APPENDIX FOR HYBRID REPRESENTATION LEARNING VIA EPISTEMIC GRAPH

Anonymous authors
Paper under double-blind review

1 VISUALIZATION

In this section, we present three graphs of the Office-31 and Office-Home datasets: the local visual graph, enhanced local graph, and global graph. Figure 1 displays these graphs, with the upper line representing the results of the Office-Home dataset and the bottom line representing the Office-31 dataset. The first column displays the local visual graph, the second column displays the enhanced local graph, and the right column displays the global graph. In these graphs, thicker edges indicate stronger relations, while node size is fixed. To avoid clutter caused by too many edges, we show the top-150 edges in the Office-Home dataset and top-70 edges in the Office-31 dataset. Additionally, we have highlighted two nodes in each graph to demonstrate the differences between the three graphs.

In the Office-Home dataset, the *Scissors* node in the global graph is positioned near two types of concepts: tools and stationery. The typical tools include *Knives*, *Hammer*, and *Screwdriver*, while the stationery items include *Eraser*, *Pencil*, and *Pen*. In the local visual graph, *Scissors* is related to the typical tools category due to their similar metallic appearance. In the enhanced local graph, *Scissors* has features from both the semantic and visual graphs. Specifically, it has some thick edges with *Knives* and *Screwdriver*, as well as a thin edge with *Pen*. Another interesting node is *Mop*, which is related to *Toothbrush*, *Bucket*, *Curtains*, and other objects in the visual graph. However, in the semantic graph, it is only related to *Toothbrush*, *Bucket*, and *Sink*. As a result, in the enhanced visual graph, *Mop* is positioned closer to the semantic graph with three edges connecting it to *Toothbrush*, *Bucket*, and *Bottle*.

In the Office-31 dataset, the *mouse* node has no edges connecting it to other nodes in the local visual graph due to its different appearance. However, it has many neighbors, including *keyboard*, *laptop computer*, and others in global graph. In the enhanced local graph, *mouse* begins to maintain some edges with other nodes, which confirms the guidance of the global graph. For *ruler*, it holds an edge with *pen* in the local visual graph, while it does not have an edge in the global graph. In the enhanced local graph, it still has an edge with *pen*, which demonstrates that visual information is preserved in the enhanced local graph.

2 ABLATION STUDIES

We conduct ablation studies in the context of universal domain adaptation using the Office-31 dataset, as presented in Table 1. These experiments involve varying values of σ in Eq. 6 and Eq. 7 to investigate the sparsity of the adjacency matrix.

As we increased σ to 0.1, we observe a marginal decline in the overall results, approximately around 1%. Further increments in the value of σ seem to introduce confusion in the model's ability to learn precise relationships.

Additionally, we delve into the impact of various loss functions. For adjacency matrix loss denoted as $\mathcal{L}_a$ in Eq. 11, we substitute it with $\mathcal{L}_1$ and $\mathcal{L}_2$ distance losses, labeling them as UAN + EGLayer w/ $\mathcal{L}_1$ and UAN + EGLayer w/ $\mathcal{L}_2$, respectively. The corresponding formulations are expressed as follows:

$$\mathcal{L}_1(\boldsymbol{A}, \boldsymbol{A}^{s\prime}) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(a_{ij} - a_{ij}^{s\prime} \right), \tag{1}$$

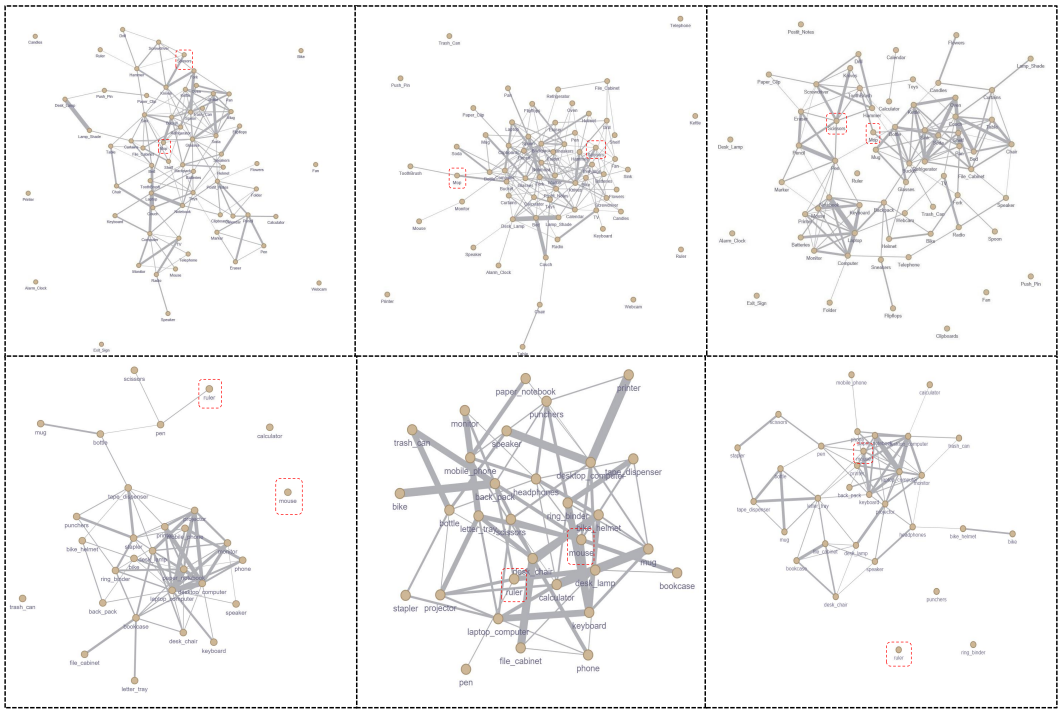

Figure 1: Visualization of three graphs.

$$\mathcal{L}_2(\boldsymbol{A}, \boldsymbol{A}^{s\prime}) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(a_{ij} - a_{ij}^{s\prime}\right)^2. \tag{2}$$

We have also employed L_1 regularization to replace L_2 regularization in Eq. 12, and we refer to this as UAN + EGLayer w/ $\mathcal{L}_{reg1}$:

$$\mathcal{L}_{reg1}(\boldsymbol{S}) = \| \left\langle \boldsymbol{S}, \boldsymbol{S}^{\mathrm{T}} \right\rangle \|_1 = \|\boldsymbol{C}\|_1 = \sum_{i=1}^{n} \sum_{j=1}^{n} |c_{ij}|. \tag{3}$$

Moreover, we introduce three versions for ablation studies: UAN + EGLayer w/o $\mathcal{L}_a$, UAN + EGLayer w/o $\mathcal{L}_{reg}$, and UAN + EGLayer w/o $\mathcal{L}_g$. In the context of these experiments, UAN + EGLayer w/o $\mathcal{L}_g$ indicates that the experiment lacks both $\mathcal{L}_a$ and $\mathcal{L}_{reg}$.

As indicated in Table 1, UAN + EGLayer w/ $\mathcal{L}_1$ demonstrates a performance improvement when compared to UAN + EGLayer w/o $\mathcal{L}_a$. Conversely, UAN + EGLayer w/ $\mathcal{L}_2$ shows a performance degradation. Both of these settings underperform in comparison to UAN + EGLayer. Notably, the L_1 regularization version exhibits a significant performance decrease. This phenomenon could be attributed to regularization causing node embeddings to become too distinct, thereby hindering the model's ability to learn relationships.

Comparing UAN + EGLayer w/o $\mathcal{L}_a$, UAN + EGLayer w/o $\mathcal{L}_{reg}$, and UAN + EGLayer w/o $\mathcal{L}_g$, it's evident that both $\mathcal{L}_{reg}$ and $\mathcal{L}_a$ play distinct roles in boosting performance. When these two losses are not utilized, there is an average performance reduction of 1.06%. Moreover, the versions with only $\mathcal{L}_{reg}$ and $\mathcal{L}_a$ both outperform UAN + EGLayer w/o $\mathcal{L}_g$. Finally, UAN + EGLayer demonstrates a clear advantage when compared with UAN + EGLayer w/o $\mathcal{L}_a$ and UAN + EGLayer w/o $\mathcal{L}_{reg}$.

Furthermore, we conduct experiments involving 2 GCN layers, with the first layer adapting the features to the same dimension, and the second layer aligning the features to the global graph dimension. This configuration is referred to as UAN + EGLayer + 2 layer GCN. Another setup involve

inserting the EGLayer after the feature extractor and before the standard linear classifier as an intermediary layer, without dimensionality transfer. Given that our proposed EGLayer can theoretically be inserted after any layer, we label this experiment as UAN + middle EGLayer.

When compared to the final version, both of these settings exhibit an average performance decrease of 3.63% and 0.26%, underscoring the simplicity and effectiveness of the final EGLayer version. It's worth noting that UAN + middle EGLayer demonstrates a 0.61% and 0.94% improvements in A→W and W→D domain adaptation, hinting at potential for further exploration when inserting the EGLayer after different layers. Consequently, we remain committed to exploring relevant solutions in this regard.

Table 1: Ablation study for universal domain adaptation experiments on Office-31 dataset

Methods	A→W	D→W	W→D	A→D	D→A	W→A	Average
UAN + EGLayer + σ 0.1	83.47	93.47	93.67	**86.11**	86.09	85.71	88.09
UAN + EGLayer + σ 0.5	15.35	28.74	24.48	20.85	21.58	11.30	20.38
UAN + EGLayer	**83.51**	**94.23**	**94.34**	**86.11**	**87.88**	**88.26**	**89.06**
UAN + EGLayer w/ $\mathcal{L}_1$	83.89	93.58	93.57	85.69	87.38	87.24	88.56
UAN + EGLayer w/ $\mathcal{L}_2$	83.92	93.86	93.74	85.69	84.28	85.79	87.88
UAN + EGLayer w/o $\mathcal{L}_a$	83.89	93.06	92.94	85.69	86.87	87.05	88.25
UAN + EGLayer w/ $\mathcal{L}_{reg1}$	48.23	29.86	72.45	85.71	83.56	44.84	60.78
UAN + EGLayer w/o $\mathcal{L}_{reg}$	**84.59**	94.07	94.03	83.63	**88.59**	**88.28**	88.87
UAN + EGLayer w/o $\mathcal{L}_g$	84.20	93.69	93.72	84.82	85.41	86.18	88.00
UAN + EGLayer	83.51	**94.23**	**94.34**	**86.11**	87.88	88.26	**89.06**
UAN + EGLayer + 2 layer GCN	82.79	88.71	92.79	84.31	81.72	82.27	85.43
UAN + middle EGLayer	**84.12**	93.12	**95.28**	85.69	87.10	87.46	88.80
UAN + EGLayer	83.51	**94.23**	94.34	**86.11**	**87.88**	**88.26**	**89.06**

3 CORRELATION LOSS STUDY

We conduct experiments to determine the optimal values for α_1 and α_2 in Eq. 13. The results on the validation set from *art* to *clipart*, with 6,000 iterations, are depicted in Figure 2. In the left chart, we keep α_2 fixed and train the model with α_1 values of 0.01, 0.05, 0.1, 0.5, 1, 5, and 10. In the right chart, we keep α_1 fixed and train the model with α_2 values of 0.001, 0.005, 0.01, 0.05, 0.1, and 0.5.

We observe that all experimental settings reach their peak performance between 3,000 and 4,500 iterations. For α_1, excessively large values ($\alpha_1 = 10$) result in significantly poorer performance. The performance of the other weights are similar, peaking at around 66%. Notably, the validation set result with a weight of 1.0 significantly outperforms other settings, leading us to choose $\alpha_1 = 1.0$.

Regarding α_2, we notice that even $\alpha_2 = 0.5$ lead to performance degradation, indicating that excessive regularization could impede the model's learning ability. The experimental performance of the other settings are relatively close, with only the 0.01 version exceeding 66%. Consequently, we select α_2 as 0.01.

4 IMPLEMENTATION DETAILS OF FEW-SHOT LEARNING

Our methods are evaluated based on the Matching Networks Vinyals et al. (2016), Prototypical Networks Snell et al. (2017), Classifier-Baseline Chen et al. (2021), and Meta-Baseline Chen et al. (2021).

Matching Networks defines a support set $\mathcal{D}^S = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}$ and the query images $\boldsymbol{x}'$ for training. The final prediction is calculated as:

$$P(\boldsymbol{y}' \mid \boldsymbol{x}', \mathcal{D}^S) = \sum_{i=1}^{k} \alpha\left(\boldsymbol{x}', \boldsymbol{x}_i\right) \boldsymbol{y}_i, \tag{4}$$

where α is attention mechanism.

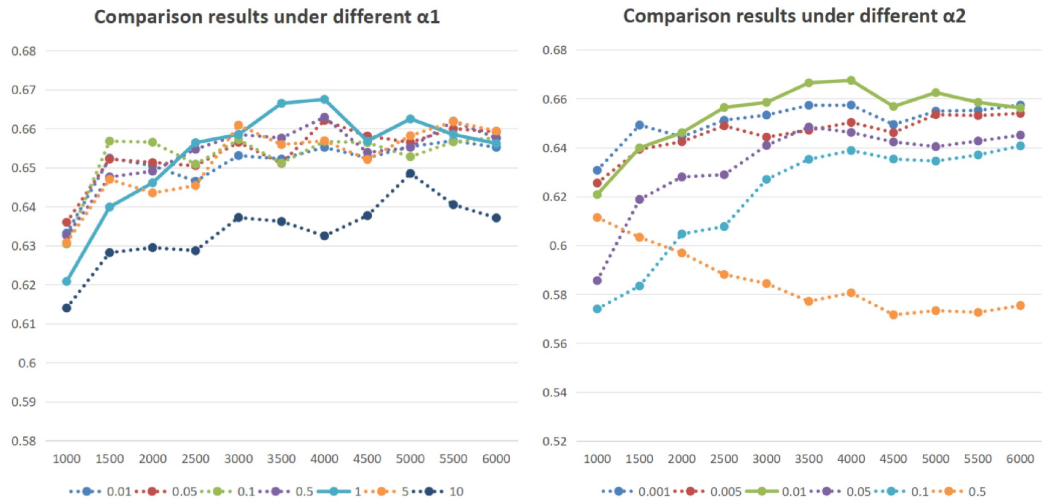

Figure 2: The results of the comparison are presented under varying values of α_1 and α_2. The validation results demonstrate that $\alpha_1 = 1.0$ and $\alpha_2 = 0.01$ yields the best performance as compared to higher and lower values of α_1 and α_2.

Prototypical Networks defines prototype w_c for training by average the embeddings of support set and exploits cosine similarity to calculate the final logits:

$$w_c = \frac{1}{|\mathcal{D}_c^S|} \sum_{\boldsymbol{x} \in \mathcal{D}_c^S} f_\theta(\boldsymbol{x}), \tag{5}$$

$$P(\boldsymbol{y}' = c \mid \boldsymbol{x}', \mathcal{D}^S) = softmax(\langle f_\theta(\boldsymbol{x}'), w_c \rangle). \tag{6}$$

Classifier-Baseline and Meta-Baseline first utilize the whole label-set for training on all base classes with cross-entropy loss. For validation, classifier is removed and feature extractor f_θ is used to computes the average embedding w_c of each class c in support set $\mathcal{D}^S$ as Prototypical Networks.

Then, for a query sample $\boldsymbol{x}'$, cosine similarity is computed between the extracted features of $\boldsymbol{x}$ and average embedding w_c for the final prediction with softmax function:

$$P(\boldsymbol{y}' = c \mid \boldsymbol{x}') = \frac{\exp\left(\tau \cdot \langle f_\theta(\boldsymbol{x}'), w_c \rangle\right)}{\sum_{c'} \exp\left(\tau \cdot \langle f_\theta(\boldsymbol{x}'), w_{c'} \rangle\right)}, \tag{7}$$

where the Meta-Baseline trains a learnable scalar τ through a meta learning way and the Classifier-Baseline fixes the τ as 1.0.

5 Transfer Learning

We have evaluated the transfer learning ability of our method by exchanging the models trained on miniImageNet and tieredImageNet in both Classifier-Baseline and Meta-Baseline settings. We name the model trained on miniImageNet for few-shot learning on tieredImageNet as miniImageNet→tieredImageNet, and vice versa. As shown in Table 2, Meta-Baseline + EGLayer achieves the best performance for both 1-shot (67.98%) and 5-shot (81.27%) in miniImageNet→tieredImageNet setting.

In the tieredImageNet→miniImageNet setting, Classifier-Baseline + EGLayer outperforms Classifier-Baseline and Classifier-Baseline + LPLayer, improving the performance by 1.04%/0.61% and 0.73%/0.88%, respectively. For Meta-Baseline, Meta-Baseline + EGLayer still have 1.74%/1.24% and 0.29%/0.36% improvements over Meta-Baseline and Meta-Baseline + LPLayer.

In general, our method demonstrates an overall advantage in transfer learning tasks, validating the generalization and reliability of the learned features.

Table 2: Transfer experiments on miniImageNet and tieredImageNet

dataset	Methods	Backbone	1-shot	5-shot
miniImageNet→tieredImageNet	Classifier-Baseline	ResNet-12	64.15 ± 0.54	79.81 ± 0.42
	Classifier-Baseline + LPLayer	ResNet-12	64.51 ± 0.33	79.83 ± 0.44
	Classifier-Baseline + EGLayer	ResNet-12	**64.89 ± 0.56**	**80.03 ± 0.43**
	Meta-Baseline	ResNet-12	67.63 ± 0.49	80.99 ± 0.42
	Meta-Baseline + LPLayer	ResNet-12	67.61 ± 0.58	80.88 ± 0.43
	Meta-Baseline + EGLayer	ResNet-12	**67.98 ± 0.59**	**81.27 ± 0.43**
tieredImageNet→miniImageNet	Classifier-Baseline	ResNet-12	76.35 ± 0.22	90.50 ± 0.12
	Classifier-Baseline + LPLayer	ResNet-12	76.66 ± 0.24	90.23 ± 0.12
	Classifier-Baseline + EGLayer	ResNet-12	**77.39 ± 0.28**	**91.11 ± 0.14**
	Meta-Baseline	ResNet-12	76.79 ± 0.24	89.53 ± 0.13
	Meta-Baseline + LPLayer	ResNet-12	78.24 ± 0.29	90.41 ± 0.22
	Meta-Baseline + EGLayer	ResNet-12	**78.53 ± 0.31**	**90.77 ± 0.13**

6 ZERO-SHOT LEARNING

Table 3: Zero-shot experiments on miniImageNet and tieredImageNet

dataset	Methods	0-shot
miniImageNet	Classifier-Baseline + EGLayer	48.34 ± 0.20
tieredImageNet	Classifier-Baseline + EGLayer	48.50 ± 0.25

We conduct zero-shot experiments to evaluate whether our proposed method could align extracted features more closely with the semantic space by leveraging external knowledge. In these experiments, we employed graph node embeddings instead of one-shot image features. The results presented in Table 3 indicate that EGLayer achieved an accuracy of approximately 50% in zero-shot tasks. This result confirms the ability of EGLayer to align features with the semantic space effectively.

REFERENCES

Yinbo Chen, Zhuang Liu, Huijuan Xu, Trevor Darrell, and Xiaolong Wang. Meta-baseline: Exploring simple meta-learning for few-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9062–9071, 2021.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.