DomainATM: Domain Adaptation Toolbox for Medical Data Analysis – *Manual*

Hao Guan, Mingxia Liu

In what follows, we introduce the workflow and usage of the Domain Adaptation Toolbox for Medical data analysis (DomainATM). The software and source code of DomainATM can be found online[1].

A. *Prerequisite & Installation & Launch*

Before using the toolbox, the "image processing toolbox" and "statistics & machine learning toolbox" of MATLAB need to be installed.

After downloading the toolbox, for Windows and Mac OS systems, click the file "DomainATM.mlappinstall" to install the toolbox; for Linux system, please click "Install App" in the "Apps" of MATLAB to install the DomainATM. Within seconds, the installed toolbox will appear in the "Apps" of MATLAB. Click the icon of "DomainATM", then the toolbox will be launched, as shown in Fig. S1. To make the toolbox run correctly, the current working directory of MATLAB needs to be set to the directory of the toolbox. For example, if the toolbox (including all the subfolders) is saved in "E:/DomainATM", then the current working directory of MATLAB should be set to "E:/DomainATM".

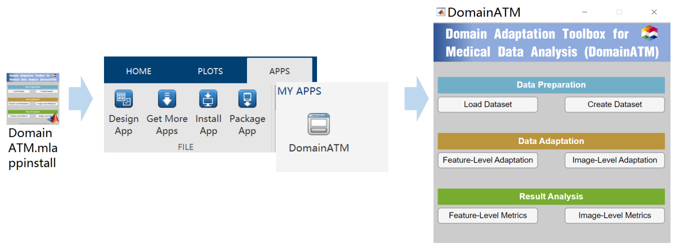


Fig. S1. Installation and launch of the DomainATM toolbox.

B. *File Organization*

There are several subfolders in the directory of the toolbox, including: "data", "algorithms_feat", "algorithms_img", "evaluation", and "tools", as shown in Fig. S2. Specifically, the "data" subfolder is used to store real-world or synthetic medical datasets. The "algorithms_feat" and "algorithms_img" subfolders are used to store feature-level and image-level domain adaptation algorithms, respectively. The "evaluation" subfolder is used to store results generated by different domain adaptation algorithms. The "tools" subfolder stores functions used by the toolbox.

[1]https://mingxia.web.unc.edu/domainatm/

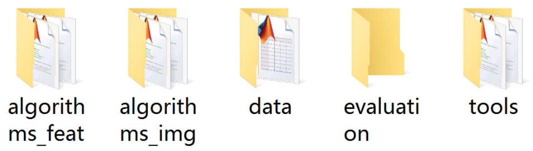
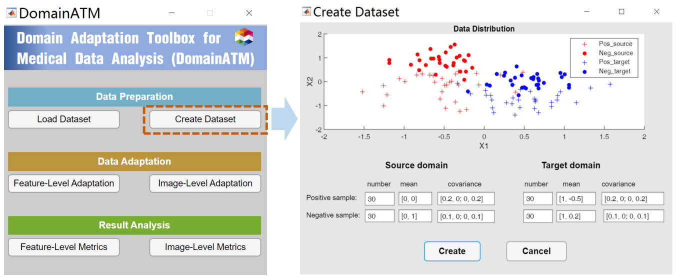


Fig. S2. Subfolders in the directory of the DomainATM toolbox.

C. *Creating Dataset*

Click the "Create Dataset" button, then the dialogue for generating synthetic dataset will be popped up as shown in Fig. S3. Users can set the number, mean value, and covariance of positive/negative samples in source domain and target domain, respectively. Otherwise, default settings will be used. Click the "Create" button, then the created dataset will be visualized and automatically saved in the "data" subfolder of the toolbox.



	Source domain			Target domain		
	number	mean	covariance	number	mean	covariance
Positive sample:	30	[0, 0]	[0.2, 0; 0, 0.2]	30	[1, -0.5]	[0.2, 0; 0, 0.2]
Negative sample:	30	[0, 1]	[0.1, 0; 0, 0.1]	30	[1, 0.2]	[0.1, 0; 0, 0.1]

Fig. S3. Creating synthetic dataset for evaluation of domain adaptation algorithms using the DomainATM toolbox.

D. *Loading Dataset*

To perform feature-level domain adaptation, the first step is to load the data into the toolbox. Both real-world medical datasets or synthetic datasets are stored in the "data" subfolder. Click the "Load Data" button and a file selection dialogue will be popped up. Select the data to be processed, then they will be loaded and automatically visualized in the user interface, as shown in Fig. S4.

To prepare the dataset for feature-level adaptation, it should be a *.mat* file that consists of the following three items. 1) **X**: source and target data, with each row representing a sample and each column representing a feature. 2) Domain label: a logical vector, with each element indicating that the corresponding sample comes from the source domain (with the value of "1") or the target domain (with the value of "0").

3) $\mathbf{Y}$: category labels for source samples and target samples. ("1" for positive samples, "2" for negative samples, "0" for unlabeled samples).

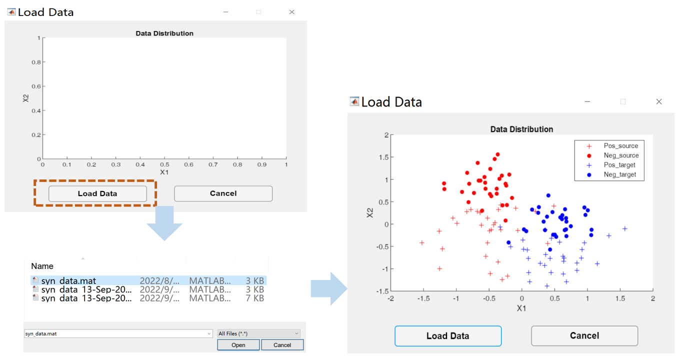


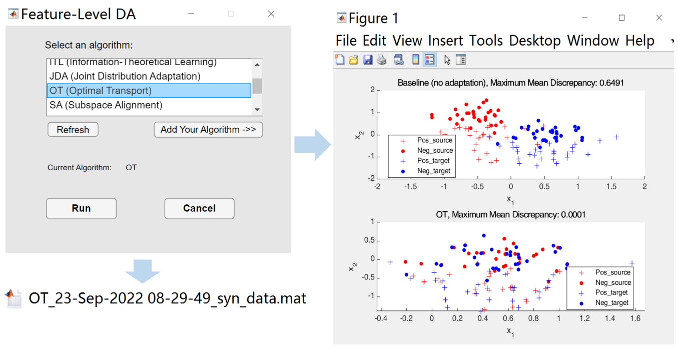
Fig. S4. Loading data into the DomainATM toolbox for further adaptation processing. After being loaded, the data distribution will be automatically visualized.

E. Performing Feature-Level Data Adaptation

After loading the medical dataset (each sample is represented by a feature vector), users can run and test different domain adaptation algorithms. Click the "Feature-Level Adaptation" button, then the corresponding dialogue will be popped up. In the dialogue, all the available adaptation algorithms are listed. User can select one and then run it for the data that have been loaded already (if no data are loaded or no algorithm is selected, there will be a warning and nothing will run). After running, the data (in feature space) before and after adaptation will be saved in the "evaluation" subfolder (the result is named as algorithm + timestamp + dataset). Meanwhile, the data before and after adaptation will be visualized in the toolbox as shown in Fig. S5. Currently, the DomainATM includes nine built-in feature-level domain adaptation methods, with details introduced as follows.

1) Subspace Alignment (SA) [1]: The key hyper-parameter is the dimension of the shared subspace (default value is 2).
2) Correlation Alignment (CORAL) [2]: No hyper-parameters.
3) Transfer Component Analysis (TCA) [3]: The key hyper-parameters are the kernel type (default setting is "rbf") and subspace dimension (default value is 2).
4) Optimal Transport (OT) [4]: The key hyper-parameter is the regularization coefficient (default value is 0.01).
5) Joint Distribution Adaptation (JDA) [5]: the key hyper-parameters include kernel type (default setting is "rbf") and subspace dimension (default value is 2).
6) Transfer Joint Matching (TJM) [6]: The key hyper-parameters include kernel type (default setting is "rbf") and subspace dimension (default value is 2).
7) Geodesic Flow Kernel (GFK) [7]: The key hyper-parameter is the subspace dimension (default setting is 2).
8) Scatter Component Analysis (SCA) [8]: The key parameter is the dimension of the transformed space (default setting is 2).
9) Information-Theoretical Learning (ITL) [9]: The key hyper-parameters include subspace dimension (default value is 2) and regularization parameter (default value is 10).

Note that the TCA and GFK algorithms were implemented based on source code provided by Ke Yan[2]. The JDA and TJM algorithms were implemented based on source code provided by the authors[3]. The users can tune the parameters of each algorithm in their scripts in the "algorithms_feat" subfolder. In each script, parameters can be set in a section with the tips of "set your parameters here".



Current Algorithm: OT

OT_23-Sep-2022 08-29-49_syn_data.mat

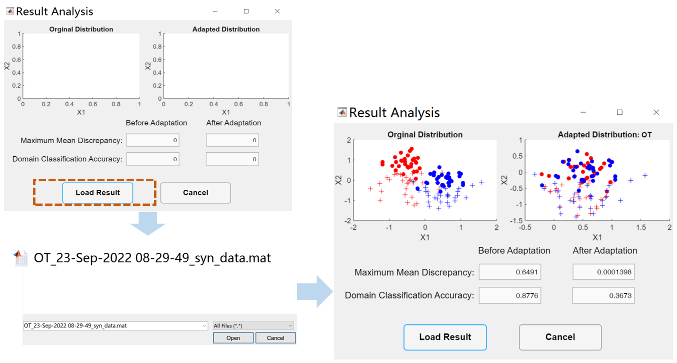
Baseline (no adaptation), Maximum Mean Discrepancy: 0.6491

OT, Maximum Mean Discrepancy: 0.0001

Fig. S5. Selecting and running a domain adaptation algorithm for the loaded medical dataset. After running, the result will be saved as .mat data file in "evaluation" subfolder.

F. Evaluating Feature-Level Adaptation Algorithms

After the processing of domain adaptation by a specific algorithm, the data before and after adaptation will be saved in the "evaluation" subfolder. Users can evaluate the adaptation performance by clicking the button of "Feature-Level Metrics". In the popped-up dialogue, click the "Load Result" button, and then select and load the feature-level adaptation result of a specific algorithm. Then, the evaluation metrics will be automatically computed and displayed in the user interface. Meanwhile, the distribution of original data and adapted data will be visualized as shown in Fig. S6.



OT_23-Sep-2022 08-29-49_syn_data.mat

	Before Adaptation	After Adaptation
Maximum Mean Discrepancy:	0.6491	0.0001398
Domain Classification Accuracy:	0.8776	0.3673

Fig. S6. Evaluation of feature-level domain adaptation algorithms in the DomainATM toolbox. After loading the result (in .mat file), performance evaluation metrics before and after adaptation will be calculated, and the data distribution will also be visualized.

[2]https://github.com/viggin
[3]https://github.com/jindongwang

G. Performing Image-Level Data Adaptation

To perform image-level domain adaptation, click the "Image-Level Adaptation" button, and the corresponding dialogue will be popped up. In the dialogue, select the source and target images, as well as the adaptation algorithm. After that, click the "Run" button to run the algorithm. With a few seconds, the adapted/harmonized source image will be displayed in the user interface. This process is illustrated in Fig. S7.

The DomainATM toolbox includes two built-in image-level domain adaptation methods. 1) Histogram Matching (HM) [10]: No hyper-parameters are needed for this method. 2) Spectrum Swapping-based Image-Level Harmonization (SSIMH) [11]: The key hyper-parameter is the threshold value for the range of low-frequency region for source and target image. The default value is set to 3 in the toolbox. Please note the image-level domain adaptation module can work well in the following two different problem settings.

1) One-to-One Image Harmonization: Given a source image and a target/reference image, one can select a specific algorithm to adapt the source image to the target image space.
2) Batch Image Harmonization: Given multiple source images and a target/reference image, we can adapt all the source images to the target image space via batch harmonization. In this setting, only the first source image is displayed in the GUI of DomainATM.

There are two manners for users to perform "Batch Image Harmonization". The first way is to click the "Load source image" button, and then select multiple MRIs in .nii format. After running, the adapted (harmonized) source images will be saved in the "evaluation" subfolder with the naming format of "source image name + adaptation method + target image name + time stamp". The second way is to first click the "Batch Image Harmonization" button, and then in the popped-up window, select the folder which stores the MRIs that to be harmonized. After running, in the "evaluation" folder, there will be a created subfolder which stores all the adapted (harmonized) source images. For example, suppose the user stores all the source MRIs in a subfolder "Batch Source Images" of the "data" folder, and the target image is Target_Image.nii. After running, all the adapted (harmonized) images can be found in a subfolder "Batch Source Image_h" of the "evaluation" folder.

Users may want to perform site-to-site image adaptation. A practical solution is to first select a "reference image" that can represent the properties of all target images. Then, one can use the "batch image harmonization" module in DomainATM (as described above) to harmonize all the source images to the reference target image. Another solution is that one can first select a predefined template such as Anatomical Automatic Labeling (AAL) [12] as the "reference image", and then adapt all source and target images to the common template space.

H. Evaluating Image-Level Adaptation Algorithms

When the image-level adaptation is finished, the adapted source images will be automatically saved in the "evaluation" subfolder. To assess the adaptation performance, click the "Image-Level Metrics". In the popped-up dialogue, select the source image, target image and adapted/harmonized image, respectively. Please note that if the users have already loaded the source and target images via the "Image-Level Adaptation" module, these images will be automatically displayed in the dialogue of "Image-Level Metrics", and users do not need to load them again. Then click the "Run" button to run the evaluation. The metrics before and after adaptation will be calculated and shown in the user interface. This process is illustrated in Fig. S8.

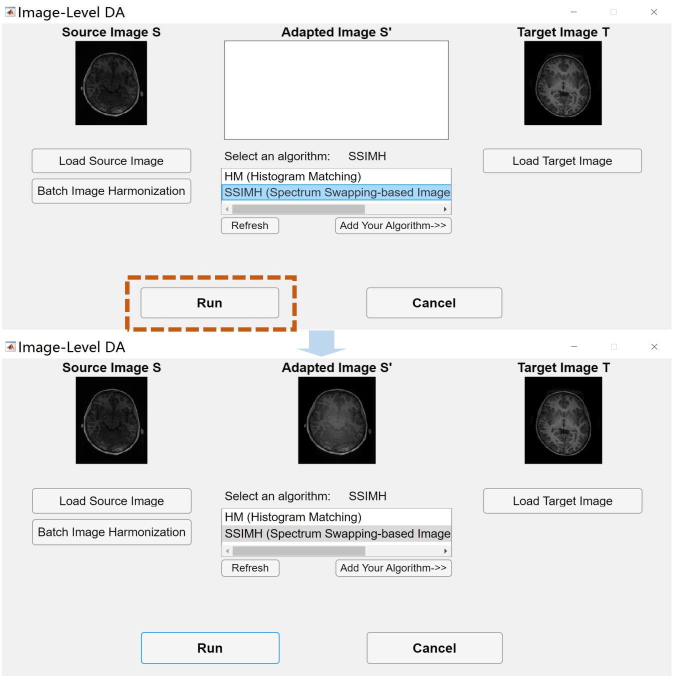


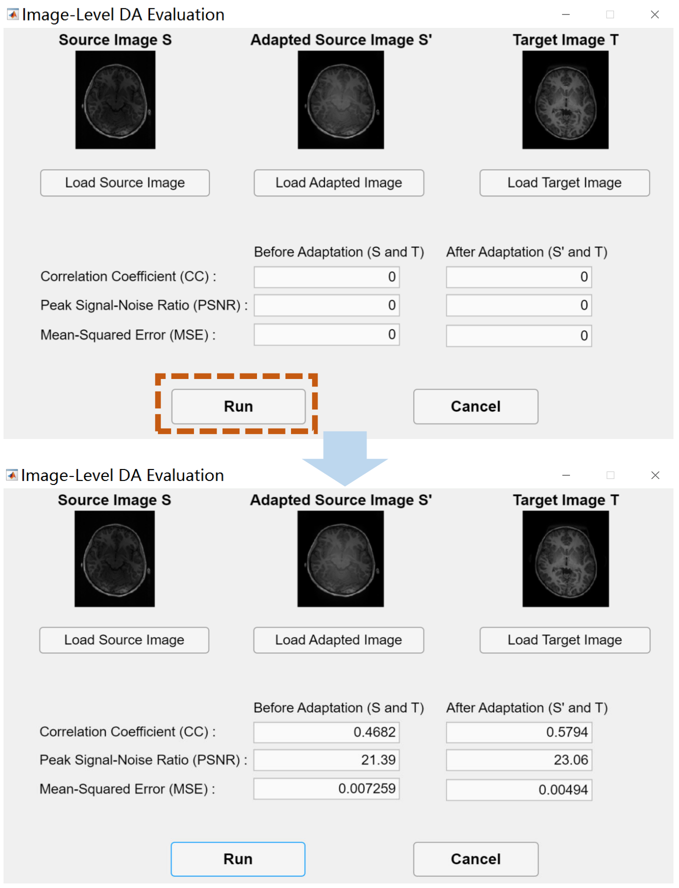
Fig. S7. Image-level domain adaptation by the DomainATM software.

I. Plugging Self-Defined Data Adaptation Algorithms

In real-world practice of medical data analysis, the users may want to develop their own domain adaptation algorithms. To cater to this demand, the DomainATM toolbox allows the users to add their own methods into the toolbox. In both of the dialogues for feature-level adaptation and image-level adaptation, the users can click the "Add Your Algorithm" button, which can guide the user to the subfolder to write their own scripts. All the algorithms should follow consistent input/output formats. After that, click "Refresh" button, the self-defined algorithm will be displayed in the toolbox. The users can select, use, and evaluate their customized algorithms just like all the other built-in algorithms as described above. Fig. S9 illustrates the operations.

For feature-level data adaptation, a self-defined algorithm should include the following four sequential parameters:

- X: All samples in the source and target domains. It is an $n \times m$ matrix, with n and m denoting the number of samples and the feature dimension, respectively.
- domain_label: It is n-dimensional logical vector indicating whether the i-th sample is from source domain or not.



	Before Adaptation (S and T)	After Adaptation (S' and T)
Correlation Coefficient (CC) :	0	0
Peak Signal-Noise Ratio (PSNR) :	0	0
Mean-Squared Error (MSE) :	0	0

	Before Adaptation (S and T)	After Adaptation (S' and T)
Correlation Coefficient (CC) :	0.4682	0.5794
Peak Signal-Noise Ratio (PSNR) :	21.39	23.06
Mean-Squared Error (MSE) :	0.007259	0.00494

Fig. S8. Evaluation of image-level domain adaptation by the DomainATM toolbox.

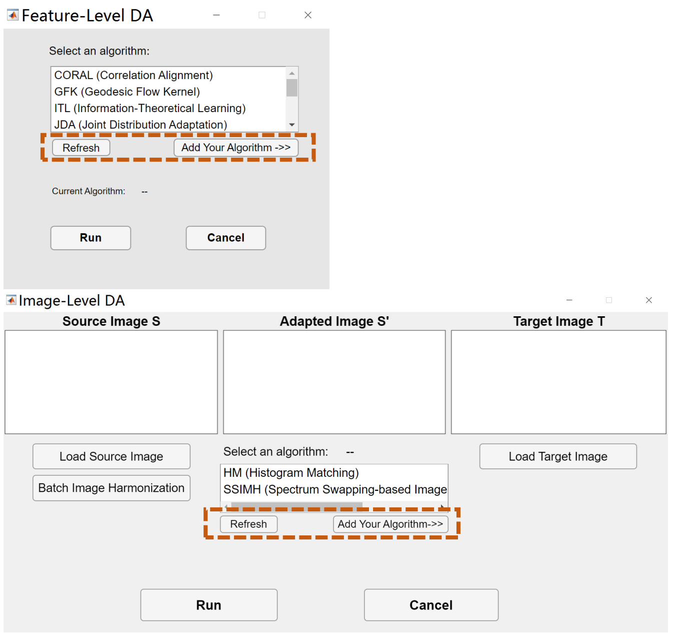


Fig. S9. Plugging self-defined domain adaptation algorithms into the DomainATM toolbox.

That is, domain_label$(i) = 1$ denotes that it is from the source domain, while domain_label$(i) = 0$ denotes that it is from the target domain.

- Y: Category labels for all the samples. For some unsupervised algorithms, this parameter may not be used. If not used, Y can be set to a zero vector.
- param: A struct containing all hyper-parameters.

So we can define a feature adaptation algorithm *FeatureDA* via $\mathrm{X}' = \mathit{FeatureDA}(\mathrm{X}, \text{domain\_label}, \mathrm{Y}, \text{param})$, where X' is the samples after adaptation (with the same type as X). If no label is needed, the value of Y can be set to zero.

For image-level data adaptation, a self-defined algorithm should include the following three parameters:

- source: A source image in .nii format.
- target: A target image in .nii format.
- param: A struct containing all hyper-parameters.

One can define an image adaptation algorithm *ImageDA* via the following: $\mathrm{S}' = \mathit{ImageDA}(\text{source}, \text{target}, \text{param})$, where S' is the source image after adaptation.

References

[1] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, "Unsupervised visual domain adaptation using subspace alignment," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2960–2967.

[2] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[3] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210, 2010.

[4] H. Guan, L. Wang, and M. Liu, "Multi-source domain adaptation via optimal transport for brain dementia identification," in *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2021, pp. 1514–1517.

[5] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2200–2207.

[6] ——, "Transfer joint matching for unsupervised domain adaptation," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1410–1417.

[7] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2066–2073.

[8] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang, "Scatter component analysis: A unified framework for domain adaptation and domain generalization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 7, pp. 1414–1430, 2016.

[9] Y. Shi and F. Sha, "Information-theoretical learning of discriminative clusters for unsupervised domain adaptation," in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, 2012, pp. 1275–1282.

[10] R. T. Shinohara *et al.*, "Statistical normalization techniques for magnetic resonance imaging," *NeuroImage: Clinical*, vol. 6, pp. 9–19, 2014.

[11] H. Guan, S. Liu, W. Lin, P.-T. Yap, and M. Liu, "Fast image-level MRI harmonization via spectrum analysis," in *International Workshop on Machine Learning in Medical Imaging*. Springer, 2022.

[12] N. Tzourio-Mazoyer *et al.*, "Automated anatomical labeling of activations in SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain," *Neuroimage*, vol. 15, no. 1, pp. 273–289, 2002.