

---

# HELM: Hierarchical and Explicit Label Modeling with Graph Learning for Multi-Label Image Classification

---

Marjan Stoimchev<sup>1,2</sup>   Boshko Koloski<sup>1,2</sup>

Jurica Levatic<sup>1</sup>   Dragi Kocev<sup>1</sup>   Sašo Džeroski<sup>1</sup>

<sup>1</sup> Jožef Stefan Institute, Ljubljana, Slovenia

<sup>2</sup> Jožef Stefan International Postgraduate School, Ljubljana, Slovenia

{marjan.stoimchev, boshko.koloski, jurica.levatic}@ijs.si

{dragi.kocev, saso.dzeroski}@ijs.si

## Abstract

Hierarchical multi-label classification (HMLC) is essential for modeling complex label dependencies in remote sensing. Existing methods, however, struggle with multi-path hierarchies where instances belong to multiple branches, and they rarely exploit unlabeled data. We introduce HELM (*Hierarchical and Explicit Label Modeling*), a novel framework that overcomes these limitations. HELM: (i) uses hierarchy-specific class tokens within a Vision Transformer to capture nuanced label interactions; (ii) employs graph convolutional networks to explicitly encode the hierarchical structure and generate hierarchy-aware embeddings; and (iii) integrates a self-supervised branch to effectively leverage unlabeled imagery. We perform a comprehensive evaluation on four remote sensing image (RSI) datasets (UCM, AID, DFC-15, MLRSNet). HELM achieves state-of-the-art performance, consistently outperforming strong baselines in both supervised and semi-supervised settings, demonstrating particular strength in low-label scenarios.

## 1 Introduction

Hierarchical multi-label classification (HMLC) addresses predictive modeling problems where samples are annotated with multiple labels that are organized in a hierarchy (e.g., a tree or a directed acyclic graph). While incorporating hierarchical information has shown promise in fields like gene function prediction [1, 2] and text categorization [3, 4], its potential in computer vision, and specifically remote sensing, remains largely unexplored.

**Limitations.** Modern HMLC methods typically translate the hierarchy into the network design [5, 6] or embed hierarchical constraints into the loss function [7–9]. However, they face critical limitations: (i) they often assume single-path hierarchies, failing to model realistic multi-path scenarios where images contain multiple object categories across different branches of the hierarchy; (ii) they underuse the hierarchy; network-based approaches are computationally heavy, and loss-based formulations often miss long-range dependencies; (iii) they focus almost exclusively on supervised learning, ignoring the vast amounts of available unlabeled data. While recent methods have started using Graph Neural Networks (GNNs) [10, 11], they remain limited, and semi-supervised learning (SSL) for HMLC in computer vision is practically non-existent.

---

\* Correspondence to marjan.stoimchev@ijs.si

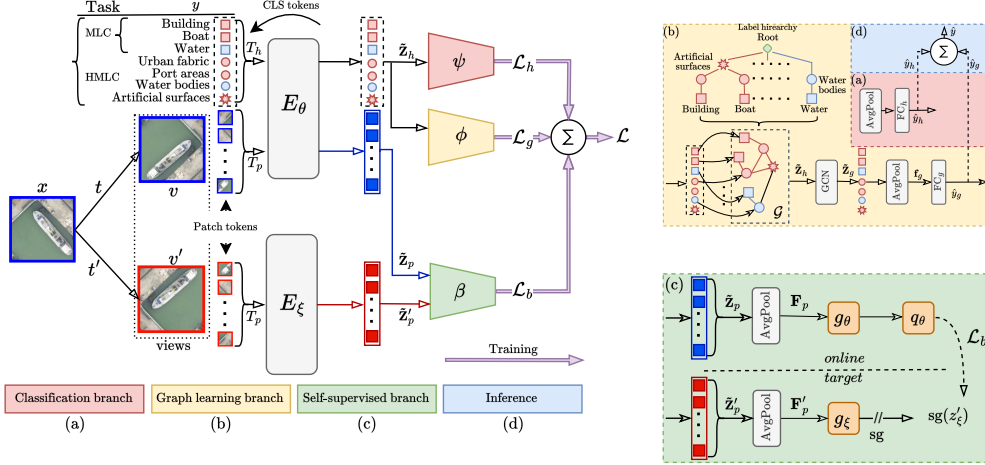


Figure 1: High-level overview of HELM. The framework integrates a ViT encoder with hierarchy-specific tokens that feed three distinct branches: (a) classification, (b) graph learning with a GCN, and (c) a BYOL self-supervised branch. The losses from each branch are combined to optimize the model end-to-end.

**Our Approach.** We propose HELM, a novel semi-supervised framework designed to address these limitations. HELM utilizes a multi-branch architecture with three key components: (i) hierarchy-specific class tokens integrated into a Vision Transformer (ViT) encoder to explicitly model label interactions; (ii) a graph learning branch that uses Graph Convolutional Networks (GCNs) to model dependencies by propagating information through parent-child relationships; (iii) a self-supervised component (BYOL) that leverages unlabeled data to learn robust representations. To our knowledge, HELM is the first semi-supervised HMLC method for images capable of handling complex multi-path hierarchies.

**Contributions.** Our main contributions are: (1) A novel multi-token transformer architecture that integrates graph-based hierarchical reasoning and self-supervised learning for HMLC. (2) Extensive experiments on four real-world remote sensing datasets demonstrating consistent and significant improvements over baselines and state-of-the-art methods. (3) A framework that effectively leverages unlabeled data, achieving substantial performance gains (up to 37%) in low-label regimes, which are common in remote sensing applications [12].

## 2 Methodology

### 2.1 Architecture Overview

HELM consists of three branches that are jointly optimized through a composite loss function  $\mathcal{L} = \mathcal{L}_s + \mathcal{L}_g + \mathcal{L}_b$  (Figure 1). These branches are: (i) a classification branch for discriminative learning on labeled data; (ii) a graph learning branch to capture hierarchical dependencies; and (iii) a self-supervised branch to leverage unlabeled data.

### 2.2 Encoder with Hierarchy-Specific Tokens

Given an input image  $x \in \mathbb{R}^{C \times W \times H}$ , we follow the standard ViT process [13], dividing it into  $N_p$  patches and projecting them to obtain a sequence of patch tokens  $T_p \in \mathbb{R}^{N_p \times d}$ . To explicitly model the label structure, we introduce  $M$  learnable, hierarchy-specific CLS tokens  $T_{\text{CLS}} \in \mathbb{R}^{M \times d}$ , where  $M$  is the total number of labels (leaf and intermediate) in the hierarchy. These tokens serve a dual purpose: they provide the output dimensionality for classification and act as initial node embeddings for the graph learning branch.

The tokens are concatenated,  $T = [T_{\text{CLS}} \| T_p] \in \mathbb{R}^{(M+N_p) \times d}$ , and processed through the ViT encoder  $E(\cdot, \theta)$ . Through the self-attention mechanism, the CLS tokens interact with the patch tokens, allowing them to evolve into semantically meaningful embeddings that represent specific labels.

### 2.3 Classification Branch

This branch performs supervised learning on labeled data. It aggregates the output hierarchy-specific token embeddings  $\tilde{\mathbf{z}}_{\text{CLS}} \in \mathbb{R}^{M \times d}$  via average pooling to get a unified representation  $\mathbf{f}_{\text{CLS}}$ . This vector is projected by a fully connected layer to the label space. The supervised loss is computed using binary cross-entropy ( $\mathcal{H}$ ) on a batch of  $B_l$  labeled samples:

$$\mathcal{L}_s = \frac{1}{B_l} \sum_{i=1}^{B_l} \mathcal{H}(y_i, p_s(y|x_i)) \quad (1)$$

### 2.4 Graph Learning Branch

To model label dependencies, we construct a directed graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  from the label hierarchy. The hierarchy-specific CLS tokens  $\tilde{\mathbf{z}}_{\text{CLS}}$  serve as initial node features. A GraphSAGE [14] operator  $\phi(\cdot)$  is applied to propagate information and generate structure-aware embeddings:  $\tilde{\mathbf{z}}_g = \phi(\tilde{\mathbf{z}}_{\text{CLS}}; \mathcal{G})$ . These are then pooled and projected to get predictions. This branch processes the entire batch (labeled and unlabeled), but the loss is computed only on labeled samples, enabling a semi-supervised flow of information through the graph structure:

$$\mathcal{L}_g = \frac{1}{B_l} \sum_{i=1}^{B_l} \mathcal{H}(y_i, p_g(y|x_i)) \quad (2)$$

### 2.5 Self-Supervised Branch

We integrate Bootstrap Your Own Latent (BYOL) [15] to leverage unlabeled data. This branch operates on the entire batch in a label-agnostic manner. For each image, two augmented views are created. An online network (sharing the main encoder’s weights  $\theta$ ) is trained to predict the representation of the same image from a target network (with weights  $\xi$  updated via an exponential moving average of  $\theta$ ). The loss encourages similarity between the predictions of the online network and the projections of the target network:

$$\mathcal{L}_b = 2 - 2 \cdot \frac{\langle q_\theta(g_\theta(\mathbf{F}_p)), g_\xi(\mathbf{F}'_p) \rangle}{\|q_\theta(g_\theta(\mathbf{F}_p))\| \cdot \|g_\xi(\mathbf{F}'_p)\|} \quad (3)$$

where  $\mathbf{F}_p$  and  $\mathbf{F}'_p$  are representations from the two views,  $g$  is a projection head, and  $q$  is a predictor network.

## 3 Results

We evaluate HELM on four public remote sensing datasets: UCM [16], AID [17], DFC-15 [18], and MLRSNet [19]. These datasets cover a wide range of scene types and hierarchical complexities. Detailed dataset statistics, hierarchy construction, implementation settings, and baseline descriptions are provided in Appendix A.1.1–A.1.7.

**Supervised Results.** Table 1 shows the supervised results of HELM and its variants. Incorporating hierarchical structure already improves performance, as HMLC surpasses the flat MLC baseline on all datasets. Adding graph reasoning (HELM<sub>g</sub>) yields further gains, particularly on UCM and DFC-15, confirming the benefit of modeling label dependencies. HELM<sub>b</sub> performs best on AID and MLRSNet, indicating that utilizing unlabeled data aids generalization. The complete HELM model achieves the overall best or second-best results, with the highest AUPRC on UCM (0.904) and the lowest Ranking Loss across all datasets (0.022, 0.017, 0.006, 0.024).

**Semi-Supervised Results.** Figure 2 shows the semi-supervised performance for different amounts of labeled data (1%, 5%, 10%, and 25%). HELM-SSL consistently surpasses both its supervised variant (HELM-SL) and the supervised HMLC baseline, with the largest AUPRC improvements in the lowest label settings. At 1% supervision, HELM achieves gains of 25.0% on UCM, 6.6% on AID, 37.0% on DFC-15, and 18.5% on MLRSNet. Complete semi-supervised tables for all datasets and model variants are provided in Appendix A.2.

Table 1: Performance comparison with loss components. Best in **bold**, second-best underlined.

Method	Loss Components			AUPRC ( $\uparrow$ )				Ranking Loss ( $\downarrow$ )			
	$\mathcal{L}_s$	$\mathcal{L}_g$	$\mathcal{L}_b$	UCM	AID	DFC-15	MLRSNet	UCM	AID	DFC-15	MLRSNet
MLC	✓			0.863	0.767	0.967	0.838	0.031	0.025	0.010	0.039
HMLC	✓			0.890	0.827	0.971	0.863	0.031	0.021	0.008	0.027
HELM <sub>g</sub>	✓	✓		0.899	0.842	<b>0.979</b>	0.869	0.024	0.019	0.007	0.025
HELM <sub>b</sub>	✓		✓	0.885	<b>0.852</b>	0.969	<b>0.873</b>	0.029	0.019	0.012	0.025
<b>HELM</b>	✓	✓	✓	<b>0.904</b>	<u>0.849</u>	0.977	<u>0.871</u>	<b>0.022</b>	<b>0.017</b>	<b>0.006</b>	<b>0.024</b>

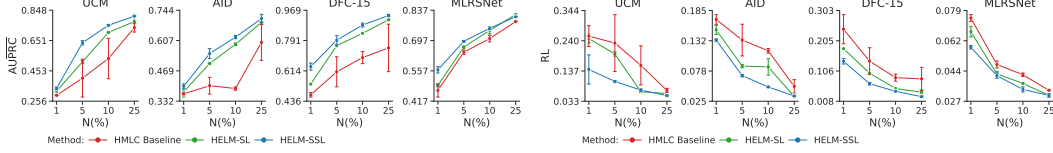


Figure 2: Semi-supervised results with different labeled proportions. HELM consistently outperforms the supervised baseline, with the largest gains at 1–5% labeled data. Full results are available in Appendix A.2.

**Comparison with State of the Art.** Table 2 compares HELM with established hierarchical multi-label classification methods: C-HMCNN [8], HiMulConE [9], and HMI [20]. HELM achieves the highest AUPRC and the lowest Ranking Loss on every dataset. It improves upon HiMulConE by 7.2% on UCM and 10.3% on AID in AUPRC, while reducing Ranking Loss by 29.0% on UCM and 15.0% on AID. These consistent improvements confirm that hierarchy-specific tokens combined with graph-based reasoning significantly enhance label consistency and predictive accuracy.

Table 2: Comparison with state-of-the-art HMLC methods in the supervised setting. Best in **bold**, second-best underlined.

Method	AUPRC ( $\uparrow$ )				Ranking Loss ( $\downarrow$ )			
	UCM	AID	DFC-15	MLRSNet	UCM	AID	DFC-15	MLRSNet
C-HMCNN [8]	0.834	0.764	0.962	0.792	0.038	0.024	0.012	0.041
HiMulConE [9]	<u>0.843</u>	<u>0.770</u>	<u>0.970</u>	<u>0.865</u>	<u>0.031</u>	<u>0.020</u>	<b>0.006</b>	<u>0.035</u>
HMI [20]	0.661	0.647	0.923	0.437	0.080	0.073	0.043	0.138
<b>HELM (Ours)</b>	<b>0.904</b>	<b>0.849</b>	<b>0.977</b>	<b>0.871</b>	<b>0.022</b>	<b>0.017</b>	<b>0.006</b>	<b>0.025</b>

## 4 Discussion

We introduced HELM, a novel semi-supervised framework for HMLC that combines hierarchy-specific tokens, graph-based structure encoding, and self-supervised representation learning. The experiments show that HELM consistently improves over strong baselines and achieves state-of-the-art performance on four RSI datasets. By modeling multi-path hierarchies and leveraging unlabeled data, it provides substantial gains, particularly in low-label regimes.

The effectiveness of HELM arises from three complementary design choices. First, hierarchy-specific CLS tokens allow the model to represent each label explicitly and capture fine-grained relationships through self-attention. Second, the GCN propagates information across parent-child links, enriching label embeddings with structural context. Third, the BYOL branch exploits unlabeled imagery to learn generalizable visual features that strengthen the model under limited supervision.

As shown in the Appendix, the graph module adds minimal computational overhead while yielding clear performance benefits. The BYOL branch increases training cost but provides important improvements when labeled data are scarce, which is often the case in remote sensing.

Future work will focus on automatic hierarchy discovery to remove manual construction, initialization of hierarchy-specific tokens using vision-language models, and extensions to multi-modal inputs such as SAR or multispectral imagery to improve generalization and applicability.

## Acknowledgments

We acknowledge the financial support of the Slovenian Research and Innovation Agency (ARIS) through the core research programme P2-0103 (Knowledge Technologies), projects J1-3033, J2-2505, J2-4452, J2-4660, J3-3070, J4-3095, J5-4575, J7-4636, J7-4637, and N2-0236. The work of the BK was supported by the Young Researcher grant PR-12394.

## References

- [1] Ping Fu Shou Feng and Wenbin Zheng. A hierarchical multi-label classification method based on neural networks for gene function prediction. *Biotechnology & Biotechnological Equipment*, 32(6):1613–1621, 2018.
- [2] Leander Schietgat, Celine Vens, Jan Struyf, Hendrik Blockeel, Dragi Koccev, and Sašo Džeroski. Predicting gene function using hierarchical multi-label decision tree ensembles. *BMC bioinformatics*, 11:1–14, 2010.
- [3] Jibing Gong, Zhiyong Teng, Qi Teng, Hekai Zhang, Linfeng Du, Shuai Chen, Md Zakirul Alam Bhuiyan, Jianhua Li, Mingsheng Liu, and Hongyuan Ma. Hierarchical graph transformer-based deep learning model for large-scale multi-label text classification. *IEEE Access*, 8:30885–30896, 2020. doi: 10.1109/ACCESS.2020.2972751.
- [4] Hao Peng, Jianxin Li, Senzhang Wang, Lihong Wang, Qiran Gong, Renyu Yang, Bo Li, Philip S. Yu, and Lifang He. Hierarchical taxonomy-aware and attentional graph capsule rcnns for large-scale multi-label text classification. *IEEE Transactions on Knowledge and Data Engineering*, 33(6):2505–2519, 2021. doi: 10.1109/TKDE.2019.2959991.
- [5] Xinqi Zhu and Michael Bain. B-cnn: branch convolutional neural network for hierarchical classification. *arXiv preprint arXiv:1709.09890*, 2017.
- [6] Brendan Kolisnik, Isaac Hogan, and Farhana Zulkernine. Condition-CNN: A hierarchical multi-label fashion image classification model. *Expert Systems with Applications*, 182:115195, 2021.
- [7] Jingzhou Chen, Peng Wang, Jian Liu, and Yuntao Qian. Label relation graphs enhanced hierarchical residual network for hierarchical multi-granularity classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4858–4867, 2022.
- [8] Eleonora Giunchiglia and Thomas Lukasiewicz. Coherent hierarchical multi-label classification networks. *Advances in neural information processing systems*, 33:9662–9673, 2020.
- [9] Shu Zhang, Ran Xu, Caiming Xiong, and Chetan Ramaiah. Use all the labels: A hierarchical multi-label contrastive learning framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16660–16669, June 2022.
- [10] Zhao-Min Chen, Xiu-Shen Wei, Peng Wang, and Yanwen Guo. Multi-label image recognition with graph convolutional networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5177–5186, 2019.
- [11] Ziwen Lan, Keisuke Maeda, Takahiro Ogawa, and Miki Haseyama. Hierarchical multi-label attribute classification with graph convolutional networks on anime illustration. *IEEE Access*, 11:35447–35456, 2023.
- [12] Anastasiia Safonova, Gohar Ghazaryan, Stefan Stiller, Magdalena Main-Knorn, Claas Nendel, and Masahiro Ryo. Ten deep learning techniques to address small data problems with remote sensing. *International Journal of Applied Earth Observation and Geoinformation*, 125:103569, 2023.
- [13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=YicbFdNTTy>.
- [14] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

- [15] Jean-Bastien Grill, Florian Strub, Florent Alché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.
- [16] B. Chaudhuri, B. Demir, S. Chaudhuri, and L. Bruzzone. "Multilabel Remote Sensing Image Retrieval Using a Semisupervised Graph-Theoretic Method. *IEEE Transactions on Geoscience and Remote Sensing*, 56(2):1144–1158, 2018.
- [17] Y. Hua, L. Mou, and X.X. Zhu. Relation Network for Multi-label Aerial Image Classification. *IEEE Transactions on Geoscience and Remote Sensing*, 2019.
- [18] Y. Hua, L. Mou, and X.X. Zhu. Recurrently exploring class-wise attention in a hybrid convolutional and bidirectional LSTM network for multi-label aerial image classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 149:188–199, 2019.
- [19] Q. Xiaoman Qi, Z. Panpan, W. Yuebin, Z. Liqiang, P. Junhuan, W. Mengfan, C. Jialong, Z. Xudong, Z. Ning, and P.M.Takis. MLRSNet: A multi-label high spatial resolution remote sensing dataset for semantic scene understanding. *ISPRS Journal of Photogrammetry and Remote Sensing*, 169:337–350, 2020.
- [20] Bo Xiong, Michael Cochez, Mojtaba Nayyeri, and Steffen Staab. Hyperbolic embedding inference for structured multi-label prediction. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 33016–33028. Curran Associates, Inc., 2022.
- [21] Y. Yang and S. Newsam. Bag-of-visual-words and spatial extensions for land-use classification. In *18th ACM SIGSPATIAL International Symposium on Advances in Geographic Information Systems*, pages 270–279, 01 2010.
- [22] G-S Xia, J. Hu, F. Hu, B. Shi, X. Bai, Y. Zhong, X. Lu, and L. Zhang. AID: A Benchmark Data Set for Performance Evaluation of Aerial Scene Classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55:3965 – 3981, 02 2017.
- [23] Referans Copernicus. Corine land cover. *Copernicus Land Monitoring Service. L. Monit. Serv.*, 2018.
- [24] Peng Xia, Xingtong Yu, Ming Hu, Lie Ju, Zhiyong Wang, Peibo Duan, and Zongyuan Ge. Hgclip: Exploring vision-language models with graph representations for hierarchical understanding. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 269–280, 2025.
- [25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

## A Appendix

### A.1 Experimental Settings

#### A.1.1 Datasets

We evaluate HELM across four public remote sensing image datasets: UCM [16, 21] (2.1k images, 17 leaf labels), AID [17, 22] (3k, 17), DFC-15 [18] (3.3k, 8), and MLRSNet [19] (109k, 60). Table 3 reports train/test splits, hierarchy sizes, and level-wise cardinality and density.

**UCM.** The UC Merced dataset contains 2,100 aerial RGB images at  $256 \times 256$  with 17 object-level (leaf) labels [16]. We construct a hierarchical extension by mapping the original categories to CORINE-inspired levels, yielding 30 labels across three levels ( $|\mathcal{L}| = \{4, 9, 17\}$ ; total  $h=30$ , leaf  $\ell=17$ ). This introduces intermediate abstractions while preserving the original leaf space.

**AID.** The Aerial Image Dataset comprises 3,000 images at  $600 \times 600$  with 17 leaf labels [17]. We construct a hierarchical version with the same depth and level sizes as UC Merced, resulting in 30 labels across three levels ( $|\mathcal{L}| = \{4, 9, 17\}$ ;  $h=30$ ,  $\ell=17$ ). Relative to UC Merced, AID

Table 3: Comprehensive statistics for remote sensing datasets.  $N$  denotes total images;  $N_{train}$  and  $N_{test}$  represent training and test set sizes.  $|\mathcal{L}|$  indicates the number of unique labels at each hierarchical level (1 to 4), where  $\ell$  corresponds to the leaf level and  $h$  refers to the total number of labels across all levels.  $Card@Level$  is the average number of active labels per image at that level (label cardinality), and  $Dens@Level$  is the mean per-label prevalence at that level (label density).

Dataset	$N$	$N_{train}$	$N_{test}$	$ \mathcal{L} $						$Card@Level$						$Dens@Level$					
				1	2	3	4	$\ell$	$h$	1	2	3	4	$\ell$	$h$	1	2	3	4	$\ell$	$h$
UCM	2,100	1,667	433	4	9	17	-	17	30	1.74	3.03	3.19	-	3.19	7.96	0.44	0.34	0.19	-	0.19	0.27
AID	3,000	2,400	600	4	9	17	-	17	30	2.18	4.51	4.93	-	4.93	11.62	0.54	0.50	0.29	-	0.29	0.39
DFC-15	3,341	2,672	669	3	7	8	-	8	17	1.61	2.73	2.80	-	2.80	6.83	0.54	0.39	0.35	-	0.35	0.40
MLRSNet	109,151	87,336	21,815	7	15	22	60	60	104	2.21	3.62	4.41	6.10	6.10	15.25	0.32	0.24	0.20	0.10	0.10	0.16

exhibits higher label co-occurrence and per-label prevalence, providing a richer supervisory signal for hierarchical learning.

**DFC-15.** The 2015 IEEE GRSS Data Fusion Contest dataset provides 3,341 image patches at  $600 \times 600$  with 8 leaf categories [18]. From the original semantic categories, we construct a compact three-level hierarchy totaling 17 labels ( $|\mathcal{L}| = \{3, 7, 8\}$ ;  $h=17$ ,  $\ell=8$ ). Although it contains fewer leaf classes than the other benchmarks, its leaf-level prevalence is highest, yielding dense supervision per image.

**MLRSNet.** MLRSNet includes 109,151 images at  $256 \times 256$  annotated with 60 leaf categories [19]. We construct the deepest hierarchical extension among the four datasets, with 104 labels across four levels ( $|\mathcal{L}| = \{7, 15, 22, 60\}$ ;  $h=104$ ,  $\ell=60$ ). Images typically carry many active leaf labels, while each individual leaf label is comparatively rare across the corpus, reflecting broad coverage over numerous fine-grained categories and stressing scalability.

In all cases, the hierarchical versions are constructed from the original multi-label datasets to introduce intermediate semantic levels while retaining the original leaf label sets.

### A.1.2 Hierarchy Construction

All hierarchical structures were constructed by systematically mapping classes to the CORINE Land Cover nomenclature (CLC) [23], which provides well-established relationships between land cover types across multiple levels. This enables meaningful hierarchical structures while preserving the multi-label nature at the leaf level. When direct CLC mapping was infeasible, we used ChatGPT-assisted mapping followed by manual validation [24]. Figure 3 presents a representative example of the constructed hierarchical structure for the UCM dataset.

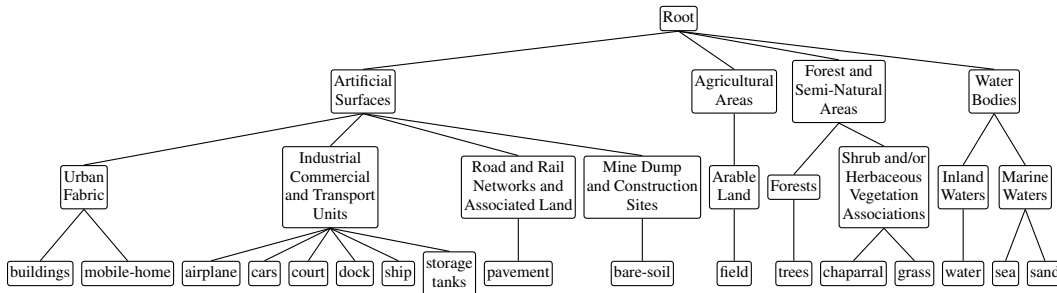


Figure 3: Example of constructed label hierarchy for the UCM dataset, derived from the CORINE Land Cover nomenclature. The hierarchy demonstrates the 3-level structure with 4 top-level categories, 9 intermediate-level categories, and 17 leaf-level labels.

### A.1.3 Implementation Details

Our HELM framework is implemented in PyTorch Lightning, leveraging the base variant of the Vision Transformer (ViT-B/16) with a patch size of  $16 \times 16$  pixels. The ViT backbone is initialized

<https://github.com/Lightning-AI/pytorch-lightning>

with pre-trained ImageNet weights to provide strong visual priors. For the graph learning component, we employ PyTorch Geometric (PyG) with a two-layer GraphSAGE network, enabling efficient message passing within the label hierarchy. The self-supervised BYOL component is integrated using LightlySSL, a specialized library for self-supervised learning in computer vision.

Models are trained on RGB images resized to  $224 \times 224$  pixels using the unified learning objective. The hierarchy-specific CLS tokens produce latent representations of dimension  $d = 768$  after average pooling. For the supervised components (classification and graph branches), we apply weak augmentations, including horizontal and vertical flips and Gaussian blur. For the BYOL component, we employ both weak and strong augmentations, with the latter incorporating color jittering, affine transformations, random cropping, and random erasing to strengthen representation learning.

Training is conducted for 100 epochs using the AdamW optimizer with a base learning rate of  $1 \times 10^{-4}$ , cosine annealing scheduler, and a mini-batch size of 16. Mixed-precision training is utilized to accelerate computation and reduce memory footprint. All experiments are performed on four NVIDIA A100 GPUs (40 GB each).

To ensure full reproducibility, all hierarchical label structures constructed for each dataset will be released as YAML configuration files, providing the exact parent-child mappings used in the graph module. These YAML files, together with the training scripts and configuration examples, will be included in the public code release accompanying the final version of this paper.

#### A.1.4 Evaluation Strategy

In the semi-supervised setting, we vary the labeled ratio in  $\{1, 5, 10, 25\}\%$ , using the remainder as unlabeled. For each configuration, we perform three independent runs with different random seeds to ensure statistical reliability. Performance is evaluated on a fixed test set using exclusively leaf labels, regardless of the training configuration. This ensures fair comparison across all methods, including those that use hierarchical information during training (HMLC, HELM variants) and those that do not (MLC baseline). We report  $\overline{\text{AUPRC}}$  and ranking loss, averaged over the three independent runs.

#### A.1.5 Computational Resources

All experiments were conducted on four NVIDIA A100 GPUs equipped with 40 GB memory each.

#### A.1.6 Evaluation Metrics

We employ two performance metrics to evaluate the effectiveness of the methods: the average area under the precision-recall curve (AUPRC) and the ranking loss measure. Given that the tasks involve multi-label classification and hierarchical multi-label classification, we use a variant of AUPRC—the area under the micro-averaged Precision-Recall curve ( $\overline{\text{AUPRC}}$ ). These metrics are chosen because they are independent of classification thresholds and provide a reliable assessment of performance.

In this context,  $\overline{\text{Prec}}$  represents the proportion of predicted labels that are correct, while  $\overline{\text{Rec}}$  corresponds to the proportion of actual labels in the dataset that are correctly predicted. These values are calculated as follows:

$$\overline{\text{Prec}} = \frac{\sum_i \text{TP}_i}{\sum_i \text{TP}_i + \sum_i \text{FP}_i}, \quad (4)$$

$$\overline{\text{Rec}} = \frac{\sum_i \text{TP}_i}{\sum_i \text{TP}_i + \sum_i \text{FN}_i}, \quad (5)$$

where  $i$  iterates over all classes. By varying the decision threshold, an average Precision-Recall (PR) curve is generated.

Ranking loss (RL), measuring the average fraction of incorrectly ordered label pairs for a given example, is defined as

---

<https://pytorch-geometric.readthedocs.io/en/latest/>  
<https://github.com/lightly-ai/lightly>



$$\text{RL} = \frac{1}{N} \sum_{i=1}^N \frac{1}{|Y_i^+| \cdot |Y_i^-|} \sum_{y_p \in Y_i^+} \sum_{y_n \in Y_i^-} \mathbb{I}(f_i(y_p) \leq f_i(y_n)), \quad (6)$$

where  $N$  is the number of examples,  $Y_i^+$  and  $Y_i^-$  are the sets of positive and negative labels for the  $i$ -th example,  $f_i(y)$  is the predicted score for label  $y$ , and  $\mathbb{I}(\cdot)$  is the indicator function, which returns 1 if the condition is true and 0 otherwise. The loss is weighted by the sizes of  $Y_i^+$  and  $Y_i^-$ , penalizing each pair of misordered labels equally. A lower ranking loss indicates better performance, as fewer label pairs are incorrectly ranked.

To compute these metrics, we use the readily available implementations from `scikit-learn` [25].

### A.1.7 Compared Methods

**HELM Variants and Baseline Methods.** To systematically evaluate the contribution of each component, we analyze different variants of HELM by selectively including loss terms from the overall objective. Table 4 summarizes all evaluated configurations and corresponding loss components.

Table 4: HELM variants and baseline methods with their loss component configurations for systematic evaluation.

Method Variant	Notation	$\mathcal{L}_s$	$\mathcal{L}_g$	$\mathcal{L}_b$	Task
MLC Baseline	MLC	✓			{SL}
HMLC Baseline	HMLC	✓			{SL}
HELM (Graph-only)	HELM <sub>g</sub>	✓	✓		{SL, SSL}
HELM (BYOL-only)	HELM <sub>b</sub>	✓		✓	{SL, SSL}
HELM (Full)	HELM	✓	✓	✓	{SL, SSL}

We establish two baseline methods to isolate the impact of hierarchical information: the MLC baseline, which uses only leaf labels during training and performs standard multi-label classification, and the HMLC baseline, which incorporates the complete hierarchical label set during training but treats all labels independently without exploiting structural relationships (essentially a flat approach to hierarchical classification). Both baselines utilize only the supervised classification loss  $\mathcal{L}_s$ , omitting the graph and self-supervised components.

The HELM variants systematically introduce additional components to assess their individual and combined contributions. HELM<sub>g</sub> adds graph-based dependency modeling to capture hierarchical label relationships, while HELM<sub>b</sub> incorporates self-supervised learning through BYOL to leverage unlabeled data. The complete HELM model integrates all three components, providing a unified framework that combines supervised classification, hierarchical reasoning, and self-supervised representation learning.

This experimental design enables precise measurement of how each component contributes to performance, particularly in scenarios with limited labeled supervision. All methods are evaluated using exclusively leaf labels of testing examples to ensure fair comparison, regardless of training configuration.

**State-of-the-Art Methods.** We also compare HELM against three state-of-the-art HMLC methods: C-HMCNN [8], HiMulConE [9], and HMI [20]. These methods represent a diverse range of approaches to hierarchical multi-label classification, highlighting the strengths and limitations of current techniques.

C-HMCNN [8] ensures consistency between parent and child predictions through a modified binary cross-entropy (BCE) loss, providing flexibility in handling complex hierarchies. Although originally designed for tabular data, we adapted it for image-based tasks by replacing its tabular encoder with our vision transformer encoder. HiMulConE [9] adopts a two-stage approach: it first employs a hierarchy-preserving contrastive loss to learn label-aware embeddings and then trains a classifier on leaf nodes using cross-entropy and softmax. Since the original method utilizes a cross-entropy loss designed for single-label classification at the leaf level, we replaced it with a BCE loss to support scenarios where multiple labels may exist at the leaf level. HMI [20] utilizes a hyperbolic

Poincaré ball model to encode logical relationships, such as implication and exclusion, through geometric constraints. This approach achieves high consistency with fewer dimensions, enhancing computational efficiency. However, similar to C-HMCNN, HMI was originally designed for tabular data and we adapted it for image-based tasks by incorporating our vision transformer encoder to enable a direct comparison.

For a fair evaluation, all methods are implemented using our vision encoder, and we utilize the official code repositories of each method, setting hyperparameters according to the original authors' recommendations.

## A.2 Complete Semi-Supervised Learning Results

Table 5 presents comprehensive semi-supervised learning results across all labeled data proportions for each dataset. The results demonstrate HELM's effectiveness in leveraging unlabeled data, with particularly strong performance gains in low-label scenarios.

Table 5: Evaluation of HELM's individual loss components in the semi-supervised learning setting compared to the supervised HMLC baseline. Models are trained with varying proportions of labeled data and evaluated on a fixed test set. Results are reported using AUPRC (higher is better) and Ranking Loss (lower is better), averaged over three runs. The best performance for each dataset and labeled data proportion is highlighted in bold.

AUPRC( $\uparrow$ )	UCM N (%)				AID N (%)				DFC-15 N (%)				MLRSNet N (%)				Avg. ranks N (%)			
Methods	1	5	10	25	1	5	10	25	1	5	10	25	1	5	10	25	1	5	10	25
HMLC Baseline	0.268	0.357	0.479	0.691	0.366	0.410	0.382	0.564	0.481	0.619	0.692	0.766	0.470	0.643	0.716	0.787	3.3	3.8	3.5	3.8
HELM <sub>b</sub>	0.290	0.554	0.656	0.770	0.362	0.461	0.557	0.671	0.522	0.717	0.824	0.909	0.380	0.643	0.709	0.788	3.3	3.0	3.0	2.8
HELM <sub>g</sub>	0.269	0.532	0.714	0.799	0.375	0.485	0.597	0.680	0.619	0.792	0.875	0.916	0.422	0.657	0.707	0.781	2.5	2.0	2.5	2.5
HELM	<b>0.335</b>	<b>0.651</b>	<b>0.730</b>	<b>0.807</b>	<b>0.390</b>	<b>0.533</b>	<b>0.631</b>	<b>0.701</b>	<b>0.659</b>	<b>0.790</b>	<b>0.885</b>	<b>0.934</b>	<b>0.557</b>	<b>0.666</b>	<b>0.743</b>	<b>0.816</b>	<b>1.0</b>	<b>1.3</b>	<b>1.0</b>	<b>1.0</b>
RL( $\downarrow$ )	UCM				AID				DFC-15				MLRSNet				Avg. ranks			
Methods	1	5	10	25	1	5	10	25	1	5	10	25	1	5	10	25	1	5	10	25
HMLC Baseline	0.266	0.232	0.178	0.080	0.169	0.122	0.137	0.062	0.232	0.164	0.101	0.088	0.074	0.048	0.042	0.033	3.0	3.8	3.8	3.5
HELM <sub>b</sub>	0.247	0.125	0.086	0.057	0.216	0.088	0.059	0.041	0.179	0.092	0.058	0.033	0.098	0.051	0.043	0.034	2.3	2.5	2.5	2.8
HELM <sub>g</sub>	0.307	0.127	0.068	0.050	0.155	0.076	0.056	0.039	0.137	0.067	0.039	0.031	0.084	0.043	0.041	0.033	3.8	2.3	2.3	2.0
HELM	<b>0.137</b>	<b>0.106</b>	<b>0.072</b>	<b>0.051</b>	<b>0.133</b>	<b>0.071</b>	<b>0.048</b>	<b>0.033</b>	<b>0.132</b>	<b>0.065</b>	<b>0.037</b>	<b>0.023</b>	<b>0.061</b>	<b>0.041</b>	<b>0.036</b>	<b>0.030</b>	<b>1.0</b>	<b>1.3</b>	<b>1.3</b>	<b>1.3</b>

The complete results demonstrate that HELM consistently outperforms both the HMLC baseline and ablated variants across all datasets and labeled data proportions. The performance gains are particularly pronounced in extremely low-resource scenarios at 1-5% labeled data. At 1% labeled data, HELM achieves substantial AUPRC improvements of 25.0% on UCM, 6.6% on AID, 37.0% on DFC-15, and 18.5% on MLRSNet compared to the HMLC baseline. As the proportion of labeled data increases, the performance gap narrows but remains consistently favorable for HELM across both metrics, demonstrating that the self-supervised component continues to provide benefits even with more abundant supervision.

### A.2.1 Computational Efficiency Analysis

Figure 4 illustrates the computational trade-offs inherent in our framework design, evaluated on the representative UCM dataset. The graph learning component proves remarkably efficient, adding only 107K parameters while delivering substantial performance gains. In contrast, the BYOL component introduces significant computational overhead due to its dual-encoder architecture.

## A.3 Qualitative Evaluation

This section presents a qualitative evaluation of HELM compared to state-of-the-art methods using the UCM dataset.

### A.3.1 Visualization of Learned Embeddings

The embeddings are derived from the hierarchy-specific CLS tokens produced by the final layer of the encoder. Since our encoder serves as a common backbone architecture across all methods, the comparison between learned embeddings is both fair and direct, ensuring that differences arise solely

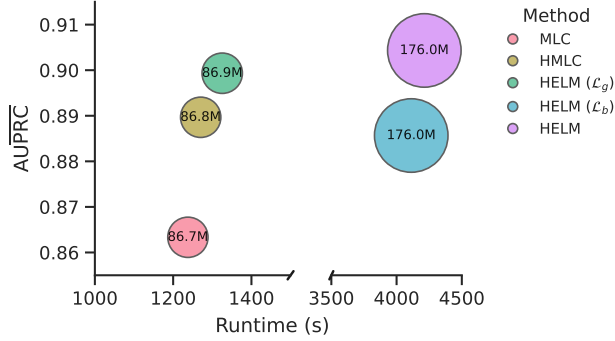


Figure 4: Comparison of training times, performance, and parameter counts for the baseline methods and HELM on the UCM dataset. The size of each bubble corresponds to the number of parameters (in millions).

from the design and learning mechanisms of the methods. We use Uniform Manifold Approximation and Projection (UMAP) to visualize the high-dimensional embeddings.

To quantify embedding quality, we use Normalized Mutual Information (NMI), which evaluates the alignment between clusters formed by embeddings and ground-truth labels. Before computing NMI, we apply  $k$ -Nearest Neighbors clustering on the original embeddings, with the number of clusters set to match the number of labels at each hierarchical level. We report the average NMI calculated by averaging NMI values across different levels of the hierarchy. This approach provides a comprehensive evaluation of both fine-grained (leaf-level) and broader (ancestor-level) clustering quality.

Figure 5 presents the UMAP analysis results across the hierarchical levels of the UCM dataset. HELM with hierarchical modeling delivers the best results, achieving NMI values of 0.411 at the first level and 0.801 at the second level of the hierarchy, with an average NMI of  $\overline{\text{NMI}} = 0.737$  across all levels. The embeddings produced are well-structured and form distinct clusters that align with the hierarchical label relationships, demonstrating the effectiveness of explicitly modeling hierarchical dependencies. This approach allows the model to capture both coarse-grained and fine-grained relationships, resulting in superior clustering quality and predictive performance. In contrast, the HMLC baseline achieves  $\overline{\text{NMI}}$  of 0.612, placing it as the second-lowest performing method in the comparison. This highlights the significant limitations of excluding hierarchical relationships, which restrict the model’s ability to organize embeddings meaningfully. While HiMulConE delivers competitive results, it falls short of capturing the hierarchical structure as effectively as HELM with hierarchical modeling.

#### A.4 Limitations

While HELM outperforms state-of-the-art models, our approach has limitations. The BYOL branch is computationally intensive, significantly increasing runtime and parameter count due to its dual-encoder architecture. This computational cost is justified by the enhanced representation quality and improved generalization capabilities observed across our experimental evaluation, particularly in low-label scenarios where annotation costs are prohibitive.

As shown in the UMAP analysis (Figure 5), while HELM outperforms state-of-the-art methods and generates well-defined clusters that align with the hierarchical structure of labels, it occasionally fails to consistently cluster hierarchical levels. For instance, labels such as "sand" and "water", which share the same parent label "Water Bodies" at higher levels of the hierarchy, are well-clustered individually but fail to fully reflect their shared broader grouping. This suggests that although HELM excels at modeling fine-grained relationships, there is room for improvement in capturing and representing broader hierarchical structures more consistently.

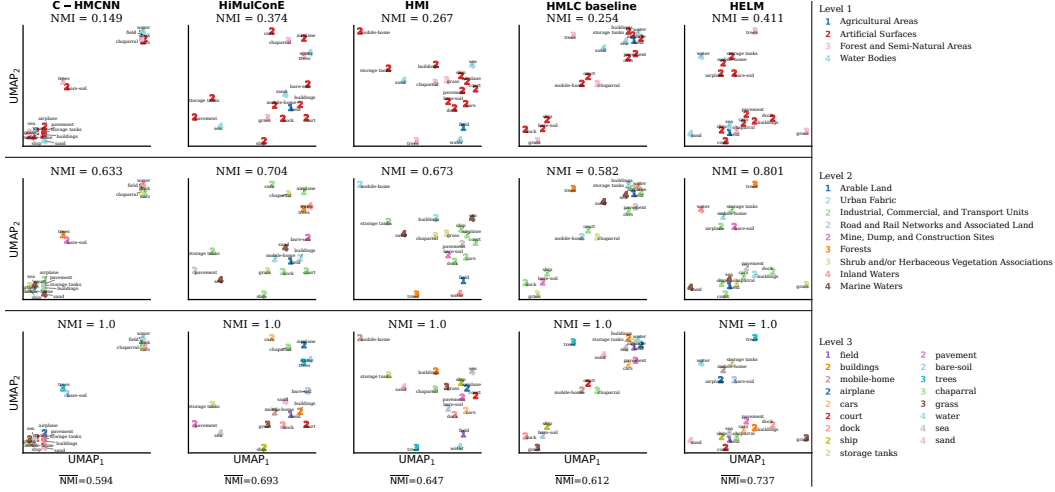


Figure 5: Comparison of 2-D UMAP embeddings between HELM and state-of-the-art methods for the UCM dataset. The learned embeddings are colored based on different levels of the UCM label hierarchy. The visualization is based on embeddings corresponding to leaf labels, while the color coding reflects the grouping and relationships at each hierarchical level. The NMI values are reported for each method, where higher values indicate better alignment between clusters and ground truth labels, reflecting the quality of hierarchical embeddings.

## EurIPS Paper Checklist

### 1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope?

Answer: [Yes]

Justification: The abstract and Introduction clearly describe the three main components of HELM: hierarchy-specific tokens, graph-based reasoning, and self-supervised learning. They also state the expected improvements in both supervised and semi-supervised settings. These claims are directly supported by the experimental results shown in Table 1 and Table 2, as well as by the learning curves in Figure 2, which confirm the accuracy and scope of the stated contributions.

### 2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: The paper explicitly discusses both computational and methodological limitations. Appendix A.2.1 presents a quantitative analysis of computational efficiency, highlighting the added cost of the BYOL branch and the minimal overhead of the graph module. The dedicated *Limitations* subsection further reflects on representational shortcomings in capturing high-level hierarchical groupings and the increased runtime associated with the dual-encoder design (see Fig. 5). These discussions provide a balanced view of HELM’s trade-offs in performance and scalability.

### 3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper is empirical and methodological without formal theorems or proofs.

### 4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: All experimental details required for reproduction are provided. The datasets, hierarchy construction process, model configurations, training setup, metrics, and evaluation protocols are described in Appendix A.1.1–A.1.7, A.1.3, and A.1.4, with model variants summarized in Table 4. To ensure full reproducibility, the exact hierarchical label structures used for each dataset will be released as YAML configuration files alongside the code.

## 5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Public datasets are cited and described (Appendix A.1.1), but an anonymized code release and scripts are not included in the current submission. We plan to release code and detailed run scripts upon acceptance.

## 6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Implementation and training details, augmentations, optimizer, schedules, batch sizes, epochs, and evaluation setup are provided in Appendix A.1.3 and A.1.4.

## 7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report averages and standard deviations over three independent runs with different random seeds in the learning curves (Fig. 2), where the variability bands make the stability of the methods clear. All tables report mean values only, while per-run results and standard deviations are omitted for brevity but are reflected in the learning curves.

## 8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We provide details on the computational setup (on single A100 40GB GPU) and report a dedicated computational efficiency analysis in Appendix A.2.1 and Figure 4. The analysis quantifies runtime, parameter counts, and trade-offs across model variants on the UCM dataset. For simplicity, runtime profiling is presented on a single dataset, but the relative scaling trends remain consistent across all benchmarks. Training times are comparable between runs, with the graph branch adding negligible overhead and the BYOL component introducing a predictable increase in runtime due to its dual-encoder structure.

## 9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: The work uses public remote sensing datasets with citations, no sensitive personal data, and follows standard evaluation practices. No violations of the Code of Ethics were identified.

**10. Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [No]

Justification: While we do not include a dedicated broader impacts section, the paper discusses potential positive implications in environmental monitoring, urban planning, and climate assessment within the Introduction and Discussion. Given the foundational and non-sensitive nature of this research, no direct negative societal impacts are anticipated.

**11. Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The work does not release high-risk foundation models or scraped web datasets. It evaluates task models on public benchmarks.

**12. Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [No]

Justification: All datasets and prior methods used in this work are properly cited in the main text and appendix. However, the specific dataset and code licenses are not explicitly listed in the current version for brevity. We will include the corresponding license information for each asset in the camera-ready version to ensure full compliance and transparency.

**13. New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: The paper does not introduce new datasets or public model checkpoints at the time of submission. Upon acceptance, all configuration files, hierarchical mappings, and trained model weights will be documented and released to support full reproducibility.

**14. Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The research does not involve crowdsourcing or human subjects.

**15. Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The research does not involve human subjects and therefore does not require IRB review.

**16. Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

Answer: [Yes]

Justification: We used ChatGPT in two limited and fully disclosed ways: (i) to assist in hierarchy mapping for classes without direct CORINE Land Cover alignment (Appendix A.1.2), and (ii) for light paraphrasing and linguistic refinement of manuscript text to improve clarity and readability. The model was not involved in designing methods, conducting experiments, or interpreting results.