remote sensing

MDPI

Article

Training Methods of Multi-Label Prediction Classifiers for Hyperspectral Remote Sensing Images

Salma Haidar [1,2,*], and José Oramas [1]

[1] Department of Computer Science, University of Antwerp, imec-IDLab, Sint-Pietersvliet 7, 2000 Antwerpen, Belgium

[2] Microtechnix BV, Anthonis de Jonghestraat 14a, 9100 Sint Niklas, Belgium

* Correspondence: salma.haidar@uantwerpen.be

1. Introduction

In this document, we introduce additional details and information about the work we conducted to facilitate the replication of the two-component network we designed and the training schemes we explored. In Section 2 we introduce all hyperparameters tuned for each scheme under the multi-label as well as the single-label classification task. In Section 3 we add the relevant hyperparameters we used to train the two methods from the literature that we reproduced and trained using the patches datasets we sampled. In Section 4, we provide insights into the inference processing time of our training schemes.

2. Hyperparameters for each training scheme

In this section, we present the relevant hyperparameters selected for training the network under each scheme we proposed. We selected the hyperparameters that generated the highest performance across the three schemes. Sections 2.1 and 2.2 relate to experiments under Sections (4.1) and (4.3) of the original paper, respectively. In them, We provide information that will allow reproducing the results achieved under the mentioned sections. Furthermore, under Section 2.3 we provide the mathematical representation of the evaluation metrics used in the multi-label classification context.

2.1. Multi-Label Classification: Performance Across Training Schemes

In this experiment we trained the multi-label classifier under the the three training schemes, *Iterative*, *Joint*, *Cascade*, using multi-labeled patches. We sampled those patches under the *multi-label sampling* approach as defined under Section (3.4) in the original paper. Table 1 presents the hyperparamters used for performing the experiment in Section 4.1. Those hyperparameters were used to train the two-component network for multi-label classification task following the three schemes, on patches sampled from PaviaU dataset. Table 2 presents the hyperparamters used for performing the experiment in Section (4.1) to train the two-component network for multi-label classification task under the three schemes, however using patches sampled from Salinas dataset.

Table S1. Multi-label classification: Hyperparameters adopted for the three training schemes using multi-labeled patches sampled from PaviaU dataset.

Parameter	***Iterative***	***Joint***	***Cascadet***
Batch size	200	200	200
Epochs-AE	95	–	95
Epochs-Clf	200	200	200
Iterative Epochs	20	NA	NA
Lr-AE	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$
Lr-Clf	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$
Lr-step_size AE	10	15	15
Lr-step_size Clf	15	15	15
Lr-schedulerγ	0.9	0.9	0.9
dropout _AE	0.3	0.3	0.3
dropout_Clf	0.6	0.6	0.6
L2-Regularization	0.0001	0.0001	0.0001

Table S2. Multi-label classification: Hyperparameters adopted for the three training schemes using multi-labeled patches sampled from Salinas dataset.

Parameter	***Iterative***	***Joint***	***Cascade***
Batch size	130	130	130
Epochs-AE	95	–	95
Epochs-Clf	200	200	240
Iterative Epochs	20	NA	NA
Lr-AE	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$
Lr-Clf	$1e^{-2}$	$1e^{-3}$	$1e^{-5}$
Lr-step_size AE	10	20	15
Lr-step_size Clf	15	20	15
Lr-scheduler γ	0.9	0.9	0.9
Dropout_Clf	0.6	0.6	0.6
L2-Regularization	0.0001	0.0001	0.0001

2.2. Single-Label Classification: Performance Across Training Schemes

Under this experiment, we trained the two-component network for the single-label classification task. For that purpose, we utilized single labeled patches sampled under the *Single label sampling* approach. Table 3 and Table 4 provide a summary of the hyperparameters tuned for the highest performance for each of the three training schemes, *Iterative*, *Joint*, *Cascade*.

Table S3. Single-label classification: Hyperparameters adopted for each training scheme using patches sampled from PaviaU dataset.

Parameter	***Iterative***	***Joint***	***Cascade***
Batch size	240	164	100
Epochs-AE	95	–	95
Epochs-Clf	260	200	240
Iterative Epochs	20	NA	NA
Lr-AE	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$
Lr-Clf	$1e^{-3}$	$1e^{-3}$	$1e^{-5}$
Lr-step_size AE	10	20	15
Lr-step_size Clf	10	20	15
Lr-scheduler γ	0.9	0.9	0.9
Dropout_Clf	0.6	0.6	0.6
L2-Regularization	0.0009	0.001	0.0001

Table S4. Single-label classification: Hyperparameters adopted for each training scheme using patches sampled from Salinas dataset.

Parameter	***Iterative***	***Joint***	***Cascade***
Batch size	240	200	200
Epochs-AE	95	–	95
Epochs-Clf	200	200	200
Iterative Epochs	20	NA	NA
Lr-AE	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
Lr-Clf	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
Lr-step_size AE	10	10	10
Lr-step_size Clf	10	10	10
Lr-scheduler γ	0.9	0.9	0.9
Dropout_Classifier	0.6	0.6	0.6
L2-Regularization	0.0009	0.0007	0.0009

2.3. Multi-label Classification: Evaluation Metrics.

The equations below define the evaluation metrics we calculated to evaluate the performance of the three training schemes, *Iterative*, *Joint*, *Cascade* within the multi-label classification task. For that we adopted the measures used in [44].

$$Hamming - Loss = \frac{1}{n, L} \sum_{i=1}^{n} \sum_{j=1}^{L} I(y_i^j \neq \hat{y_i}^j)$$

$$Accuracy = \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i \cap \hat{y_i}|}{|y_i \cup \hat{y_i}|}$$

$$Precision = \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i \cap \hat{y_i}|}{|y_i|}$$

$$Recall = \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i \cap \hat{y_i}|}{|\hat{y_i}|}$$

3. Comparison with existing work

We selected two relevant methods in the literature with the purpose of evaluating their performance, on the patches dataset we generated. Hence, we reproduced the same architecture of the two methods however, the training process takes into consideration the shape of our patches dataset. Since the new data differs from the original one used to train those those methods, we had to tune the hyperparameters to achieve a good results on our dataset.

3.1. Patches dataset of densely sampled pixels with neighbourhood

In the section we present the hyperparameters we used to train the method reproduced from [36]. As mentioned in the original paper, we reproduced the same architecture, however, due to lack of complete details of implementation we fine-tuned our own hyperparameters to train this network using our patches datasets from from PaviaU and Salinas. Table 5 presents the details.

Table S5. Hyperparameters for training the method developed by [36].

Parameter	***PaviaU***	***Salinas***
Batch size	100	164
Epochs	200	200
Lr	$1e^{-3}$	$1e^{-3}$
Lr-Scheduler	StepLR	ReduceLROnPlateau
Lr-step/patience	20	5
γ*/factor*	0.9	0.9
Weight_decay	0.09	0.09

3.2. Joint training using small sized dataset

In the section we present the hyperparamerts we used to train the method reproduced from [23]. The purpose was to reproduce their architecture and train it using the patches datasets we sampled. Although in their paper they do not test their method using Salinas datasets, however, we chose to perform an experiment using patches sampled from Salinas dataset since aim is to test the performance of such method on the data that we sampled and used to train our two-component network.

Table S6. Hyperparameters for training the method developed by [23].

Parameter	***PaviaU***	***Salinas***
Batch size	100	100
Epochs	200	200
Lr	$1e^{-2}$	$1e^{-2}$
Lr-step	10	20
γ	0.9	0.9
L2-Regularization	0.0001	0.0001

4. Time cost analysis

Table 7 and Table 8 provide details on the average inference time incurred for the processing of each example (patch). They cover the computation time of the model produced by each training scheme when applied to our test dataset under both PaviaU and Salinas datasets and for the two tasks: multi-label and single-label prediction.

Table S7. Inference Time (Average ± Standard Deviation) in Milliseconds for the three training schemes on the PaviaU and Salinas test datasets for the Multi-label Classification task

	Iterative	***Joint***	***Cascade***
PaviaU	1.199 ± 0.416	1.633 ± 0.512	1.505 ± 0.659
Salinas	1.273 ± 0.482	1.262 ± 0.428	0.930 ± 0.262

Table S8. Inference Time (Average ± Standard Deviation) in Milliseconds for the three training schemes on the PaviaU and Salinas test datasets for the Single-label Classification task

	Iterative	***Joint***	***Cascade***
PaviaU	0.520 ± 0.506	0.778 ± 0.451	0.642 ± 0.484
Salinas	0.532 ± 0.521	0.817 ± 0.456	0.557 ± 0.498

The processing times for each task and dataset, as detailed in the two tables above, highlight notable trends. In both datasets, inference times for multi-label tasks are consistently longer than those for single-label tasks. This suggests that multi-label tasks are more computationally demanding, likely due to their more complex decision boundaries and a greater number of output classes. Additionally, variations in the datasets, particularly in the number of classes they contain, further contribute to the increased complexity observed in multi-label predictions. In a multi-label classification problem with more classes, there will be more operations involved, leading to longer computation times. In conclusion, computational demands differ between training schemes and task types, with multi-label tasks requiring more time across both datasets. The standard deviation suggests that some training schemes are more consistent in their inference times than others. These conclusions highlight the importance of considering both the nature of the classification task (multi-label vs. single-label) and the characteristics of the dataset in terms of the number of classes to output when selecting a training scheme for hyperspectral image classification.