Ablation Studies for A Novel Unsupervised Approach for Training Convolutional DSSNs on Remote Sensing

João P K Ferreira, João P L Pinto, Júlia S Moura and Cristiano L Castro

I. Metrics

In our experiments, we evaluate different metrics for different purposes. The metrics reported in the main paper are quantitative and used to assess the actual performance of our model in a supervised manner. In contrast, we also obtained other qualitative metrics to evaluate if the obtained segmentation provides meaningful information.

A. Supervised Quantitative

Starting with the supervised quantitative metrics, we may first define some concepts before defining the metrics. Given a set of pixels, the predicted class is the one the classifier assigns, and the target class is the real ground truth label of the pixel. Therefore:

- a True Positive (TP) is predicted positive for a given class ($\hat{y} = 1$), and the target label is also positive for the same class ($y = 1$).
- a True Negative (TN) is predicted negative for a given class ($\hat{y} = 0$), and the target label is also negative for the same class ($y = 0$).
- a False Positive (TN) is predicted positive for a given class ($\hat{y} = 1$), but the target label is negative for the same class ($y = 0$).
- a False Negative (TN) is predicted negative for a given class ($\hat{y} = 0$), but the target label is positive for the same class ($y = 1$).

The first and most common metric is Accuracy (Acc). This metric measures the total number of TPs and TNs divided by the total number of pixels:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

The Precision (P) is a metric that measures the proportion of correct positive predictions. In other words, this indicates how many classified pixels are valid.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

The Recall (R), or Sensitivity, or True Positive Rate, is a metric that measures the proportion of actual positive predictions that were correctly identified. In other words, this indicates how many correct pixels were classified.

J. Ferreira, J. Pinto, J. Moura, and C. Castro are with the Graduate Program in Electrical Engineering, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil, e-mail: jpklock@ufmg.br

Manuscript received April 19, 2005; revised September 17, 2014.

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

The Specificity (S), or Selectivity, or True Negative Rate, is a metric that measures the proportion of actual negatives that was correctly identified. In other words, this indicates how many incorrect pixels were classified.

$$Specificity = \frac{TN}{FP + TN} \tag{4}$$

The F1-Score metric, or Dice Coefficient, is a harmonic mean of both the Precision and the Recall. Optimizing this metric means finding the best balance between FPs and FNs. And another way of thinking about this metric is the ratio between the area of intersection between two segmentations, where they are equal, divided by the total number of pixels of both images.

$$F1\text{-}Score = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2TP}{2TP + FP + FN} \tag{5}$$

The Intersection over Union (IoU) metric, or Jaccard Index, is given by the ratio between the area of overlap between two segmentations, where they are equal, divided by the union between the predicted segmentation and the ground truth. This is similar to the F1-Score, but the F1 considers the overlap area twice, once in each segmentation, whereas the IoU considers it only once.

$$IoU = \frac{TP}{TP + FP + FN} \tag{6}$$

These metrics so far are calculated individually for each class. So in a multiclass problem, this would result in several IoU calculations, one for each class. So, given C classes, the Mean IoU (mIoU) is the mean of the IoU for each class:

$$mIoU = \frac{1}{C} \sum_{c=1}^{C} IoU_c \tag{7}$$

This leads directly to the next metric, which is the Weighted Mean IoU (WmIoU), or the Mean IoU weighed by the contribution of each class. This weight can have multiple meanings, but it usually is the number of pixels of a given class in the ground truth divided by the total number of pixels.

$$WmIoU = \frac{1}{C} \sum_{c=1}^{C} w_c \cdot IoU_c \tag{8}$$

B. Segmentation Quantitative

The segmentation quantitative metrics are widely used to analyze the segmentation quality as a clustering problem, which is often useful in an unsupervised semantic segmentation context. We start with the Variation of Information (VI) metric [1]. This is usually used to evaluate clustering in general, but that has also been widely used to assess unsupervised semantic segmentation [2, 3, 4, 5, 6, 7]. It measures the distance between two segmentations regarding their average conditional entropy. In other words, it roughly measures the amount of randomness in one segmentation, which cannot be explained by the other. The equation is given by

$$VI(S_1, S_2) = H(S_1) + H(S_2) - 2I(S_1, S_2) \tag{9}$$

where $H(S)$ is the entropy of a given segmentation S, and $I(S_1, S_2)$ is the mutual information between segmentations S_1 and S_2.

In a practical perspective, given that a segmentation X is composed of several subsets, $X = \{X_1, X_2, \ldots, X_k\}$, where each X_i is a set of pixels composing a given partition of the complete segmentation. Equally, a second segmentation Y is $Y = \{Y_1, Y_2, \ldots, Y_l\}$ And suppose that $|X_i|$ is the number of elements inside a subset. We can define the total number of elements as

$$n = \sum_i |X_i|$$

Now consider the proportion of a single subset relative to the others as

$$p_i = \frac{|X_i|}{n} \text{ and } q_j = \frac{|Y_j|}{n}$$

And finally, consider the number of elements in common between a subset of both segmentations as $|X_i \cap Y_j|$. Its proportion related to the total is given by

$$r_{ij} = \frac{|X_i \cap Y_j|}{n}$$

The variation of information is then given by

$$VI(X, Y) = -\sum_{i,j} r_{ij} \left[log\left(\frac{r_{ij}}{p_i}\right) + log\left(\frac{r_{ij}}{q_j}\right) \right] \tag{10}$$

Larger values of this metric correspond to the more significant dissimilarity between the clusterings; therefore, lower VI values mean better segmentation results.

Another metric is the Probability Rand Index (PRI) [8], also used by a number of works [2, 3, 5, 6, 7]. This metric counts the fraction of pairs of pixels whose labelings are consistent between the computed segmentation and the ground truth, averaging across multiple ground truth segmentations. In other words, it measures the similarity between two data clusters.

The Rand Index can be understood as the number of agreements between two segmentations divided by the number of agreements summed with the number of disagreements. Given a set of n elements $S = o_1, \ldots, o_n$, and two partitions of S to compare, $X = X_1, \ldots, X_r$, a partition of S into r subsets, and $Y = Y_1, \ldots, Y_s$, a partition of S into s subsets, consider:

- a is the number of pairs of elements in S that are in the same subset in X and Y
- b is the number of pairs of elements in S that are in different subsets in X and Y
- c is the number of pairs of elements in S that are in the same subset in X and different subsets in Y
- d is the number of pairs of elements in S that are in different subsets in X and in the same subset in Y

The Rand Index is calculated by

$$R = \frac{a+b}{a+b+c+d} = \frac{a+b}{\frac{n}{2}} \tag{11}$$

where $a + b$ is the number of agreements between X and Y and $c + d$ is the number of disagreements between X and Y.

The Probabilistic Rand Index is the Rand Index averaged by every image, and a higher PRI value indicates a more accurate segmentation.

The Global Consistency Error (GCE) metric [9] is also used by some works [5, 6]. It measures the extent to which one segmentation can be viewed as a refinement of another. Segmentations related in this manner are considered consistent since they could represent the same natural image segmented at different scales.

Given that segmentation is a division of the pixels of an image into sets and that the segments are sets of pixels, if one segment is a proper subset of the other, then the pixel lies in an area of refinement, and the error should be zero. If there is no subset relationship, then the two regions overlap inconsistently.

Given two segmentations of the same size, S_1 and S_2, consider the measure of the error at each pixel p_i

$$E(S_1, S_2, p_i) = \frac{|R(S_1, p_i) \setminus R(S_2, p_i)|}{|R(S_1, p_i)|} \tag{12}$$

where $R(S_j, p_i)$ is the region in segmentation j that contains pixel p_i, $\setminus$ denotes set difference, and $|\cdot|$ denotes set cardinality. This measure evaluates to 0 if all the pixels in S_1 are also contained in S_2, thus achieving the tolerance to refinement discussed above. Given the error measures at each pixel, the error between two segmentations is defined as

$$GCE = \frac{1}{n}\min\left\{\sum_i E(S_1, S_2, p_i), \sum_i E(S_2, S_1, p_i)\right\} \tag{13}$$

For this metric, lower values mean closer segmentations.

Another metric is the Boundary Displacement Error (BDE) [10], also used to compare semantic segmentations [2, 5, 6]. It measures the average displacement error of boundary pixels between two segmented images. In other words, it defines the error of one boundary pixel as the distance between the pixel and the closest pixel in the other boundary image.

Given the boundaries of two segmentations, obtained by taking the absolute values of their gradients and checking for values above zero, a bipartite graph matching problem is solved by first constructing the distance matrix between

the boundary elements in each image, associating with the nearest pixel in the other boundary. Then, the percentage of matched edge elements is computed and taken as the boundary error. This error is calculated for both segmentations, and its mean will be the BDE value. The lower it is, the closer both segmentations are.

Finally, the last metric is the Segmentation Covering (SC) metric [11], also used by some works to compare segmentations [6, 7]. It measures the overlap of regions from the segmentation output and the ground truth.

The overlap between two regions R and R' is given by

$$\mathcal{O}(R, R') = \frac{|R \cap R|}{|R \cup R|}$$

where $|\cdot|$ is the area; in other words, it is the intersection over the union. The covering of a segmentation S by another segmentation S' is:

$$SC = \mathcal{C}(S \rightarrow S') = \frac{1}{N}|R| \cdot \max_{R' \in S'} \mathcal{O}(R, R') \qquad (14)$$

where N denotes the total number of pixels in the image. In other words, the area of every segment in S is multiplied by the highest overlap with the segments in S'. The sum for every segment is taken and pondered by the number of pixels. The higher the SC value, the better the quality of segmentation.

C. Metrics Summary

We comprise all metrics in Table I to facilitate the reading, organization, and posterior consultation. The table is composed of the name of the metrics, a brief description of what the metrics try to measure, and how to analyze the metric, where " $\uparrow$ " indicates that the higher the metric value is, the better is the segmentation result, and similarly, for " $\downarrow$ " the lower the metric value, the better is the segmentation.

II. Feature Extractor

Besides the DIC feature extractor from [6], we also tested two other feature extractors: Kan [12] and SEEK [4]. Before getting to the experiments, we briefly explain each feature extractor.

A. Kan

The first feature extractor tested is Kan from [12], and comprises three blocks; each block includes a Convolutional layer, a ReLU activation layer, and a Batch Normalization layer, except for the last block that does not have a ReLU layer. The constant K is the number of classes chosen, which the authors set to $K = 100$. A simplified version of their pipeline, highlighting the feature extractor, can be seen in Figure 1.

B. SEEK

The second feature extractor tested is derived from [4]. This feature extractor consists of a simple architecture, with an additional squeeze and excite block, similar to what is found in a MobileNet. Figure 2 shows a simplified view of the architecture. Each block comprises a Convolution Layer, a Batch Normalization Layer, and a ReLU layer. As for the Squeeze and Excite block, the activation function for the Linear 1 layer is ReLU, and for the Linear 2 layer is Sigmoid. Note that the block names are repeated twice; the authors use the same blocks twice to extract features and sum the contributions from each of the two forward passes through the blocks.

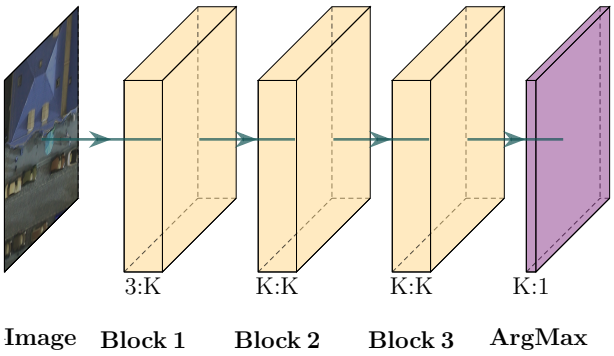

Fig. 1: Simplified pipeline from [12]. Each block comprises Convolutional, ReLU, and Batch Normalization concatenated layers.

C. DIC

The third architecture is derived from [6]. Although this architecture has already been deeply explained in Section II-A of the main paper, we replicate the general architecture in Figure 3. It mainly comprises a few blocks from U-Net, precisely a combination of the first pooling and final upsampling, becoming a simplified version of U-Net, for the first part. Each block comprises a Convolutional layer, a Batch Normalization Layer, and a ReLU layer, except Block 7, which does not have a ReLU activation. The DSC block explanation can be found in Section II-A of the main paper.

III. Experimental Setup

Our experiments aim to analyze different methodologies in literature by modifying the contents of every block in the common semantic segmentation pipeline, to understand in detail how each contributes to the network and what changes in them could lead to more complex semantic information learned. For that purpose, in this section, we train several networks from scratch, without any form of transfer learning, so that we can evaluate each of them individually, focused only on our desired tasks. After defining our metrics and feature extractors, we fix the following parameters as our default for each block :

- Preprocessing: the images are normalized between 0 and 1. Since data augmentation is random and we wanted to make the comparisons fair while analyzing the unbiased ability of the feature extractors to learn semantics, we disabled data augmentation for these experiments.
- Feature extractor: the three feature extractor blocks described above will be tested individually without any modifications.

TABLE I: Metrics used to evaluate semantic segmentation in this work.

Metric	What it measures	How to analyze
Accuracy	The percentage of correctly assigned pixels in a predicted segmentation	↑
IoU	The overlap between a target and predicted segmentation	↑
WIoU	The same as IoU, but weighted by the number of pixels in each class	↑
F1-Score	The correct pixels considering the number of wrong positives and wrong negatives	↑
VoI	The distance between the two segmentations in terms of their average conditional entropy	↓
PRI	The accuracy of labels assigned to pairs of pixels in segments of two segmentations	↑
GCE	The extent to which one segmentation can be viewed as a refinement of the other	↓
BDE	The average displacement error of boundary pixels between two segmented images	↓
SC	The overlap of regions of the segmentation output, and the regions of the ground truth	↑

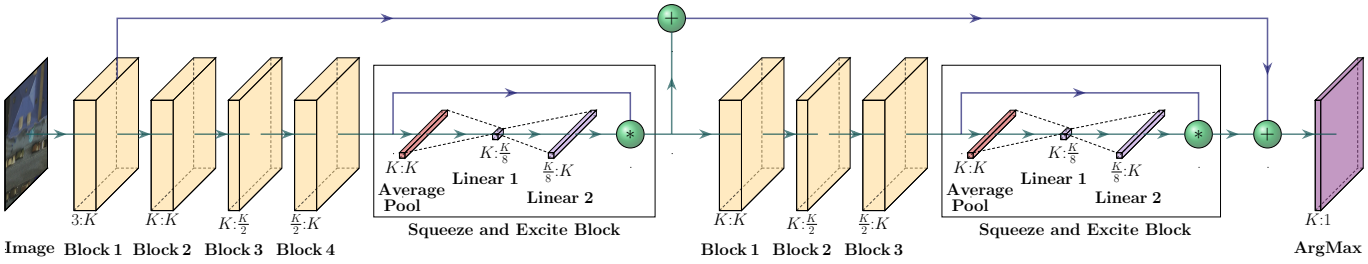

Fig. 2: Simplified pipeline from [4]. Each block comprises Convolutional, Batch Normalization, and ReLU concatenated layers.

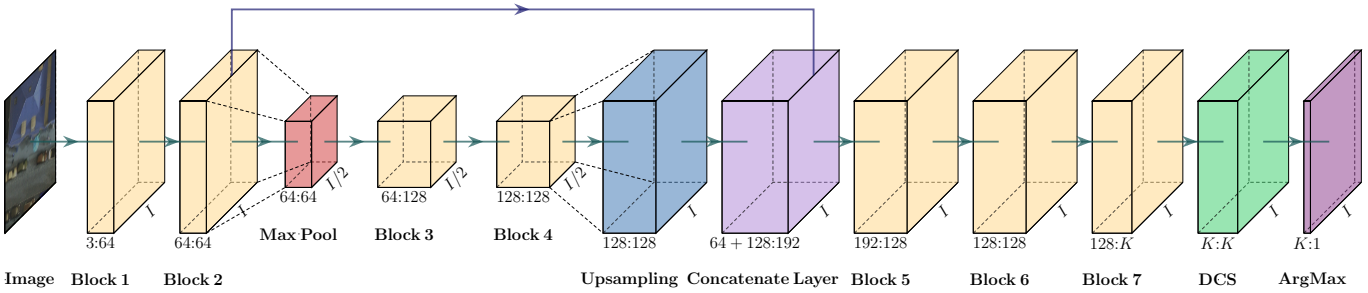

Fig. 3: Simplified pipeline from [6]. Each block comprises Convolutional, ReLU, and Batch Normalization concatenated layers.

- Classifier: a softmax layer with the proposed cross-entropy loss, weighted by the class weights of each batch.
- Class refinement: the output of the unsupervised semantic segmentation from [13] is used to guide the training, and the networks all have the number of classes set to $K = 100$. Although in our main paper we use superpixels, in our first experiments we intended to use a less accurate semantic segmentation, but we later found that using superpixels led to better results.

Due to computational resource limitations, we limit each experiment to 10 epochs and the batch size to 10 images. The batch size is chosen due to GPU limitations. As for the number of epochs, it was selected because we empirically observed that the networks mainly converged with this number of epochs, probably because of the high number of images, which will be shown next, leading to a faster generalization.

Moreover, we used the Potsdam dataset for every experiment, as explained in the main paper. For the training and testing division, according to the literature, we used the subset of the dataset provided by [14], where the authors used a total of 8550 images of 200×200 pixels, with 855 images for testing, with complete ground truth, and 7695 images for training, where 4545 also had a complete ground truth and 3150 images had no source of supervision. We perform our tests on both variations of the dataset, with 6 and 3 classes.

IV. Learning Rate

The first experiment we performed was regarding the learning rate (lr) we would use to train our proposed network in future experiments. This is important to check whether our chosen feature extractors could learn properly during training in the remaining experiments. We decided to use an exponential decaying learning rate from the beginning of the training until the end so that the network could perform smaller steps toward the gradient as the clustering process reached convergence. Three experiments were performed:

- 1 - $lr = 1e^{-3}$ to $lr = 1e^{-4}$
- 2 - $lr = 1e^{-4}$ to $lr = 1e^{-5}$
- 3 - $lr = 1e^{-5}$ to $lr = 1e^{-6}$

We start by checking the loss curves for each model training. Since this is an unsupervised problem, in practice, we would not have any supervision to compare; therefore, we only evaluate the loss curve of the training set in this parameter optimization experiment, and skip the supervised metrics curves.

An example of the loss curves generated during training can be seen in Figure 4. Since our batch size equals 10 and the dataset has 7550 images, each epoch will have 755

batches. And since we train for 10 epochs, the total number of batches for each training is 7550. In this figure, as can be seen in the upper plots, the actual losses returned from the network contain a high amount of noise, which is caused both because the training is unsupervised, therefore subject to noise, and also because of the weighted cross entropy which for every batch assigns distinct weights to different classes. To alleviate this noise in the visualization, we use a moving average filter with a window size of 100 observations, resulting in the bottom plots, which will be the default visualization for the remainder of this work.

The loss curves for each experiment are aggregated in Figure 5, where the results for experiment 1 are replicated for convenience.

From the loss curves, we can see that both Kanezaki and SEEK converged fast in experiment 1, indicating that the learning rate might be too high. As for experiment 3, the losses suggest that no algorithm is learning correctly. Therefore, although the algorithms might not have entirely achieved convergence in experiment 2 due to its smoothness, this experiment seems to have achieved the best result.

We also evaluate our networks' metrics by dividing the experiments for each feature extractor. The results for the test dataset can be seen in Table II, where "Exp" stands for the experiment number, "FE" stands for "Feature Extractor," and the remaining columns are the abbreviation of each metric, as described in Section I. The "Kanezaki" feature extractor was also abbreviated for "Kan." For every feature extraction, there are two results, one for the complete dataset with all six classes and another where the name of the feature extractor has a "3", meaning that it is the variation of the dataset with only three classes. In this table, the IoU and wIoU metrics are the mean over all the classes, and the segmentation quantitative metrics (VoI, PRI, GCE, BDE, and SC) are all the mean of the metric value for every image in the test dataset. We also separate the types of metrics with a thicker vertical line between $F1$ and VoI, where the left half represents the supervised metrics, and the right half represents the segmentation metrics.

As the results show, experiment 2 obtained the best results overall. In experiment 3, DIC got the best Acc for the six classes experiment, but its IoU, wIoU, and F1-Score were better in experiment 2. Checking the other metrics, we can see that VoI, PRI, and GCE are better for experiment 1 in general, while GCE and SC are better for experiment 2.

Given these results, we chose to follow the experiments with our learning rate decaying from $1e^{-4}$ to $1e^{-5}$ since it obtained the more consistent supervised metrics for every feature extractor, competitive segmentation metrics compared to the other experiments, and it also presented a smooth loss function curve.

V. Number of classes

In our proposed architectures, we used several classes, or clusters, of $K = 100$ since having a large number of classes should theoretically allow the network to learn a wider variety of semantics. But we also tested this hypothesis by training our network with a lower number of classes, $K = 10$, to investigate the effects of having a smaller number of classes for the network to learn. Here we used only the best learning rate achieved in the previous experiment. The loss curves can be seen in Figure 6.

Clustering images with a lot of content variation, which is the case for remote sensing images, is a challenging problem. The loss functions show that the network had more difficulty learning to segment using only 10 classes since the loss values are higher and the curves are less steep. The curves show that a higher number of classes allows the network to distribute the semantics between more classes and learn more specific semantic features. Next, in Table III, we evaluate the metrics for this experiment, where "K" is the number of classes.

Overall, many segmentation quality metrics are better for the 10-class problem. This is indicative that even though the classes learn more specific semantics in the case with more classes, when mapping this information to the particular classes of the dataset, part of the information is lost, meaning that the usage of the classes without the mapping might make more sense.

For the supervised metrics on the left side of the table, the 10 classes experiment obtained a better Acc. Still, the IoU, wIoU, and F1-score all were better for the 100-class network, which is more important in a semantic segmentation problem than just accuracy. With these results, we choose to stay with 100 classes in our network.

VI. Cross-Entropy

The third experiment that we performed was regarding cross-entropy loss. We proposed using a weighted version of this loss, where the weights are recalculated at every batch. But this causes the loss function to become very unstable between batches, as seen in Figure 4. To explore this problem, we trained our network with our best learning rate and 100 classes, varying between our loss function and the original cross-entropy, and compared the results. The loss function plots can be seen in Figure 7, where "LF" stands for "Loss Function."

As we can see, both networks trained similarly, but the original cross-entropy loss was smoother than our proposed loss. Here the values cannot be compared since they are in different scales. Next, in Table IV, we evaluate the metrics for each loss.

The supervised metrics on the left side of the table show that the only case where the CE performed better was the accuracy of the six classes dataset, by 0.1%. The remaining metrics were all the better for the WCE loss, and here we highlight an increase of 4.5% in the IoU value. As for the segmentation metrics on the right side of the table, results vary between methods. The CE method obtained a slightly better SC value in both experiments, indicating that some regions overlap better for this loss, but this could be the case because since there are no class weights, the larger classes might be favored, and if these contain the largest homogeneous regions in the dataset images, it makes sense that the metric evaluating the region overlap would have a better value. For the remaining segmentation metrics, most obtained a better result for the

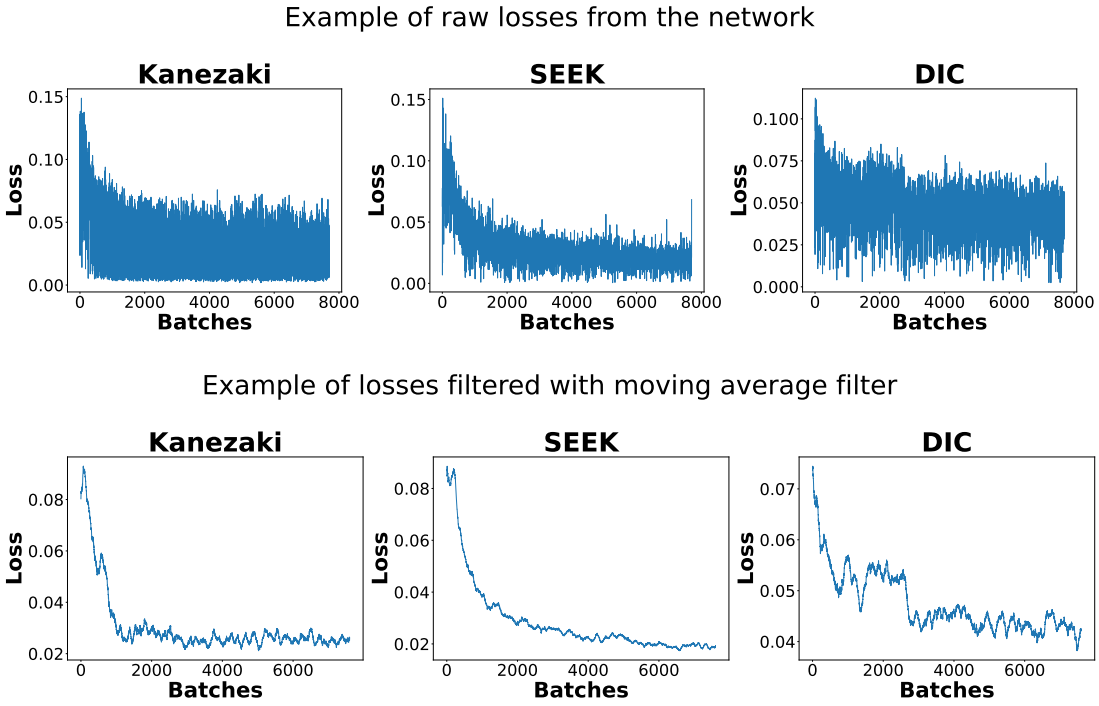

Fig. 4: Example of losses from an average training.

TABLE II: Metrics for the learning rate experiment.

Exp	FE	Acc↑	IoU↑	wIoU↑	F1↑	VoI↓	PRI↑	GCE↓	BDE↓	SC↑
1	SEEK	47.9	18.6	28.4	27.3	**1.50**	0.634	**0.167**	18.0	0.504
	DIC	59.3	28.5	41.8	39.4	1.65	**0.703**	0.260	11.8	0.538
	Kan	57.6	28.6	39.0	40.6	1.70	0.691	0.254	11.9	0.547
2	SEEK	58.6	24.6	38.6	33.0	1.58	0.678	0.217	12.6	**0.567**
	DIC	59.4	**32.4**	**43.2**	**45.7**	1.84	0.698	0.293	**11.1**	0.548
	Kan	58.4	26.8	40.0	37.3	1.83	0.686	0.293	11.2	0.535
3	SEEK	54.6	22.0	34.4	30.5	1.68	0.663	0.239	12.6	0.544
	DIC	**60.0**	30.2	42.2	42.3	1.76	0.698	0.274	11.8	0.558
	Kan	57.3	23.7	37.2	32.2	1.76	0.676	0.273	11.4	0.537
1	SEEK3	63.3	44.1	45.3	60.2	1.16	0.704	**0.159**	20.2	0.654
	DIC3	72.5	55.4	56.3	70.8	**1.15**	**0.748**	0.173	15.7	0.677
	Kan3	71.6	54.7	55.8	70.0	1.35	0.724	0.214	14.7	0.664
2	SEEK3	72.8	56.4	57.4	71.6	1.27	0.733	0.202	14.5	**0.688**
	DIC3	**74.1**	**58.4**	**59.4**	73.2	1.27	0.740	0.203	**13.8**	0.679
	Kan3	72.7	56.4	57.2	71.7	1.40	0.717	0.225	14.8	0.667
3	SEEK3	69.2	51.5	52.6	67.1	1.35	0.714	0.210	15.2	0.669
	DIC3	73.9	58.3	59.2	**73.3**	1.32	0.725	0.211	15.0	0.681
	Kan3	71.3	54.7	55.6	70.3	1.48	0.700	0.235	15.0	0.657

TABLE III: Metrics for the class number experiment.

K	FE	Acc↑	IoU↑	wIoU↑	F1↑	VoI↓	PRI↑	GCE↓	BDE↓	SC↑
100	SEEK	58.6	24.6	38.6	33.0	1.58	0.678	0.217	12.6	0.567
	DIC	59.4	**32.4**	**43.2**	**45.7**	1.84	0.698	0.293	**11.1**	0.548
	Kan	58.4	26.8	40.0	37.3	1.83	0.686	0.293	11.2	0.535
10	SEEK	50.2	19.5	30.2	28.0	1.70	0.652	0.243	13.0	0.545
	DIC	**59.9**	25.6	40.1	33.8	**1.41**	**0.709**	**0.186**	14.1	**0.585**
	Kan	54.4	21.9	34.3	30.4	1.72	0.659	0.254	12.9	0.549
100	SEEK3	72.8	56.4	57.4	71.6	1.27	0.733	0.202	14.5	0.688
	DIC3	74.1	**58.4**	**59.4**	**73.2**	1.27	**0.740**	0.203	**13.8**	0.679
	Kan3	72.7	56.4	57.2	71.7	1.40	0.717	0.225	14.8	0.667
10	SEEK3	65.8	47.6	48.9	63.4	1.38	0.702	0.213	15.3	0.652
	DIC3	**74.5**	58.1	59.1	73.0	**1.12**	0.765	**0.173**	14.9	**0.703**
	Kan3	68.6	51.4	52.4	67.3	1.42	0.699	0.219	15.4	0.659

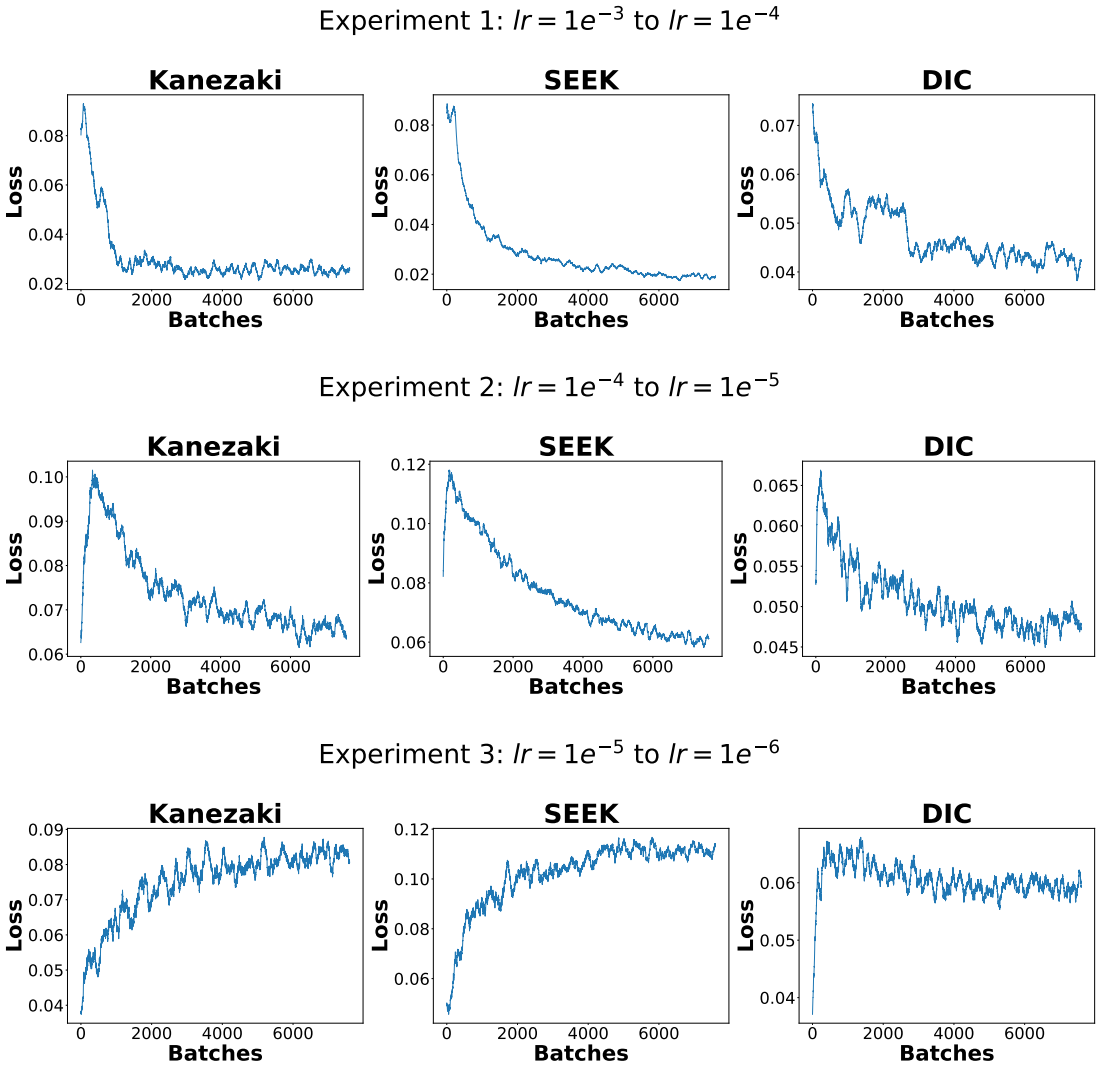

Fig. 5: Losses for the learning rate experiment.

TABLE IV: Metrics for the cross-entropy experiment.

LF	FE	Acc↑	IoU↑	wIoU↑	F1↑	VoI↓	PRI↑	GCE↓	BDE↓	SC↑
WCE	SEEK	58.6	24.6	38.6	33.0	**1.58**	0.678	**0.217**	12.6	0.567
	DIC	59.4	**32.4**	**43.2**	**45.7**	1.84	0.698	0.293	**11.1**	0.548
	Kan	58.4	26.8	40.0	37.3	1.83	0.686	0.293	11.2	0.535
CE	SEEK	50.3	19.3	29.7	27.5	1.67	0.658	0.234	12.9	0.543
	DIC	**59.5**	27.9	41.3	38.7	1.61	**0.708**	0.254	11.8	0.549
	Kan	57.2	23.8	37.3	32.2	**1.58**	0.683	0.223	12.8	**0.579**
WCE	SEEK3	72.8	56.4	57.4	71.6	1.27	0.733	0.202	14.5	0.688
	DIC3	**74.1**	**58.4**	**59.4**	**73.2**	1.27	0.740	0.203	**13.8**	0.679
	Kan3	72.7	56.4	57.2	71.7	1.40	0.717	0.225	14.8	0.667
CE	SEEK3	65.1	45.4	46.7	60.1	**1.15**	0.736	**0.169**	14.6	0.692
	DIC3	72.9	56.5	57.5	71.5	1.17	**0.752**	0.181	14.6	**0.700**
	Kan3	71.5	54.4	55.3	69.9	1.29	0.731	0.202	15.2	0.683

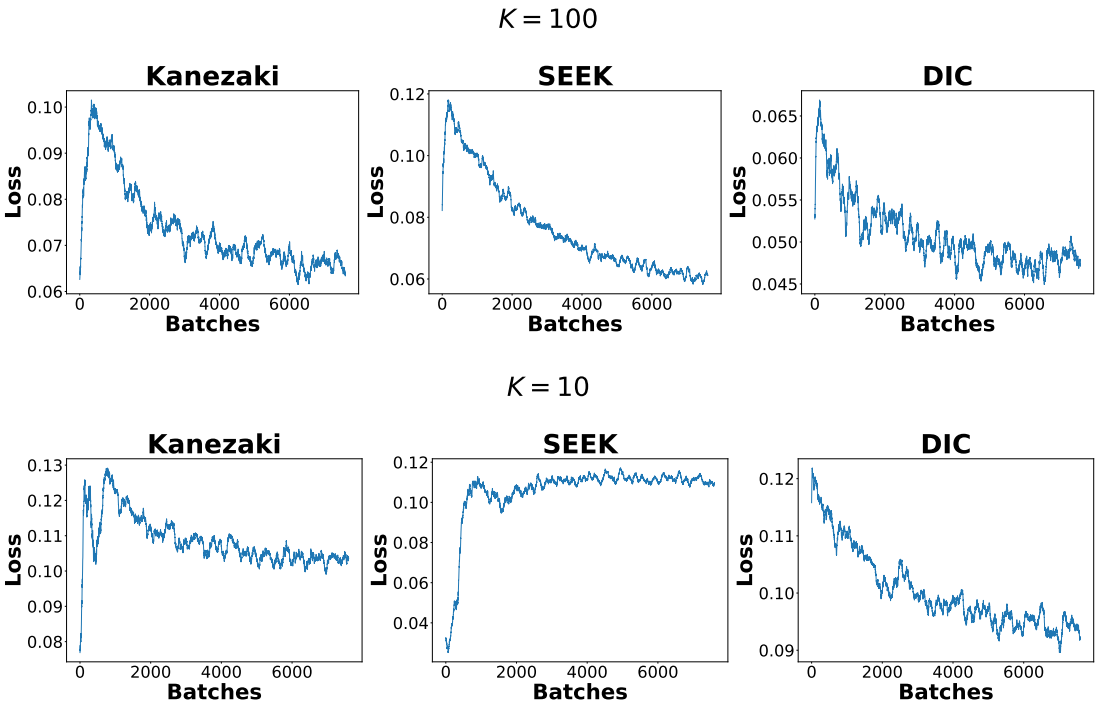

Fig. 6: Losses for the class number experiment.

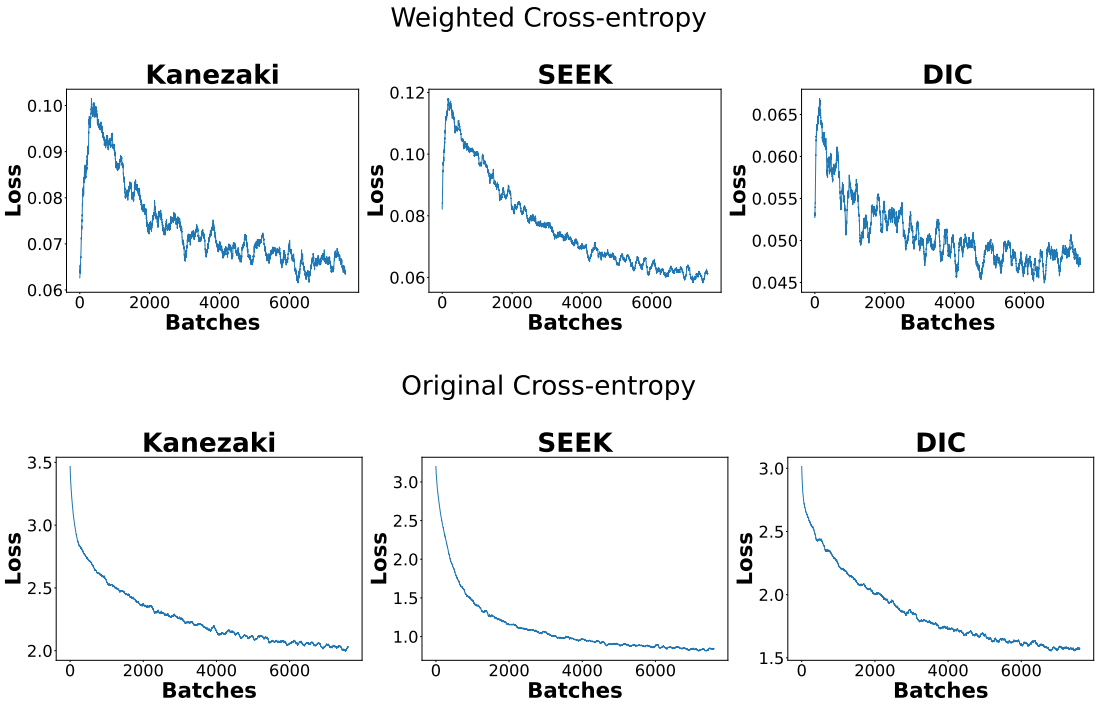

Fig. 7: Losses for the cross-entropy experiment.

CE loss in the three classes dataset. Since the classes are better balanced in this case, the impact of favoring smaller classes is lower, explaining why the segmentation metrics are better. Still, the supervised metrics indicate that the WCE loss performed better.

In general, this experiment proves that using a class-weighted loss approach can favor the smaller classes, and since the training is unsupervised, balancing the weights is an essential way of regularizing the learning and avoiding that larger classes overfit the network by assigning every pixel to them, mainly at the beginning of the training.

VII. Feature Extractor Comparison

With our previous experiments, we also settle for the DIC feature extractor, since at this point there is enough information, when comparing all of the previous experiments' metrics tables, that the DIC feature extractor has the best overall results and can generalize better than the other classifiers, probably because of both the more extensive network and the DCS block.

For simplicity, we aggregate the metric results for the Cross-Entropy experiment in Table V, where "FE" stands for "Feature Extractor," and the remaining columns are the abbreviation of each metric, as described in Section I. The "Kanezaki" feature extractor was also abbreviated for "Kan".

As the results show, the DIC feature extractor obtained the best supervised metrics for both datasets, as depicted on the left side of the table. Checking the segmentation metrics, we can see that VoI, GCE, and SC are better for SEEK, while PRI and BDE are better for DIC.

In a more in-depth analysis of the segmentation metrics only (right half of the table), the metrics highlighted for SEEK mostly show that the segmentations achieved are closer to the ground truth in their shape. As for DIC, the highlighted metrics indicate that the segmentation is closer to the ground truth in content.

Given these results, we chose to use the DIC feature extractor since it obtained the most consistent supervised and competitive segmentation metrics compared to the others.

VIII. Second Classifier

With the previous experiment, we had already decided on our feature extractor, but we decided to test another idea that could potentially improve our methodology. Many different works use two classifiers during learning, either to maximize their mutual information [14, 15] or as a form of data augmentation and regularization, to learn multiple representations of an image at the same time [16]. Either way, learning with two classifiers simultaneously can be beneficial and create the possibility of exploiting this information as a loss function since both would theoretically be learning to segment the same images.

With this idea in mind, we modified our feature extractors to allow two classifiers in parallel. In practice, our solution is close to [14], except that he uses a different number of classes for each classifier while we set the same number for both.

To add the second classifier, we modify the DIC feature extractor, shown in Figure 3, by adding a replicate of Block 7, the DSC block, and the ArgMax layer, working in parallel with the original blocks. The output of Block 6 goes to both the original blocks and the replicate blocks simultaneously, and accordingly, two classifications can be performed simultaneously.

We first show in Figure 8 the loss curves for both classifiers for the training, where "C1" stands for "Classifier 1" and "C2" for "Classifier 2". We used the optimal learning rate and loss function defined in the previous experiments.

As expected, every loss function achieves a similar loss and shape at the end of the training, even though they start differently due to the weights initialization, showing that all classifiers are learning correctly and indicating that they are learning the same segmentation, which can only be proved when comparing the actual segmentations. Then we show the metrics results in Table VI, where "NC" stands for "Number of Classifiers." In the Feature Extractor column, the "_1" indicates the first classifier, and equally, "_2" is the second classifier.

Overall, the experiment with two classifiers obtained a better result in almost every metric, with increases of about 1.5% for the accuracies and similar increases for every other supervised metric. The only metrics with a better outcome for the one classifier experiment were the F1-score and SC for the six-class dataset and BDE for the three-class dataset. However, the results are still close, with none surpassing differences of 0.2%.

Additionally, if we analyze the results of both classifiers of the second experiment, they are almost equal in every metric, with slight variations. This confirms that they are both learning the same segmentation and that it is coherent with what we expected. As for how they are learning this segmentation, this could vary for each classifier. Since it is an unsupervised problem, the cluster might differ from each classifier in practice since the metrics only evaluate the results after the softmax activation. This can only be assessed when checking the actual segmentations, which will be performed later.

IX. Pixel Classifier

After defining the optimal feature extractor for our network, along with several hyperparameters, we test the addition of other losses to our proposed weighted cross-entropy loss (L_{wce}) to check if it is possible to further enhance the semantic extraction with additional constraints to the training.

The first loss we propose to use is the spatial continuity loss (L1 loss), presented by [17]. Since clustering primarily aims to group pixels, creating homogeneous regions in the segmentation mask, the network output should benefit from being spatially continuous, forcing the cluster labels to be similar for neighbor pixels. For that, the authors use the L1-norm of the horizontal and vertical differences of the response map (r) or the logits of the network classification before the softmax activation. The loss is given by

TABLE V: Metrics obtained for different feature extractors.

FE	Acc↑	IoU↑	wIoU↑	F1↑	VoI↓	PRI↑	GCE↓	BDE↓	SC↑
SEEK	58.6	24.6	38.6	33.0	**1.58**	0.678	**0.217**	12.6	**0.567**
DIC	**59.4**	**32.4**	**43.2**	**45.7**	1.84	**0.698**	0.293	**11.1**	0.548
Kan	58.4	26.8	40.0	37.3	1.83	0.686	0.293	11.2	0.535
SEEK3	72.8	56.4	57.4	71.6	**1.27**	0.733	**0.202**	14.5	**0.688**
DIC3	**74.1**	**58.4**	**59.4**	**73.2**	**1.27**	**0.740**	0.203	**13.8**	0.679
Kan3	72.7	56.4	57.2	71.7	1.40	0.717	0.225	14.8	0.667

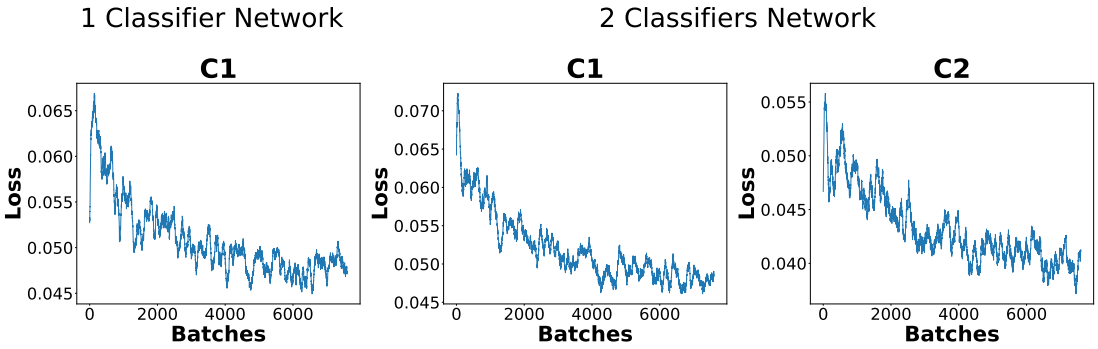

Fig. 8: Losses for the multiple classifiers experiment.

TABLE VI: Metrics for the multiple classifiers experiment.

NC	FE	Acc↑	IoU↑	wIoU↑	F1↑	VoI↓	PRI↑	GCE↓	BDE↓	SC↑
1C	DIC_1	59.4	32.4	43.2	**45.7**	1.84	0.698	0.293	11.1	**0.548**
2C	DIC_1	**60.9**	32.5	44.1	45.5	**1.75**	**0.708**	**0.281**	11.4	0.541
	DIC_2	**60.9**	**32.6**	**44.4**	45.5	1.79	0.706	0.286	**11.0**	0.547
1C	DIC3_1	74.1	58.4	59.4	73.2	1.27	0.740	0.203	**13.8**	0.679
2C	DIC3_1	**75.7**	**60.3**	**61.2**	**74.8**	**1.19**	**0.754**	**0.190**	14.0	**0.693**
	DIC3_2	**75.7**	**60.3**	**61.2**	**74.8**	1.23	0.748	0.196	14.6	**0.693**

$$L_{con} = \frac{1}{K\cdot(H-1)\cdot(W-1)} \sum_{n=1}^{K} \sum_{i=1}^{(H-1)(W-1)} \sum_{j=1} \|r_{n,i+1,j} - r_{n,i,j}\|_1 + \|r_{n,i,j+1} - r_{n,i,j}\|_1 \tag{15}$$

where the $\|\cdot\|$ is the norm-1, or simply the absolute value of the difference, $r_{n,i,j}$ is position (i, j) of the n-th response map, K is the number of clusters, or classes, H is the image height and W is the image width.

The second loss we implemented is the Mumford-Shah loss (MS loss), proposed by [3]. Their loss is based on the Mumford-Shah functional and tries to minimize the energy variation within the pixels in each cluster. In practice, for each of the K classes, a centroid is calculated using its probability map applied to the color image. Then each pixel is subtracted from the color of its centroid, and this difference is minimized. Along with this minimization, the authors also add a constraint to enhance the bias field continuity, that is, the L1-norm of the horizontal and vertical differences of the activations of the response map. This is almost the same as the spatial continuity loss from [17], except that the L1-norm is taken after the softmax activation output instead of the response map.

Given a pixel (r), an input image (x) and the softmax output (y), the loss is calculated by

$$L_{MScnn}(\Theta, x) = \sum_{n=1}^{K} \sum_{r=1}^{N} |x(r_i) - c_n|^2 y_n(r_i) + |\nabla y_n(r_i)| \tag{16}$$

where N is the total number of pixels, Θ represents the network parameters, $|\nabla y_n(r)|$ is the L1-norm of the softmax output, and c_n is the centroid calculated using the input image and softmax activations as membership probabilities for each class

$$c_n = \frac{\sum_{r=1}^{N} x(r_i) y_n(r_i)}{\sum_{r=1}^{N} y_n(r_i)}$$

The resulting loss curves for the network trained with each of these losses are shown in Figure 9. Note that the total loss of our network is composed of either the sum of our weighted cross-entropy loss with the spatial continuity loss.

$$loss_1 = L_{wce} + L_{con}$$

or by the sum of our loss with the Mumford-Shah loss

$$loss_2 = L_{wce} + L_{MScnn}$$

In the following Figures, we evaluate how adding another loss affected the shape of the curve.

When adding the L1 loss, the curves indicate that the network had more difficulty learning since it converges slower. As for the MS loss, the curve seems to converge similarly, with only the loss value being higher because both losses are being summed up. Next, we show the metrics in Table VII, where we abbreviate the classifier (C) as being either 1 (C1) or 2 (C2). Also, we did not specify the dataset here for lack of space,

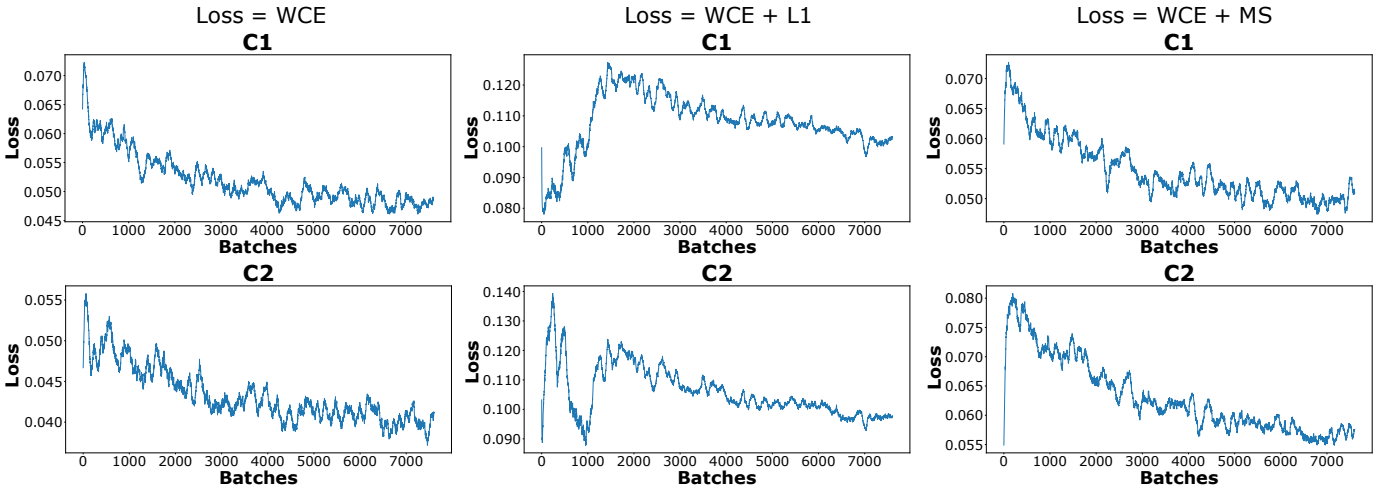

Fig. 9: Comparison between different losses used to train the network.

but the upper half shows the metrics for the dataset with six classes and the bottom half for the dataset with three classes.

We can see that with the L1 loss, the segmentations improved a bit since the Acc, VoI, PRI, and GCE were generally better in the six classes. By enforcing homogeneous regions, we can assume that larger classes got a better representation, but at the cost of sacrificing smaller classes. This is similar to the dataset with three classes, except the Acc. As for the MS metrics, almost no metric improved, indicating that this loss did not improve the model. With these results, we opt to stay with WCE loss only.

X. Class Refinement

Our final experiment with our network regards the class refinement step. All previous studies in the literature have adopted a superpixel segmentation to enforce the pixels within each superpixel to share a common class. We proposed using our class refinement, with the segmentation provided by the external algorithm of [13], where it uses superpixels to segment the image but applies several pre and post-processes to obtain a more consistent segmentation of the image instead of the raw superpixels.

In this experiment, we compare our network's training using superpixels and our proposed mask. The losses for each training are shown in Figure 10, where "GT" indicates which type of Ground Truth labels was used to train the network.

We can see by the results that the loss is much smoother for the superpixel experiment, achieving convergence with less noise. This can be explained because when using an external algorithm, a previous clustering is performed, so the labels enforce classes that can have significant differences in size. As for the second experiment, since multiple superpixels are not enforced to be in the same class, the network can choose the class enforcement by itself, resulting in a more negligible difference in the class sizes. This is directly reflected in the noise in each loss function, much of which can be attributed to our weighted cross-entropy loss. We then compare the metrics for each experiment in Table VIII, where "Seg" stands for Segmentation method, "EX" for "External algorithm," and "SP" for "Superpixels."

Overall, only a few segmentation metrics had a better value for the External Algorithm method, and even those were close to the Superpixels results. These results show that even though providing a pre-grouping of the classes could help to lead the network training, letting the network learn these groupings by itself is even better, generating a final semantic segmentation closer to supervised ground truths. Therefore, we abandon the External Algorithm idea for now and stay with the broadly used superpixel segmentation.

XI. Image Preprocessing

After defining the final specification of the network, the only block missing is the Image Preprocessing. We leave the preprocessing experiment for last because we needed the complete architecture.

To test whether the network needs a preprocessing block or not, we perform an additional experiment were we train 4 different networks for 100 epochs instead of 10, so that we may see the effects of the data augmentation. In each experiment, we train with a different configuration of data augmentations:

- 1 - No data augmentation.
- 2 - Only geometric augmentations (for images and labels).
- 3 - Only color augmentations (for images only).
- 4 - All data augmentations.

This way, we expect to be able to observe which type of data augmentation further impacts our model. The results for this experiment can be seen in Table IX.

As seen, the experiment without data augmentations had the best results. Therefore, we may conclude that in this specific case of unsupervised semantic segmentation, adding data augmentation might have introduced more noise to the model, rather than generalizing. Given this result, we choose to exclude data augmentation from our final architecture.

TABLE VII: Metrics for the different losses experiment. The first half shows the results for the Potsdam dataset, and the second half for the Potsdam-3 dataset.

Loss	C	Acc↑	IoU↑	wIoU↑	F1↑	VoI↓	PRI↑	GCE↓	BDE↓	SC↑
WCE	C1	60.9	32.5	44.1	**45.5**	1.75	0.708	0.281	11.4	0.541
	C2	60.9	**32.6**	**44.4**	**45.5**	1.79	0.706	**0.286**	**11.0**	**0.547**
WCE+L1	C1	**61.7**	29.5	43.7	40.0	**1.62**	**0.715**	0.257	11.4	0.534
	C2	61.4	29.3	43.5	39.8	**1.62**	0.714	0.257	11.7	0.539
WCE+MS	C1	60.5	29.1	42.5	40.1	1.74	0.703	0.284	11.2	0.535
	C2	60.8	29.3	42.8	40.2	1.72	0.706	**0.286**	11.2	0.528
WCE	C1	**75.7**	**60.3**	**61.2**	**74.8**	1.19	**0.754**	0.190	**14.0**	0.693
	C2	**75.7**	**60.3**	**61.2**	**74.8**	1.23	0.748	0.196	14.6	0.693
WCE+L1	C1	74.1	58.4	59.2	73.4	**1.17**	0.752	**0.185**	14.8	0.703
	C2	74.1	58.2	59.0	73.2	**1.17**	**0.754**	**0.185**	15.1	**0.704**
WCE+MS	C1	73.8	57.6	58.5	72.7	1.27	0.741	0.201	14.5	0.687
	C2	74.2	58.1	59.1	72.9	1.23	0.748	0.194	**14.0**	0.683

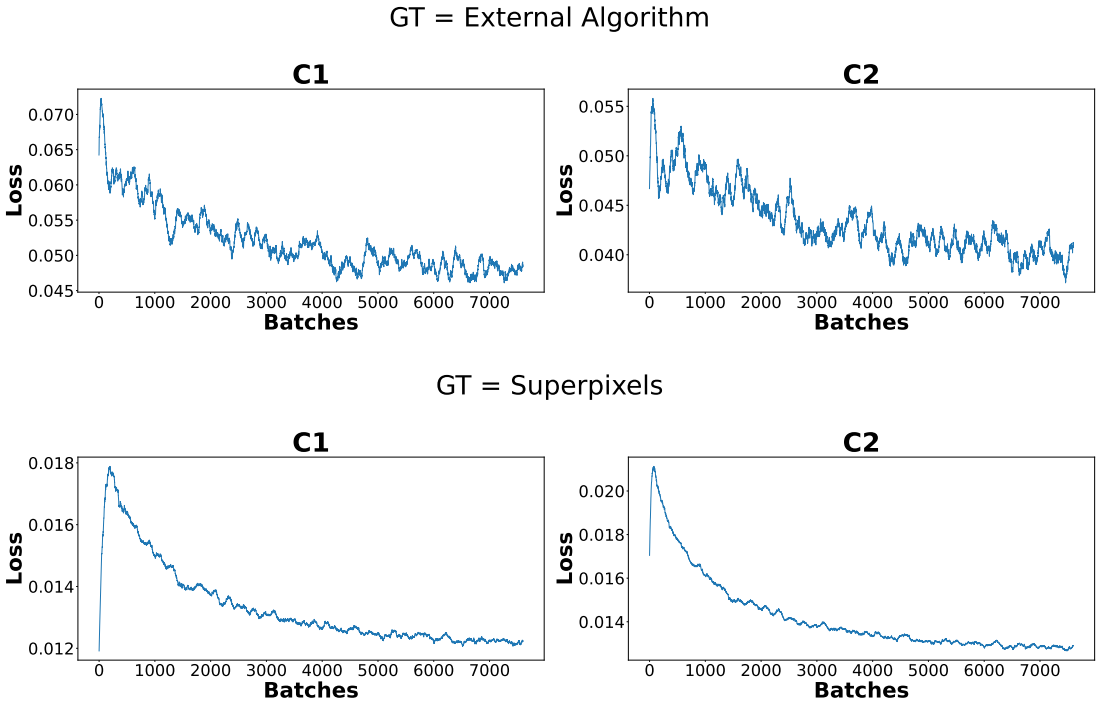

Fig. 10: Losses for different types of ground truth labels.

TABLE VIII: Metrics for the class number experiment.

Seg	C	Acc↑	IoU↑	wIoU↑	F1↑	VoI↓	PRI↑	GCE↓	BDE↓	SC↑
EX	C1	60.9	32.5	44.1	45.5	1.75	**0.708**	0.281	11.4	0.541
	C2	60.9	32.6	44.4	45.5	1.79	0.706	0.286	11.0	0.547
SP	C1	62.6	32.3	45.2	44.7	**1.74**	0.705	0.281	10.8	0.551
	C2	**62.9**	**33.1**	**45.8**	**45.8**	1.78	0.706	**0.278**	**10.5**	**0.566**
EX	C1	75.7	60.3	61.2	74.8	**1.19**	0.754	**0.190**	**14.0**	0.693
	C2	75.7	60.3	61.2	74.8	1.23	0.748	0.196	14.6	0.693
SP	C1	75.9	60.4	61.3	74.9	1.26	**0.747**	0.199	14.1	0.690
	C2	**76.3**	**61.0**	**61.8**	**75.4**	1.22	0.750	**0.190**	14.1	**0.696**

TABLE IX: Experiments with different types of data augmentation.

Exp	Acc↑	IoU↑	wIoU↑	F1↑	VoI↓	PRI↑	GCE↓	BDE↓	SC↑
1	**63.5**	**37.0**	**46.3**	51.6	**1.75**	**0.710**	**0.278**	**10.4**	**0.562**
2	62.1	36.7	45.2	**51.8**	1.80	0.701	0.283	11.0	0.547
3	62.7	33.7	45.5	46.7	**1.75**	0.703	0.280	10.9	0.559
4	60.8	29.2	43.0	40.1	**1.75**	0.707	0.282	11.3	0.545
1	**76.5**	**61.4**	**62.2**	**75.8**	**1.23**	0.747	**0.193**	**14.1**	**0.702**
2	76.1	60.7	61.5	75.3	1.27	0.743	0.200	14.8	0.696
3	76.3	61.2	62.0	75.6	**1.23**	**0.748**	0.194	**14.1**	0.695
4	75.5	59.7	60.5	74.4	1.25	0.744	0.198	15.2	0.686

XII. Final architecture

After analyzing every building block and every step, a summary of our final network is presented:

- Preprocessing
  - Image Normalization only.
- Feature Extractor
  - DIC feature extractor [6] modified to accommodate two classifiers instead of one.
- Classifier
  - Weighted cross-entropy loss, followed by argmax classifier.
- Class Refinement
  - Superpixels ground truth.
- Training
  - Exponential decaying learning rate from $1e^{-4}$ to $1e^{-5}$ and $K_1 = K_2 = 100$ classes.

References

[1] M. Meilă, "Comparing clusterings - An axiomatic view," in *ICML 2005 - Proceedings of the 22nd International Conference on Machine Learning*, 2005, pp. 577–584. 2

[2] X. Xia and B. Kulis, "W-net: A deep model for fully unsupervised image segmentation," *CoRR*, vol. abs/1711.08506, 2017. 2

[3] B. Kim and J. C. Ye, "Mumford-shah loss functional for image segmentation with deep learning," *IEEE Transactions on Image Processing*, vol. 29, pp. 1856–1866, 2020. 2, 10

[4] T. Ilyas, A. Khan, M. Umraiz, and H. Kim, "SEEK: A framework of superpixel learning with cnn features for unsupervised segmentation," *Electronics (Switzerland)*, vol. 9, no. 3, 2020. 2, 3, 4

[5] Z. Khan and J. Yang, "Bottom-up unsupervised image segmentation using FC-Dense u-net based deep representation clustering and multidimensional feature fusion based region merging," *Image and Vision Computing*, vol. 94, 2020. 2

[6] L. Zhou and W. Wei, "DIC: Deep Image Clustering for Unsupervised Image Segmentation," *IEEE Access*, vol. 8, pp. 34 481–34 491, 2020. 2, 3, 4, 13

[7] Q. Lin, W. Zhong, and J. Lu, "Deep superpixel cut for unsupervised image segmentation," *Proceedings - International Conference on Pattern Recognition*, pp. 8870–8876, 2020. 2, 3

[8] A. Blasiak, "A Comparison of Image Segmentation Methods," Carnegie Mellon University, Tech. Rep. May, 2007. 2

[9] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2, 2001, pp. 416–423. 2

[10] J. Freixenet, X. Muñoz, D. Raba, J. Martí, and X. Cufí, "Yet another survey on image segmentation: Region and boundary information integration," *Lecture Notes in Computer Science*, vol. 2352, pp. 408–422, 2002. 2

[11] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 898–916, 2011. 3

[12] A. Kanezaki, "Unsupervised image segmentation by backpropagation," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2018-April, pp. 1543–1547, 2018. 3

[13] B. R. A. Jaimes, J. P. K. Ferreira, and C. L. Castro, "Unsupervised Semantic Segmentation of Aerial Images with Application to UAV Localization," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, 2022. 4, 11

[14] X. Ji, A. Vedaldi, and J. Henriques, "Invariant information clustering for unsupervised image classification and segmentation," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2019-October, pp. 9864–9873, 2019. 4, 9

[15] R. Harb and P. Knöbelreiter, "InfoSeg: Unsupervised Semantic Image Segmentation with Mutual Information Maximization," *Lecture Notes in Computer Science*, vol. 13024 LNCS, pp. 18–32, 2021. 9

[16] J. H. Cho, U. Mall, K. Bala, and B. Hariharan, "Picie: Unsupervised Semantic Segmentation using Invariance and Equivariance in Clustering," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 16 789–16 799, 2021. 9

[17] W. Kim, A. Kanezaki, and M. Tanaka, "Unsupervised Learning of Image Segmentation Based on Differentiable Feature Clustering," *IEEE Transactions on Image Processing*, vol. 29, pp. 8055–8068, 2020. 9, 10