A Comparative Analysis of Multi-Task Learning Approaches in the Context of Multi-Label Remote Sensing Image Retrieval

Jun Xiang
404895
MSc. Geoinformation & Geodesy

Abstract

We studied the retrieval performance of three multi-task methods on remote sensing dataset BigEarthNet and MLRSNet, and find that multi-task approaches outperformed a simple single-task approach very moderately. However, this conclusion is not for sure because these multi-task approaches were experimented with non-optimized margin value and improper label-by-label sampling method, further experiments should be carried out to get a more grounded conclusion.

1. Introduction

This project aims to compare the performance of multi-task approaches in content-based remote sensing image retrieval (CBIR). The goal of all the methods in this work is to learn a metric for multi-label images, such that samples with maximum overlap in label sets are close. The three multi-task methods we compared are:

1. Diverse Visual Feature Aggregation for Deep Metric Learning (**Diva** [4])

2. Divide and Conquer the Embedding Space for Metric Learning (**D&C** [7])

3. Deep Metric Learning with BIER: Boosting Independent Embedding Robustly (**Bier** [6])

The baseline for this work was Margin baseline [9], a single task approach which only used margin loss (Eq.1) for metric learning, the batch miner for Margin baseline was online semihard triplet mining, which tended to generate semihard positive and negative samples in current mini-batch if these samples were available, otherwise it would randomly choose any easy positive or easy negative samples. When there were fewer semihard samples on the fly, the train loss would plateau. The common mathematical notations used in this work are listed in Table 1.

Table 1: Mathematical notations

Notations	Description
X	the image data for an image patch
Y	a set of category labels for an image patch
y	a category label for an image patch, $y \in Y$
ϕ	the embedding function
f	the feature encoder (ResNet50)
x	L2 normalized embedding vector $x = \phi(f(X))$
d_{ap}	L2 embedding distance between X_a, X_p
d_{an}	L2 embedding distance between X_a, X_n
t	an image triplet $\{X_a, X_p, X_n\}$
T	the image triplets set, $t \in T$
|T|	the size of image triplets set T
ζ	loss function

2. Methods

2.1. Diva

Diva proposed four tasks as illustrated in Fig 1 to capture different aspects of the image data. The discriminative task captured features which allow to accurately separate one class from all others, the anchor and positive samples were from the same class while the negative sample was from a different class; The class-shared task learned the commonalities among different classes, its anchor, positive and negative samples were all from different classes; Intra-class task described variations within a given class, its anchor, positive and negative samples were from the same class; And finally sample-specific task learned the features invariant to transformations and rotations, its positive sample was a flipped and augmented version of the anchor sample, and its negative sample was randomly selected.

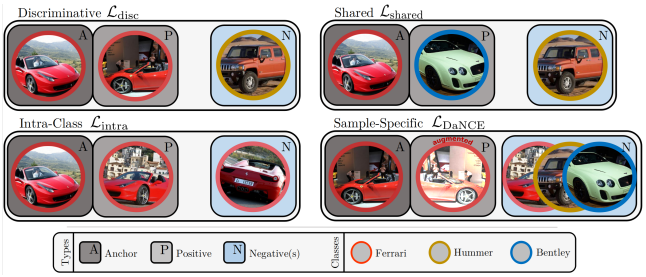


Figure 1: Diva tasks [4].

The architecture of Diva is presented in Fig 2. In this work we used ResNet50 as the backbone, in the forward pass, image patches were fed into the backbone, then the last linear layer feature was split into four parts, each part went through a regressor and came out as a sub-embedding with size 128, then margin loss (Eq.1) was applied for discriminative, intra-class, and class-shared tasks, and fast momentum contrastive loss (Eq.6) was used for self-similarity task, because there were some overlap between embedding vectors of different tasks, an adversarial loss (Eq.2) was designed to maximize the difference between these embedding pairs. In the backward pass, the gradients from all the loss functions propagated back to the embedding layer.

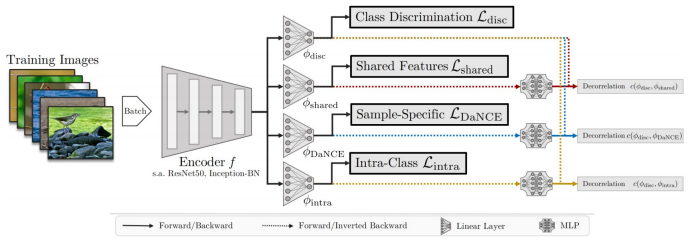


Figure 2: Diva architecture [4].

- **Margin Loss [9]** Let the l2 distance between embedding vector x_i and embedding vector x_j be denoted as d_{ij}, the margin loss for this embedding pair can be presented as:

$$\zeta^{margin} = (\alpha + y_{ij}(d_{ij} - \beta))_+ \qquad (1)$$

where $y_{ij} = 1$ when this embedding pair shares the same class label, otherwise $y_{ij} = -1$. α controls the separation of the margin, β determines the boundary between positive pairs and negative pairs, so the threshold value for positive samples is β-α, for negative samples is β+α. We set $\alpha = 0.2$ and β trainable with initial value 1.2, β is the same for all the classes, the default learning rate for β is 5e-4.

- **Adversarial Loss** The adversarial loss used for task decorrelation is presented in Fig 3, Learner i and Learner j are embedding vectors(red), in the forward pass, the vector of embedding j passed through a regressor (blue), and was projected to the vector space of embedding i, the loss function tried to maximize the similarity of projected vector and embedding i. In the backward pass, the sign of gradients of both embedding vectors were flipped by the gradient reverse layer, therefore the reversed gradients back-propagated to embedding layers were minimizing the similarity of this embedding pair.
Let the adversarial regressor (blue) be denoted as a function g(i,j): $R^{d_j} \rightarrow R^{d_i}$, projecting x_j into the space of x_i, let the length of x_i be denoted as dim_i, the adversarial loss can be written as:

$$\zeta^{adv} = -\frac{1}{dim_i}\sum(g(i,j)(x_j) \odot x_i)^2 \qquad (2)$$

where $\odot$ is the element-wise multiplication.

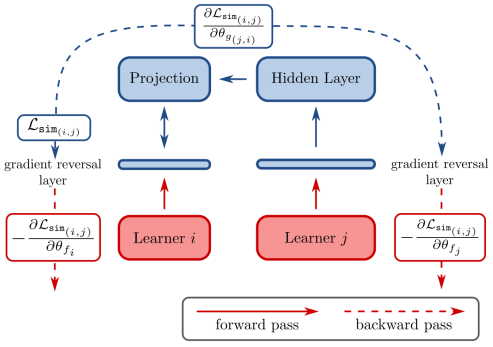


Figure 3: Adversarial Loss [6]

- **Loss setup for Diva** Let T_{disc} be the image triplets set for discriminative task, for t in T_{disc} , $y_a = y_p$ and $y_a \neq y_n$, distance weighted sampling [9] was applied when selecting negative samples, this batch miner chose positive samples randomly but chose negative samples based on reversed L2 distance to the anchors, it tended to choose the negatives which were more close to the anchors than other negatives in the mini-batch, which resulted in a variety of negatives (easy, semihard, and hard), and improved performance. The margin loss for discriminate embedding function ϕ_{disc} is :

$$\zeta_{disc} = \frac{1}{|T_{disc}|}\sum_{t \in T_{disc}} \zeta^{margin}(t) \qquad (3)$$

Let T_{shared} be the image triplets set for class-shared task, for t in T_{shared}, $y_a \neq y_p \neq y_n$, the margin loss

for class-shared embedding function ϕ_{shared} is:

$$\zeta_{shared} = \frac{1}{|T_{shared}|} \sum_{t \in T_{shared}} \zeta^{margin}(t) \quad (4)$$

Let T_{intra} be the image triplets set for intra-class task, for t in $y_a = y_p = y_n$, the margin loss for intra-class embedding function ϕ_{intra} is:

$$\zeta_{intra} = \frac{1}{|T_{intra}|} \sum_{t \in T_{intra}} \zeta^{margin}(t) \quad (5)$$

For self-similarity task, data augmentation was applied to the anchor X_a to generate its positive surrogate $\hat{X}_a$, then it was passed to another feature encoder (Resnet50) which denoted as $\hat{f}$ to get its embedding vector $\hat{x}_a = \hat{\phi}(\hat{f}(\hat{X}_a))$. Next, we use noise contrastive estimation (NCE) to increase the correlation between the anchor embedding vector x_a and its positive surrogate $\hat{x}_a$ by contrasting against a set of its negative embedding vector x_n. We denote all the anchor embedding vectors as I, and its negative counterparts as N, The loss function for self-similarity embedding function ϕ_{NCA} is:

$$\zeta_{NCE} = \frac{1}{|I|} \sum_{x_a \in I} -log \frac{exp(x_a \cdot \hat{x}_a/\tau)}{\sum_{x_n \in N} exp(x_a \cdot \hat{x}_n/\tau)} \quad (6)$$

where the temperature parameter τ is set to 0.07
The final loss of Diva is minimized by performing joint training of all tasks:

$$\begin{aligned} \zeta_{Diva} = \zeta_{disc} + \alpha_1\zeta_{shared} + \alpha_2\zeta_{intra} + \alpha_3\zeta_{NCE} \\ +\rho_1\zeta^{adv}(\phi_{disc}, \phi_{NCE}) \\ +\rho_2\zeta^{adv}(\phi_{disc}, \phi_{shared}) \\ +\rho_3\zeta^{adv}(\phi_{disc}, \phi_{intra}) \end{aligned} \quad (7)$$

where α is the weight for the loss of each task, it was set to {1, 0.3, 0.3, 0.3} by default. ρ is the weight for decorrelation between embedding vectors from different tasks, it was set to {1500, 1500, 1500} by default. Table 2 is the setup summary of Diva. The final embedding vector of Diva is concatenated by ϕ_{disc}, ϕ_{NCE}, ϕ_{shared} and ϕ_{intra} with weights {0.5, 1, 1, 1}.

2.2. D&C

D&C as showed in Fig 4 tried to learn different aspects of image features by dividing the dataset into different clusters. Similar to Diva architecture, the D&C model also used ResNet50 as the backbone, the training set was fed into ResNet50 to produce the initial embedding vectors (size 512), then the training set was divided into several clusters based on the L2 distance among initial embedding vectors. Then the data from each cluster was fed into ResNe50 and the associated sub-embedding layer exclusively, margin loss (Eq.1) was applied for the embedding vectors from each cluster. In the backward pass, the gradient for each cluster was only propagated to the respective sub embedding layer. The recommended number of clusters was 8 [7].

Table 2: Diva summary

Diva tasks			
Tasks	Loss	Batch minner	Embed size
ζ_{disc}	Margin [9]	Distance weighted [9]	128
ζ_{NCE}	MOCO [2]	None	128
ζ_{shared}	Margin [9]	Random distance [1]	128
ζ_{intra}	Margin [9]	Intra random [1]	128

Diva task decorrelation		
Task decorrelation	Loss	hidden layer size
ζ_{disc}, ζ_{NCE}	Adversarial	512
ζ_{disc}, shared	Adversarial	512
ζ_{disc}, ζ_{shared}	Adversarial	512

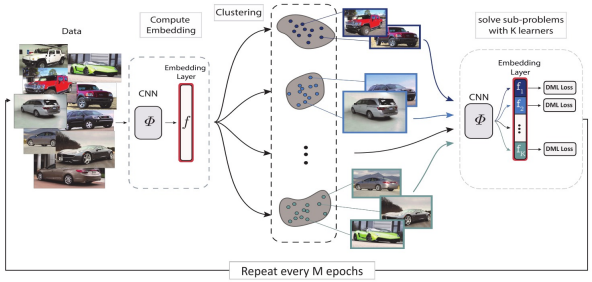


Figure 4: D&C architecture [7].

- **Loss setup for D&C** All the image data were grouped into 8 clusters $\{C_k | 1 \leq k \leq 8\}$ according to their pairwise distance in embedding space with K-means, for each cluster C_k there was a dedicated embedding function ϕ_k, triplets set T_k were sampled from cluster C_k, the positive samples were randomly selected for each anchor while the negative samples were chosen by distance weighted sampling [9]. Then T_k were fed into margin loss to minimize the loss of ϕ_k. So the final loss for D&C is:

$$\zeta_{D\&C} = \frac{1}{|T_k|} \sum_{t \in T_k} \zeta_k^{margin}(t) \quad (8)$$

Table 3: D&C summary

D&C parameters	
Loss	Margin [9]
Batch minner	Semihard
Clusters	8
Re-clustering	every 10 epochs
Fine tune epoch	110
Sub-embedding size	64

Table 3 summarizes the setup of D&C, the original paper [7] has tested the influence of the re-clustering frequency on retrieval performance, and it didn't make big difference to re-cluster the train set every 2 epochs or 10 epochs, to reduce the training time we set re-clustering every 10 epochs. After 110 epochs, re-clustering will be stopped on train set, and all the sub-embedding vectors were concatenated for fine-tuning.

2.3. Bier

Bier [6] as showed in Fig 5 used a boosted way to make the loss function focus more on samples which violated the margin. Similar as Diva architecture, the Bier model also used ResNet50 as the backbone, in the forward pass, image patches were fed into the backbone to produce the initial embedding vectors (size 512), they were split into three groups of sub-embedding vectors with size 96, 160 and 256 by three learners, then binomial deviance loss (Eq.9) was applied on them in a boosted way: 1) A similarity function (Eq.10) was applied to the sub-embedding vector pairs to get similarity matrix for each group. 2) different combination of these similarity matrix were fed into binomial loss to get the gradients, then the negative values of these gradients were served as the weights for the loss of next learner as showed in (Eq.9). The boosted loss for Bier is a sum of weighted loss from each learner. In order to capture more diversified feature characteristics, an adversarial loss (Eq.2) was applied to maximize the difference between pairs of sub-embedding vectors from each learner.

- **Binomial Loss** Let s_{ij} be the cosine similarity score between normalized embedding vector x_i and x_j, and y_{ij} be the indicator, $y_{ij} = 1$ when this embedding pair shares same class label, otherwise $y_{ij} = -1$, C_{ij} is the cost for this embedding pair, the binomial loss function can be written as:

$$\zeta^{bin} = \sum log\left(1 + e^{-y_{ij}\beta_1(s_{ij}-\beta_2)C_{ij}}\right) \quad (9)$$

where:
$s_{ij} = x_i {x_j}^T$

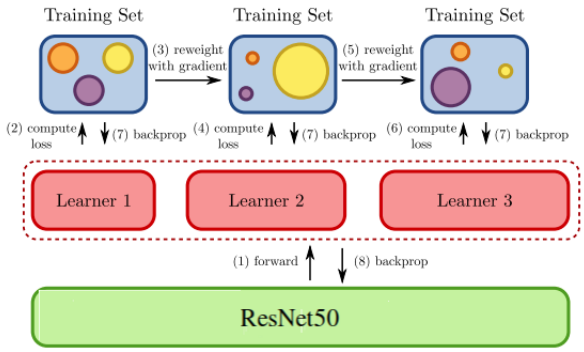


Figure 5: Bier architecture [6].

$$C_{ij} = \begin{cases} 1, & \text{if } y_{ij} = 1 \\ 25, & \text{if } y_{ij} = -1 \end{cases}$$

β_1 is the scaling parameter and set to 2 by default. β_2 is the margin parameter with default initial value 0.5, and it was set non-trainable in the original paper [6]. Here We denote $S(\phi)$ as the similarity matrix of embedding vector pairs from embedding function ϕ:

$$S(\phi) = y_{ij}\beta_1(x_i \cdot x_j - \beta_2)C_{ij} \quad (10)$$

- **Loss setup for Bier** Let the similarity matrices (Eq.(10)) of sub-embedding vector pairs for these three learner be denoted as $S(\phi_1)$, $S(\phi_2)$ and $S(\phi_3)$, the gradient of binomial loss towards similarity matrix of embedding pairs as $\partial(\zeta^{bin}(\hat{S}_i))$, the boosted weight as w_i, the boosted loss function for Bier is:

$$\begin{aligned} \zeta_{boosted} = w_1\zeta^{bin}(S(\phi_1)) + w_2\zeta^{bin}(S(\phi_2)) \\ +w_3\zeta^{bin}(S(\phi_3)) \end{aligned} \quad (11)$$

where:
$w_1 = 1$
$\hat{S}_2 = \frac{1}{3}S(\phi_1) + \frac{2}{3}S(\phi_2), w_2 = -\partial(\zeta^{bin}(\hat{S}_2))$
$\hat{S}_3 = \frac{1}{2}\hat{S}_2 + \frac{1}{2}S(\phi_3), w_3 = -\partial(\zeta^{bin}(\hat{S}_3))$

Bier added a weight constrain to its embedding layer ϕ and its adversarial layer, this weight constrain makes all the weights to have a squared L2 norm of 1 for all row vectors, and this weight loss only back-propagate to the embedding layer. Let the weight matrix for the embedding layer be denoted as W and for the adversarial layer be denoted as $\hat{W}$, the bias vector for the adversarial layer be denoted as $\hat{B}$, the

weight loss can be written as:

$$\zeta_{weight} = max(0, \hat{b}^T \hat{b} - 1) + \sum_i (\hat{w_i}^T \hat{w_i} - 1)^2 + \sum_i (w_i^T w_i - 1)^2 \tag{12}$$

Where $\hat{w_i}$ denotes the i-th row of the weight matrix $\hat{W}$, w_i denotes the i-th row of the weight matrix W. Finally, the loss function for Bier is:

$$\zeta_{Bier} = \zeta_{boosted} + \lambda_{div}(\lambda_{weight}\zeta_{weight} + \rho_1\zeta^{adv}(\phi_1, \phi_2) + \rho_2\zeta^{adv}(\phi_1, \phi_3) + \rho_3\zeta^{adv}(\phi_2, \phi_3)) \tag{13}$$

Where λ_{weight} is the regularization parameter for weights, it was set to 1e5 by default. ρ controls the strength of the decorrelation of sub-embedding vectors, it was set to {1e5, 1e5, 1e5} by default. λ_{div} controls the strength of the auxiliary loss functions, it was set to 5e-5 by default. Table 4 summarizes the setup of Bier.

Table 4: Bier summary

Bier tasks	
Loss	Binomial deviance loss
Learner1 (ϕ_1)	embed size 96
Learner2 (ϕ_2)	embed size 160
Learner3 (ϕ_3)	embed size 256
Batch miner	None

Bier task decorrelation		
Decorrelation	Loss	Hidden layer size
ϕ_1, ϕ_2	Adversarial	512
ϕ_1, ϕ_3	Adversarial	512
ϕ_2, ϕ_3	Adversarial	512

3. Evaluation Metrics

The metrics used in this work are Recall@K, R-Precision@K (R-P@K) and Mean Average Precision at K (MAP@K). The scores of these three metrics are averaged over all examples of the evaluation dataset. To define these measures, let the ground truth label set of a query image be denoted as Y, its predicted label set is denoted as Z_i, $1 \leq i \leq K$, K is the number of K nearest neighbors for a query image.

- **Recall@K** is the percentage of correctly predicted labels in all the ground truth labels. Recall@K for a single query:

$$Recall@K = \frac{|Y \cap \{Z_1 \cup Z_2 ... \cup Z_K\}|}{|Y|} \tag{14}$$

- **R-Precision@K** is defined as followed: Let K be the K retrieved images for a single query image, let r be the number of retrieved images which are correctly predicted, thus R-Precision@K for a single query is:

$$R - Precision@K = \frac{1}{K}\sum_{i=1}^{K} r(i) \tag{15}$$

where: $r(i) = \begin{cases} 1, & \{Y \cap Z_i\} = Z_i \\ 0, & otherwise \end{cases}$

- **MAP@K** is Mean Average Precision with the number of nearest neighbors for each retrieved sample set to K. MAP@K for a single query:

$$MAP@K = \frac{1}{K}\sum_{i=1}^{K} P(i) \tag{16}$$

$$P(i) = \begin{cases} R - Precision@i, & r(i) = 1 \\ 0, & otherwise \end{cases} \tag{17}$$

Suppose a query image has got 10 (K=10) retrieved images as it showed in Table 5, Recall@K is failed to capture the difference between different retrieval results. R-Precision@K can't reflect of the rank of correctly retrieved images, MAP@K takes the ranking of correctly predicted results into account, higher-ranked true positives lead better MAP, thus it is more informative than Recall@K and R-Precision@K.

Table 5: Comparison of metrics [5]

Retrieved	**Correct**	**R@1**	**R@10**	**R-P@10**	**MAP@10**
10 results	1st	100	100	10	10
10 results	1st, 10th	100	100	20	12
10 results	1st, 2th	100	100	20	20
10 results	All	100	100	100	100

4. Experiments

4.1. Datasets

Two remote sensing datasets BigEarthNet and MLRSNet were used in this work.

BigEarthNet [8] is a benchmark archive, consisting of 590,326 pairs of Sentinel-1 and Sentinel-2 image patches, after removing 71,042 patches which are covered by clounds, 519,284 patches are left for train/val/test dataset. Each patch is a section of i) 120×120 pixels for 10m bands; ii) 60×60 pixels for 20m bands; and iii) 20×20 pixels for 60m bands. Each patch in BigEarthNet is associated by class labels from CORINE Land Cover (CLC) database of

the year 2018 (CLC 2018), in total there are 43 class labels.

MLRSNet [10] contains 109,161 remote sensing images that are annotated into 60 predefined class labels. Each image patch has 3 channels (RGB) and a fixed size of 256×256 pixels with a resolution between 10m to 0.1m.

Table 6 illustrates some basic statistics of these two datasets, such as the number of patches (N), the number of labels(|L|), along with multilabel data statistics, such as label cardinality (LC) which represents the average number of labels per patch, also label density (LD) which is $\frac{LC}{|L|}$.

For BigEarthNet, cubic interpolation was applied to 20m bands and 60ms bands to make all channels have the same resolution (120×120). Both datasets are split into train/val/test set with ratio around 50%/10%/40%, the patch size for BigEarthNet is $12 \times 120 \times 120$, for MLRSNet is $3 \times 256 \times 256$, all the patches were channel-wise standardized, during training time, they were randomly cropped and flipped, during test time only the center crop of the patches were used. The cropped patch size for BigEarthNet is $12 \times 100 \times 100$, for MLRSNet is $3 \times 224 \times 224$.

Table 6: Multi-label statistics

Datasets	**N**	**|L|**	**LC**	**LD**
BigEarthNet	519,284	43	2.94	0.068
MLRSNet	109,161	60	4.97	0.083

Table 7: Dataset splits

Split	**BigEarthNet**	**MLRSNet**	**Preprocess**
Train	269,695	49,928	random crop random flip
Val	60,824	9,816	central crop
Test	188,765	49,407	

4.2. Implementation Details

The feature extractor used in this work is pretrained ResNet50. All the hyper-parameters for training in Table 8 such as the learning rate and weight decay were followed in the default settings in original papers. Batches of BigEarthNet were constructed by randomly sampling 4 images for each of the 43 classes, and batches of MLRSNet were constructed by randomly sampling 2 images for each of the 60 classes, thus the batchsize was 172 for BigEarthNet and 120 for MLRSNet, when we sampled the images, we only consider one label per image.

Table 9 lists different settings for backbone and margin. The margin β for margin loss and β_2 for binomial loss generally have two settings: 1) fixed value, 2) trainable, however margin β has two cases when it is trainable: class-specific and the same for all classes, we denote class-specific as trainable (class). The backbone has two settings: 1) frozen, 2) unfrozen. The combinations of these settings were tried out on MLRSNet train set by each method for 120 epochs (the train loss of all methods in this work were already converged when trained for 120 epochs), and the checkpoint file was saved by evaluating recall@1 on the val set every 10 epochs. Then the best model weights were applied to MLRSNet and BigEarthNet test set to get retrieval results.

During test time, the best model was loaded to generate sub-embedding vectors for image patches from test set, then all the sub-embedding vectors were concatenated up to embedding vectors with length of 512, these final embedding vectors were L2 normalized before evaluation. A KNN classifier was applied to find K most similar images

Table 8: Training Setup

Training Setup	
Number of samples per class	2 for MLRSNet 4 for BigEarthNet
Embedding size	512
Backbone	ResNet50
Weight decay of backbone	1e-4
Learning rate of backbone	1e-5
Weight decay of embedding layer	1e-4
Learning rate of embedding layer	1e-5
Learning rate of adversial layer	1e-5
initial β for Margin loss	1.2
initial β_2 for Binominal loss	0.5
Learning rate of β, β_2	5e-4
Learning rate scheduler	step gamma 0.3 tau [55]
Optimizer weight decay	4e-4
Optimizer	Adam

Table 9: Settings for backbone and margin

Setting	Backbone	Margin (β)	Binomial (β_2)
1	frozen	fixed value 1.2	fixed value 0.5
2	frozen	trainable (class)	
3	frozen	trainable	trainable
4	unfrozen	fixed value 1.2	fixed value 0.5
5	unfrozen	trainable (class)	
6	unfrozen	trainable	trainable

for a given query image, the predicted labels were made up by the labels associated to those retrieved images.

4.3. Retrieval Results

Table 10: Retrieval scores on MLRSNet

Methods	Setting	MLRSNet test set		
		R@1	R-P@8	MAP@8
Baseline	1	58.5	24.3	17.5
Baseline	2	56.5	20.7	14.4
Baseline	3	59.2	24.8	17.7
Baseline	4	**74.9**	**39.6**	**31.6**
Baseline	5	68.9	31.5	24.1
Baseline	6	70.8	34.2	26.6
Diva	1	55.2	21.0	14.7
Diva	2	56.3	20.0	13.9
Diva	3	56.3	20.2	14.0
Diva	4	53.5	19.5	13.5
Diva	5	67.2	29.3	21.7
Diva	6	**72.8**	**36.1**	**29.3**
D&C	1	58.0	24.3	17.5
D&C	2	57.7	22.3	15.7
D&C	3	58.1	24.0	17.1
D&C	4	**73.6**	**37.6**	**29.7**
D&C	5	68.9	31.6	24.2
D&C	6	73.3	37.3	29.3
Bier	1	55.3	21.7	15.3
Bier	3	57.5	23.4	16.5
Bier	4	67.7	33.4	25.6
Bier	6	**73.4**	**38.6**	**30.5**

Table 10 lists retrieval scores on MLRSNet test set under different settings listed in Table 9, the bold represents the best result. Baseline got its best recall@1 score (74.9%) with unfrozen backbone and fixed margin (setting 4). Diva and Bier obtained their best scores with unfrozen backbone and trainable margin (setting 6), and D&C got its best scores with setting 4 though it also obtained similarly good scores with setting 6. The common trend for each method is that when the backbone was frozen (setting 1, 2 and 3) their recall@1 scores were between 55% to 60%, which were far below the case when the backbone was unfrozen (setting 4, 5 and 6). Though Baseline slightly outperformed three multi-task methods, they actually obtained very similar best retrieval scores on MLRSNet.

Table 11 shows the retrieval scores on BigEarthNet when the best setting was applied. Unlike MLRSNet where each multi-task method got roughly similar results at its best setting, on BigEarthNet Bier significantly outperformed Diva and D&C on recall@1, however it had very similar R-Precision@8 and MAP@8 to Baseline.

Table 11: Retrieval scores on BigEarthNet

Methods	Setting	BigEarthNet Test set		
		R@1	R-P@8	MAP@8
Baseline	4	80.5	**43.6**	36.0
Diva	6	68.7	35.0	27.1
D&C	4	76.5	40.4	32.6
Bier	6	**82.0**	43.4	**36.3**

Fig 10 and Fig 11 show the retrieved images from MLRSNet test set by each method when the best setting was applied. The query images are at the first column, and their corresponding retrieved images are at the rest 4 columns, they were ordered by their L2 embedding distance from the query image in ascending order. The texts on the left side of each image are class names, and the red presents irrelevant class labels to the query images.

We find that all four methods have got good predictions for class labels which are relatively simple and large continuous objects like grass, bare soil, forest and water, however for small or very thin discontinuous and sparse objects like wind turbine (the 3rd query image), Bier seemed to be able to pay more attention on these small objects than Diva and D&C, its four corresponding retrieved images all contain wind turbine, while Diva and D&C have got one or two retrieved images which contain big area of bare soil without any wind turbine, Baseline mistook the transmission tower to wind turbine in one of its retrieved images.

There problem in the retrieved images is that, though all the retrieved images from each method were relevant to the queries images, they were not well-ordered according to their semantic similarity with the query images, many retrieved images were semantically exactly the same as the query images, but they were not more close to the query images in embedding space. For example, in the 2nd row of D&C results, the 4th retrieved image shared 6 labels as the query image, but it was more far away from the query image in embedding space than the 2nd and 3th retrieved images which shared 4 labels with the query image. All the methods in this work have this bad rank problem on their retrieved images, because when using margin loss or binomial loss for metric learning, the triplets or pairs were selected by considering one category label for each sample, even though each sample was associated with a set of different labels simultaneously.

4.4. Comparison on embedding distance

In our work the retrieved images were obtained by checking their distance related to query images in embedding space, in order to get correct retrieved results we need the images pairs which shared same class labels more close

than those shared different labels. In order to check how different methods work on the distance of embedding pairs, we selected the samples which contain the most common label from the val set of both datasets, and calculated their L2 distance towards the whole val set (not include itself), then we separated the distance data into two groups: intra and inter, intra group contained distance data from embedding pairs which shared at least 1 class label, while inter group were from those shared 0 labels.

All the distance data were grouped into histogram bins by counting their presences in each bin interval, then the counts for each bin were normalized to make sure that the whole area under the curve is 1. We calculated the overlap area of two different normalized histograms to measure their difference, with the possible value of the intersection lying between 0 (no overlap) and 1 (identical distributions), in our case we hope this overlap area become smaller during the training, so that the intra group will always have smaller distance than the inter group, which will result in better recall@1.

The most common class label in MLRSNet val set is class 56 (trees), about 6410 samples (65%) from MLRSNet val set contain this class. The most common class label in BigEarthNet val set is class 24 (mixed forest), about 22,729 samples (38%) from BigEarthNet val set contain this class. Table 12 shows the statistic of intra and inter pairs generated among these samples and the whole val set (not include itself), about 27.4% embedding pairs from MLRSNet val set shared 0 class label, while about 44.7% embedding pairs from BigEarthNet val set shared 0 class label. At epoch 0, the inter and intra group for both datasets were almost 100% overlapped as it showed in Fig 6. The initial mean pair distance of intra and inter group was almost the same, it was around 0.03 for MLRSNet and 0.07 for BigEarthNet. In our work the initial margin β and binomial β_2 were 1.2 and 0.5 by default, which might not properly set for either MLRSNet or BigEarthNet.

Fig 12 and Fig 13 show distance density at Epoch 120 on MLRSNet val set. Baseline and D&C got very similar density plot at same settings of margin β and backbone, their intra-inter overlap values were the same around 0.77 when margin β was fixed and the backbone was unfrozen, also the span of their embedding distance data was quite similar when they had the same setting, which is is not surprising because they all only used Margin loss for metric learning and Semihard for batch mining, the only difference was that D&C grouped the train dataset into 8 different subgroups by KMeans and was trained on the samples from each separated subgroup where samples were more close to each other, so D&C can get much harder negative samples than Baseline, which theoretically would result in better retrieval performance, but we see that D&C didn't really outperform Baseline, they have got very similar best retrieval scores as listed in Table 10.

Table 12: Statistics of embedding pairs on val set

Type	Shared labels	MLRSNet Count	Percent
Inter	0	11,629,210	27.4
Intra	1	12,263,061	28.9
	2	7,505,954	17.7
	3	4,648,709	11
	4	3,404,608	8
	>4	2,921,763	7
Total		42,373,305	100

Type	Shared labels	BigEarthNet Count	Percent
Inter	0	502,076,436	44.7
Intra	1	372,328,100	33.1
	2	190,863,827	16.9
	3	52,255,466	4.68
	4	6,226,744	0.58
	>4	403,038	0.04
Total		1,124,153,611	100

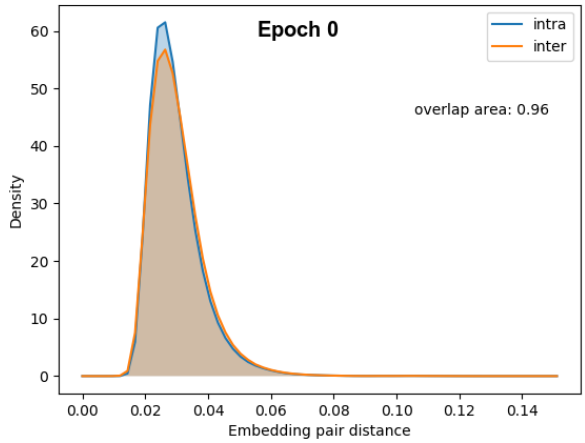
Diva used margin loss for metric learning, however at the same time it tried to capture the different aspects of the data by feeding specially defined triplets to margin loss. We find that Diva worked bad at fixed margin with initial value 1.2. When keeping margin fixed, the intra-inter overlap area was around 0.9 (frozen backbone), and it increased to 0.92 (unfrozen backbone), which was quite unusual since we have seen big improvement of Baseline and D&C with unfrozen backbone setting. The reason could be that fixed initial β value was not right for Diva on MLRSNet set, this can be further proved by Diva's retrieval scores in Table 10, it got the lowest recall@1 (53.5%) when the backbone was unfrozen and margin β was fixed.

Bier used boosted binomial loss for metric learning, and unlike Baseline, Diva and D&C worked on triplets, Bier worked on embedding pairs. And we find that the performance of Bier is sensitive to its margin β_2 value. As showed in Fig 13 (b) when working with fixed margin, Bier kept both of the inter and intra distance above 1.0 at epoch 120 regardless of the setting of backbone, this was not a coincidence. For binomial loss β_2 is the cosine distance of normalized embedding vectors, we can convert it back to L2 distance using the following functions:

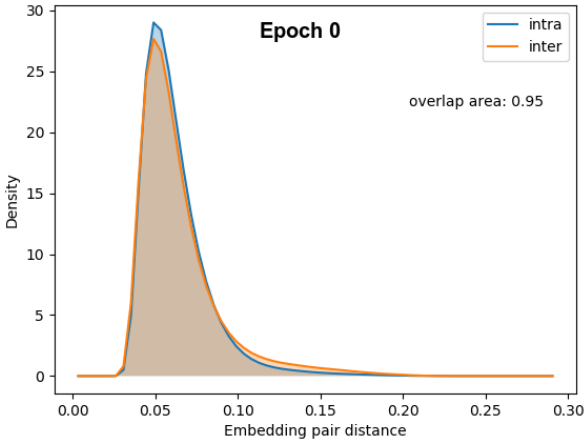
$$\begin{aligned} {d_{ij}}^2 &= (x_i - x_j)(x_i - x_j)^T \\ &= \|\mathbf{x_i}\|^2 - 2x_i{x_j}^T + \|\mathbf{x_j}\|^2 \\ &= 2 - 2s_{ij} \end{aligned} \tag{18}$$

Figure 6: Initial L2 embedding distance

(a) MLRSNet



(b) BigEarthNet



So the initial L2 margin for binomial loss in our work was $\sqrt{2-2*0.5} = 1$, however the initial L2 embedding distance of MLRSNet val set is between 0.02 and 0.07, which was far less than our binomial L2 margin, when the margin was fixed at a value which was way larger than all the negative pairs in the dataset, the binomial loss will push the distance of negative distance above 1.0. But we also find that all the intra distance also above 1.0, this is because these pairs were sampled by only considering one label per pair, this label-by-label sampling method ignored other co-existed labels of samples, thus many of the positive pairs were actually negative pairs if another category label was considered. Also, binomial loss penalized negative pairs way much harder than positive pairs, so in the end the distance of most pairs were pushed above 1.0, which resulted in very high intra-inter overlap (0.87). When this margin became trainable, the intra-inter overlap decreased to 0.80 (Fig 13 (b)). In order to further improve Bier's retrieval performance, an optimal initial binomial margin should be further explored for MLRSNet.

Fig 14 shows the density plot on BigEarthNet when the best setting was applied to each method. We find that Bier had a slightly bigger intra-inter overlap area (0.77) than Diva(0.71), however Bier had got a significantly better recall@1 (82%) than Diva (68%). This can be explained by the fact that metric learning doesn't care about the level of separation of the intra and inter group, as long as most of the samples from intra group have smaller distance than that from inter group, then a good recall@1 will be guaranteed, Bier pushed all the intra and inter distance to a span with very small variance, but still the inter group had a slightly bigger mean value than the intra group, which was enough to make most of intra samples correctly separate from the inter group. The intra and inter plot of Diva had a way bigger variance than Bier, though they had similar intra-inter overlap area, their retrieval performance differed significantly.

4.5. Ablation study

4.5.1 Margin β

We find the different settings of margin β make a very moderate influence on the retrieval performance of multi-task approaches when the backbone is frozen. Fig 7 shows recall@1 on MLRSNet test set with different β settings. Fig 7 (a) represents the case when the backbone was set frozen, the recall@1 scores of each method with different β settings are roughly the same around 55%.

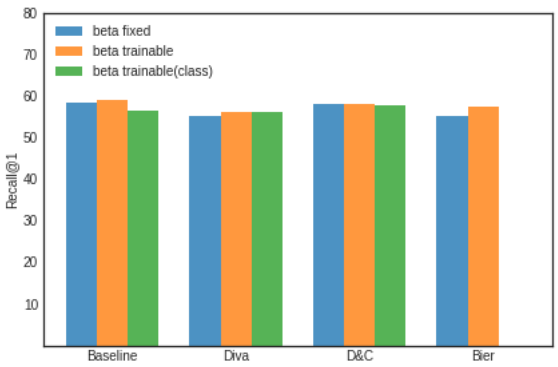
Fig 7 (b) represents the case when the backbone was set unfrozen, the recall@1 scores of each method become more different with different β setting, Baseline worked well on fixed β (blue), while Diva and Bier performed better on trainable β (orange), and D&C worked well on both fixed and trainable β settings. But it's not sure that trainable β is definitely better than fixed β because we didn't investigate different initial values for this margin value. The default margin value (1.2 for margin loss, 0.5 for bionmial loss) might not be an optimal choice for both β settings when the initial embedding distance of MLRSNetval set was between 0.02 to 0.06 in Fig 6.

4.5.2 Resnet50 frozen VS unfrozen

We find that the frozen option of the backbone played a big role in the retrieval performance of multi-task approaches. As it can be observed in Table 10, when the backbone is completely frozen (setting 1, 2 and 3), no matter which setting of the margin β it is, all three multi-task approaches yield similar recall@1 scores ranged from 55% to 58% on MLRSNet test set, however when they run with unfrozen

Figure 7: Fixed β VS Trainable β

(a) Recall@1 on MLRSNet test set
(Frozen backbone)



(b) Recall@1 on MLRSNet test set
(Unfrozen backbone)

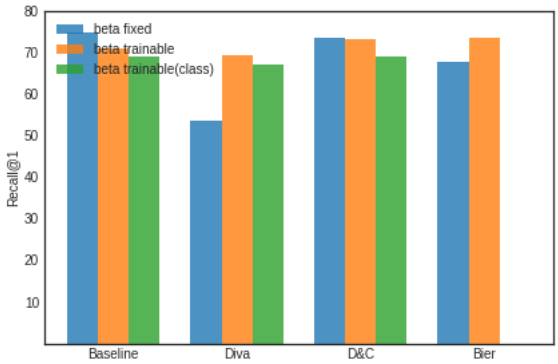
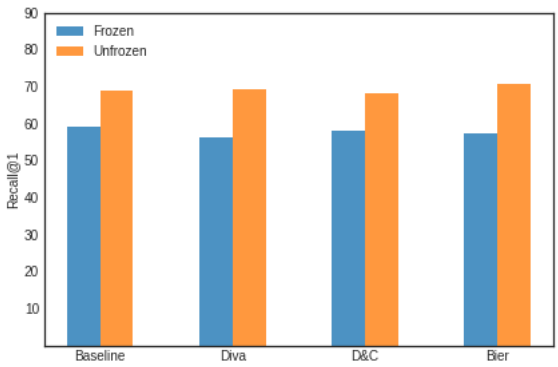
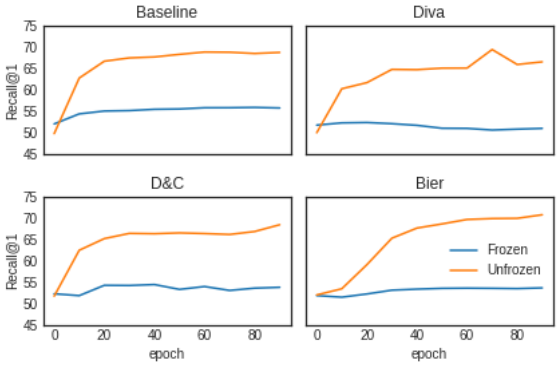


Figure 8: Frozen backbone VS Unfrozen backbone

(a) Recall@1 on MLRSNet test set
(β trainable)



(b) Recall@1 on MLRSNet test set
(β trainable)



backbone (setting 4, 5 and 6), the recall@1 scores of all the methods are greatly improved.
Fig 8 (a) shows the recall@1 on MLRSNet test set, it can be seen directly from the bar chart that the recall@1 of each method gains about 10% improvement when the backbone is unfrozen. Fig 8 (b) illustrates the change of recall@1 on MLRSNet val set during training, it is clear that frozen setting of the backbone stopped all the methods to improve its learning on embedding distance, because the recall@1 for each method didn't change much over time.

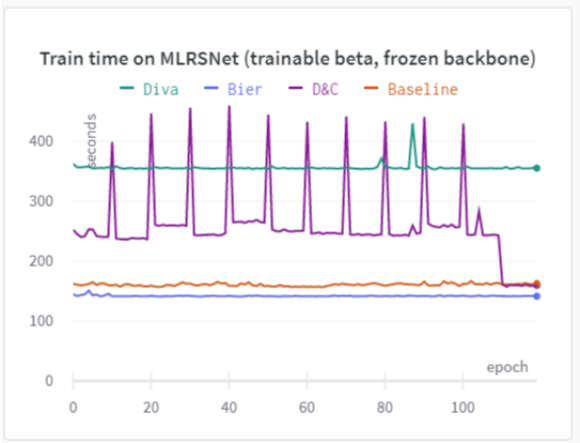
4.6. Training cost

When consider the training cost, Diva is the most expensive one among all the methods in this work. First it has a big number of hyper parameters for tuning, such the weights of four loss functions during training, the weights of each sub-embedding vectors during evaluation, and the weights of decorrelation between different type of embeddings; also it cost more time and computing resources than others, because it has another feature extractor (Resnet50) running in the memory to learn self-specific features. In a word a complex method like Diva takes a lot of time and computing resources to experiment out its best configurations, and it is not sure that Diva will outperform other simpler method. D&C has fewer hyper-parameters than Diva, but D&C also takes many training time because it re-clusters the embeddings of the whole train dataset every T epochs, we set T =10 in this work. Fig 9 shows the training time when each method run with the PC configuration: GeForce GTX 1080 8G and RAM 32G. Bier and Baseline had the least training time, while Diva and D&C almost took twice more time than Bier and Baseline.

5. Conclusion

In this work we studied three mulit-task multi-label retrieval methods on remote sensing dataset MLRSNet and BigEarthNet. We find that the pretrained frozen Resnet50 is very limited to learn features on remote sensing dataset,

Figure 9: Train time for each method



all the methods obtained roughly similar poor recall@1 on MLRSNet when the backbone was set to frozen, which made it hard to compare their retrieval performance.

Another important hyper parameter is the margin beta of margin loss and binomial loss, currently we find that Diva and Bier worked better with trainable margin, while D&C worked well with the default fixed margin, however this is not for sure because we haven't investigated different initial values for the margin beta, the current default initial value might not be an optimal choice for dataset MLRSNet and BigEarthNet.

We find all three multi-task approaches obtained roughly similar scores on MLRSNet, however Bier significantly outperformed Diva and D&C on BigEarthNet, and Diva had got the lowest retrieval score on both datasets. The simple margin baseline worked unexpectedly well on both datasets, it slightly outperformed all three multi-task methods on MLRSNet, and got a very similar retrieval score to Bier on BigEarthNet.

Another thing need to mention is that, all the multi-task methods in this work used a label-by-label sampling style, which ignored the coexistence of the other labels when selecting triplets. One consequence resulted from this label-by-label sampling style is that the embedding distance between retrieved images and query images were not well aligned with their semantic similarity. We should consider a triplet mining method which consider multiple labels jointly. Suppose the L2 distance between embedding x_i and x_j is d_{ij}, the similarity between its multi-hot labels is $s_{ij} = y_i y_j^T$, a valid triplet can be defined as: let y_a as the anchor, if $s_{ai} > s_{aj}$, then the positive sample is X_i, the negative sample is X_j, and only select hard triplets which has $d_{an} < d_{ap}$. The positive margin and negative margin for triplet loss should be trainable for this multi-label triplet mining.

Some other methods like SNDL [3] can also help to improve the alignment of semantic distance and embedding distance between retrieved images and query images. This method maintains an augmented memory bank of embedding during the training, at each epoch it calculates the distance similarity matrix and label similarity matrix between the mini-batch and the memory bank, the loss is calculated by aligning these two similarity matrix together, so that those pairs shared more category labels will be more close than those shared less category labels. The big drawback of this method is that the demand of GPU memory is linear with the train data size, which is a big challenge to apply it to remote sensing dataset.

It should be emphasized that, our current conclusion that multi-task methods didn't outperform simple single-task method (margin baseline), is based on our limited experiments in this work. For a more grounded comparison, other metric loss functions such as classic triplet and contrastive loss should be further experimented out in multi-task settings, also optimal margin value should be explored for each dataset.

References

[1] Schroff Florian, Kalenichenko Dmitry, and Philbin James. Facenet: A unified embedding for face recognition and clustering. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015. 3

[2] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv:1911.05722*, 2020. 3

[3] J. Kang, R. Fernandez-Beltran, D. Hong, J. Chanussot, and A. Plaza. Graph relation network: Modeling relations between scenes for multilabel remote-sensing image classification and retrieval. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–15, 2020. 11

[4] Timo Milbich, Karsten Roth, Homanga Bharadhwaj, Samarth Sinha, Yoshua Bengio, Björn Ommer, and Joseph Paul Cohen. Diva: Diverse visual feature aggregation for deep metric learning. *arXiv:2004.13458*, 2020. 1, 2

[5] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. 2020. 5

[6] M. Opitz, G. Waltner, H. Possegger, and H. Bischof. Deep Metric Learning with BIER: Boosting Independent Embeddings Robustly. *arXiv:cs/1801.04815*, 2018. 1, 2, 4

[7] Artsiom Sanakoyeu, Vadim Tschernezki, Uta Büchler, and Björn Ommer. Divide and conquer the embedding space for metric learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1, 3, 4

[8] G. Sumbul, M. Charfuelan, B. Demir, and V. Markl. Bigearthnet: A large-scale benchmark archive for remote sensing image understanding. *IEEE International Geoscience and Remote Sensing Symposium, Yokohama, Japan*, 2019. 5

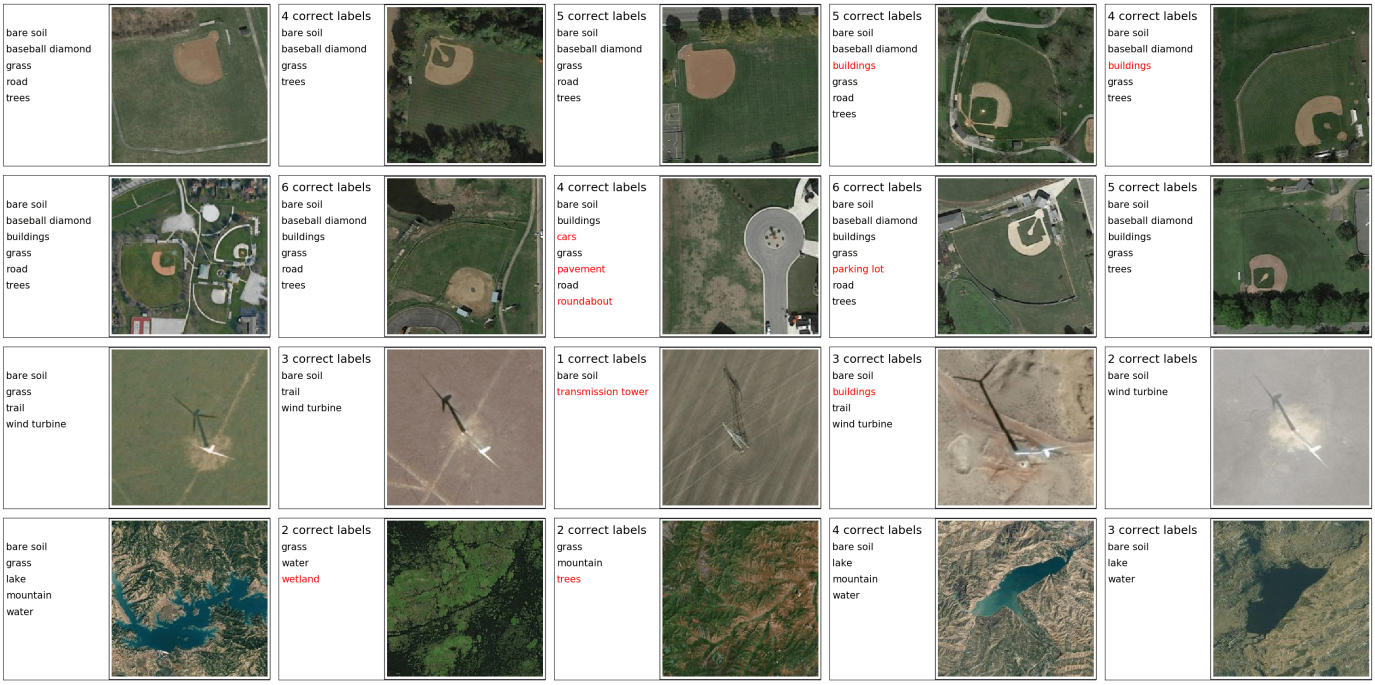
[9] Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krähenbühl. Sampling matters in deep embedding learning. *ICCV*, 2017. 1, 2, 3, 4

[10] Qi Xiaoman, Zhu Panpan, Wang Yuebin, Zhang Liqiang, Peng Junhuan, Wu Mengfan, Chen Jialong, Zhao Xudong,

Zang Ning, and Mathiopoulos P.Takis. Mlrsnet: A multi-label high spatial resolution remote sensing dataset for semantic scene understanding. *Mendeley Data, V2, doi: 10.17632/7j9bv9vwsx.2*, 2020. 6

Figure 10: Retrieved images on MLRSNet part1

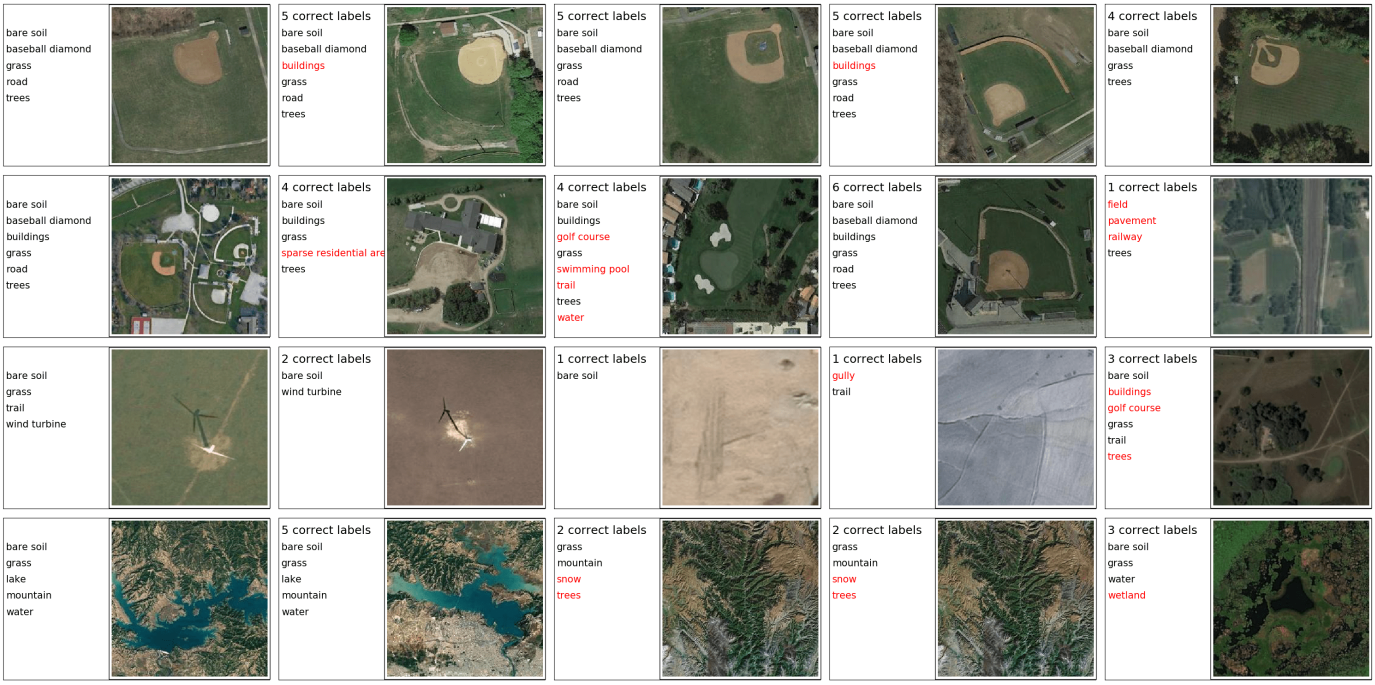
(a) Baseline



(b) Diva

Figure 11: Retrieved images on MLRSNet part2

(a) D&C



(b) Bier

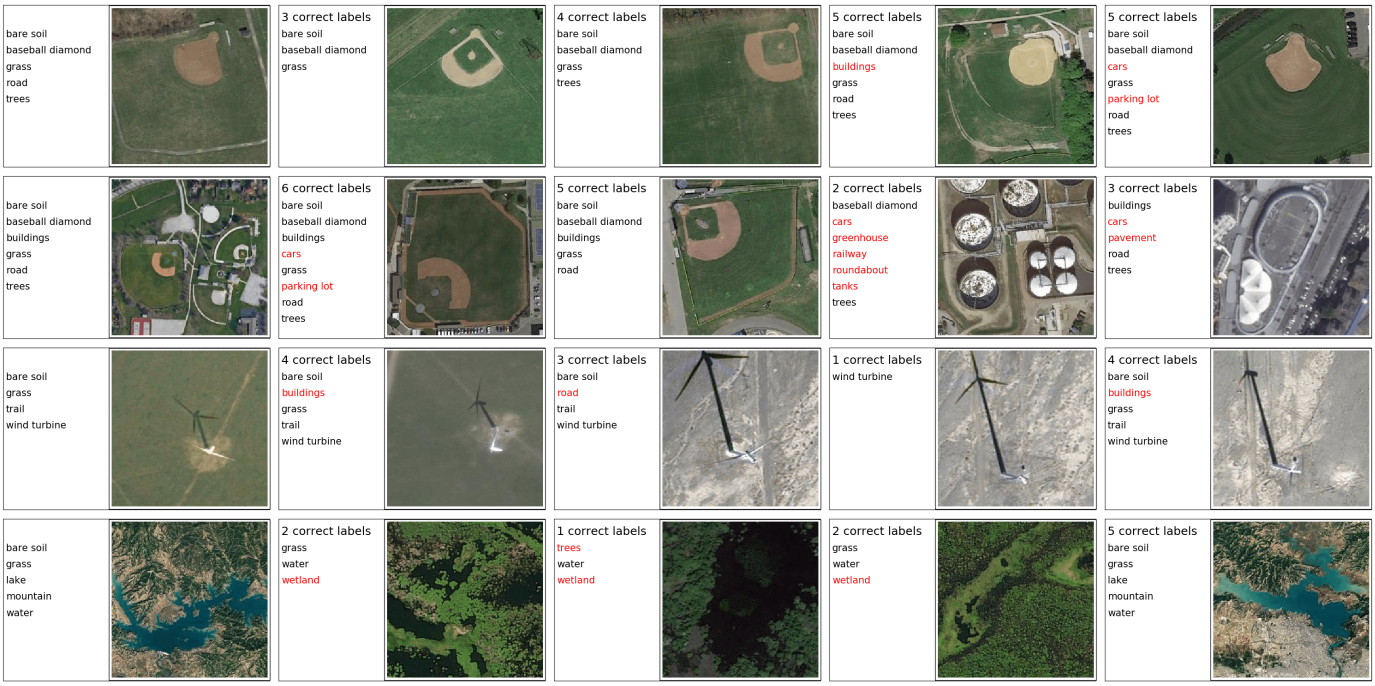


Figure 12: Embedding distance on MLRSNet val set (part1)

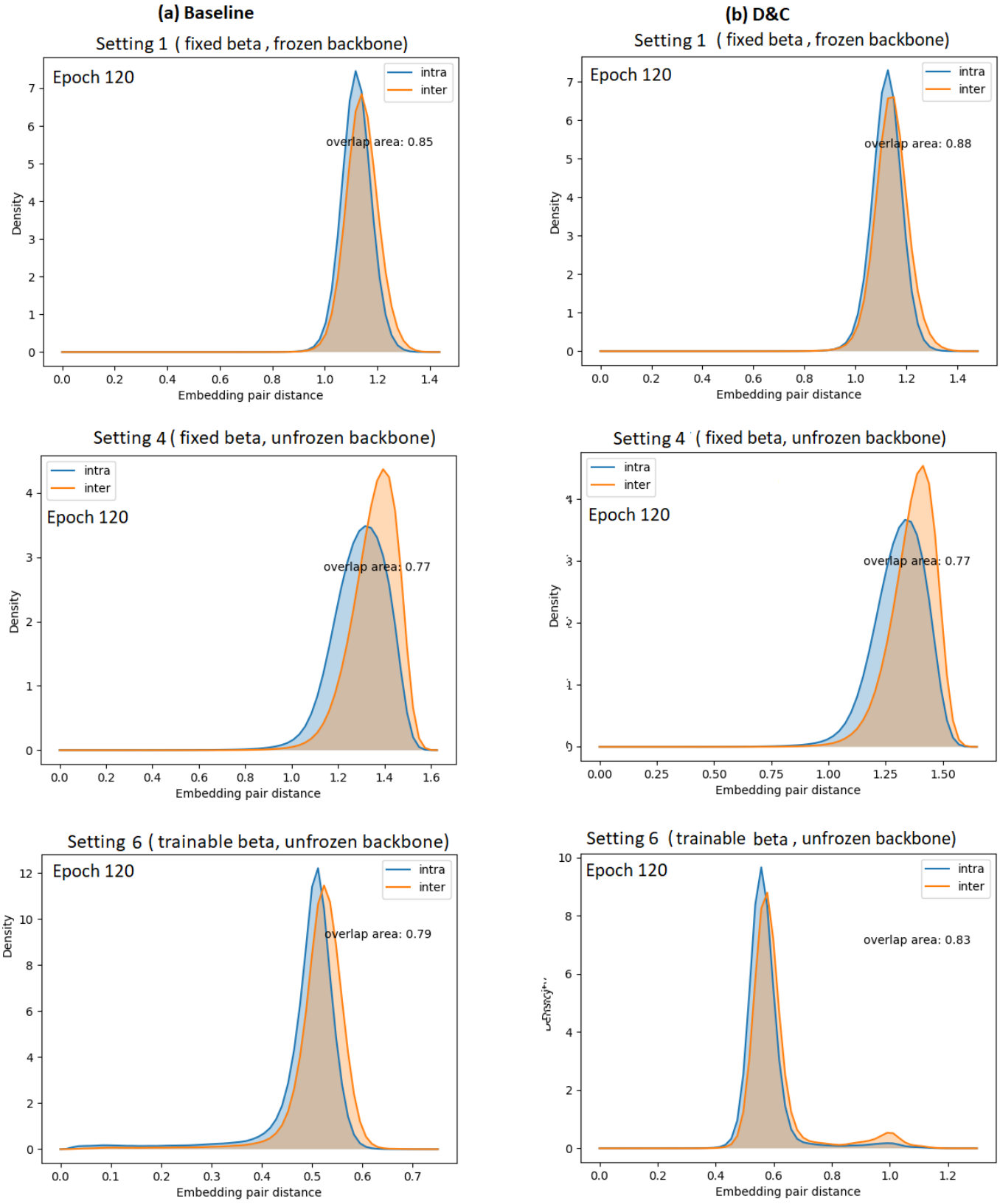


Figure 13: L2 embedding distance on MLRSNet val set (part2)

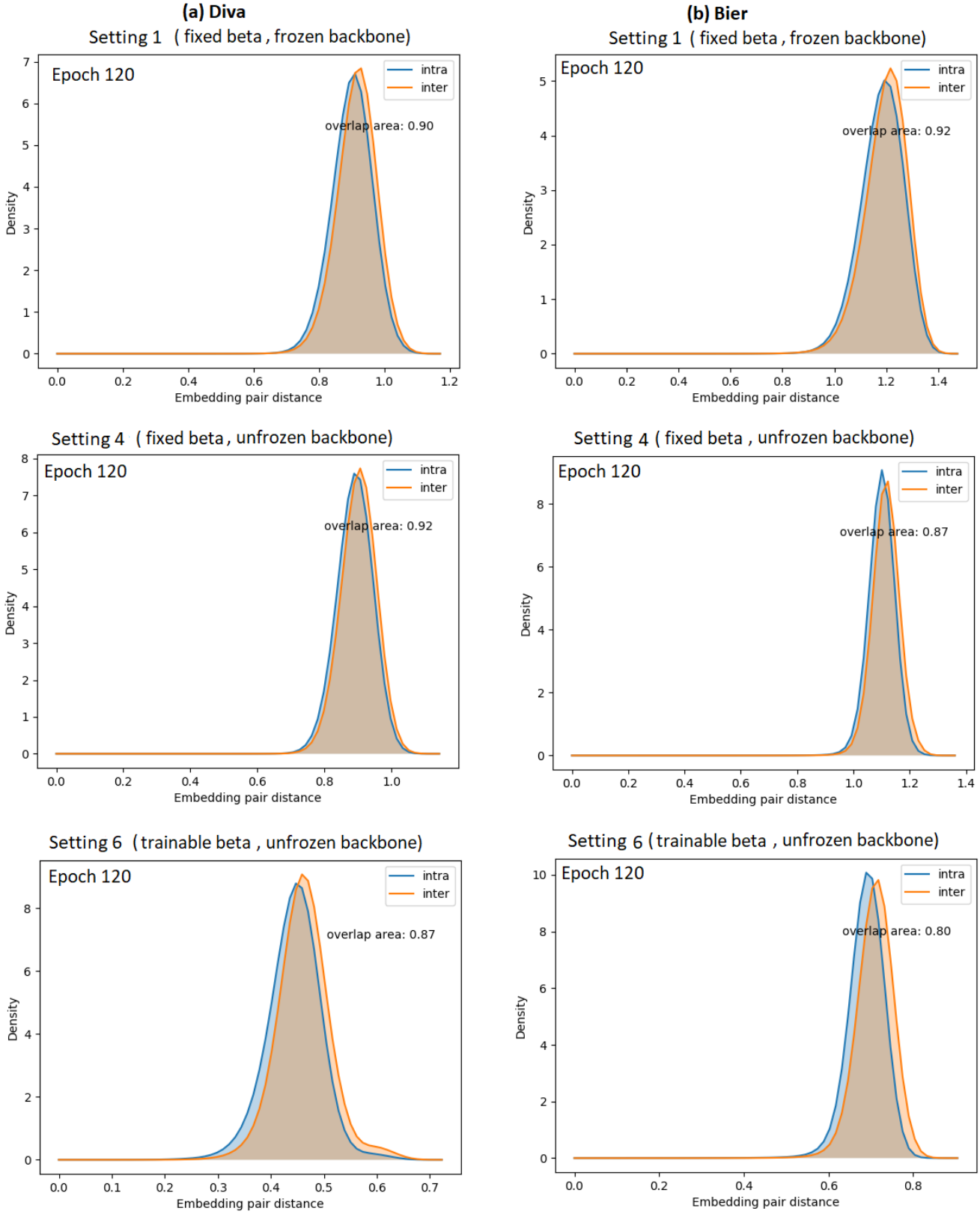


Figure 14: L2 embedding distance on BigEarthNet val set