DC3DCD: Unsupervised learning for multiclass 3D point cloud change detection

Iris de Gélis, Sébastien Lefèvre, Thomas Corpetti

DC3DCD: UNSUPERVISED LEARNING FOR MULTICLASS 3D POINT CLOUD CHANGE DETECTION

A PREPRINT

Iris de Gélis[1,2*], **Sébastien Lefèvre**[2], **and Thomas Corpetti**[3]

[1]Magellium, Toulouse, France
[2]IRISA, UMR 6074, Université Bretagne Sud, Vannes, France
[3]CNRS, LETG, UMR 6554, Rennes, France

ABSTRACT

In a constant evolving world, change detection is of prime importance to keep updated maps. To better sense areas with complex geometry (urban areas in particular), considering 3D data appears to be an interesting alternative to classical 2D images. In this context, 3D point clouds (PCs) obtained by LiDAR or photogrammetry are very interesting. While recent studies showed the considerable benefit of using deep learning-based methods to detect and characterize changes into raw 3D PCs, these studies rely on large annotated training data to obtain accurate results. The collection of these annotations are tricky and time-consuming. The availability of unsupervised or weakly supervised approaches is then of prime interest. In this paper, we propose an unsupervised method, called DeepCluster 3D Change Detection (DC3DCD), to detect and categorize multiclass changes at point level. We classify our approach in the unsupervised family given the fact that we extract in a completely unsupervised way a number of clusters associated with potential changes. Let us precise that in the end of the process, the user has only to assign a label to each of these clusters to derive the final change map. Our method builds upon the DeepCluster approach, originally designed for image classification, to handle complex raw 3D PCs and perform change segmentation task. An assessment of the method on both simulated and real public dataset is provided. The proposed method allows to outperform fully-supervised traditional machine learning algorithm and to be competitive with fully-supervised deep learning networks applied on rasterization of 3D PCs with a mean of IoU over classes of change of 57.06% and 66.69% for the simulated and the real datasets, respectively. The code is available at `https://github.com/IdeGelis/torch-points3d-DC3DCD`.

Keywords 3D point clouds · Change detection · Unsupervised Deep learning · Deep clustering

1 Introduction

While urban environments are continuously and rapidly evolving, change detection between several temporal acquisitions is a way to quickly highlight modified areas in order to update maps [Demir et al., 2012] or to identify damaged objects [Dong and Shan, 2013] in case of natural disasters. With regard to the complexity of urban landscapes, sensing the area using also vertical information appears to be judicious. Indeed, such three-dimensional (3D) information allows to better characterize environment geometry and to avoid two-dimensional (2D) image problems such as the difference of viewing angles between distinct acquisitions, spectral variability of objects over time, perspective, and distortion effects [Qin et al., 2016]. 3D data are acquired thanks to Light Detection And Ranging (LiDAR) sensor or photogrammetric process, both resulting in 3D Point Clouds (PCs). No matter the type of acquisition (LiDAR, photogrammetry for example), multi-temporal 3D data are generalizing. Indeed, more and more national mapping agencies opt for full territory Aerial Laser Scanning (ALS) campaign, as in the Netherlands with Actueel Hoogtebestand Nederland (AHN) multi-temporal campaigns [Sande et al., 2010], or in France with the LiDAR high density (HD) project whose objective is to propose a complete 3D high resolution coverage of France with regular updates in the

*iris.de-gelis@irisa.fr

future. In the same time satellites mission for 3D sensing are multiplying, e.g., Pléiades [Bernard et al., 2012], Pléiades Néo [Jérôme, 2019], 3D Optical Constellation (CO3D) [Lebègue et al., 2020] missions. In civil engineering, Terrestrial Laser Scanning (TLS) or unmanned aerial vehicle (UAV) photogrammetry are becoming unavoidable for an accurate sense of complex objects. Thereby, this calls for methods able to analyze these multi-temporal 3D data.

Whether related to urban environment [Stilla and Xu, 2023] or geosciences [Okyay et al., 2019], many studies have been focused on handling 3D data for accurate change extraction. Recently, some methods based on deep learning proved their efficiency over traditional distance-based methods and machine learning. While the first deep learning methods for 3D change detection where based on 2.5D rasterization of PCs into Digital Surface Model (DSM) [Zhang et al., 2018a, 2019] or range image [Nagy et al., 2021], most recent works are dealing directly with the raw 3D data. Indeed, although easing the computation, any rasterization process implies a significant loss of information for instance on building facades but also due to aggregation of several points in a cell. Ku et al. [2021] propose to represent PCs by graphs and apply graph convolution operator (EdgeConv) [Wang et al., 2019] to extract discriminative features. However, their proposed method, called Siamese Graph Convolutional Network (SiamGCN), results only in change detection at the scene level, i.e., solving a change classification task. On the opposite, de Gélis et al. [2023] proposed a network to solve change segmentation task, i.e., providing multiclass change information at point level (so coarser results than change classification). To handle raw 3D PCs, they rely on Kernel Point Convolution (KPConv) [Thomas et al., 2019]. Their network, named Siamese KPConv enables to outperform other machine learning methods or DSM-based deep learning methods. More recently, a study [de Gélis et al., 2023] suggests improving Siamese KPConv by making the network focusing more on change-related features. To do so, the authors propose to provide an hand-crafted feature related to change as input to Siamese KPConv along with 3D point coordinates. They also developed three other architectures for this particular task. OneConvFusion and Triplet KPConv contain specific change-related encoder that takes as input only feature difference. Encoder Fusion SiamKPConv (EFSKPConv) concatenates both change and mono-date information directly in the encoder. This latter achieves the best results compared to previous state-of-the-art methods.

Although efficient, these deep learning-based methods are supervised, i.e., require large databases for the training of the network. For change segmentation, millions of points should be annotated according to the change type. This annotation is often performed manually because any automatization process is not obvious due to PCs characteristics such as the lack of point-to-point correspondence, sparsity, or occlusions (see the example of AHN Change Detection (AHN-CD) dataset [de Gélis et al., 2023]). Therefore, it is interesting to develop methods that require none or at least less annotations to perform 3D change segmentation. As stated in recent surveys [Kharroubi et al., 2022, Xiao et al., 2023], the literature still lacks of methods for unsupervised or weakly supervised learning when dealing with 3D PCs change detection. Indeed, nowadays unsupervised methods are mostly traditional rule-based methods that are often very specific to a dataset. Also, recently an adaptation of Deep Change Vector Analysis (DCVA) [Saha et al., 2019] to 3D PCs change detection has been proposed based on self-supervised learning [de Gélis et al., 2023]. However, this unsupervised method only deals with binary change segmentation, thus the method is not able to distinguish between multiple classes of changes. Thereby, in this paper, we propose an unsupervised learning strategy to deal with multiclass 3D PCs change segmentation. The strategy is based on deep clustering principle and in particular on DeepCluster [Caron et al., 2018]. Deep clustering consists in jointly optimizing deep representation of the data and performing clustering with learned features [Ren et al., 2022, Zhou et al., 2022]. This strategy has received increasing interest for 2D image unsupervised representation learning in computer vision [Caron et al., 2018, Cho et al., 2021]. However, as far as the change detection task is concerned, the use of deep clustering is less common. Zhang et al. [2018b] and Dong et al. [2021] propose to rely on an unsupervised clustering of deep feature representations to further train their network to perform change detection. In Zhang et al. [2018b] a stack of fully-connected layers is used to learn Gaussian-distributed and discriminative difference representations for non-change and different types of changes. Dong et al. [2021] further improve the latter by using a Convolutional Neural Network (CNN) relying on multi-scale self-attention. Another study partially uses deep clustering principle for 2D binary change detection by introducing a deep clustering loss jointly with contrastive and appealing losses to make a CNN network learn discriminative mono-date features. These features are further compared using DCVA principle to extract binary changes in multi-modal optical and Synthetic Aperture Radar (SAR) images [Saha et al., 2021]. This principle was adapted in de Gélis et al. [2023] for 3D PCs binary change detection. Finally, to the best of our knowledge, Zhang et al. [2021] is the first and unique study using the deep clustering principle on 3D data. However, the approach relies on a voxelisation of the PC instead of dealing with the raw PC directly, and performs classification at the scene level and not at the point level (a.k.a. segmentation).

The contributions of this work are thus as follows:

1. To the best of our knowledge, we propose the first unsupervised learning strategy for multiclass change segmentation into raw 3D PCs.

2. We build upon DeepCluster [Caron et al., 2018] to tackle the change segmentation task in 3D PCs. To do so, we experiment our model with several 3D PCs change segmentation architectures.

3. After analyzing the learning behavior, we evaluate our method on both real and simulated public datasets, and provide comparisons with supervised methods and weakly-supervised methods.

The description of our method is given in Section 2. The results are provided in Section 3 and discussed in Section 4. Finally, the conclusion is provided in Section 5. The implementation of our method is available at `https://github.com/IdeGelis/torch-points3d-DC3DCD`.

2 Unsupervised 3D change detection

Before describing our method for 3D PCs multiple change segmentation, an overview of DeepCluster [Caron et al., 2018] is given, as it is an important component of our model.

2.1 DeepCluster principle

Among the variety of studies related to deep clustering [Ren et al., 2022, Zhou et al., 2022], DeepCluster appears to be among the most fundamental ones. Proposed by Caron et al. [2018], this method resides in a rather simple idea of alternatively clustering deep latent representation of data to obtain pseudo-labels further used to train a CNN. In particular, the convolutional network is trained in a supervised manner using pseudo-labels as objective for prediction. In a traditional supervised approach, giving a set of N images x_n ($n \in [1, N]$), a parametrized classifier g_W predicts the correct labels (y_n) using the features extracted by $f_\theta(x_n)$ (W and θ being the parameters from the classifier and the back-bone convolutional model, respectively). They are optimized according to the following problem:

$$\min_{\theta,W} \frac{1}{N} \sum_{n=1}^{N} \ell(g_W(f_\theta(x_n)), y_n) \tag{1}$$

where ℓ is the loss function, a classical negative log-likelihood (NLL) in their method. This cost function is minimized using standard mini-batch stochastic gradient descent and backpropagation to compute the gradient. The difficulty in an unsupervised setting is therefore to define y_n.

In DeepCluster, Caron et al. [2018] proposed to rely on a classical clustering algorithm such as k-means [MacQueen, 1967] or power iteration clustering (PIC) [Lin and Cohen, 2010] to obtain a pseudo-label (y_{PL_n}) that is used instead of y_n. Caron et al. [2018] showed that the choice of the clustering algorithm is not crucial. Thereby, for illustration purposes, we continue with the example of k-means algorithm. This clustering method matches data to k groups (pseudo-clusters) by minimizing distance between each data and its corresponding cluster center, called centroid (and contained in the centroid matrix C in practice).

Finally, the unsupervised training process alternates between i) clustering the output features of the back-bone convolutional model ($f_\theta(x_n)$) (clustering step), and ii) updating parameters θ and W using the obtained pseudo-labels (y_{PL_n}) thanks to Equation 1 (training step). This relies on the fact that a Multi-Layer Perceptron (MLP) classifier on top of a standard CNN with randomly initialized weights (θ) provides results far above from the chance (i.e., random) level [Noroozi and Favaro, 2016].

In practice, a few tricks are required to avoid trivial solutions, e.g., assigning all the inputs to the same cluster. First, the authors get rid of empty clusters by randomly dividing in two groups the largest cluster when an empty cluster appears. Second, if the pseudo-cluster representation of data is largely imbalanced, the deep model will tend to assign all data to the most represented pseudo-cluster. To counter this, they propose to sample input images during the training based on a uniform distribution over the pseudo-labels.

They showed the robustness of their method by training different architectures (Alexnet [Krizhevsky et al., 2017] and VGG-16 [Simonyan and Zisserman, 2014]) on ImageNet [Deng et al., 2009] or YFCC100M [Thomee et al., 2016] images datasets.

In the following, we adapt this principle to 3D PCs change detection.

2.2 DC3DCD: unsupervised learning for 3D multiple change extraction

Whereas the task and the data (2D image classification) of DeepCluster is far from 3D PCs multiple change segmentation, we nevertheless decided to adapt this method to our task and particular data. By replacing the CNN by a 3D PCs change detection back-bone, some change-related features can be extracted. Thereby, the clustering of these deep features results in change-related pseudo-clusters. We further rely on these pseudo-clusters to optimize the trainable parameters θ of the change detection back-bone. Figure 1 illustrates our method called DeepCluster 3D Change Detection (DC3DCD).

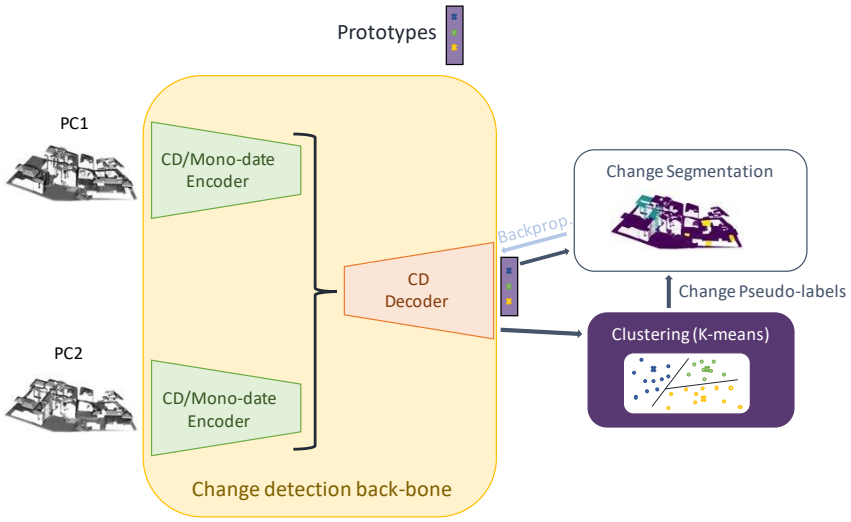


Figure 1: **Illustration of our proposed method: DC3DCD.** It is trained by alternatively clustering deep features to match a pseudo-label to each point of PC 2. These pseudo-labels are used to optimize the back-bone trainable parameters.

Algorithm 1 Fully unsupervised DeepCluster 3DCD training

Initialize the back-bone trainable parameters θ
for $e \leftarrow 1$ to $\mathcal{E}$ **do**
  Run mini-batch k-means to obtain centroids C on the whole training set
  Assign to each point of the training set a pseudo-cluster
  Replace parameters of the prototype layer by C
  Compute the weights W_k considering pseudo-label distribution in the training set
  Training sample selection: random drawing considering W_k
  for $i \leftarrow 1$ to $\mathcal{I}$ **do**
    Use $\mathcal{L}_{NLL}$ (weighted by W_k) to modulate the back-bone trainable parameters (θ) considering pseudo-labels

The overall training process of our method is given in the Algorithm 1. Even if the general idea of DeepCluster remains, some specific features of DC3DCD distinguishing it from the original DeepCluster idea should be noted, as described in the following.

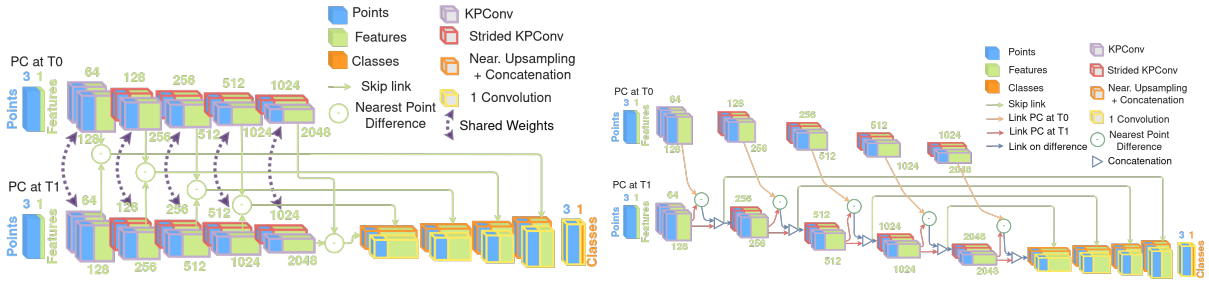
2.2.1 Back-bone model

Because of the unstructured nature of 3D PCs, traditional CNN models cannot be applied on them and furthermore, they do not output change-related deep features, but they rather characterize each input independently. To cope these issues, some studies have recently proposed different architectures for supervised change detection in 3D PCs (see section 1). These architectures can therefore be used as a back-bone to our unsupervised method. In practice, both *Siamese KPConv* [de Gélis et al., 2023] and *Encoder Fusion SiamKPConv* [de Gélis et al., 2023] will be experimented. *Siamese KPConv* architecture is chosen because it is the first architecture to perform change segmentation into raw 3D PCs. It extends the idea of Siamese networks with the KPConv convolution principle, as can be seen in Figure 2a. The very recent *Encoder Fusion SiamKPConv* is experimented as well since it has been shown in de Gélis et al. [2023] that it outperforms other state-of-the-art methods including *Siamese KPConv*. The main idea behind this architecture is no more to encode the two PCs independently (as *Siamese KPConv* do) but rather to include the change information along the encoding process, as can be seen in Figure 2b where the encoding in the bottom part takes into account also the differences between features.

Thus, parameters θ to be optimized are parameters of these back-bone architectures.

2.2.2 Use of a prototype layer

In the original version of DeepCluster, the final classification layer of g_W is re-initialized before each parameter optimization session (i.e., training steps) because there is no correspondence between two consecutive cluster assignments. The authors further proposed an improved version of DeepCluster (DeepCluster-V2) by replacing the classifier g_W by the prototypes, i.e., cluster centers. This ends up with an explicit comparison of the features and the centroid matrix



(**a**) Siamese KPConv [de Gélis et al., 2023] (**b**) Encoder Fusion SiamKPConv [de Gélis et al., 2023]

Figure 2: **Back-bone architectures used in our experiments.**

C, and tends to improve stability and performance of DeepCluster [Caron et al., 2020]. According to preliminary experiments, we also decided to use the centroid matrix C defining pseudo-cluster centers. Therefore, the last fully connected layer parameters of the back-bone are set using the centroid matrix C. This so-called prototype layer is updated after each clustering step and fixed during the training step.

2.2.3 Input data

We aim at detecting changes into raw 3D PCs. In addition of using a specific back-bone able to compute features directly in 3D PCs, some pairs of bi-temporal vertical cylinders need to be used as input to the network. In a context of change detection, cylinders are preferred to spheres when point clouds have a privileged orientation, as for airborne 3D PCs where the vertical direction embeds different information than horizontal ones [de Gélis et al., 2023].

Furthermore, in their experimentation, Caron et al. [2018] provide Sobel-filtered images as input to the CNN instead of Red Green Blue (RGB) images. Sobel filtering acts as an edge detector thanks to the computation of gradients on the image. This seems an important step in their method [Caron, 2021, Mustapha et al., 2022] and acts as a pre-computation of relevant features. However, when dealing with 3D PCs, there is no direct equivalent to Sobel filtering. Thus, we rely on some predefined features commonly used to characterize 3D PCs. Let us note that using such features can help the network, as already shown in a supervised scenario [de Gélis et al., 2023]. As such, we consider here some handcrafted features already used in a Random Forest (RF)-based change detection context Tran et al. [2018]:

- Point distribution represented by point normals and information on the distribution of points in the neighborhood (i.e., linearity, planarity, and omnivariance).
- Height information characterized by rank of the point on vertical axis in the neighborhood, maximum range of elevation of points in the neighborhood, and normalized height according to the local Digital Terrain Model (DTM) (rasterization of the PC at the ground level).
- Change information described through a feature called stability (ratio of the number of points in the neighborhood of the current PC to the number of points in the neighborhood in the other PC).

We refer the reader to the original paper Tran et al. [2018] for a detailed description of these features.

2.2.4 Size of the training set

Change segmentation task implies assigning a pseudo-label to each point of the second PC (of pairs of the training set). To fit in memory, a mini-batch k-means [Sculley, 2010] clustering is used. The principle of splitting the largest cluster when an empty cluster appears is used as in DeepCluster.

2.2.5 Imbalanced dataset

Change detection datasets are highly imbalanced. To avoid falling in a trivial solution where the back-bone predicts all points with the same label, after each clustering step weights W_k (considering pseudo-labels distribution) are computed. These weights are further used to both select training cylinders and weight the loss ($\mathcal{L}_{NLL}$). Let us note that this cylinder selection process was also applied in the supervised context [de Gélis et al., 2023] (on the real labels though). It aims at giving more training samples of underrepresented pseudo-clusters. It also acts as a kind of data augmentation because from one epoch to another, selected cylinders differ according to the random drawing of the cylinder's central point. Without this trick, the method is likely to collapse to a single class prediction.

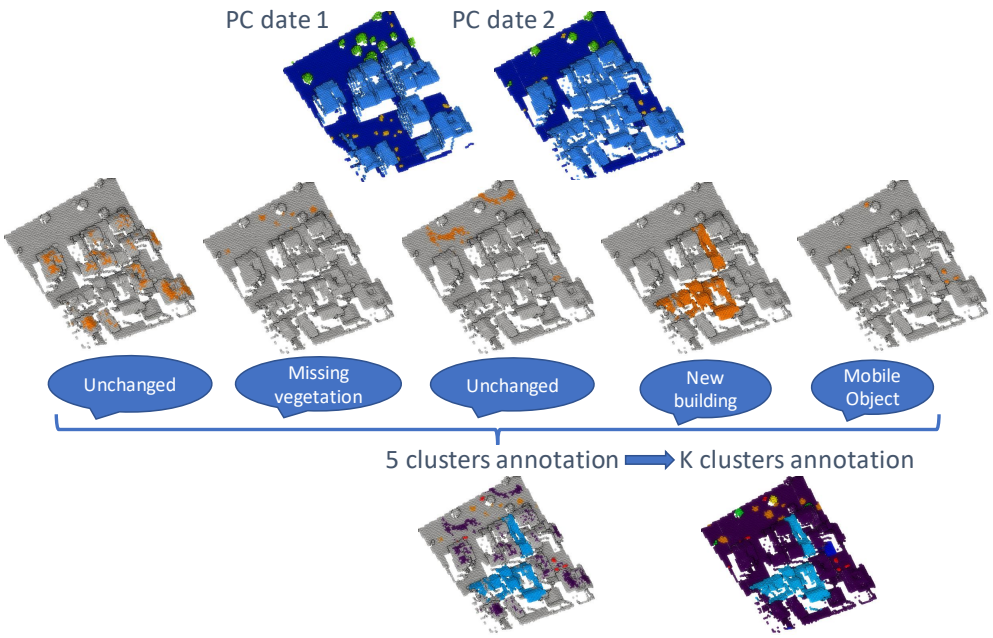


Figure 3: **User guided mapping of predicted clusters to real classes.** For the K predicted clusters, a mapping with the corresponding real class is performed by a user to obtain the final change segmentation of the PC. 5 mappings are provided for the sake of illustration. Segmenting the whole dataset requires K annotations only. This is far less than the millions of points that need to be annotated in order to build training and validation sets in a supervised setting.

2.2.6 3D PCs data augmentation

During the training step, data augmentation appears to be crucial for stability and performance of the method. In particular, the following data augmentation strategies are used: random cylinders rotation around the vertical axis (same angle for both cylinders of a pair), and addition of a Gaussian noise at point level.

2.3 From predicted pseudo-labels to real labels

The above training using DC3DCD method is fully unsupervised, thereby no use of a ground truth is required. At the end of the overall training process, the back-bone predicts labels for all points of the second PC according to the change. Predicted labels do not directly correspond to the real labels. There is an oversegmentation of PCs inducted by the choice of K, the number of pseudo-clusters, which is often large compared to the number of real classes. By opting for such an oversegmentation setting, we expect to be able to address various use cases with different size and precision of classes. One real class is then composed of several predicted clusters, while we assume a predicted cluster to contain only one real class. To map a real label onto each predicted label, a mapping step is necessary. For this mapping, we consider that the user should be involved in order to select the kind of changes that is of interest given the use case. DC3DCD enables to train a back-bone to segment the PC into small areas containing the same types of change or unchanged objects. Thus, the user just has to select for each predicted cluster a corresponding real class. It can be viewed as a kind of active learning process. This is illustrated in Figure 3. This strategy finally involves K annotations to obtain a final change segmentation over the whole testing set (no matter its size). K corresponds to the number of pseudo-clusters used during the training. This hyper-parameter has to be set beforehand. For this reason, our method can be viewed as weakly supervised since K annotations by a user are required in the end of the process. However, these annotations (smaller than the millions of points contained in the dataset) are not taken into account in the learning process, and we therefore classify our approach in the family of unsupervised methods. In practice, for the experimental assessment of our method, we map a predicted cluster onto a real class taking into account the real majority class it contains.

3 Experimental results

3.1 Experimental settings and protocol

We detail here how we set the main hyper-parameters and we describe our experimental settings.

3.1.1 Datasets

Both simulated and real datasets will be experimented, considering Urb3DCD-V2 in low density LiDAR configuration [de Gélis et al., 2021, 2023] and AHN-CD [de Gélis et al., 2023]. To the best of our knowledge, these are the only public datasets for change segmentation. The Urb3DCD dataset is composed of simulated ALS PCs over a french city model (Lyon, France). A simulator is used in order to obtain an accurate annotation of 3D points concerning multiclass changes. Conversely, AHN-CD dataset is made of real ALS PC from AHN campaigns over the Netherland [Sande et al., 2010]. The change ground-truth is obtained thanks to a semi-automatic process. In de Gélis et al. [2023], it has been shown that this process is not perfect since it results in a lot of misclassifications. Therefore, the test set of this dataset will be used for qualitative assessment only. As for quantitative assessment, AHN-CD comes with a manually annotated sub-part of the test set, on which we will report and compare quality scores. Note that during the training, we make no use of the ground truth unless for the method assessment purpose (see Section 3.2). The first sub-sampling rate dl_0 is set to 1 m and the cylinder radius to 50 m for Urb3DCD-V2-1. While for AHN-CD dl_0 is set to 0.5 m and the cylinder radius to 20 m because of the difference of density between both datasets.

3.1.2 Number of pseudo-clusters K

As shown in different studies related to DeepCluster [Caron et al., 2018, Mustapha et al., 2022], choosing the number of pseudo-clusters is important. We experimented several values for K on the simulated dataset and found that $K = 1000$ was an adequate compromise to ensure a stable training and a reasonable oversegmentation. A too small value may not reflect all different types of changes (and thus not allow the user to select the changes of interest), while a too large value leads to high training and annotation times. The same value will also be used with the real dataset AHN-CD.

3.1.3 Training step and parameters optimization

For the training process, a Stochastic Gradient Descent (SGD) with momentum of 0.98 is applied to minimize a point-wise NLL loss using the pseudo-labels defined in the clustering step. A batch size of 10 is used. The initial learning rate is set to 10^{-3} and scheduled to decrease exponentially. As in Caron et al. [2018], we experimentally verified that reassigning the clustering after each epoch is better than an update after each n epochs. Indeed, if several training epochs are conducted, the model seems to converge in the first local minimum associated with a non-optimal pseudo-clustering. In each epoch, $3{,}000$ cylinder pairs are seen by the model. A total of 55 epochs, i.e., 55 clustering and training steps, is performed.

3.1.4 Comparisons with supervised methods

A comparison with the only existing deep supervised methods is provided, namely *Siamese KPConv* [de Gélis et al., 2023], *Encoder Fusion SiamKPConv* [de Gélis et al., 2023], DSM-based deep learning methods (adaptation of Daudt et al. [2018] networks to DSM inspired by Zhang et al. [2019]). Results of a RF algorithm trained on hand-crafted features [Tran et al., 2018] are also given.

3.1.5 Adaptation of a supervised method to a weakly supervised setting

To evaluate the benefit of our method, we propose a comparison with deep learning-based supervised methods tuned to a weakly supervised setting. In practice, we use *Encoder Fusion SiamKPConv*, the best of supervised techniques according to de Gélis et al. [2023]. To this end, we trained it with the same amount of annotated data as our DC3DCD setting. However, this is not straightforward since this network cannot be trained with only $1{,}000$ points. Indeed, as during the supervised training of this network, labels should be provided for each point of the second PC of the pair, and as a cylinder contains more than $1{,}000$ points (about $3{,}500$), we cannot directly compare the supervised training with the same amount of labels (i.e., $K = 1000$). Thus, for both training and validation sets, we chose 7 cylinders, each one centered on one of the 7 classes contained in Urb3DCD-V2 to be sure that each class is represented. Note that with this minimal training configuration, the number of annotated points in the 14 cylinders is around $50{,}000$. As 7 cylinders are less than the batch size of 10 used for all other deep learning-based methods, we also provide results with a batch size of 2.

3.1.6 Comparisons with unsupervised methods

To the best of our knowledge, there is no other weakly supervised or unsupervised deep learning method tackling 3D PCs multiple change segmentation. Thereby, we provide a comparison with a k-means algorithm applied on the ten hand-crafted features of Tran et al. [2018] dedicated to change detection in 3D PCs. Note that LiDAR specific features (e.g., intensity or number of echoes) used in Tran et al. [2018] are ignored here since the simulated dataset does not contain such information. For fair comparison, the k-means is set to predict also $K = 1,000$ pseudo-clusters. The same user-guided mapping is done as proposed in the previous section (Figure 3) to assign the final classes.

Before presenting the quantitative results, we analyze in the next section the behavior of the network during the training process.

3.2 Analysis of the learning process

Before presenting the results on the different datasets, we propose to study the behavior of DC3DCD during the training phase. To do so, we rely on the *Encoder Fusion SiamKPConv* back-bone and the configuration without the use of the ten hand-crafted features as input, so considering only 3D points coordinates. Note that the same tendencies are obtained with hand-crafted features or with *Siamese KPConv* back-bone, but we prefer to show results with a network that takes into account the minimum information regarding changes. In practice, we compute criteria associated with the clustering quality and the pseudo-cluster distribution along the training process.

3.2.1 Clustering quality

The evolution of the clustering quality during training epochs is computed by comparing the pseudo-clusters obtained thanks to the k-means on deep features, and the real classes. More precisely, we compute the normalized mutual information (NMI) given by the following formula:

$$\text{NMI}(Y, Y_C) = \frac{I(Y, Y_C)}{\sqrt{H(Y)H(Y_C)}} \tag{2}$$

where Y and Y_C contain the probabilities p_i, p_{C_i}, of each label $i = \{1...N\}$ associated with the true and pseudo-labels. H is the entropy defined as:

$$H(Y) = -\sum_{i=1}^{N} p_i \log_2 p_i \tag{3}$$

and I is the mutual information, defined as:

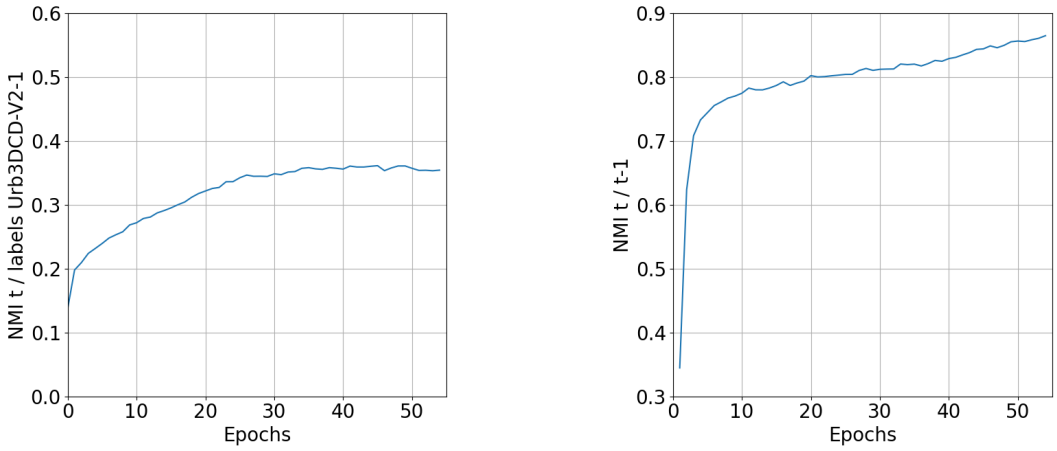
$$I(Y, Y_C) = H(Y) - H(Y|Y_C) \tag{4}$$

Intuitively, the NMI is a measure of the information shared between two clusterings, i.e., in our case the clustering of deep features and real classes. If the NMI is equal to 0, the two clusterings are totally independent. On the opposite, if the NMI is equal to 1, there is a perfect correlation between the two clusterings, i.e., one of them is deterministically predictable from the other.

We present the evolution of the clustering quality along the epochs in Figure 4a by giving the NMI between the clustering and the real labels of Urb3DCD-V2 dataset. As can be seen, the clustering tends to get closer to real classes along with the training process. It seems to stabilize after 30-40 epochs. Let us remark that at the end of the training, the NMI is around 0.35. It is still far to 1, but the same trend was observed in DeepCluster training quality assessment by Caron et al. [2018]. In Figure 4b, we evaluate the number of reassignments of cluster from one epoch to the following using the NMI between the clustering of the two epochs. It seems that during the first epochs, there is an important evolution of the clustering, but the training rapidly converges to a rather stable clustering ($NMI > 0.8$). Again, the same tendency was obtained by Caron et al. [2018] for DeepCluster on ImageNet dataset.

3.2.2 Pseudo-cluster distribution

As for the pseudo-cluster distribution, we remind that, ideally, a pseudo-cluster contains only one real class, and a real class can be distributed into several pseudo-clusters. To measure the purity of a pseudo-cluster, we propose to investigate the entropy H (Equation 3) of each pseudo-cluster. If it is near 0, then the pseudo-cluster contains almost only one real class. However, if a pseudo-cluster is divided into several classes, the entropy is higher. In Figure 5 is given the pseudo-cluster distribution at epoch 10 and 50. Pseudo-clusters are sorted in increasing entropy values. First, as can be seen, there is an improvement between epoch 10 and 50. The area under the entropy curves (dotted points in Figure 5) is indeed smaller at epoch 50 (0.24 of mean entropy) than at epoch 10 (0.39 of mean entropy), meaning that



(**a**) Clustering quality (**b**) Cluster reassignment

Figure 4: **Analysis of the behavior of DC3DCD during the training.** The evolution of clustering quality (**a**) is given thanks to the NMI between the clustering and the real labels of Urb3DCD-V2 dataset. The NMI between the clustering at epoch t and the clustering at epoch $t-1$ gives the cluster reassignment (**b**).

entropy values are globally smaller. Then, after 50 epochs of training, 80% of the pseudo-clusters have a "purity level" greater or equal to 93%. These results confirmed the relevance of the proposed user-guided strategy to automatically map a pseudo-cluster onto the majority real class for the evaluation, as stated in the method description section (see Section 2.3).

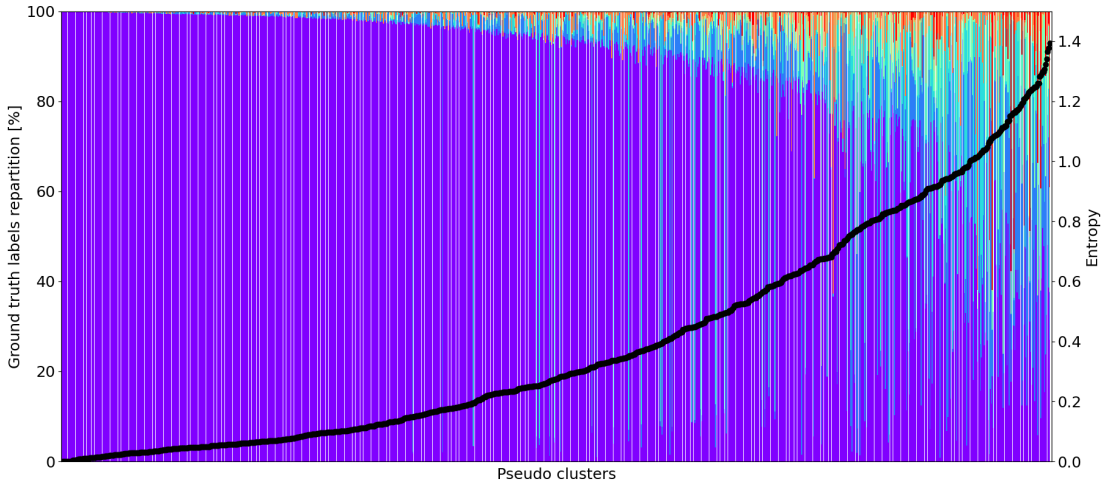
After having studied the training behavior of the DC3DCD method, we will now compare it to the state-of-the-art on the testing set of both simulated and real datasets.

3.3 Results on simulated Urb3DCD dataset

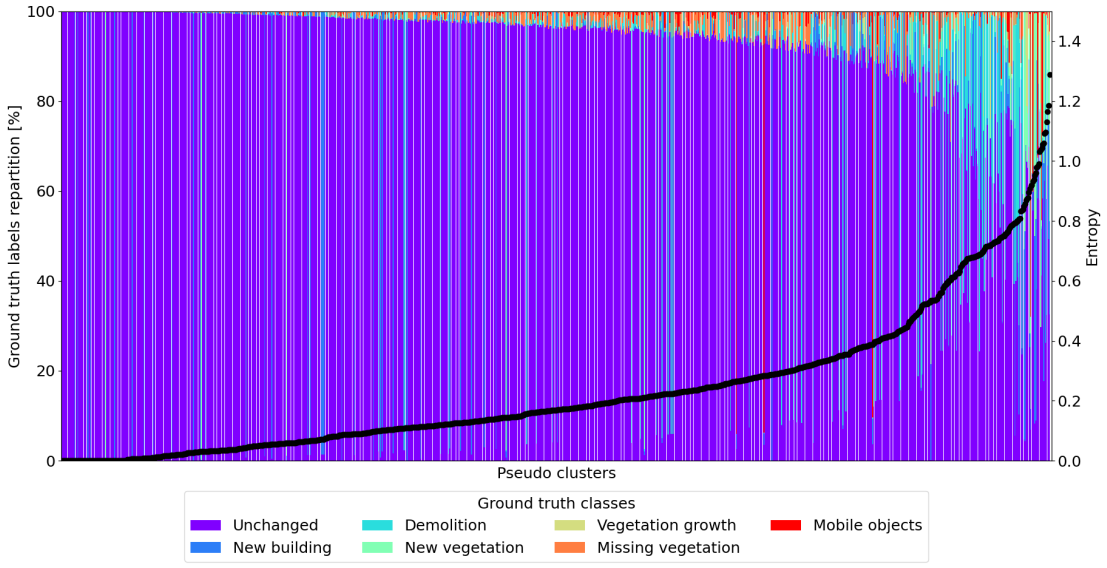
Quantitative evaluation of DC3DCD on the simulated Urb3DCD-V2-1 dataset is given in Tables 1 and 2. In these tables, we also recall supervised results for the sake of comparison. Let us first analyze DC3DCD results without hand-crafted features (two first lines of the bottom part of Tables 1 and 2). As can be seen, results for both *Siamese KPConv* and *Encoder Fusion SiamKPConv* back-bones are rather low. Indeed, while requiring the same annotation effort, k-means algorithm trained on the same hand-crafted as the RF method proposed in Tran et al. [2018] provides better results (cf. first line of middle part in Table 1). However, these results are interesting because the two experimented back-bones provide significantly different results. In particular, *Encoder Fusion SiamKPConv* ends up with a mIoU$_{ch}$ 1.5 times higher than *Siamese KPConv*. While in a supervised setting, the improvement of *Encoder Fusion SiamKPConv* was of 5 points of mIoU$_{ch}$, in the unsupervised context the choice of the architecture seems even more crucial.

Then, when hand-crafted features are added to the input of the network, results are largely improved (cf. the two last lines of Table 1). While DC3DCD with *Siamese KPConv* architecture and hand-crafted features provides results comparable to the k-means algorithm, DC3DCD provide interesting results with both hand-crafted features and the *Encoder Fusion SiamKPConv* architecture. Indeed, in this configuration there is more than 15 points of mIoU$_{ch}$ of improvement compared to k-means. Furthermore, DC3DCD with this configuration is better than a fully supervised RF, and provides results comparable with fully supervised deep architectures trained on 2.5D rasterization of 3D PCs. Thereby, providing hand-crafted features is an important step in unsupervised settings. One possible interpretation is that the unsupervised version is very tricky to train in reason of the large number of possible local minima. Adding hand-crafted features probably helps the initialization to be closer to the global minimum.

Notice that the *Encoder Fusion SiamKPConv* in a weakly supervised setting provides rather low results given the higher annotation effort required (about 50,000 annotated points). Results with a batch size of 10 are not stable. This is explained by the fact that only one batch is seen per epoch, and the same learning rate scheduler as with a batch size of 2 is used. Thus, this training is more prone to fall in a local minimum. Even with a reduced batch size, leading to more stable results, we can see the benefit of using DC3DCD for the training of *Encoder Fusion SiamKPConv* network because the effort of annotation is lower.



(**a**) Epoch 10



(**b**) Epoch 50

Figure 5: **Ground truth class distribution in pseudo-clusters** sorted by increasing entropy value at epoch 10 (**a**) and 50 (**b**). Each pseudo-cluster entropy is given by the dotted black curve.

	Method	mAcc (%)	mIoU$_{ch}$ (%)
Supervised	Siamese KPConv [de Gélis et al., 2023]	91.21 ± 0.68	80.12 ± 0.02
	Encoder Fusion SiamKPConv [de Gélis et al., 2023]	**94.23** ± 0.88	**85.19** ± 0.24
	DSM-Siamese	80.91 ± 5.29	57.41 ± 3.77
	DSM-FC-EF	81.47 ± 0.55	56.98 ± 0.79
	RF [Tran et al., 2018]	65.82 ± 0.05	52.37 ± 0.10
Weakly sup.	k-means	56.15 ± 0.62	41.46 ± 0.53
	Encoder Fusion SiamKPConv (batch size 10)	29.03 ± 22.46	12.84 ± 18.49
	Encoder Fusion SiamKPConv (batch size 2)	53.09 ± 3.73	36.60 ± 3.18
	DC3DCD *Siamese KPConv*	28.28 ± 3.73	14.43 ± 3.70
	DC3DCD *Encoder Fusion SiamKPConv*	52.30 ± 2.41	37.75 ± 2.11
	DC3DCD *Siamese KPConv* (with input features)	54.91 ± 5.45	42.27 ± 6.64
	DC3DCD *Encoder Fusion SiamKPConv* (with input features)	**68.45** ± 1.10	**57.06** ± 0.41

Table 1: **Quantitative evaluation of DC3DCD on Urb3DCD-V2 low density LiDAR dataset.** *Top*: supervised methods. DSM-based methods are adaptation of Daudt et al. [2018] networks to DSM inspired by Zhang et al. [2019] and RF refers to Random Forests. *Middle*: Weakly supervised methods with k-means and *Encoder Fusion SiamKPConv* results using 7 training cylinders in the training and validation set (equivalent to about 50,000 annotated points). *Bottom*: Weakly supervised methods with our proposed DC3DCD evaluated in 4 different settings: with *Siamese KPConv* or *Encoder Fusion SiamKPConv* architectures and with or without the addition of 10 hand-crafted features as input to the network.

		Per class IoU (%)						
	Method	Unchanged	New building	Demolition	New veg.	Veg. growth	Missing veg.	Mobile Object
Supervised	SKPConv [de Gélis et al., 2023]	95.82 ± 0.48	86.68 ± 0.47	78.66 ± 0.47	93.16 ± 0.27	65.17 ± 1.37	65.46 ± 0.93	91.55 ± 0.60
	EFSKPConv [de Gélis et al., 2023]	**97.47** ± 0.04	**96.68** ± 0.30	**82.29** ± 0.16	**96.52** ± 0.03	**67.76** ± 1.51	**73.50** ± 0.81	**94.37** ± 0.54
	DSM-Siamese	93.21 ± 0.11	86.14 ± 0.65	69.85 ± 1.46	70.69 ± 1.35	8.92 ± 15.46	60.71 ± 0.74	8.14 ± 5.42
	DSM-FC-EF	94.39 ± 0.12	91.23 ± 0.31	71.15 ± 0.99	68.56 ± 3.92	1.89 ± 2.82	62.34 ± 1.23	46.70 ± 3.49
	RF [Tran et al., 2018]	92.72 ± 0.01	73.16 ± 0.02	64.60 ± 0.06	75.17 ± 0.06	19.78 ± 0.30	7.78 ± 0.02	73.71 ± 0.63
Weakly sup.	k-means	91.82 ± 0.05	70.46 ± 0.25	59.83 ± 0.11	59.20 ± 0.48	6.00 ± 0.19	0.00 ± 0.00	53.26 ± 3.19
	EFSKPConv (b. s. 10)	58.38 ± 46.55	13.22 ± 16.83	26.15 ± 23.54	11.04 ± 19.12	0.41 ± 0.71	7.29 ± 11.99	19.48 ± 33.46
	EFSKPConv (b. s. 2)	89.88 ± 0.53	29.26 ± 16.83	48.66 ± 2.57	52.91 ± 9.19	7.59 ± 6.40	14.35 ± 15.16	66.84 ± 5.42
	DC3DCD SKPConv	84.51 ± 0.70	13.33 ± 3.05	29.50 ± 13.19	40.31 ± 9.15	3.03 ± 1.80	0.08 ± 0.01	0.33 ± 0.29
	DC3DCD EFSKPConv	90.90 ± 0.79	64.06 ± 5.13	54.35 ± 3.84	58.14 ± 20.03	1.45 ± 2.05	0.94 ± 0.78	47.57 ± 2.58
	DC3DCD SKPConv (i. f.)	92.90 ± 0.21	76.61 ± 2.09	67.22 ± 2.63	61.33 ± 10.07	8.66 ± 6.54	16.39 ± 14.95	23.44 ± 40.54
	DC3DCD EFSKPConv (i. f.)	**93.96** ± 0.11	**79.26** ± 0.68	**67.88** ± 0.49	**75.34** ± 2.81	**19.48** ± 4.00	**20.29** ± 2.90	**80.10** ± 3.16

Table 2: **Per-class IoU scores of DC3DCD on Urb3DCD-V2 low density LiDAR dataset.** *Top*: supervised methods. DSM-based methods are adaptation of Daudt et al. [2018] networks to DSM inspired by Zhang et al. [2019] and RF refers to Random Forests. *Middle*: Weakly supervised methods with k-means and *Encoder Fusion SiamKPConv* results using 7 training cylinders in the training and validation set (equivalent to about 50,000 annotated points). *Bottom*: Weakly supervised methods with our proposed DC3DCD evaluated in 4 different settings: with *Siamese KPConv* (SKPConv) or *Encoder Fusion SiamKPConv* (EFSKPConv) architectures and with or without the addition of 10 hand-crafted features as input to the network. Veg. stands for vegetation; b. s. for batch size; i.f. for input features.

Two different examples are given for a qualitative assessment of the method in Figures 6 and 7. As visible in Figure 7, main changes (e.g., new buildings or demolitions) seem quite well retrieved by the k-means and both DC3DCD configurations. However, when going more into details, some misclassifications can be seen on new building facades (Figure 6) or vegetation. For new building facades, a slight improvement over k-means is reached by DC3DCD, but it is still not perfect. The k-means technique has the same tendency as the RF method (see de Gélis et al. [2023]) and confuses small new buildings with new vegetation, surely because they have the same height as visible in Figure 6. As depicted in Table 2, main difficulties of the DC3DCD method concern vegetation growth and missing vegetation. Note that this was already the most difficult classes in the supervised context. The missing vegetation is almost always confused with demolition in DC3DCD with hand-crafted input features and the *Encoder Fusion SiamKPConv* architecture. This is even worse with the k-means and missing vegetation is never predicted with DC3DCD without hand-crafted input features. However, this make sense, since the 'missing vegetation' and 'demolition' classes are both negative changes. Surprisingly, mobile objects are quite well retrieved, especially for DC3DCD with hand-crafted input features and the *Encoder Fusion SiamKPConv* architecture (Table 2).

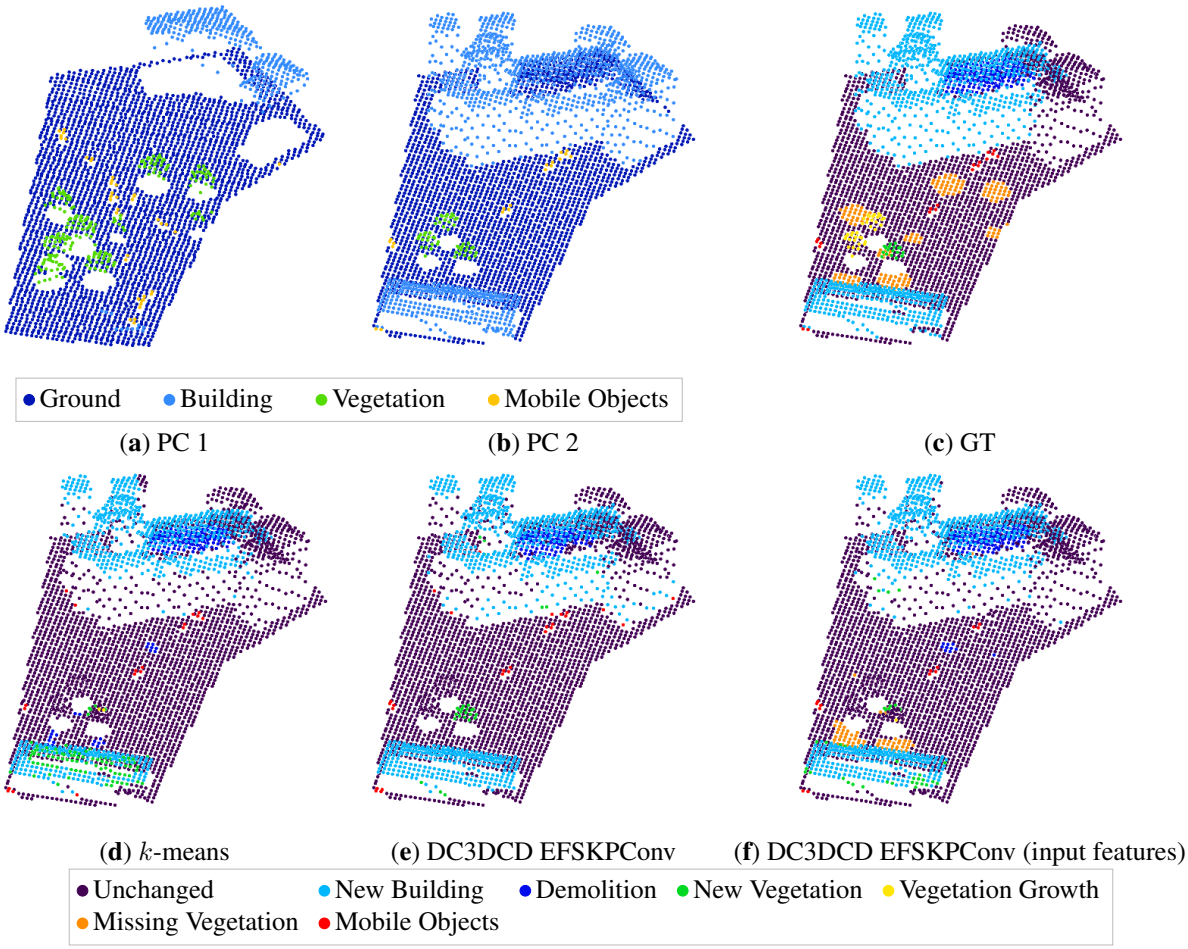


Figure 6: **Visual change detection results on Urb3DCD-V2 low density LiDAR sub-dataset (area 1):** (**a**-**b**) the two input point clouds; (**c**) ground truth (GT): simulated changes; (**d**) k-means results; (**e**) DC3DCD with the *Encoder Fusion SiamKPConv* architecture results; (**f**) DC3DCD with the *Encoder Fusion SiamKPConv* architecture and the addition of 10 hand-crafted features as input results.

3.4 Results on real AHN-CD dataset

Concerning the real AHN-CD dataset, quantitative results on the manually annotated testing set are given in Table 3, and Table 4 for per class results. For comparison purpose, we also provide results of supervised methods. However, we recall that they have been trained on the semi-automatically annotated AHN-CD dataset containing several ground truth errors [de Gélis et al., 2023]. This explains lower results of the RF compared to the k-means which have been mapped onto real classes using the manually annotated set (as for DC3DCD method). As already observed with the simulated dataset, we can see that DC3DCD provides better results than the k-means algorithm. Figure 8 shows that main changes are well retrieved for both methods. However, in the k-means results, larger objects of the clutter class such as trucks are mixed up with buildings. There are also lots of misclassifications in unchanged vegetation and unchanged building facades (see region of interest in Figure 8f). As far as DC3DCD is concerned, unchanged vegetation is well classified. A few mistakes are visible in some 'new clutter' objects. We recall that this class is a mix of a lot of objects, from vegetation to cars or garden sheds, surely explaining why its classification score is lower. Complementary results on a larger AHN-CD testing tile are shown in Figure 9. The ground truth is given by the semi-automatic process detailed in de Gélis et al. [2023]. The mapping onto the real classes is performed using this ground truth for both k-means method and DC3DCD. As visible in this example, most of 'new clutter' class objects are omitted or mixed up with the new building class, also the demolition class is totally omitted by the k-means algorithm (Figure 9d). In the DC3DCD results in Figure 9e, clutter class seems better retrieved, even though it is not perfect implying the main differences with the ground truth (Figure 9g). In the areas of interest depicted by the black rectangles in Figure 9, we observe that DC3DCD seems to better adapt to the user context (i.e., by the ground truth defined by the user) than the k-means, even

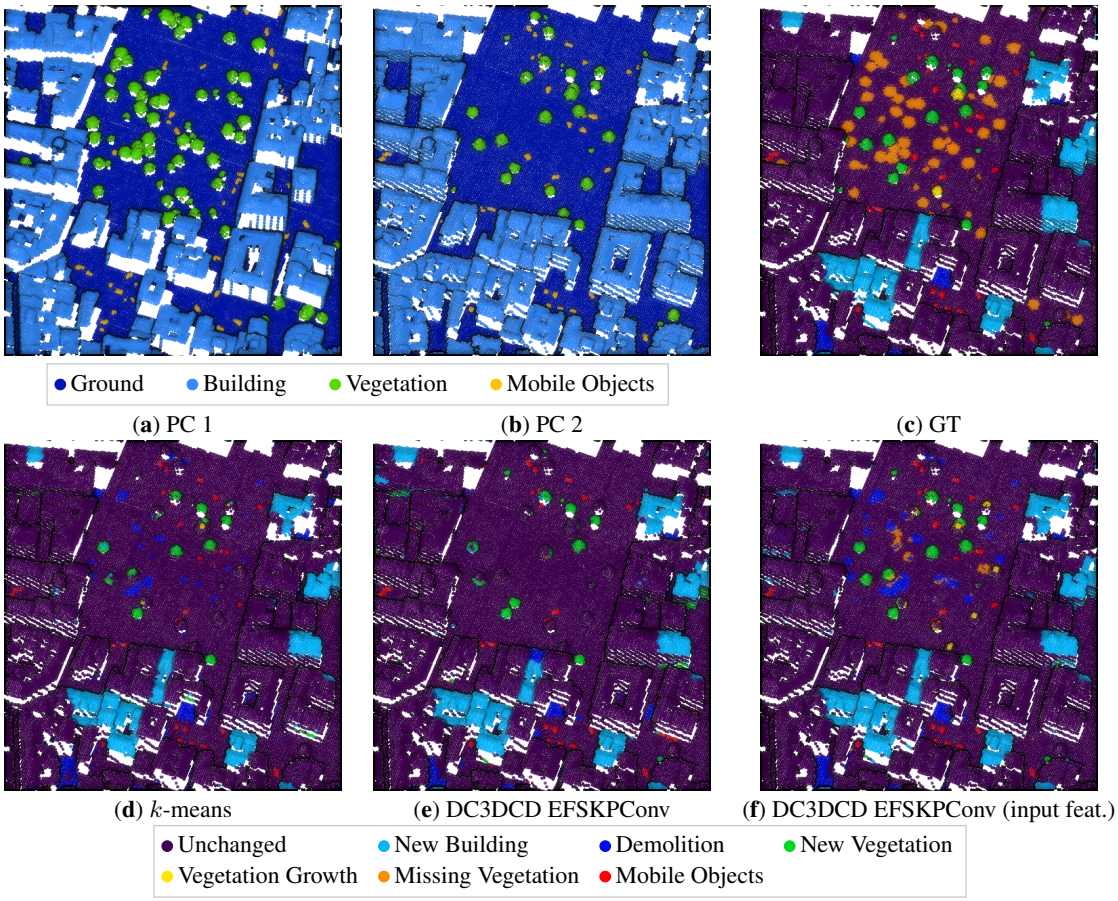


Figure 7: **Visual change detection results on Urb3DCD-V2 low density LiDAR sub-dataset (area 2):** (**a**-**b**) the two input point clouds; (**c**) ground truth (GT): simulated changes; (**d**) k-means results; (**e**) DC3DCD with the *Encoder Fusion SiamKPConv* architecture results; (**f**) DC3DCD with the *Encoder Fusion SiamKPConv* architecture and 10 hand-crafted input features.

though the same ground truth-guided mapping step has been performed. Indeed, here buildings are not set as new in the ground truth and in DC3DCD, conversely to k-means. Finally, on this tile, if we compare to the ground truth, DC3DCD obtains 55.91% of mIoU$_{ch}$, while the k-means only 24.63%.

4 Discussion

In this section, we have proposed an unsupervised change detection method with a weakly user-guided mapping to real classes providing interesting results. This problem is still open and complex and in the following, we point out some observations and discussions about possible improvements.

4.1 Importance of network's architectures and input features

We saw in the result section that the choice of the back-bone architecture and the addition of hand-crafted features as input along with 3D point coordinates are crucial. This is in agreement with the original publication of DeepCluster, where authors provided gradient of images as input to obtain interesting results [Caron et al., 2018]. These results in an unsupervised context also emphasizes conclusions of de Gélis et al. [2023] on the necessity of applying convolution on features difference. To explain this, let us note that the unsupervised context is a largely unconstrained problem. While the annotation allows counterbalancing architectures weaknesses, this is indeed no longer possible for the unsupervised setting. Thereby the choice of an architecture that more specially extracts change-related features through convolutions of difference of features from both inputs at multiple scales, and the addition of well-designed hand-crafted features, allows guiding the training of the network toward a relevant minimum, leading to a reliable change segmentation.

	Method	mAcc (%)	mIoU$_{ch}$ (%)
Supervised	Siamese KPConv [de Gélis et al., 2023]	85.65 ± 1.55	72.95 ± 2.05
	Encoder Fusion SiamKPConv [de Gélis et al., 2023]	**90.26** ± 0.22	**75.00** ± 0.74
	DSM-Siamese	50.87 ± 1.15	30.96 ± 2.48
	DSM-FC-EF	71.47 ± 1.43	45.57 ± 0.98
	RF [Tran et al., 2018]	47.94 ± 0.02	29.45 ± 0.02
WS	k-means	70.07 ± 0.56	53.12 ± 0.79
	DC3DCD *Encoder Fusion SiamKPConv* (with input features)	**83.18** ± 1.10	**66.69** ± 2.19

Table 3: **Qualitative assessment of DC3DCD on the manually annotated sub-part of AHN-CD dataset.** *Top*: supervised methods. DSM-based methods are adaptation of Daudt et al. [2018] networks to DSM inspired by Zhang et al. [2019] and RF refers to Random Forests. In supervised settings, the training is performed on the semi-automatically annotated AHN-CD dataset containing some errors (see de Gélis et al. [2023]). *Bottom*: Weakly supervised methods with k-means and our proposed DC3DCD with *Encoder Fusion SiamKPConv* architecture and with the addition of 10 hand-crafted features as input to the network.

	Method	Per class IoU (%)			
		Unchanged	New building	Demolition	New clutter
Supervised	Siamese KPConv [de Gélis et al., 2023]	89.75 ± 2.18	82.77 ± 5.38	86.44 ± 0.88	**46.65** ± 0.16
	Encoder Fusion SiamKPConv [de Gélis et al., 2023]	**94.79** ± 0.34	**95.31** ± 1.95	**88.87** ± 1.59	41.16 ± 1.30
	DSM-Siamese	77.10 ± 1.51	76.77 ± 0.79	4.91 ± 8.33	11.20 ± 1.71
	DSM-Pseudo-Siamese	78.00 ± 5.09	75.32 ± 8.59	47.46 ± 11.92	23.76 ± 0.56
	DSM-FC-EF	70.77 ± 1.13	90.32 ± 0.61	30.58 ± 1.76	15.81 ± 0.81
	RF [Tran et al., 2018]	78.24 ± 0.00	74.64 ± 0.03	0.00 ± 0.00	13.72 ± 0.06
WS	k-means	84.13 ± 0.49	83.13 ± 0.89	55.40 ± 0.50	20.84 ± 1.00
	DC3DCD *Encoder Fusion SiamKPConv* (with input features)	**91.34** ± 1.21	**89.91** ± 0.72	**69.52** ± 4.97	**40.63** ± 0.97

Table 4: **Per class IoU DC3DCD results on the manually annotated testing part of AHN-CD dataset** given in %. *Top*: supervised methods. In supervised settings, the training is performed on the semi-automatically annotated AHN-CD dataset containing some errors (see de Gélis et al. [2023]). *Bottom*: Weakly supervised methods with k-means and our proposed DC3DCD with *Encoder Fusion SiamKPConv* architecture and with the addition of 10 hand-crafted features as input to the network.

4.2 Improving DC3DCD with contrastive learning

As mentioned, the problem in an unsupervised setting is to train a network to extract appropriate features for a specific task. In a task involving comparison of similar and dissimilar data (like change detection task), the contrastive loss is often used to force the network to extract identical features for similar data. Therefore, an idea can be to force the network to predict similar features for unchanged areas. To test this principle, we propose to introduce the following contrastive term in the loss function:

$$\mathcal{L}_{cont} = 0.5 \times y_{sim} \times F_{CD}^2 \quad \text{with} \quad y_{sim} = \begin{cases} 1 & \text{if similar} \\ 0 & \text{else.} \end{cases} \tag{5}$$

where F_{CD} is the L_2-norm of output features and y_{sim} is the similarity term (set to 1 for unchanged points, and 0 elsewhere). Using this contrastive term in the loss aims at forcing to 0 change-related features in unchanged areas. To test this idea, we combine the contrastive loss in Equation 5 with the deep clustering loss (NLL using the pseudo-label) taking the mean, and train the *Encoder Fusion SiamKPConv* network since it gave the best results.

We first carried out experiments by taking the similarity y_{sim} from the ground truth (as y_{sim} is not available in practice, we first test the idea by taking real values of y_{sim}). Results were really interesting since, as visible in Tables 5 and 6, DC3DCD reached 73.51% of mIoU$_{ch}$ without the use of hand-crafted features and 82.63% of mIoU$_{ch}$ with the use of hand-crafted features on Urb3DCD-V2-1 dataset. We recall that on this dataset and in a fully supervised setting, *Siamese KPConv* and *Encoder Fusion SiamKPConv* networks obtained 80.12% and 85.19% of mIoU$_{ch}$ respectively. Thus, the addition of the contrastive part allows meeting fully supervised results (in an ideal case where y_{sim} is known).

This first experience validated the idea of using the contrastive loss. However in practice, y_{sim} needs to be estimated. To obtain the similarity y_{sim}, we first used the significant changes given by Multi-Scale Model-to-Model Cloud Comparison (M3C2) or a binary thresholding of cloud-to-cloud (C2C) distance. Results are mitigated (see Tables 5 and 6) since the best results obtained using M3C2 for y_{sim} allow us to improve by only 2 points DC3DCD without hand-crafted input features. With hand-crafted input features, results are worsened when the contrastive term is added during the training (using y_{sim} based on M3C2, obtained mIoU$_{ch}$ is 46.42%).

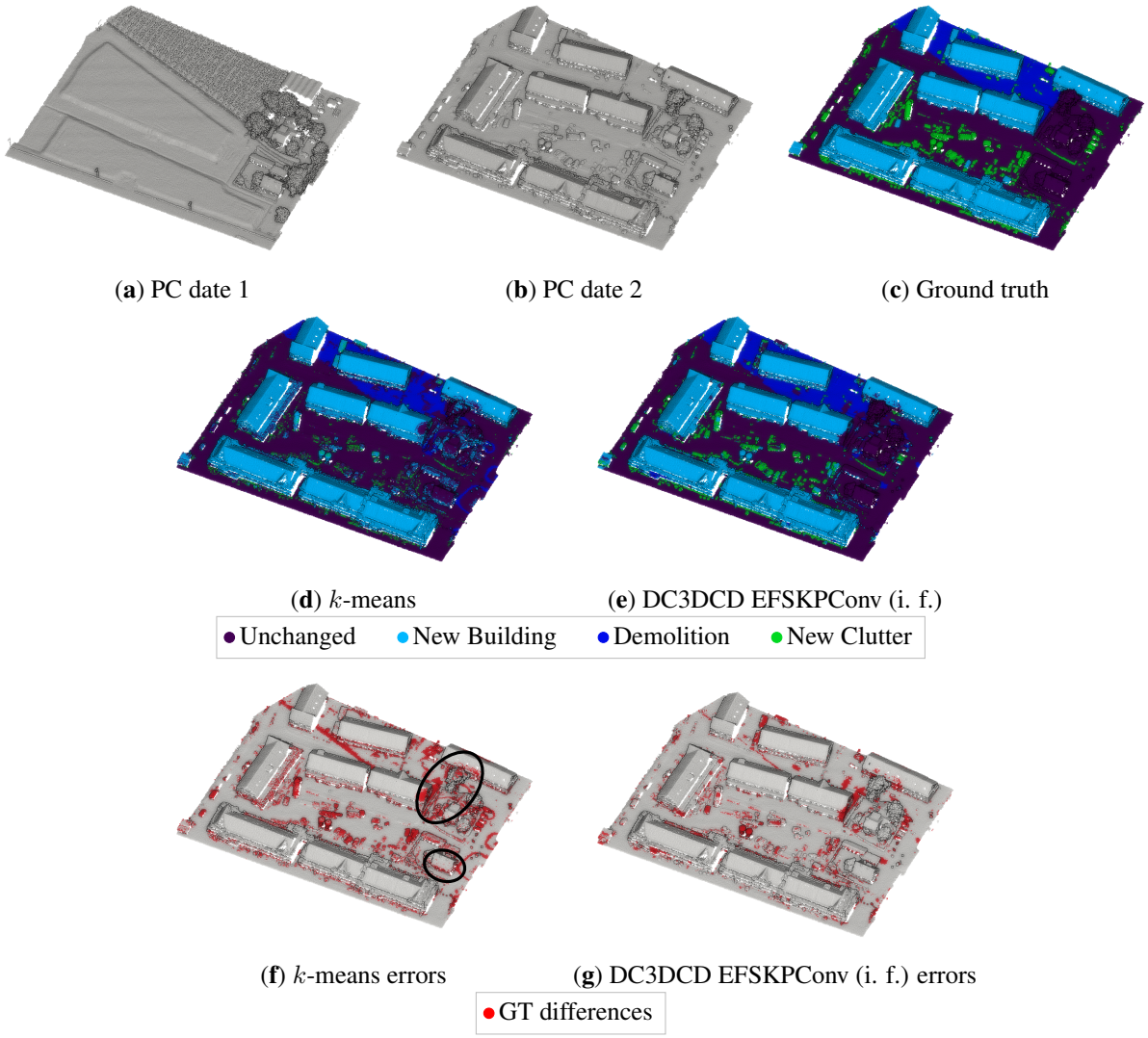


Figure 8: **Qualitative results on the manually annotated sub-part of AHN-CD dataset:** (**a**-**b**) PCs at date 1 and 2; (**c**) ground truth; k-means results (**d**) and errors (**f**); DC3DCD results (**e**) and errors (**g**) using the *Encoder Fusion SiamKPConv* architecture and the 10 hand-crafted features in input. Regions of interest specifically discussed in the text are highlighted with ellipses.

Another idea is to rely on multi-task learning [Vandenhende et al., 2021, Zhang and Yang, 2021]: a multi-task framework based on DC3DCD that jointly extracts mono-date features that can be used for similarity computation has been designed. As illustrated in Figure 10, we added decoders for mono-date semantic segmentation to the back-bone architectures to also obtain semantic segmentation of PCs. Both *Siamese KPConv* and *Encoder Fusion SiamKPConv* have encoders to extract mono-date features, therefore we just added a decoder taking as input these mono-date features instead of feature differences for *Siamese KPConv* for example. Thereby, we used the same idea as before to train the network but with two separate clusterings, performed on output features of the change decoder on one side, as in previous experiments, and on output features of mono-date decoders on the other side. This results in both change pseudo-labels and mono-date pseudo-labels which are used to modulate change encoder-decoder and mono-date encoders-decoders respectively. In practice, we shared trainable parameters between mono-date encoders and decoders. We trained first the semantic segmentation part and then apply a binary clustering (using k-means) on the nearest point mono-date features difference to obtain the similarity y_{sim} used in the contrastive term of the change detection loss. Concerning, the number of pseudo-clusters for mono-date $K_{mono-date}$, 4 and 500 have been tested (4 is the number of semantic segmentation classes in Urb3DCD-V2 dataset, 500 is considered as a sample large bound). While semantic segmentation scores, using the same user-guided mapping for mono-date semantic segmentation, are very promising

	Method	i. f.	y_{sim}	mAcc (%)	mIoU$_{ch}$ (%)
Sup.	SKPConv [de Gélis et al., 2023]			91.21 ± 0.68	80.12 ± 0.02
	EFSKPConv [de Gélis et al., 2023]			**94.23** ± 0.88	**85.19** ± 0.24
Weakly supervised	DC3DCD EFSKPConv			52.30 ± 2.41	37.75 ± 2.11
	DC3DCD-V2 EFSKPConv		GT	83.45 ± 2.22	73.51 ± 3.74
	DC3DCD-V2 EFSKPConv		M3C2	54.01 ± 3.54	39.59 ± 4.18
	DC3DCD-V2 EFSKPConv		C2C	39.74 ± 1.84	25.57 ± 1.97
	DC3DCD-V2 EFSKPConv		Multi-task ($K_{seg.sem.} = 4$)	34.31 ± 5.85	19.21 ± 5.81
	DC3DCD-V2 EFSKPConv		Multi-task ($K_{seg.sem.} = 500$)	47.62 ± 6.76	32.07 ± 6.15
	DC3DCD EFSKPConv	✓		68.45 ± 1.10	57.06 ± 0.41
	DC3DCD-V2 EFSKPConv	✓	GT	**89.04** ± 0.70	**82.63** ± 0.73
	DC3DCD-V2 EFSKPConv	✓	M3C2	58.80 ± 2.14	46.42 ± 2.45
	DC3DCD-V2 EFSKPConv	✓	C2C	42.01 ± 0.67	28.04 ± 0.60
	DC3DCD-V2 EFSKPConv	✓	Multi-task ($K_{seg.sem.} = 4$)	53.04 ± 8.22	38.90 ± 8.51
	DC3DCD-V2 EFSKPConv	✓	Multi-task ($K_{seg.sem.} = 500$)	62.95 ± 1.81	50.14 ± 3.85

Table 5: **Quantitative evaluation of DC3DCD-V2 on Urb3DCD-V2 low density LiDAR dataset.** *Top*: supervised methods. *Middle*: Weakly supervised methods with our proposed DC3DCD and DC3DCD-V2 with Encoder Fusion SiamKPConv architecture without the addition of 10 hand-crafted features as input to the network. *Bottom*: Weakly supervised methods with our proposed DC3DCD and DC3DCD-V2 with Encoder Fusion SiamKPConv architecture and with the addition of 10 hand-crafted features (i. f.) as input to the network.

(90.79% of mean of IoU (mIoU) on the 4 semantic segmentation classes of Urb3DCD-V2 with hand-crafted input features and $K_{mono-date} = 500$), using the associated y_{sim} still leads to unsatisfactory results (see Tables 5 and 6). Indeed, when $K_{mono-date}$ is set to 500, we obtain 50.14% of mIoU$_{ch}$ (with hand-crafted features) which is better than with distance-based methods (M3C2 or C2C) but worse than without the contrastive part of the loss. Thereby, computing similarity from the nearest point mono-date features difference does not seem adapted. Two main reasons for this non-success can be advanced: i) the nearest point comparison is not optimal in occluded parts as well as in dense urban areas (which is the case for Urb3DCD datasets that are acquired on models of Lyon city center), and ii) the method predicts 500 different mono-date semantic classes (far more than existing real classes) and nothing forces that two clusters that are near in semantics (e.g., belonging to the same real class) are near in the feature space, leading to differences of features likely to be very high. To counterbalance this last point, we tested using only 4 mono-date pseudo-clusters, but results are even worse, probably due to the fact that the DeepCluster strategy to train a network required a number of pseudo-clusters greater than the number of real classes.

Qualitative assessment of these results is supported by Figure 7(g-h) and 11. When the similarity comes from the binary change ground truth, visual results really lookalike the multi-change ground truth (Figure 7h and 11f). Qualitative results of multi-task learning are quite encouraging. Surprisingly, borders of 'missing vegetation' seems well retrieved, but the center is still confused with demolition.

All these experiments aim at evaluating the potential of contrastive losses to improve our unsupervised results. Results with the similarity y_{sim} issued from ground truth are very promising since they reach comparable results than the fully supervised networks. However, the method is highly dependent on the quality of this binary change annotation, and in case of mitigated binary annotation, it worsens DC3DCD results. These first perspective experimentation are, to our opinion, encouraging to limit the annotation effort while preserving interesting results. Our future work will concentrate on the estimation of precise y_{sim}.

5 Conclusion

In this paper, we proposed an unsupervised learning method based on the DeepCluster principle to tackle multiclass change segmentation in raw 3D PCs. Following the unsupervised training, we propose a user-guided mapping of pseudo-clusters to real class in order to better to fit to the use case. Given the experiments on both synthetic and real dataset, we saw the importance of the choice of an appropriate architecture to extract valuable change-related features. Also, guiding the network using hand-crafted input features along with 3D points coordinates is advocated. Using these recommended configuration, our proposed method, DeepCluster 3D Change Detection (DC3DCD), allows obtaining better results than a fully supervised traditional machine learning algorithm relying on hand-crafted features and to reach scores of fully supervised deep networks trained on 2.5D rasterization of PCs (i.e., the only existing models in the literature before this thesis). We further proposed to improve DC3DCD by introducing a contrastive loss leading to comparative results as fully supervised deep network (89.04% of mean of accuracy) in an ideal case where the similarity

	Method	i. f.	y_{sim}	Per class IoU (%) Unchanged	New building	Demolition	New veg.	Veg. growth	Missing veg.	Mobile Object
Sup.	SKPConv			95.82 ± 0.48	86.68 ± 0.47	78.66 ± 0.47	93.16 ± 0.27	65.17 ± 1.37	65.46 ± 0.93	91.55 ± 0.60
	EFSKPConv			**97.47** ± 0.04	**96.68** ± 0.30	**82.29** ± 0.16	**96.52** ± 0.03	**67.76** ± 1.51	**73.50** ± 0.81	**94.37** ± 0.54
Weakly supervised	DC3DCD			90.90 ± 0.79	64.06 ± 5.13	54.35 ± 3.84	58.14 ± 20.03	1.45 ± 2.05	0.94 ± 0.78	47.57 ± 2.58
	DC3DCD-V2		GT	97.28 ± 0.10	93.66 ± 1.30	75.24 ± 4.34	83.78 ± 6.00	49.52 ± 7.00	53.60 ± 11.98	85.28 ± 3.54
	DC3DCD-V2		M3C2	93.02 ± 0.03	75.05 ± 5.28	57.43 ± 2.03	69.02 ± 5.77	10.45 ± 8.60	4.47 ± 4.22	21.11 ± 10.22
	DC3DCD-V2		C2C	90.73 ± 0.45	64.93 ± 3.79	56.52 ± 3.74	10.76 ± 5.80	0.41 ± 0.56	0.44 ± 0.51	20.37 ± 6.96
	DC3DCD-V2		Multi-task ($K_{seg.sem.} = 4$)	87.15 ± 2.38	29.55 ± 16.82	39.72 ± 7.84	28.11 ± 2.95	0.00 ± 0.00	0.08 ± 0.09	17.78 ± 11.09
	DC3DCD-V2		Multi-task ($K_{seg.sem.} = 500$)	88.56 ± 2.11	40.44 ± 12.79	45.16 ± 12.73	50.12 ± 6.53	4.03 ± 5.09	0.43 ± 1.19	51.93 ± 6.90
	DC3DCD	✓		93.96 ± 0.11	79.26 ± 0.68	67.88 ± 0.49	75.34 ± 2.81	19.48 ± 4.00	20.29 ± 2.90	80.10 ± 3.16
	DC3DCD-V2	✓	GT	**97.73** ± 0.05	**94.50** ± 0.20	**81.10** ± 0.38	**92.22** ± 0.92	**61.32** ± 2.39	**73.39** ± 1.18	**90.12** ± 1.30
	DC3DCD-V2	✓	M3C2	93.17 ± 0.13	77.48 ± 1.85	65.34 ± 0.55	76.96 ± 5.26	25.65 ± 4.94	0.31 ± 0.54	32.76 ± 4.15
	DC3DCD-V2	✓	C2C	92.25 ± 0.15	70.01 ± 2.77	66.56 ± 1.12	27.11 ± 5.78	0.00 ± 0.00	4.59 ± 0.64	0.00 ± 0.00
	DC3DCD-V2	✓	Multi-task ($K_{seg.sem.} = 4$)	91.96 ± 1.10	69.01 ± 7.24	63.07 ± 1.49	42.76 ± 13.31	4.11 ± 6.71	17.23 ± 7.28	31.20 ± 25.63
	DC3DCD-V2	✓	Multi-task ($K_{seg.sem.} = 500$)	93.68 ± 0.48	78.23 ± 2.94	66.02 ± 1.46	71.49 ± 2.49	0.00 ± 0.00	16.07 ± 4.77	69.06 ± 16.59

Table 6: **Per class quantitative evaluation of DC3DCD-V2 on Urb3DCD-V2 low density LiDAR dataset.** *Top*: supervised methods. *Middle*: Weakly supervised methods with our proposed DC3DCD and DC3DCD-V2 with Encoder Fusion SiamKPConv architecture without the addition of 10 hand-crafted features as input to the network. *Bottom*: Weakly supervised methods with our proposed DC3DCD and DC3DCD-V2 with Encoder Fusion SiamKPConv architecture and with the addition of 10 hand-crafted features (i. f.) as input to the network.

boolean used in the contrastive loss is faultless. However, there are still improvements to be made to find a right manner to obtain the similarity.

Acknowledgements

This research was funded by Magellium, Toulouse and the CNES, Toulouse. This work was granted access to the HPC resources of IDRIS under the allocation 2022-AD011011754R2 made by GENCI.

References

Begüm Demir, Francesca Bovolo, and Lorenzo Bruzzone. Updating land-cover maps by classification of image time series: A novel change-detection-driven transfer learning approach. *IEEE Transactions on Geoscience and Remote Sensing*, 51(1):300–312, 2012.

Laigen Dong and Jie Shan. A comprehensive review of earthquake-induced building damage detection with remote sensing techniques. *ISPRS Journal of Photogrammetry and Remote Sensing*, 84:85–99, 2013.

R. Qin, J. Tian, and P. Reinartz. 3D change detection – approaches and applications. *ISPRS Journal of Photogrammetry and Remote Sensing*, 122:41–56, 2016.

Corné Van Der Sande, Sylvie Soudarissanane, and Kourosh Khoshelham. Assessment of relative accuracy of ahn-2 laser scanning data using planar features. *Sensors*, 10(9):8198–8214, 2010.

M Bernard, D Decluseau, L Gabet, and P Nonin. 3d capabilities of pleiades satellite. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci*, 39:B3, 2012.

Soubirane Jérôme. Shaping the future of earth observation with pléiades neo. In *2019 9th International Conference on Recent Advances in Space Technologies (RAST)*, pages 399–401. IEEE, 2019.

L Lebègue, E Cazala-Hourcade, F Languille, S Artigues, and O Melet. Co3d, a worldwide one-meter accuracy dem for 2025. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, 2020.

Uwe Stilla and Yusheng Xu. Change detection of urban objects using 3d point clouds: A review. *ISPRS Journal of Photogrammetry and Remote Sensing*, 197:228–255, 2023.

U. Okyay, J. Telling, C.L. Glennie, and W.E. Dietrich. Airborne lidar change detection: An overview of earth sciences applications. *Earth-Science Reviews*, 198:102929, 2019.

Z. Zhang, G. Vosselman, M. Gerke, D. Tuia, and M. Y. Yang. Change detection between multimodal remote sensing data using siamese cnn. *arXiv preprint arXiv:1807.09562*, 2018a.

Z. Zhang, G. Vosselman, M. Gerke, C. Persello, D. Tuia, and M.Y. Yang. Detecting building changes between airborne laser scanning and photogrammetric data. *Remote sensing*, 11(20):2417, 2019.

Balázs Nagy, Lóránt Kovács, and Csaba Benedek. Changegan: A deep network for change detection in coarsely registered point clouds. *IEEE Robotics and Automation Letters*, 6(4):8277–8284, 2021.

Tao Ku, Sam Galanakis, Bas Boom, Remco C. Veltkamp, Darshan Bangera, Shankar Gangisetty, Nikolaos Stagakis, Gerasimos Arvanitis, and Konstantinos Moustakas. Shrec 2021: 3d point cloud change detection for street scenes. *Computers & Graphics*, 99:192–200, 2021. ISSN 0097-8493. doi:https://doi.org/10.1016/j.cag.2021.07.004. URL `https://www.sciencedirect.com/science/article/pii/S0097849321001369`.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions On Graphics (TOG)*, 38(5):1–12, 2019.

Iris de Gélis, Sébastien Lefèvre, and Thomas Corpetti. Siamese kpconv: 3d multiple change detection from raw point clouds using deep learning. *ISPRS Journal of Photogrammetry and Remote Sensing*, 197:274–291, 2023. ISSN 0924-2716. doi:https://doi.org/10.1016/j.isprsjprs.2023.02.001. URL `https://www.sciencedirect.com/science/article/pii/S0924271623000394`.

Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6411–6420, 2019.

Iris de Gélis, Thomas Corpetti, and Sébastien Lefèvre. Change detection needs change information: improving deep 3d point cloud change detection. *arXiv preprint arXiv:2304.12639*, 2023. doi:10.48550/arXiv.2304.12639.

Abderrazzaq Kharroubi, Florent Poux, Zouhair Ballouch, Rafika Hajji, and Roland Billen. Three dimensional change detection using point clouds: A review. *Geomatics*, 2(4):457–485, 2022.

Wen Xiao, Hui Cao, Miao Tang, Zhenchao Zhang, and Nengcheng Chen. 3d urban object change detection from aerial and terrestrial point clouds: A review. *International Journal of Applied Earth Observation and Geoinformation*, 118: 103258, 2023.

Sudipan Saha, Francesca Bovolo, and Lorenzo Bruzzone. Unsupervised deep change vector analysis for multiple-change detection in vhr images. *IEEE Transactions on Geoscience and Remote Sensing*, 57(6):3677–3693, 2019.

Iris de Gélis, Sudipan Saha, Muhammad Shahzad, Thomas Corpetti, Sébastien Lefèvre, and Xiao Xiang Zhu. Deep unsupervised learning for 3d als point clouds change detection. *arXiv preprint arXiv:2305.03529*, 2023. doi:10.48550/arXiv.2305.03529.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

Yazhou Ren, Jingyu Pu, Zhimeng Yang, Jie Xu, Guofeng Li, Xiaorong Pu, Philip S Yu, and Lifang He. Deep clustering: A comprehensive survey. *arXiv preprint arXiv:2210.04142*, 2022.

Sheng Zhou, Hongjia Xu, Zhuonan Zheng, Jiawei Chen, Jiajun Bu, Jia Wu, Xin Wang, Wenwu Zhu, Martin Ester, et al. A comprehensive survey on deep clustering: Taxonomy, challenges, and future directions. *arXiv preprint arXiv:2206.07579*, 2022.

Jang Hyun Cho, Utkarsh Mall, Kavita Bala, and Bharath Hariharan. Picie: Unsupervised semantic segmentation using invariance and equivariance in clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16794–16804, 2021.

Puzhao Zhang, Maoguo Gong, Hui Zhang, Jia Liu, and Yifang Ban. Unsupervised difference representation learning for detecting multiple types of changes in multitemporal remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 57(4):2277–2289, 2018b.

Huihui Dong, Wenping Ma, Licheng Jiao, Fang Liu, and LingLing Li. A multiscale self-attention deep clustering for change detection in sar images. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–16, 2021.

Sudipan Saha, Patrick Ebel, and Xiao Xiang Zhu. Self-supervised multisensor change detection. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–10, 2021.

Jinming Zhang, Xiangyun Hu, and Hengming Dai. Unsupervised learning of als point clouds for 3-d terrain scene clustering. *IEEE Geoscience and Remote Sensing Letters*, 19:1–5, 2021.

J MacQueen. Classification and analysis of multivariate observations. In *5th Berkeley Symp. Math. Statist. Probability*, pages 281–297. University of California Los Angeles LA USA, 1967.

Frank Lin and William W Cohen. Power iteration clustering. In *Proceedings of the 27th International Conference on Machine Learning*, 2010.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI*, pages 69–84. Springer, 2016.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi:10.1109/CVPR.2009.5206848.

Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33: 9912–9924, 2020.

Mathilde Caron. *Self-supervised learning of deep visual representations*. PhD thesis, Université Grenoble Alpes, 2021.

Ahmad Mustapha, Wael Khreich, and Wasim Masr. A deep dive into deep cluster. *arXiv preprint arXiv:2207.11839*, 2022.

T.H.G. Tran, C. Ressl, and N. Pfeifer. Integrated change detection and classification in urban areas based on airborne laser scanning point clouds. *Sensors*, 18(2):448, 2018.

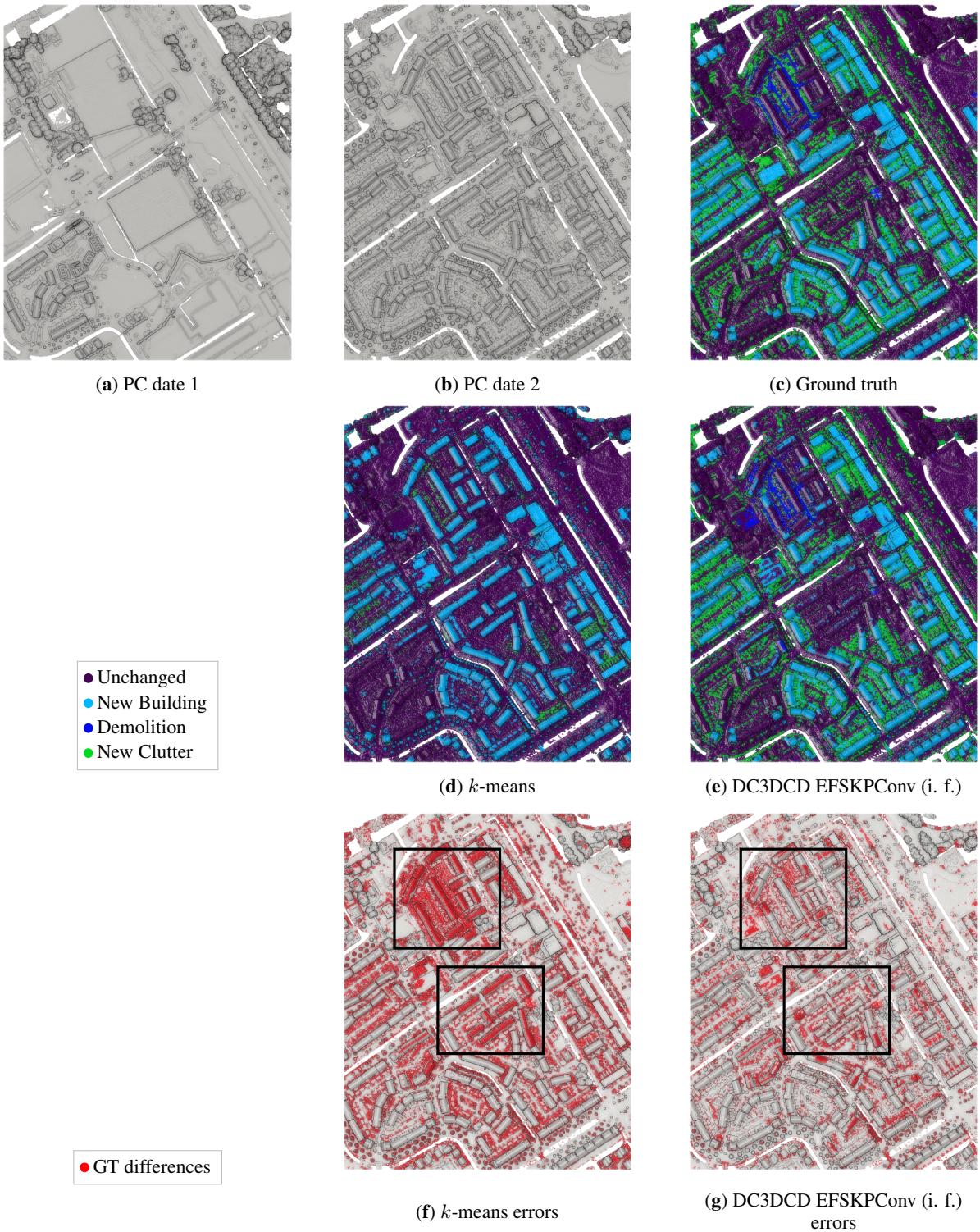
David Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, pages 1177–1178, 2010.

Iris de Gélis, Sébastien Lefèvre, and Thomas Corpetti. Change detection in urban point clouds: An experimental comparison with simulated 3d datasets. *Remote Sensing*, 13(13), 2021. ISSN 2072-4292. doi:10.3390/rs13132629. URL `https://www.mdpi.com/2072-4292/13/13/2629`.

Rodrigo Caye Daudt, Bertrand Le Saux, and Alexandre Boulch. Fully convolutional siamese networks for change detection. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 4063–4067. IEEE, 2018.

Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3614–3633, 2021.

Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609, 2021.



(**a**) PC date 1 (**b**) PC date 2 (**c**) Ground truth

(**d**) k-means (**e**) DC3DCD EFSKPConv (i. f.)

(**f**) k-means errors (**g**) DC3DCD EFSKPConv (i. f.) errors

Figure 9: **Qualitative results on the semi-automatically annotated AHN-CD dataset:** (**a**-**b**) PCs at date 1 and 2; (**c**) ground truth; k-means results (**d**) and corresponding errors (**f**); DC3DCD results (**e**) and corresponding errors (**g**) using the *Encoder Fusion SiamKPConv* architecture and the 10 hand-crafted features in input. Regions of interest specifically discussed in the text are highlighted with rectangles.

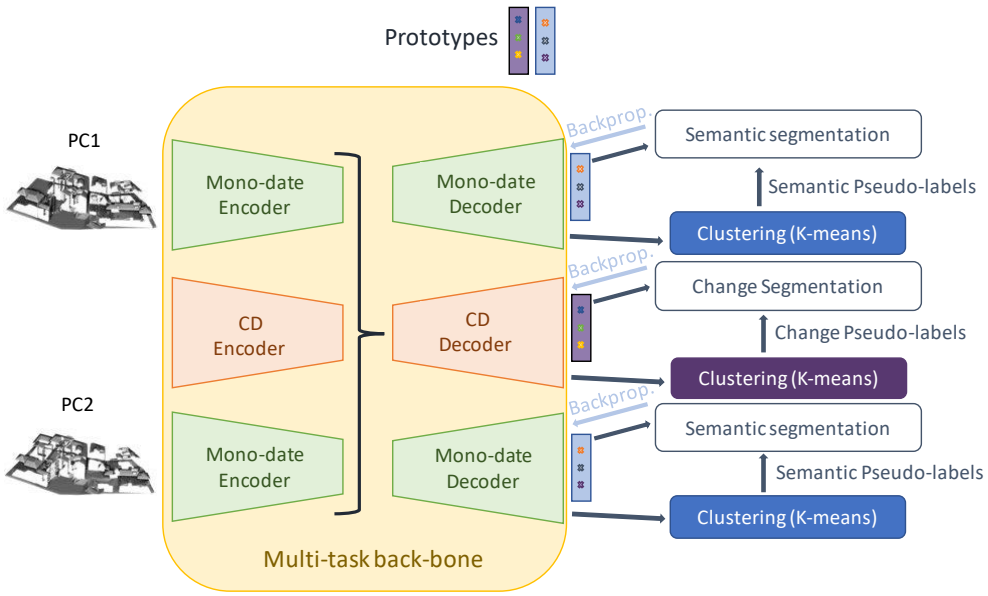


Figure 10: **DC3DCD-V2 using multi-task learning.**

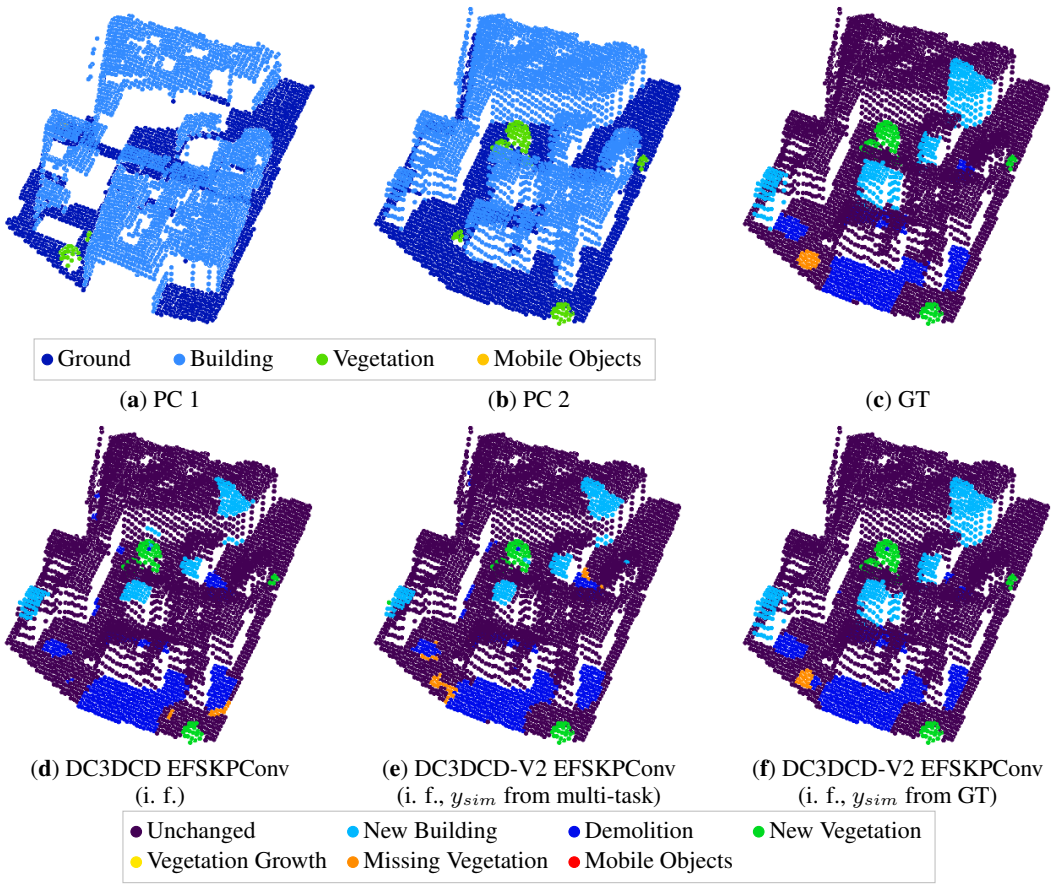
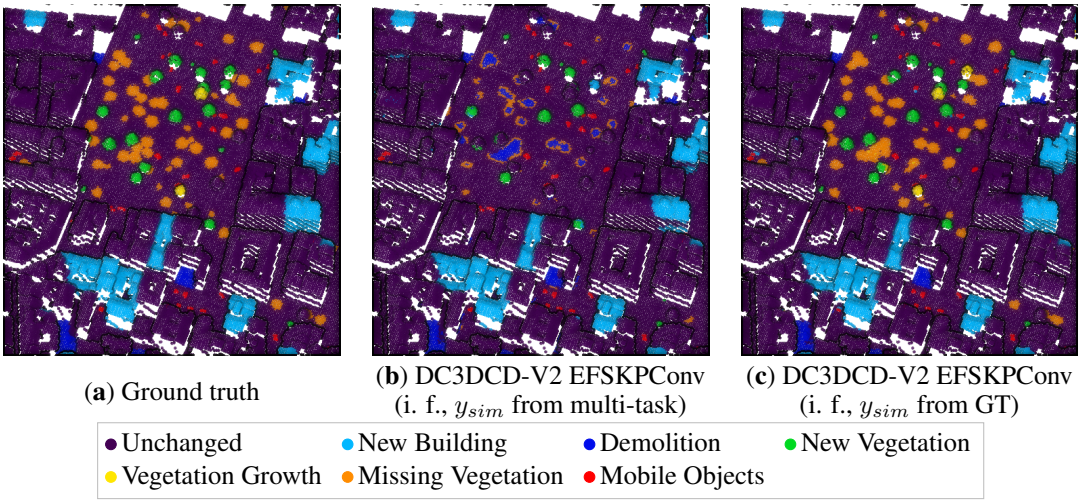


Figure 11: **Visual change detection results on Urb3DCD-V2 low density LiDAR sub-dataset:** (**a**-**b**) the two input point clouds; (**c**) ground truth (GT): simulated changes; (**d**) DC3DCD with the *Encoder Fusion SiamKPConv* architecture and 10 hand-crafted input features (i. f.) results; (**e**) DC3DCD-V2 with the *Encoder Fusion SiamKPConv* architecture, 10 hand-crafted input features results and the similarity y_{sim} computed from the multi-task configuration($K_{mono-date} = 500$); (**f**) DC3DCD-V2 with the *Encoder Fusion SiamKPConv* architecture, 10 hand-crafted input features results and the similarity y_{sim} computed from the ground truth (GT).



(**a**) Ground truth (**b**) DC3DCD-V2 EFSKPConv (i. f., y_{sim} from multi-task) (**c**) DC3DCD-V2 EFSKPConv (i. f., y_{sim} from GT)

Figure 12: **Visual change detection results on Urb3DCD-V2 low density LiDAR sub-dataset (area 2):** (**a**) ground truth (GT): simulated changes; (**b**) DC3DCD-V2 with the *Encoder Fusion SiamKPConv* architecture, 10 hand-crafted input features (i. f.) and the similarity y_{sim} computed from the multi-task configuration($K_{mono-date} = 500$); (**c**) DC3DCD-V2 with the *Encoder Fusion SiamKPConv* architecture, 10 hand-crafted input features and the similarity y_{sim} computed from the ground truth (GT). For comparison, one can refer to Figure 7 providing k-means and DC3DCD results over the same area.