Checklist

1. **Claims:** Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
2. **Code Of Ethics:** Have you read the NeurIPS Code of Ethics and ensured that your research conforms to it? [Yes]
3. **Broader Impacts:** If appropriate for the scope and focus of your paper, did you discuss potential negative societal impacts of your work? [Yes]
4. **Limitations:** Did you describe the limitations of your work? [Yes]
5. **Theory:** If you are including theoretical results, did you state the full set of assumptions of all theoretical results, and did you include complete proofs of all theoretical results? [N/A]
6. **Experiments:** If you ran experiments, did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
7. **Training Details:** If you ran experiments, did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
8. **Error Bars:** If you ran experiments, did you report error bars (e.g., with respect to the random seed after running experiments multiple times), or other information about the statistical significance of your experiments? [N/A]
9. **Compute:** Did you include the amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
10. **Reproducibility:** If the contribution is a dataset or model, what steps did you take to make your results reproducible or verifiable? [Yes]
11. **Safeguards:** Do you have safeguards in place for responsible release of models with a high risk for misuse (e.g., pretrained language models)? [N/A]
12. **Licenses:** If you are using existing assets (e.g., code, data, models), did you cite the creators and respect the license and terms of use? [Yes]
13. **Assets:** If you are releasing new assets, did you document them and provide these details alongside the assets? [Yes]
14. **Human Subjects:** If you used crowdsourcing or conducted research with human subjects, did you include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)? [N/A]
15. **IRB Approvals:** Did you describe any potential participant risks and obtain Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your institution), if applicable? [N/A]

Appendix

In the supplementary material, we provide additional visualization results, limitations, potential negative societal impacts and compute requirements of the MemSPM. In the pursuit of reproducible research, we will make the demo and network weights of our code available to the public.

This supplementary is organized as follows:

- Section A: Notations
- Section B: Limitation
- Section C: Potential societal impact
- Section D: Implementation details
  - Baseline details
  - Compute requirements
- Section E: Visualization Results

A Notations

Table 1:

	Symbol	Description
Model	$f_{encode}^{fixed}(\cdot)$	Fixed image encoder
	$f_{decode}^{unfixed}(\cdot)$	Unfixed reconstruction decoder
	f_{class}^{UniDA}	UniDA classifier
	M	Memory unit
	W	Weight vector
Space	$\mathcal{D}^s$	Labeled source dataset
	$\mathcal{D}^t$	Unlabeled target dataset
	C	Common label set
	C_s	Source label set
	C_t	Target label set
	$\hat{C}_s$	Source private label set
	$\hat{C}_t$	Target private label set
Samples	X	Input image
	$\hat{X}$	Reconstruction of image
	Z	Input-oriented embedding
	$\hat{Z}$	Task-oriented embedding
	L	Label of the image
	$\hat{L}$	Prediction of image
Measures	$w_{i,j}$	Attention weight measurement between Z and sub-prototype
	$d(\cdot,\cdot)$	Cosine similarity measurement
	$\hat{w_{i,j}}$	Adaptive threshold operation on $w_{i,j}$
Hyperparameters	N	Number of memory items
	S	Number of sub-prototypes partitioned in each memory item
	D	Dimension of each sub-prototype
	K	Top-K relevant sub-prototypes of Z

B Limitation

Training memory unit of MemSPM is challenging when adopting the commonly used ResNet-50 as the backbone. This is due to the memory unit's composition of massive randomly initialized tensors.

During the early stage of training, there is a lack of discriminability in the input-oriented embedding, which leads to addressing only a few sub-prototypes. This decoupling of the memory unit from the input data necessitates using a better pre-trained model (ViT-B/16 pre-trained on CLIP) and fixing the encoder to reduce computation requirements. Additionally, the number of sub-prototypes in one memory item might need to be adjusted for the diversity of the category.

C Potential Societal Impact

Our finding of the intra-class concept shift may influence the future work on domain adaption or other tasks. They can optimize the construction and refinement of the feature space by considering the intra-class distinction. The MemSPM also provides a method can be used to demonstrate the interpretability of model for further deployment. However, the utilization of MemSPM method for illegal purposes may be facilitated by their increased availability to organizations or individuals. And the MemSPM method may be susceptible to adversarial attacks as all contemporary deep learning systems. Although we demonstrate increased performance and interpretability compared to the state-of-the-art methods, negative transfer is still possible in extreme cases of domain-shift or category-shift. Therefore, our technique should not be employed in critical applications or to make significant decisions without human supervision.

D Implementation details

DCC. We use ViT-B/16 [1] as the backbone. The classifier is made up of two FC layers. We use Nesterov momentum SGD to optimize the model, which has a momentum of 0.9 and a weight decay of 5e-4. The learning rate decreases by a factor of $(1+\alpha\frac{i}{N})^{-\beta}$, where i and N represent current and global iteration, respectively, and we set $\alpha = 10$ and $\beta = 0.75$. We use a batch size of 36 and the initial learning rate is set as 1e-4 for Office-31, and 1e-3 for Office-Home and DomainNet. We use the settings detailed in [2]. PyTorch [3] is used for implementation.

GLC. We use ViT-B/16 [1] as the backbone. The SGD optimizer with a momentum of 0.9 is used during the target model adaptation phase of GLC [6]. The initial learning rate is set to 1e-3 for Office-Home and 1e-4 for both VisDA and DomainNet. The hyperparameter ρ is fixed at 0.75 and $|L|$ at 4 across all datasets, while η is set to 0.3 for VisDA and 1.5 for Office-Home and DomainNet, which corresponds to the settings detailed in [6]. PyTorch [3] is used for implementation.

Existing code used.

- DCC [2]: `https://github.com/Solacex/Domain-Consensus-Clustering`
- GLC [6]: `https://github.com/ispc-lab/GLC`
- PyTorch [3]: `https://pytorch.org/`

Existing datasets used.

- Office-31 [7]: `https://www.cc.gatech.edu/âĹijjudy/domainadapt`
- Office-Home [8]: `https://www.hemanthdv.org/officeHomeDataset.html`
- DomainNet [4]: `http://ai.bu.edu/M3SDA`
- VisDA [5]: `http://ai.bu.edu/visda-2017/`

Compute Requirements. For our experiments, we used a local desktop machine with an Intel Core i5-12490f, a single Nvidia RTX-3090 GPU and 32GB of RAM. When we adapt the batch-size used in DCC [2], our MemSPM only occupies 4GB of GPU memory during training in result of fixing the encoder.

E Visualization

We provid more results of visualization in Figure 1 and Figure 2 to reveal sub-prototypes stored in the memory unit, which demonstrate that our MemSPM approach can learn the intra-class concept shift.

Figure 1: The reconstruction visualization shows what have been learned in the memory, which demonstrates the intra-class diversity have been learned by MemSPM.

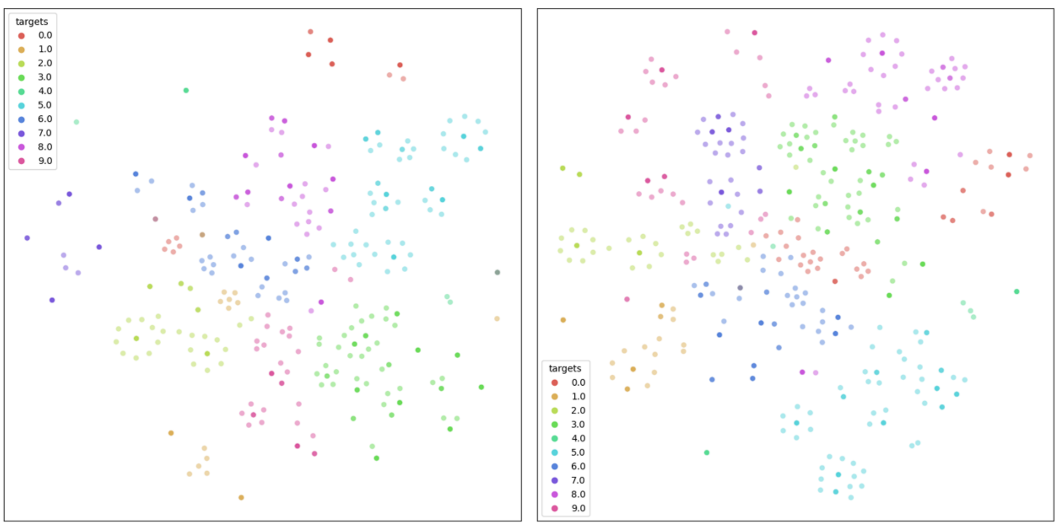

Figure 2: The tSNE visualization shows the distribution of the retrieved sub-prototypes and demonstrates that the sub-classes have been learned by MemSPM.

References

[1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[2] Guangrui Li, Guoliang Kang, Yi Zhu, Yunchao Wei, and Yi Yang. Domain consensus clustering for universal domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9757–9766, 2021.

[3] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[4] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[5] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017.

[6] Sanqing Qu, Tianpei Zou, Florian Röhrbein, Cewu Lu, Guang Chen, Dacheng Tao, and Changjun Jiang. Upcycling models under domain and category shift. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[7] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *Proceedings of the European Conference on Computer Vision*, pages 213–226, 2010.

[8] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.