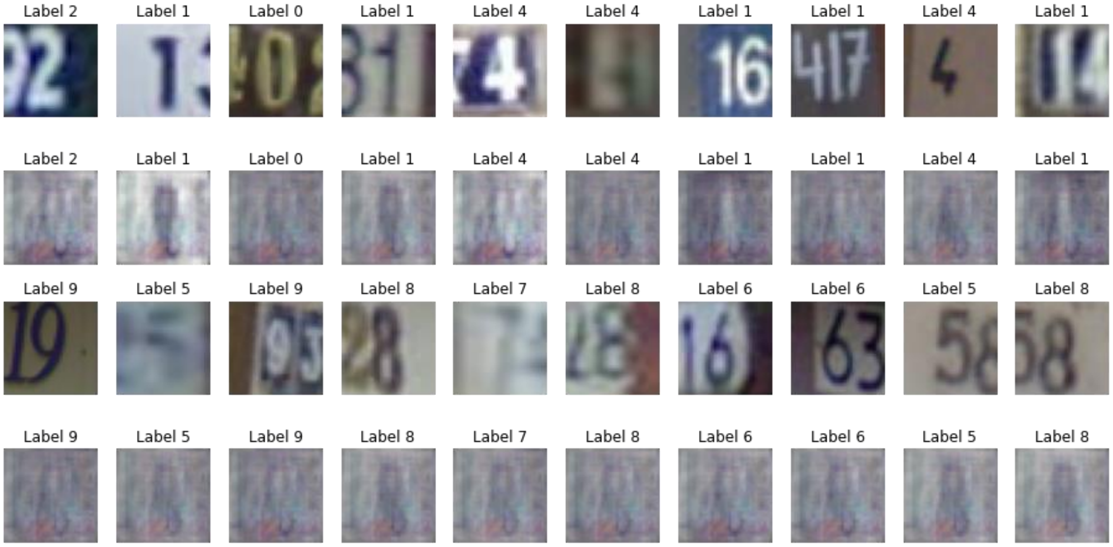

Figure S7. Same as **Figure S3**, but for the SVHN->MNIST case.

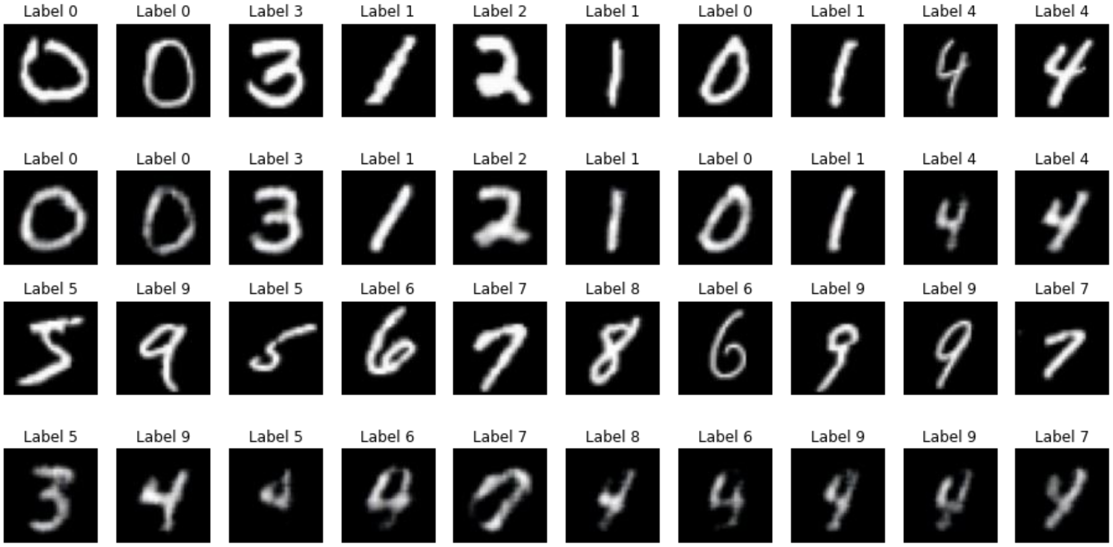

Figure S8. Same as **Figure S4**, but for the SVHN->MNIST case.

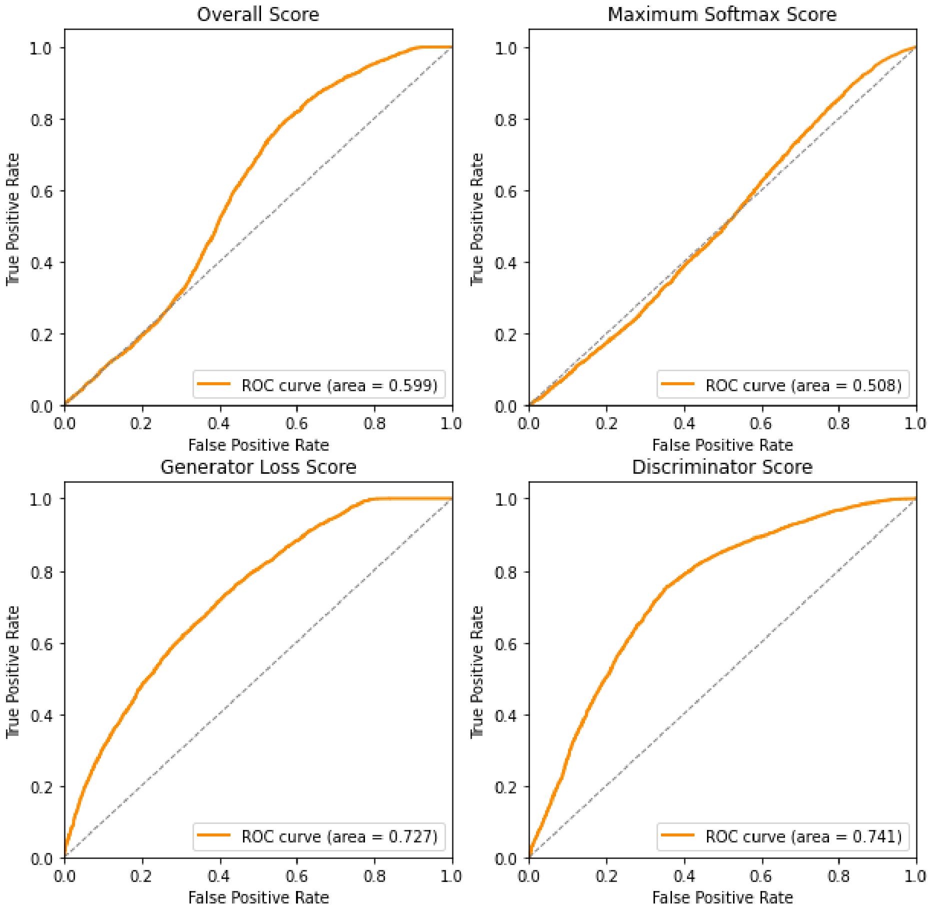

Figure S9. Same as **Figure S5**, but for the SVHN->MNIST case.

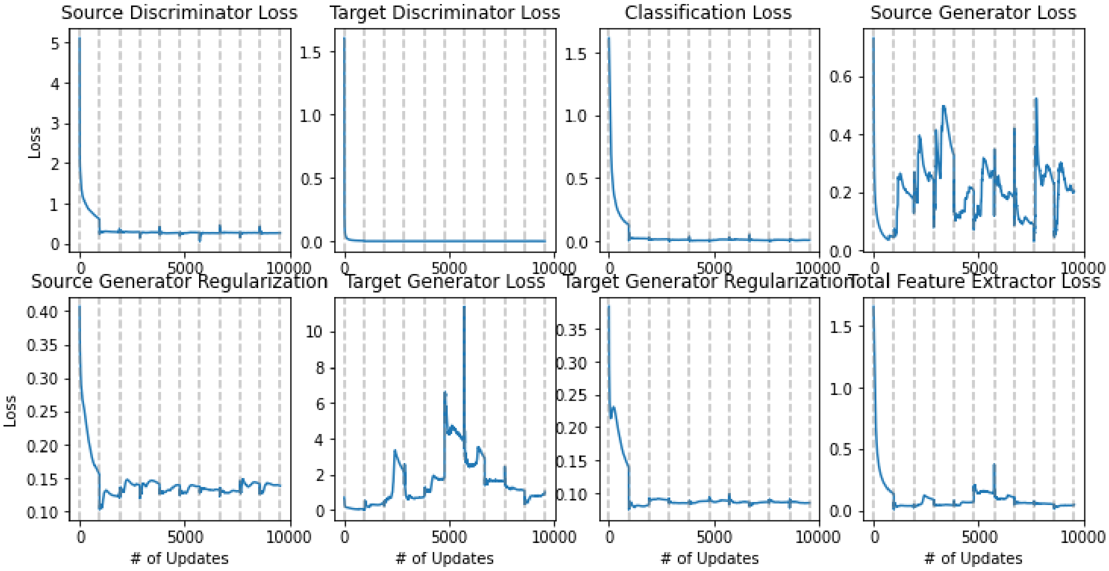

Figure S10. Same as **Figure S2**, but for the USPS->MNIST case.

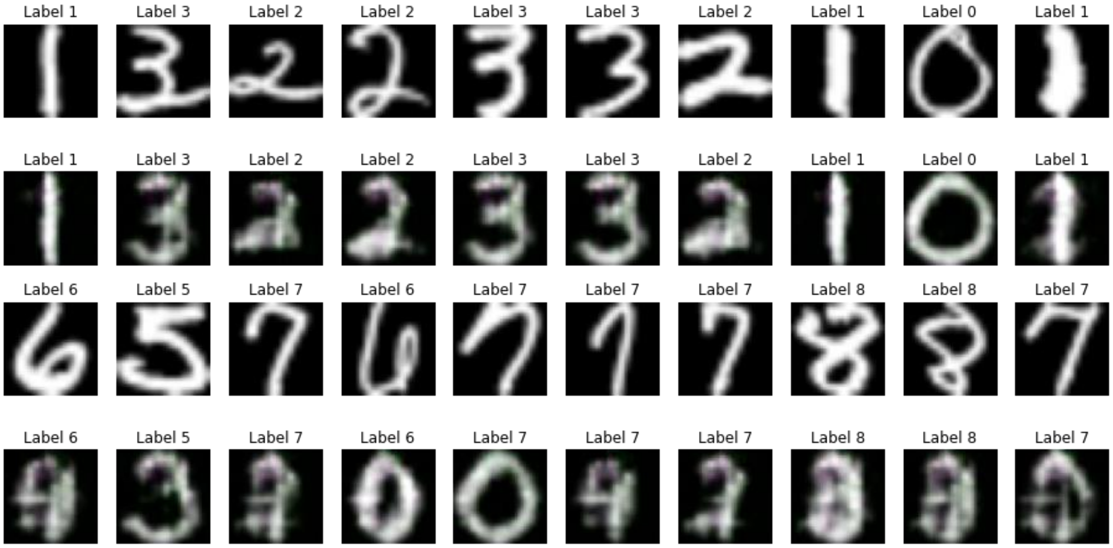

Figure S11. Same as **Figure S3**, but for the USPS->MNIST case.

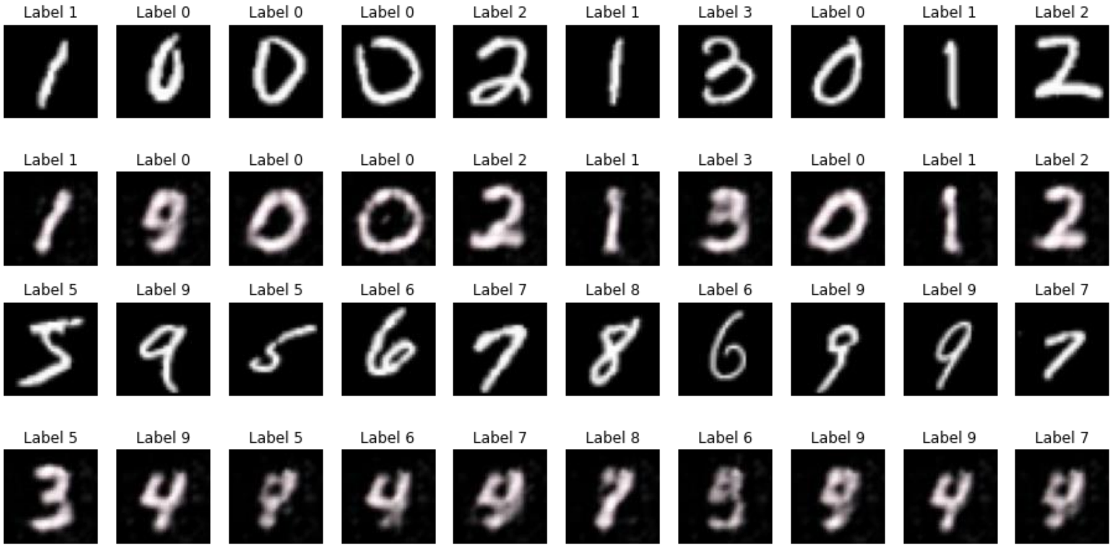

Figure S12. Same as **Figure S4**, but for the USPS->MNIST case.

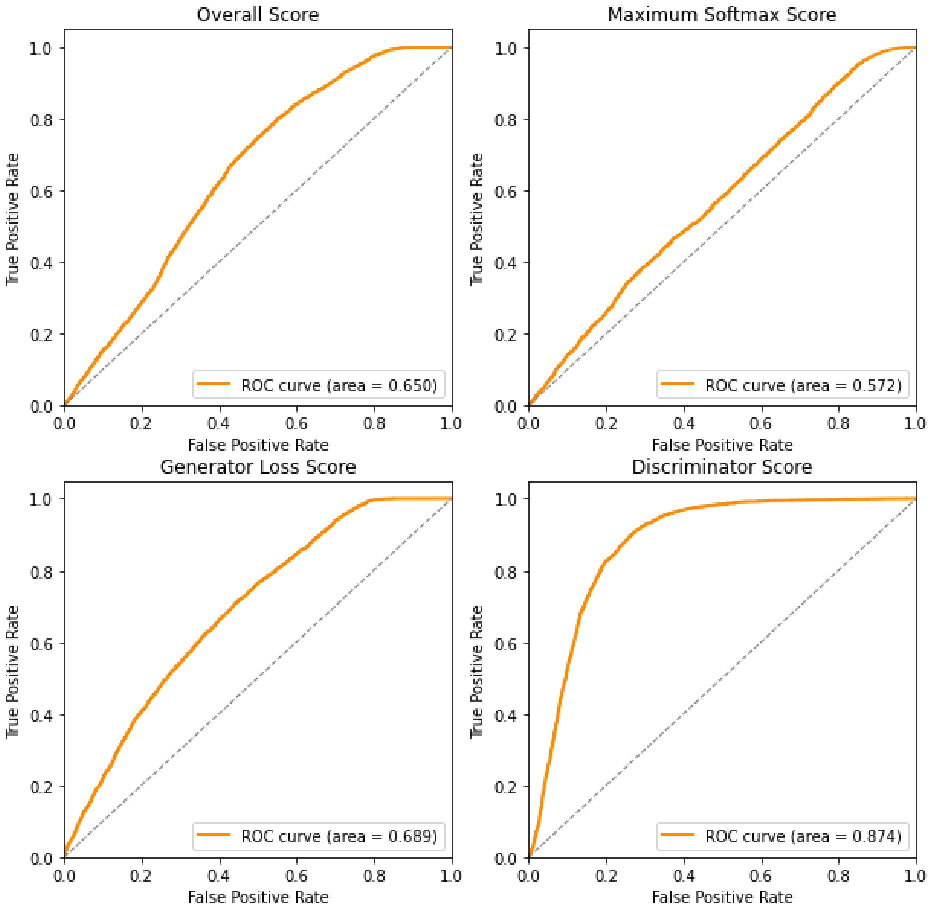

Figure S13. Same as **Figure S5**, but for the USPS->MNIST case.

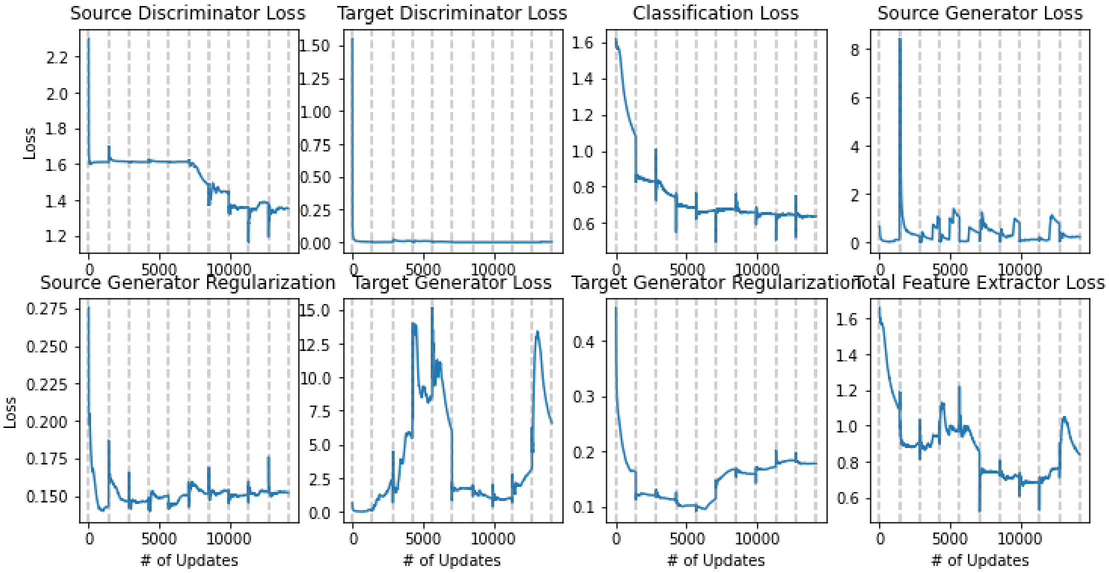

Figure S14. Same as **Figure S2**, but for the SVHN->USPS case.

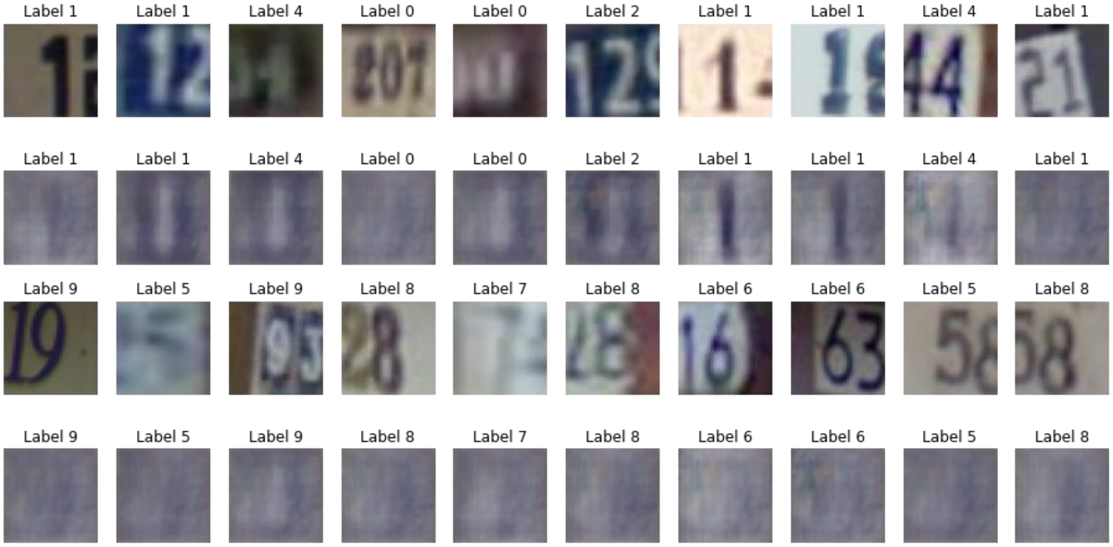
Label 1 Label 1 Label 4 Label 0 Label 0 Label 2 Label 1 Label 1 Label 4 Label 1
Label 1 Label 1 Label 4 Label 0 Label 0 Label 2 Label 1 Label 1 Label 4 Label 1
Label 9 Label 5 Label 9 Label 8 Label 7 Label 8 Label 6 Label 6 Label 5 Label 8
Label 9 Label 5 Label 9 Label 8 Label 7 Label 8 Label 6 Label 6 Label 5 Label 8

Figure S15. Same as **Figure S3**, but for the SVHN->USPS case.

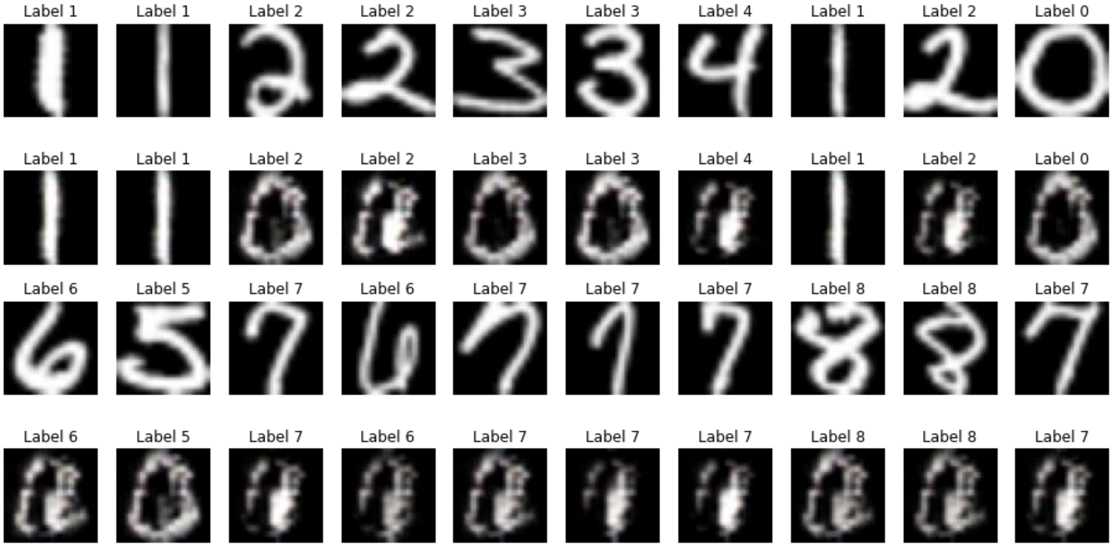
Label 1 Label 1 Label 2 Label 2 Label 3 Label 3 Label 4 Label 1 Label 2 Label 0
Label 1 Label 1 Label 2 Label 2 Label 3 Label 3 Label 4 Label 1 Label 2 Label 0
Label 6 Label 5 Label 7 Label 6 Label 7 Label 7 Label 7 Label 8 Label 8 Label 7
Label 6 Label 5 Label 7 Label 6 Label 7 Label 7 Label 7 Label 8 Label 8 Label 7

Figure S16. Same as **Figure S4**, but for the SVHN->USPS case.

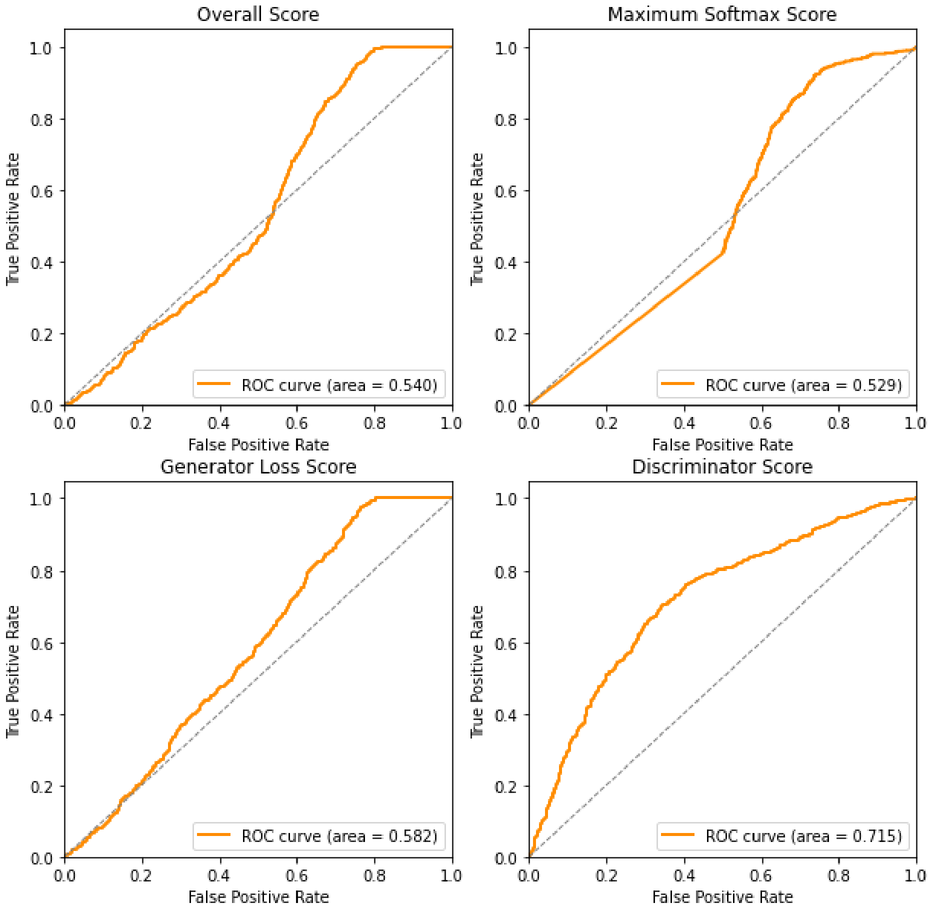

Figure S17. Same as **Figure S5**, but for the SVHN->USPS case.

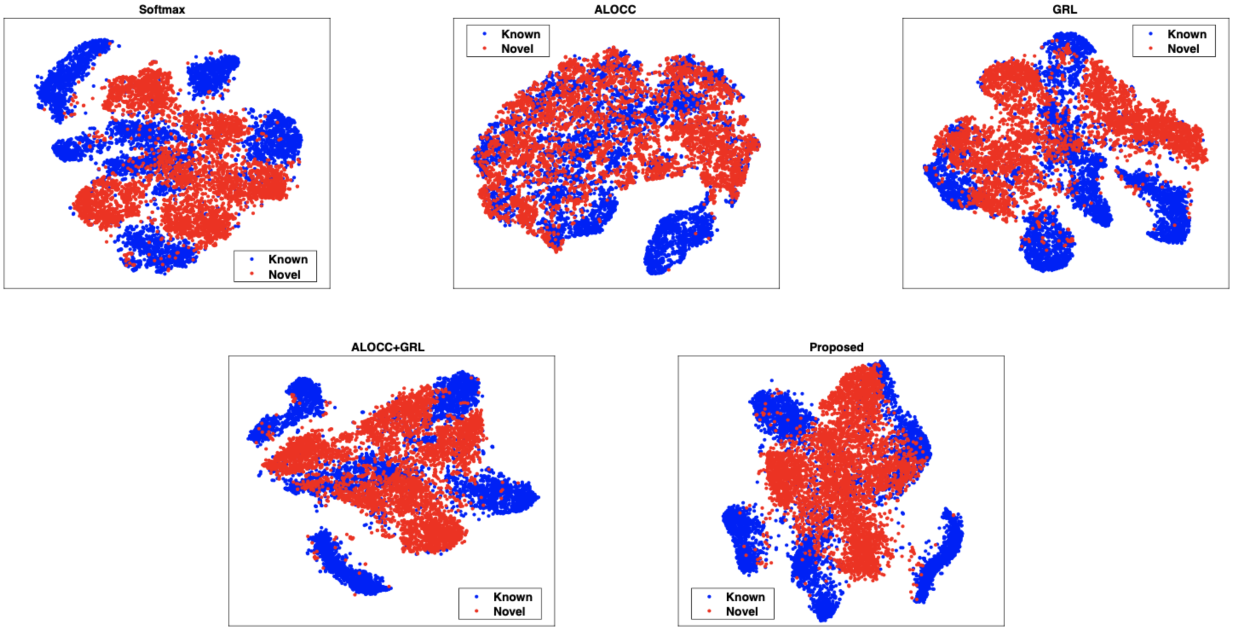

Figure S18. t-SNE plots of known and novel class features for target domain MNIST in the adaptation scenario SVHN->MNIST (reproduced from Oza et al., 2020).

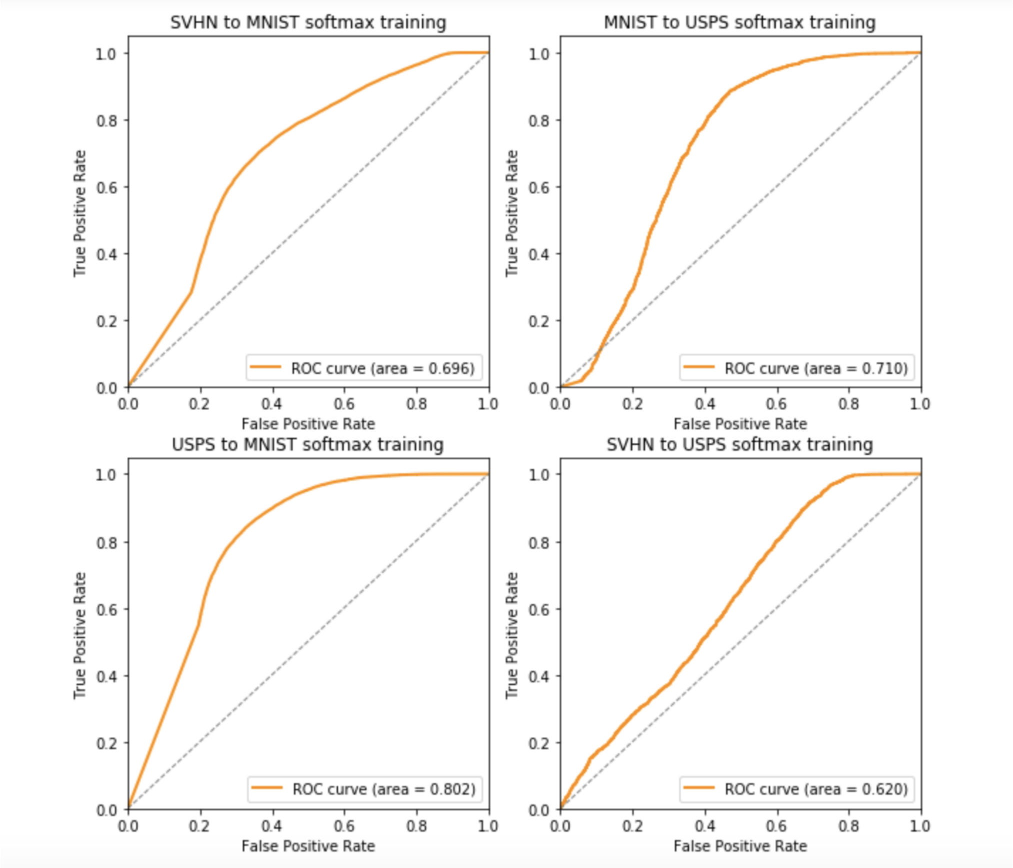

Figure S19. ROC curve based on GRL model trained on four cases. Take SVHN to MNIST as an example, in this case the model is trained using SVHN as source data and MNIST as target data. The ROC curve is generated using digit 0-4 from MNIST as original data, and digit 5-9 from MNINST as novel data.

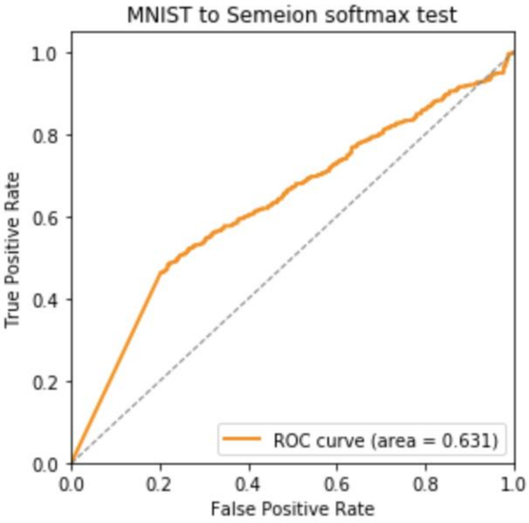

Figure S20. ROC curve based on GRL model using novel dataset Semeion.

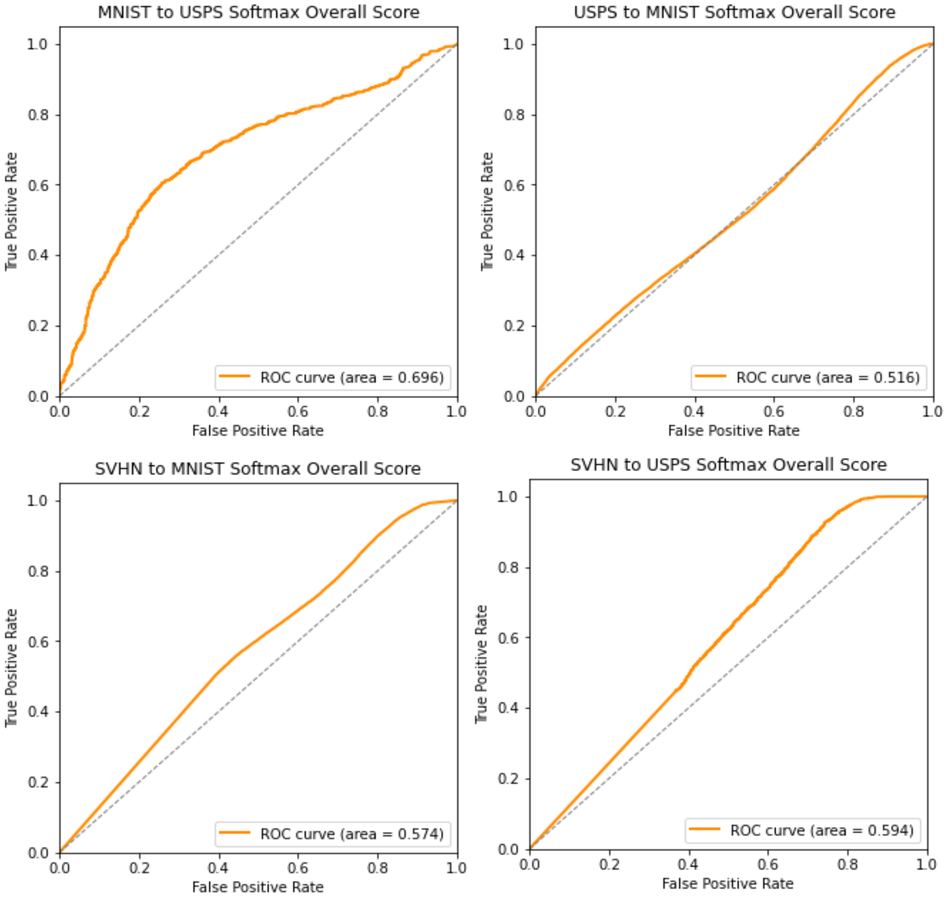

Figure S21. ROC curve based on Softmax model trained on four cases.

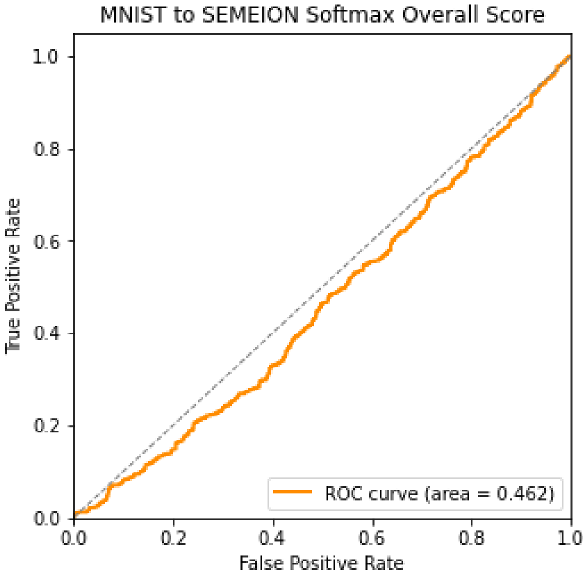

Figure S22. ROC curve based on Softmax model using novel dataset Semeion.

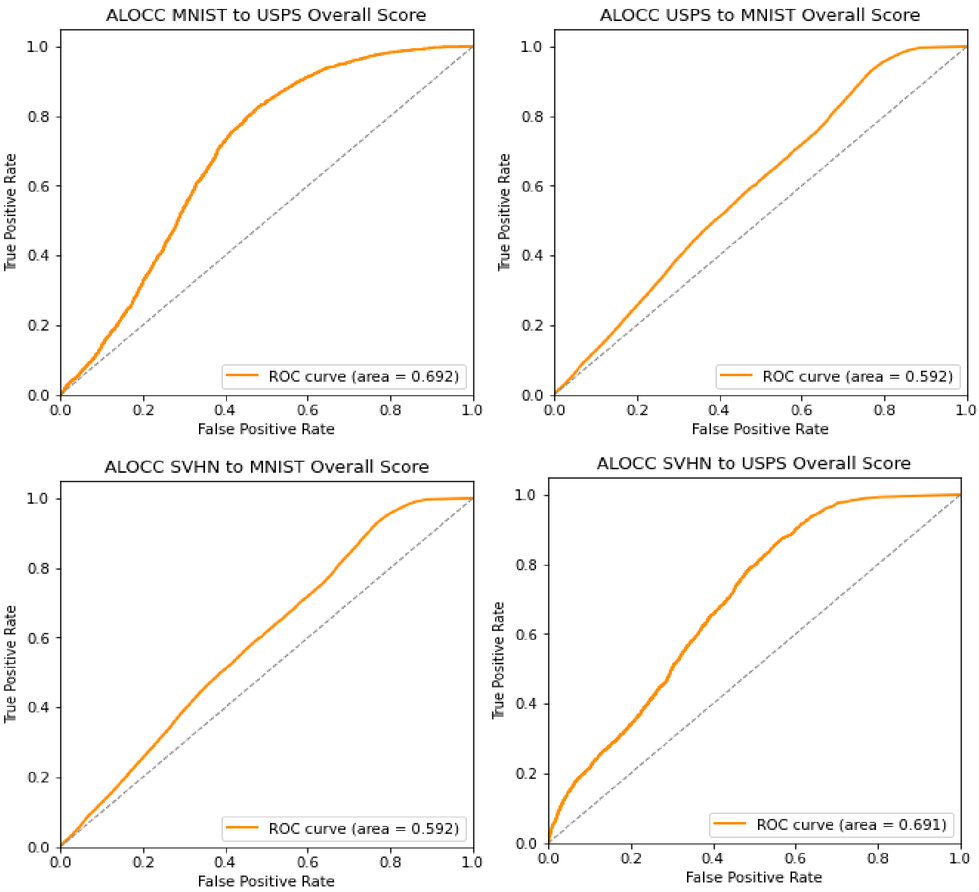

Figure S23. ROC curve based on ALOCC model trained on four cases.

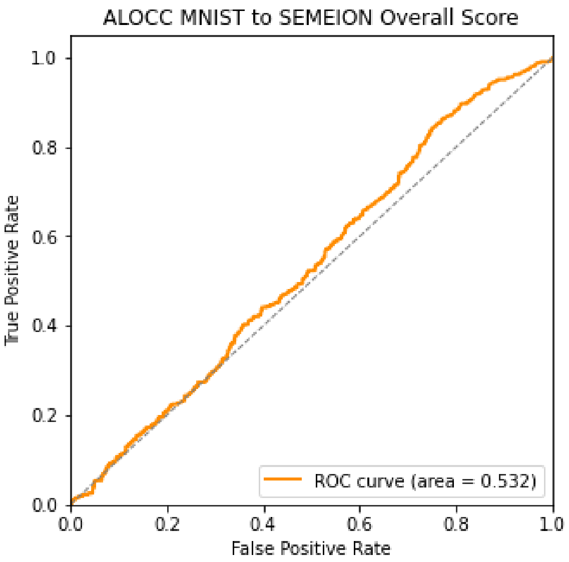

Figure S24. ROC curve based on ALOCC model using novel dataset Semeion.

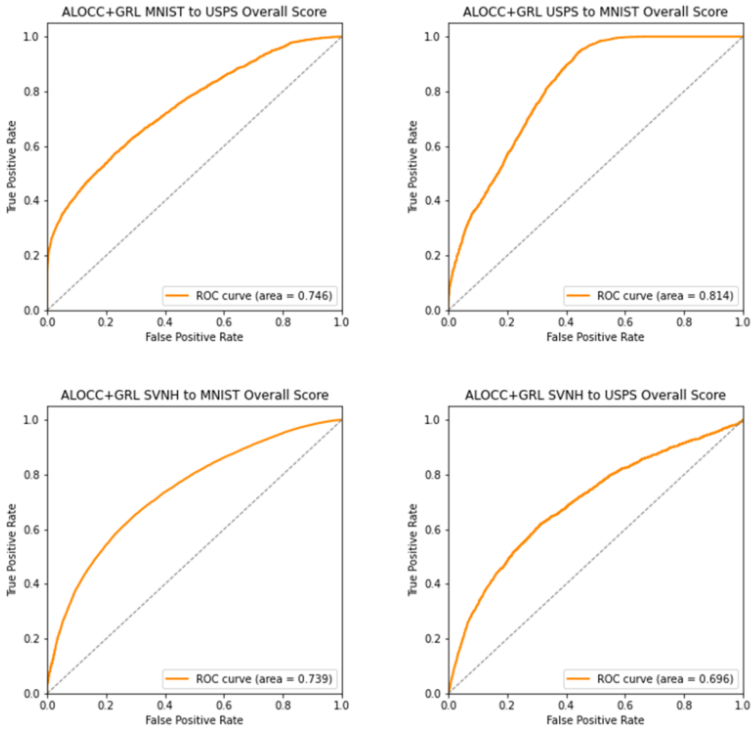

Figure S25. ROC curve based on ALOCC+GRL model trained on four cases.

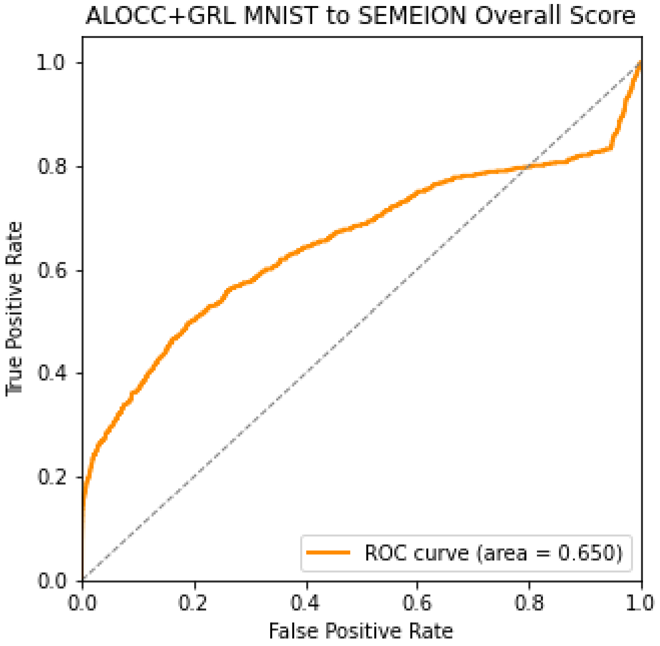

Figure S26. ROC curve based on ALOCC+GRL model using novel dataset Semeion.