Deep Image Translation with an Affinity-Based Change Prior for Unsupervised Multimodal Change Detection

Luigi Tommaso Luppino, Michael Kampffmeyer,Filippo Maria Bianchi,
Gabriele Moser,Sebastiano Bruno Serpico,Robert Jenssen,and Stian Normann Anfinsen,

Abstract—**Image translation with convolutional neural networks has recently been used as an approach to multimodal change detection. Existing approaches train the networks by exploiting supervised information of the change areas, which, however, is not always available. A main challenge in the unsupervised problem setting is to avoid that change pixels affect the learning of the translation function. We propose two new network architectures trained with loss functions weighted by priors that reduce the impact of change pixels on the learning objective. The change prior is derived in an unsupervised fashion from relational pixel information captured by domain-specific affinity matrices. Specifically, we use the vertex degrees associated with an absolute affinity difference matrix and demonstrate their utility in combination with cycle consistency and adversarial training. The proposed neural networks are compared with state-of-the-art algorithms. Experiments conducted on three real datasets show the effectiveness of our methodology.**

Index Terms—**unsupervised change detection, multimodal image analysis, heterogeneous data, image regression, affinity matrix, deep learning, adversarial networks**

I. INTRODUCTION

A. Background

THE goal of change detection (CD) methods based on earth observation data is to recognise changes on Earth by comparing two or more satellite or aerial images covering the same area at different times [1]. Multitemporal applications include the monitoring of long term trends, such as deforestation, urban planning, and earth resources surveys, whereas bi-temporal applications mainly regard the assessment of natural disasters, for example earthquakes, oil spills, floods, and forest fires [2]. This paper will focus on the latter case, and more specifically on the scenario where the changes must be detected from two satellite images with high to medium spatial resolution (10 to 30 meters). These resolutions allow

Luigi Tommaso Luppino, Michael Kampffmeyer, Robert Jenssen and Stian Normann Anfinsen are with the Machine Learning Group, Department of Physics and Technology, UiT The Arctic University of Norway, 9037 Tromsø, Norway (e-mail: luigi.t.luppino@uit.no).

Filippo Maria Bianchi is with the Department of Mathematics and Statistics, UiT The Arctic University of Norway, 9037 Tromsø, Norway and NORCE (the Norwegian Research Centre), 5008 Bergen, Norway.

Gabriele Moser and Sebastiano Bruno Serpico are with DITEN Department, University of Genoa, 16145 Genoa, Italy.

to detect changes in ground coverage (forest, grass, bare soil, water etc.) below hectare scale, but are not suitable to deal with changes affecting small objects on meter scale (buildings, trees, cars etc.). At these resolutions it is common to assume that co-registration can be achieved by applying simple image transformations such as translation, rotation, and re-sampling [3], [4], [5], [6]. This means that each pixel in the first image and its corresponding one in the second image represent the same point on the Earth. Consequently, even a simple pixel-wise operation (e.g. a difference or a ratio) would highlight changes when working with homogeneous data [4], [7], [8], i.e. data collected by the same sensor, under the same geometries and seasonal or weather conditions, and using the same configurations and settings. More robust and efficient approaches consider complex algorithms rather than simple mathematical operations to detect changes, and many examples of homogeneous CD methods can be found in the literature [8], [9], [10], [11], [12].

B. Motivation

To rely on only one data acquisition modality represents a limitation, both in terms of response time to sudden events and in terms of temporal resolution when monitoring long-term trends. To exemplify, heterogeneous change detection algorithms facilitate rapid change analyses by being able to utilise the first available image, regardless of modality [13], [14]. They also allow to increase the number of samples in a time series of acquisitions by inserting images from multiple sensors. On one hand, this allows to exploit the images acquired by all the available sensors, but on the other hand raises additional challenges. Heterogeneous sensors usually measure different physical quantities, meaning that one terrain type might be represented by dissimilar statistical models from sensor to sensor, while surface signatures and their internal relations may change completely across different instruments [4], [7], [15]. For example, optical and synthetic aperture radar (SAR) payloads are dominantly used for CD in remote sensing [16], [17] and they are often seen as complementary: the use of optical instruments is affected by solar illumination and limited to low cloud coverage, whilst SAR can operate at any time and under almost any weather conditions, because clouds are transparent to electromagnetic waves at SAR frequencies. On the other hand, optical data take real values affected by a modest additive Gaussian noise

(mainly due to atmospheric disturbance, thermal and shot noise inside the sensor), whose effect can be easily accounted for [18], whereas SAR feature vectors take complex values representing the coherent sum of the backscattered echoes, which can present high fluctuations from one pixel to the next both in amplitude and phase, resulting in the so-called speckle, a multiplicative effect which is more challenging to mitigate [19]. In few words, it is not guaranteed that the data acquired by heterogeneous sources lie in a common domain, and a direct comparison is meaningless without processing and co-calibrating the data first [2].

Heterogeneous CD methods are meant to cope with these issues, and as discussed in [20], [21], there is not a unique way to categorize them. However, two general criteria to group them are the following: 1) unsupervised methods or supervised methods; 2) deep learning methods or traditional signal processing methods. The analysis in this paper will exclusively cover unsupervised frameworks. Since they do not require any supervised information about the change, they are usually more appealing than the supervised counterparts. Indeed, collecting labelled data is often costly and nontrivial, both in terms of the time and competence required [3], [17]. Concerning the second distinction, deep learning has become the state-of-the-art in many image analysis tasks, including in the field of remote sensing [4], [6]. Deep learning methods can achieve high performance thanks to the flexibility of neural networks, which are able to apply highly nonlinear transformations to any kind of input data. For these reasons, the analysis of the literature will mainly focus on deep learning, although many important methods, based on minimum energy [22], non-linear regression [21], dictionary learning [20], [23], manifold learning [24], fractal projections [25], or copula theory [26] are worth mentioning. We refer the interested readers to [21] for a state-of-the-art analysis on heterogeneous CD based on more classical methods.

We point out that heterogeneous CD can be framed within the general context of multimodal data fusion, which broadly encompasses all processing, learning, and analysis methodologies aimed at jointly exploiting different data modalities. In remote sensing, these modalities most typically correspond to different sensors, missions, spatial resolutions, or acquisition properties (e.g., incidence angle, radar polarization, and spectral channels) [27]. Note that heterogeneous CD methods are effective also to deal with the simpler case in which the heterogeneity between the images is merely due to different environmental conditions at the moment of the acquisitions (weather, time of the day, season, and so forth). We refer the reader to the review paper in [27] for a general taxonomy of multimodal fusion in remote sensing, with examples of multiresolution, multiangular, multisensor, multitemporal, and spatial-spectral fusion using a variety of methodological approaches, including deep learning and also discussing a CD case study. In the case of image classification, recent examples of multimodal approaches based on deep neural networks include the multimodal deep learning framework in [28], the multisensor and multiscale method in [29] for semantic labeling in urban areas, and the technique in [30] for land cover mapping from multimodal satellite image time series.

The role of shallow and deep learning approaches in the area of feature extraction – with focus on hyperspectral imagery and involving various data fusion concepts – has recently been reviewed in [31]. The scientific outcome of a recent international contest in the area of multimodal fusion with open satellite and ancillary/geospatial data has been presented in [32].

C. Proposed method

We propose to combine traditional machine learning and pattern recognition techniques with deep image translation architectures to perform unsupervised CD based on heterogeneous remote sensing data. More specifically, a comparison of domain-specific affinity matrices allows us to retrieve in a self-supervised manner the *a priori* change indicator, referred to as the prior, driving the training process of our deep learning methods. In particular, our aim is to provide a reliable and informative prior, representative of the whole feature space, which is an alternative with respect to other priors previously used for heterogeneous CD, such as randomly initialised change maps, clustering/post-classification-comparison outputs, or supervised sample selection. The proposed prior computation method is an efficient approach that provides more useful information than randomly-initialised change maps, which are associated with convergence problems and inconsistent overall performance. It is directly and automatically obtainable from the input data without need of any tuning and, as opposed to clustering methods, it does not require to select sensible hyperparameters such as the number of clusters, which strictly depends on the area under investigation and the number of land covers present in the scene. The advantage with respect to post-classification and supervised sample selection is that the latter make use of prior information which can be difficult to obtain, or user prompt or in-situ measurements, which are time-consuming and/or expensive. Instead, none of the aforementioned information are required by the proposed approach.

Two architectures are proposed: The X-Net is composed of two fully convolutional networks, each dedicated to mapping the data from one domain to the other; The ACE-Net consists of two autoencoders whose code spaces are aligned by adversarial training. Their performance and consistency are tested against two recent state-of-the-art methods on three benchmark datasets, illustrating how the proposed networks perform favourably as compared to them. Summing up, the main contributions of this work are:

- A novel procedure to obtain a priori information on structural changes between the images based on a comparison of intramodal information on pixel relations.
- Two neural network architectures designed to perform unsupervised change detection, which explicitly incorporate this prior.

Moreover, this work represents a valuable contribution to the field of study as the proposed framework for heterogeneous change detection is made publicly available at this link: https://github.com/llu025/Heterogeneous_CD, together with the re-implementation of the two reference methods, as well as the three datasets used in this paper.

The remainder of this article is structured as follows: Section II describes the theoretical background and the related work. Section III introduces the reader to the notation, the proposed procedure and the architectures. Results on three datasets are presented in Section IV. Section V includes a discussion of the main features and drawbacks of each method used in this work. Section VI concludes the paper and summarises the proposed method and obtained results.

II. RELATED WORK

The most common solution to compare heterogeneous data is to transform them and make them compatible. This is the main reason why many of the heterogeneous CD methods are related to the topics of domain adaptation and feature learning. In the following we list the main deep learning architectures that are found in the heterogeneous CD literature, along with some examples of methods implementing them.

A. Stacked Denoising Autoencoders

1) Background: The autoencoder (AE) is a powerful deep learning architecture which has proven capable of solving problems like feature extraction, dimensionality reduction, and clustering [33]. A denoising AE (DAE) is a particular type of AE trained to reconstruct an input signal that has been artificially corrupted by noise. The stacked denoising autoencoder (SDAE) is probably the most used model to infer spatial information from data and learn new representations and features. SDAEs are trained following the same procedure as DAEs, but their ability of denoising is learned in a layerwise manner by injecting noise into one layer at the time, starting from the outermost layer and moving on towards the innermost one [34]. In the following, some examples from the heterogeneous change detection literature are presented.

2) Applications: Su *et al.* [35] used change vector analysis to distinguish between three classes: unchanged areas, positive changes and negative changes, as defined in [36]. They exploit two SDAEs to extract relevant features and transfer the data into a code space, where code differences from co-located patches are clustered to achieve a preliminary distinction between samples from the three classes. These samples are then used to train three distinct mapping networks, each of which learns to take the features extracted from one image as input and transform them into plausible code features related to another image. The goal of the first network is to reproduce the expected code from the latter image in case of a positive change, the second aims to do the same in case of a negative change, and the last takes care of the *no-change* case. A pixel is eventually assigned to the class corresponding to the reproduced code showing the smallest difference with the original code from the second image.

In a very similar fashion, Zhang *et al.* [37] first use a spatial details recovery network trained on a manually selected set to coregister the two images, but then extract relevant features from them with two SDAEs trained in an unsupervised fashion. Starting from these transformed images, manual inspection, post-classification comparison or clustering provides a coarse change map. This is used to select examples of unchanged pairs of pixels, which are used to train a mapping network. Once the data are mapped into a common domain, feature similarity analysis highlights change pixels, which are isolated from the rest by segmentation;

In a paper by Zhan *et al.* [17], SAR data are log-transformed and stacked together with the corresponding optical data. Next, a SDAE is used to extract two relevant feature maps from the stack, one for each of the input modalities. These are then clustered separately and the results are compared to obtain a difference image. The latter is segmented into three clusters: pixels certain to belong to changed areas, pixels certain to belong to unchanged areas, and uncertain pixels. Finally, the pixels labelled with certainty are used to train a classification network, which is then able to discriminate the uncertain pixels into the *change* and *no-change* clusters, providing the final binary change map.

Zhan *et al.* [3] proposed to learn new representative features for the two images by the use of two distinct SDAEs. A mapping network is then trained to transform these extracted features into a common domain, where the pixels are forced to be similar (dissimilar) according to their probability of belonging to the unchanged (changed) areas. The probability map is initialised randomly and the training alternates between two phases: updating the parameters of the mapping network according to the probabilities, and updating the map according to the output of the network. Once the training reaches its stopping criterion, the difference between the two feature maps is obtained. Instead of producing a binary change map, this method introduces a hierarchical clustering strategy that highlights different types of change as separate clusters.

The symmetric convolutional coupling network (SCCN) was proposed by Liu *et al.* [4]: After two SDAEs are pretrained separately on each image, their decoders are removed, one of the encoders is frozen, and the other is fine-tuned by forcing the codes of the pixels most likely to not represent changes to be similar. The pixel probability of *no-change* is initialised randomly, and is updated iteratively and alternately together with the parameters of the encoders. A stable output of the objective function is eventually reached and the probability map is finally segmented into the usual binary change map. This method was later improved in [16] by modifying slightly the objective function and the probability map update procedure.

B. Generative Adversarial Networks

1) Background: Among the most important methods in the literature of domain adaptation and data transformation are the generative adversarial networks (GANs). Proposed by Goodfellow *et al.* in [38], these architectures consist of two main components competing against each other. Drawing samples from a random distribution, a generator aims at reproducing samples from a specific target distribution as output. On the other hand, a discriminator has the goal to distinguish between *real* data drawn from the target distribution and *fake* data produced by the generator. Through an adversarial training phase, the generator becomes better at producing fake samples and it is rewarded when it fools the discriminator, whereas the latter improves its discerning skills and is rewarded when

it is able to detect fake data. Both the two parts try to overcome their opponent and become better, benefiting from this competition.

A drawback of this method is the difficulty in balancing the strength of the two components. Their efforts have to be equal, otherwise one will start to dominate the other, hindering the simultaneous improvement of both. Conditional GANs [39] are a particular case, where fake data is generated from a distribution conditioned on the input data. This architecture is suitable for the task of *image-to-image translation*: images from one domain are mapped into another (e.g. drawings or paintings into real pictures, winter landscapes into summer ones, maps of cities into aerial images).

2) Applications: The potential of this method to transform data acquired from one satellite sensor into another is striking, and it was first explored in [40] to match optical and SAR images. The dataset used consists of pairs of co-located optical and SAR images acquired at the same time. The generator learns during training to produce a plausible SAR image starting from the optical one, without knowing what the corresponding real SAR data look like. The same optical image and one of the two SAR images, either the generated or the original, are provided to the discriminator, which has to infer whether the images are a *real* or *fake* pair. For testing, the generator takes the optical images as input and provides the synthetic SAR data, whereas the original SAR data become the ground truth.

In [7], the same concept is applied to perform heterogeneous CD. The scheme is always the same: a generator tries to reproduce SAR patches starting from the corresponding optical ones, and a discriminator aims at detecting these *fake* patches. In order to facilitate a direct comparison, they introduce an approximation network which learns to transform the original SAR patches into the generated ones. Note that the training of all these networks must be carried out on patches not containing change pixels, and any other patch must be flagged and excluded from this process. At first, all the flags are set to *no-change*. Then these steps are iterated: the conditional GAN is updated, the approximation network is tuned accordingly, and finally the generated and approximated patches are compared to flag the ones containing changes. Once the training phase is over, the generated image and the approximated image are pixel-wise subtracted and segmented binarily.

C. Cyclic Generative Adversarial Networks

1) Background: A more complex framework than the conditional GAN is the cycle GAN [41]. The idea is simple: instead of using just one generator-discriminator couple dealing with the transformation from domain $\mathcal{X}$ to domain $\mathcal{Y}$, another tandem generator-discriminator is added to do the vice versa. This means that the framework can be tested for so-called *cycle consistency*: It should be possible to perform a composite translation of data from domain $\mathcal{X}$ to domain $\mathcal{Y}$, and then onwards to domain $\mathcal{X}$ (denoted $\mathcal{X} \to \mathcal{Y} \to \mathcal{X}$), and the full translation cycle should reproduce the original input. Equivalently, the cycle $\mathcal{Y} \to \mathcal{X} \to \mathcal{Y}$ should reproduce the original input in domain $\mathcal{Y}$.

In [42], this framework is applied and extended further: Along with the two input domains $\mathcal{X}$ and $\mathcal{Y}$, a latent space $\mathcal{Z}$ is introduced in between them. Data from the original domains are transformed to $\mathcal{Z}$, where they should ideally not be discernible. Thus, four generators are used to map data across domains: from $\mathcal{X}$ to $\mathcal{Z}$, from $\mathcal{Z}$ to $\mathcal{Y}$, from $\mathcal{Y}$ to $\mathcal{Z}$, and from $\mathcal{Z}$ to $\mathcal{X}$. The accurate reconstruction of the images is the first enforced principle: Data mapped from domain $\mathcal{X}$ ($\mathcal{Y}$) to $\mathcal{Z}$ must be mapped back correctly to $\mathcal{X}$ ($\mathcal{Y}$). The next requirement is cycle-consistency: Starting from $\mathcal{X}$ ($\mathcal{Y}$) and going first to $\mathcal{Z}$ and then to $\mathcal{Y}$ ($\mathcal{X}$), the images must go back to $\mathcal{X}$ ($\mathcal{Y}$) passing through $\mathcal{Z}$ again and match exactly with the original input. Concerning the discriminators, there are three: one should distinguish whether data mapped into $\mathcal{Z}$ come originally from $\mathcal{X}$ or $\mathcal{Y}$; another discriminates between original images from $\mathcal{X}$ and images which started from $\mathcal{Y}$ and performed half a cycle; the third does the same in domain $\mathcal{Y}$.

2) Applications: Inspired by these concepts, Gong *et al.* proposed the coupling translation networks to perform heterogeneous CD [15]. However, their architecture is simpler. Two variational AEs are combined so that their encoders separately take as input optical and SAR patches, respectively, and the two codes produced are stacked together. The stacked code is then decoded by both decoders and each of those yields two output patches: one is the reconstruction of the input patch from the same domain, the other is the transformation of the input patch from the opposite domain. The latter must be detected by a discriminator which is taught to discern reconstructed data from *fake* transformed data. This framework has only two discriminators, one after each decoder, whereas the code spaces of the two AEs are aligned throughout the training, eventually becoming the common latent domain, namely $\mathcal{Z}$. Together with the adversarial loss, the reconstruction and the cycle-consistency drive the learning process, which enables the two networks to translate data across domains, such that a direct comparison is feasible.

In the following section we explain how our methodology fits in this picture, framed in-between cycle-consistency and adversarial training.

III. METHODOLOGY

The same geographical region is scanned by two sensors whose pixel measurements lie in domains $\mathcal{X}$ and $\mathcal{Y}$, respectively. The first sensor captures an image $\mathcal{I}_\mathcal{X} \in \mathcal{X}^{H \times W}$ at time t_1, and the other sensor an image $\mathcal{I}_\mathcal{Y} \in \mathcal{Y}^{H \times W}$ at time t_2. H and W denote the common height and width of the images, that are obtained through coregistration and resampling. The feature spaces $\mathcal{X}$ and $\mathcal{Y}$ have dimensions $|\mathcal{X}|$ and $|\mathcal{Y}|$.

We further assume that a limited part of the image has changed between time t_1 and t_2. The final goal of the presented method is to transform data consistently from one domain to the other. To do so, it is crucial to learn a one-to-one mapping between the land cover signatures of one domain and the corresponding signatures in the other. Since no prior information is available, a reasonable option is to learn a mapping from every pixel in $\mathcal{I}_\mathcal{X}$ to the corresponding pixel in $\mathcal{I}_\mathcal{Y}$ and vice versa.

A possibility would be to train two regression functions

$$\hat{Y} = F(X) : \mathcal{X}^{h \times w} \to \mathcal{Y}^{h \times w}$$
$$\hat{X} = G(Y) : \mathcal{Y}^{h \times w} \to \mathcal{X}^{h \times w}$$

to map image patches $X \in \mathcal{X}^{h \times w} \subseteq \mathcal{I_X}$ and $Y \in \mathcal{Y}^{h \times w} \subseteq \mathcal{I_Y}$ between the image domains by using the entire images $\mathcal{I_X}$ and $\mathcal{I_Y}$ as training data. However, the presence of areas affected by changes would distort the learning process, because they would promote a transformation from one land cover in one domain to a different land cover in the other domain. For example, forests and fire scars may be erroneously connected, as may land and flooded land. To reduce the impact of these areas on training, we first perform a preliminary analysis to highlight changes. Then, the contribution of each pixel to the learning process is inversely weighted with a score expressing the chance of it being affected by a change. In this section, we first describe the algorithm providing the preliminary change analysis. We then propose two deep learning architectures and, finally, explain how they can exploit the prior computed in the change analysis.

A. Prior computation

To compute a measure of similarity between multimodal samples based on affinity matrices, we adopt an improved version of the original method proposed in our previous work [21]. Please notice that the following procedure is totally unsupervised and does not require any ancillary information or knowledge about the data nor about the acquiring sensors.

A $k \times k$ sliding window covers an area p of both $\mathcal{I_X}$ and $\mathcal{I_Y}$, from which a pair of corresponding patches X and Y are extracted. X_i (Y_i) and X_j (Y_j) stand for feature vector i and j of patch X (Y), with $i, j \in \{1, \ldots, k^2\}$. The distance between a pixel pair (i, j) is defined as $d_{i,j}^m$, where the modality $m \in \{\mathcal{X}, \mathcal{Y}\}$ depends on whether the samples are taken from X or Y. The appropriate choice of distance measure depends on the domain and the underlying data distribution. The hypothesis of Gaussianity for imagery acquired by optical sensors is commonly assumed [36], [43]. Concerning SAR intensity data, a logarithmic transformation is sufficient to bring it to near-Gaussianity [2], [17]. We use the computationally efficient Euclidean distance, as it is suitable for (nearly) Gaussian data.

Once computed, the distances between all pixel pairs can be converted to affinities, intended as values describing how close two points are in some feature space according to a metric [44], for instance by the Gaussian kernel:

$$A_{i,j}^m = \exp\left\{-\frac{\left(d_{i,j}^m\right)^2}{h_m^2}\right\} \in (0, 1], \quad i, j \in \{1, \ldots, k^2\}. \quad (1)$$

$A_{i,j}^m$ are the entries of the affinity matrix $A^m \in \mathbb{R}^{k^2 \times k^2}$ for the given patch and modality m. Here, the term affinity is used as synonym for similarity, as it has been widely used in the machine learning literature, especially with methods based on graph theory, such as spectral clustering and Laplacian eigenmaps [45], [46], [47], [48], [49], [50]. The kernel width h_m is domain-specific and can be determined automatically.

Our choice is to set it equal to the average distance to the Kth nearest neighbour for all data points in the relevant patch (X or Y), with $K = \frac{3}{4}k^2$. In this way, a characteristic distance within the patch is captured by this heuristic, which is robust with respect to outliers [51]. Silverman's rule of thumb [52] and other common approaches to determine the kernel width have not proven themselves effective in our experimental evaluation, so they were discarded. Once the two affinity matrices are computed, a matrix D holding the element-wise absolute differences $D_{i,j} = |A_{i,j}^{\mathcal{X}} - A_{i,j}^{\mathcal{Y}}|$ can be obtained.

Our previous algorithm [21] would at this point evaluate the Frobenius norm of D and assign its value to all the pixels belonging to p. Then, the $k \times k$ window is shifted one pixel and the procedure is iterated for the set $\mathcal{P}$ of all overlapping patches p that can be extracted from the image. The final result for each pixel is derived by averaging the set $\mathcal{S}^F$ of Frobenius norms obtained with all the patches covering that pixel. Clearly, the loop over the patches in $\mathcal{P}$ is computationally heavy, although when shifting a patch one pixel, most of the already computed pixel distances can be reused. If $N = H \cdot W$ is the total number of pixels in the images, the cardinality of $\mathcal{P}$ is

$$|\mathcal{P}| = (H - k + 1) \cdot (W - k + 1)$$
$$= N - (H + W)(k - 1) + (k - 1)^2. \quad (2)$$

Shifting the sliding window by a factor larger than one will speed up the algorithm, but with the result that the final map of averaged Frobenius norms exhibits an unnatural tile pattern.

To address this issue, we propose to compute the following mean over the rows of D (or columns, since $A^{\mathcal{X}}$ and $A^{\mathcal{Y}}$ are symmetrical, hence so is D):

$$\alpha_i = \frac{1}{k^2} \sum_{j=1}^{k^2} |A_{i,j}^{\mathcal{X}} - A_{i,j}^{\mathcal{Y}}|, \quad i \in \{1, \ldots, k^2\} \quad (3)$$

The main rationale for this operation is that pixels affected by changes are the ones perturbing the structural information captured by the affinity matrices, and so, on average, their corresponding rows in D should present larger values.

We can also choose to look at D as the affinity matrix of a change graph, with change affinities $D_{i,j}$ that indicate whether the relation between pixel i and j has changed. The row sums of D become vertex degrees of the graph that sum the change affinities of individual pixels. A high vertex degree suggests that many pixel relations have changed, and that the pixel itself is subject to a change. The scaling of the vertex degree by $1/k^2$ normalises and fixes the range of α_i to $[0, 1]$, which simplifies both thresholding and probabilistic interpretation. Another advantage of the vertex degree is that it isolates evidence about change for a single pixel, whereas the Frobenius norm of D accumulates indications of change for an entire patch and provides change evidence that is less localised. In conclusion, α_i contains more reliable information and, most importantly, relates only to a single pixel i. It is therefore possible to introduce a shift factor $\Delta > 1$, which on one hand means that the final result becomes an average over a smaller set $\mathcal{S}^\alpha$, but on the other hand speeds up the

computations considerably. Potentially, this shift can be as large as the patch size, reducing the amount of patches by a factor of k^2. However, this is not desirable, since each pixel will be covered only once, leaving us with a set S^α of one element and no room for averaging.

The toy example in Fig. 1 helps to explain the effectiveness of the proposed approach. To make this case easier to explain, Δ is set equal to k: each pixel in the image is covered only once. Fig. 1a simulates a patch X of 8×8 pixels extracted from a SAR image captured at t_1. It consists of four blocks representing four different classes, whose pixel intensities are affected by speckle (large variability associated with the multiplicative signal model of SAR images). The corresponding patch Y extracted from an optical image at t_2 is depicted in Fig. 1b; The same classes are disposed in the same way and the pixel intensities are affected by additive Gaussian noise. Changes are introduced by placing 4 pixels representing each class in the bottom right of each block of Y. In this way, all the possible transitions between one class and the others occur between t_1 and t_2. Clearly, a transition from one class to another represents a change, whereas no change occurs when the same class is present at the two dates. The 64×64 affinity matrices $A^\mathcal{X}$ and $A^\mathcal{Y}$ computed from X and Y are depicted in Fig. 1c and 1d. They both show a regular squared pattern, with high affinities in red and low affinities in blue, which corresponds to the the block structure of X and Y. Moreover, the latter presents the expected irregularities and perturbations due to the introduced changed pixels that are breaking the block pattern in Fig. 1b. Once the change affinity matrix D is evaluated (Fig. 1e), it can be transformed by (3) into the 8×8 image of the prior α_i shown in Fig. 1f, where dark (bright) pixels indicate small (large) values of α_i. This prior image is denoted $\boldsymbol{\alpha}$. Finally, one may retrieve a CD map by thresholding $\boldsymbol{\alpha}$, which in this case matches the ground truth with 100% accuracy, as shown in Fig. 1g by the confusion map where only true positives (white) and true negatives (black) are present.

Algorithm 1 Evaluation of $\boldsymbol{\alpha}$:

---

for all patches $p_\ell, \ell \in \{1, \dots, |\mathcal{P}|\}$ **do**
    Compute $d_{i,j}^m \, \forall i,j \in p_\ell^m, \, m = \boldsymbol{X}, \boldsymbol{Y}$
    Determine $h_\ell^{\boldsymbol{X}}$ and $h_\ell^{\boldsymbol{Y}}$
    Compute $A_{i,j}^m = \exp\left\{-\left(\frac{d_{i,j}^m}{h_\ell^m}\right)^2\right\}, \, m = \boldsymbol{X}, \boldsymbol{Y}$
    Compute $\alpha_{i,\ell} = \frac{1}{k^2} \sum_j |A_{i,j}^{\boldsymbol{X}} - A_{i,j}^{\boldsymbol{Y}}| \, \forall i \in p_\ell$
    Add $\alpha_{i,\ell}$ to the set $\mathcal{S}_i^\alpha \, \forall i \in p_\ell$
end for
for all pixels $i \in \{1, \dots, N\}$ **do**
    Compute $\alpha_i = \frac{1}{|\mathcal{S}_i^\alpha|} \sum_{\{\ell \,|\, \alpha_{i,\ell} \in \mathcal{S}_i^\alpha\}} \alpha_{i,\ell}$
end for

Given the set $\mathcal{P}$ of all the image patches of size $k \times k$ spaced by a step size Δ, Algorithm 1 summarises the procedure to obtain a set of priors $\{\alpha_i\}_{i=1}^N$ for the whole dataset, which can be rearranged into the image $\boldsymbol{\alpha} \in \mathbb{R}^{H \times W}$. For each pixel $i \in \{1, \dots, N\}$ in the image, the mean over $\mathcal{S}_i^\alpha$ is computed,

where $\mathcal{S}_i^\alpha$ is the set of the $\alpha_{i,\ell}$ obtained with all the patches $p_\ell \in \mathcal{P}$ covering pixel i. If Δ is a factor of k, this average is calculated over $\left(k/\Delta\right)^2$ values.

The size k has an important role in the effectiveness of this methodology, because the patches p could be too small or too big to capture the shapes and the patterns within them. To reduce the sensitivity to this parameter, one may suggest to use different values of k for Algorithm 1 and combine the results in an ensemble manner. For example, once k is defined, the method can be applied also for $k_{small} = k/2$ and $k_{big} = 2 \cdot k$. However, the size of the matrices containing first $d_{i,j}^m$ and then $A_{i,j}^m$ exhibits a quadratic growth with respect to k, thus becoming quickly unfeasible in terms of memory usage and computational time. Hence, instead of applying the method to the original images with k_{big}, we suggest to down-sample the images by a factor of 2, apply the algorithm with k, and re-scale the output to the original size. This procedure might introduce artifacts and distortions, but their effects are mitigated when combined with the results obtained with k_{small} and k.

In the following subsections, we explain how to exploit the outcome of Algorithm 1 to train the proposed deep learning architectures in absence of supervision.

B. X-Net: Weighted Translation Network

The main goal of our approach is to map data across two domains. As Fig. 2 illustrates, this means to train a function $F(\boldsymbol{X}) : \mathcal{X}^{h \times w} \to \mathcal{Y}^{h \times w}$ to transform data between the domains of $\boldsymbol{X}$ and $\boldsymbol{Y}$, and a second function $G(\boldsymbol{Y}) : \mathcal{X}^{h \times w} \to \mathcal{Y}^{h \times w}$ to do the opposite. The two mapping functions can be implemented as convolutional neural networks (CNNs). Hence, the training can be carried out by the minimisation of an objective function with respect to the set ϑ of parameters of the two networks. The objective function, commonly referred to as the loss function $\mathcal{L}(\vartheta)$, is defined *ad hoc* and usually consists of a weighted sum of loss terms, where each relates to a specific objective or property that we want from the solution. For this particular framework, we introduce three loss terms. Note that from now on we refer to training patches of much larger size than the patch size k of Section III-A used to compute the affinity-based prior.

1) Weighted translation loss: For a pair of patches $\{\boldsymbol{X}, \boldsymbol{Y}\}$, we want in general the domain translation to satisfy:

$$\begin{aligned} \hat{\boldsymbol{Y}} &= F(\boldsymbol{X}) \simeq \boldsymbol{Y}\,, \\ \hat{\boldsymbol{X}} &= G(\boldsymbol{Y}) \simeq \boldsymbol{X}\,, \end{aligned} \tag{4}$$

where $\hat{\boldsymbol{Y}} = F(\boldsymbol{X})$ and $\hat{\boldsymbol{X}} = G(\boldsymbol{Y})$ stand for the data transformed from one domain into the other. However, pixels that are likely to be changed shall not fulfill the same requirements, i.e., condition (4) should be satisfied in unchanged areas but should not be enforced in changed ones in order not to hinder the capability of the proposed method to discriminate changes. More formally, if H_0 and H_1 indicate the "no-change" and "change" hypotheses, respectively, then in a least mean-square error (MSE) framework, it would be desired that the network

(a) $\boldsymbol{X}$ (SAR) at t_1

(c) $A^{\mathcal{X}}$

(e) $D = |A^{\mathcal{X}} - A^{\mathcal{Y}}|$

(f) Prior image $\boldsymbol{\alpha}$

(b) $\boldsymbol{Y}$ (optical) at t_2

(d) $A^{\mathcal{Y}}$

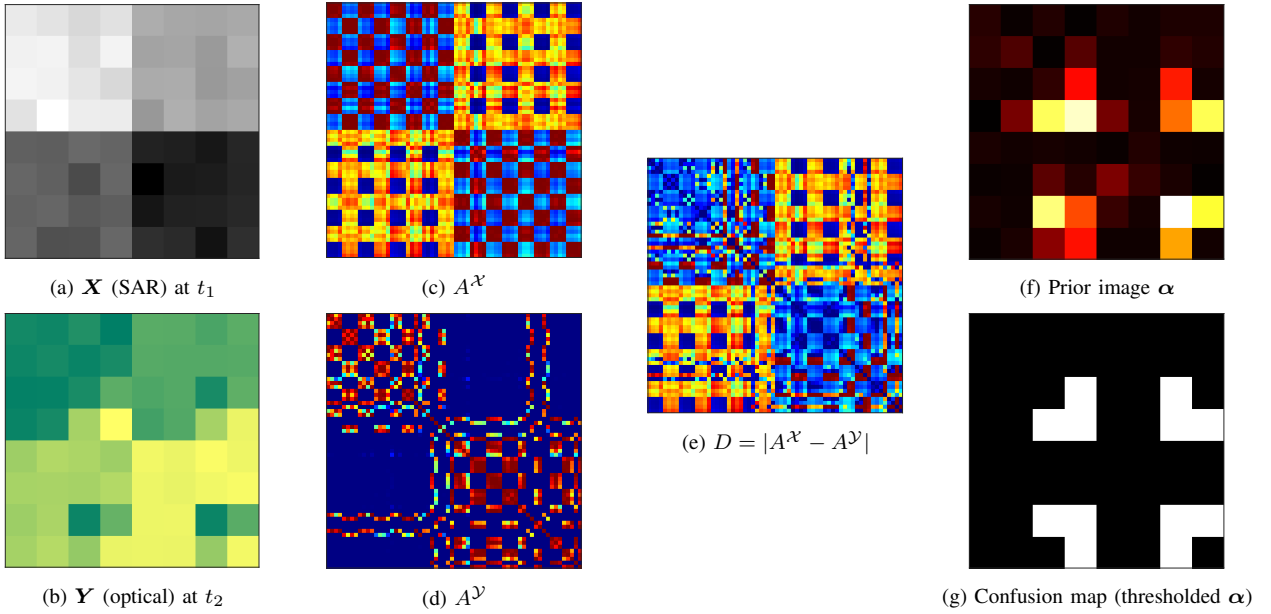(g) Confusion map (thresholded $\boldsymbol{\alpha}$)

Fig. 1: Toy example. a) Patch from the SAR image at time t_1; b) Corresponding patch in the optical image at time t_2; c-e) Affinity matrices and their absolute difference; f) Prior image $\boldsymbol{\alpha}$ obtained from D by applying (3); g) Confusion map obtained by thresholding $\boldsymbol{\alpha}$, with true positives (white) and true negatives (black). Best viewed in colour.

parameters ideally minimized the following MSE conditioned to "no-change":

$$\mathcal{L}_{H_0}(\vartheta) = \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[\delta(\boldsymbol{X},\hat{\boldsymbol{X}})\,\middle|\,H_0\right] + \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[\delta(\boldsymbol{Y},\hat{\boldsymbol{Y}})\,\middle|\,H_0\right],\tag{5}$$

where $\delta(\boldsymbol{A},\boldsymbol{B})$ indicates the squared L_2 distance between two equal-sized $h \times w$ patches $\boldsymbol{A}$ and $\boldsymbol{B}$, i.e.:

$$\delta(\boldsymbol{A},\boldsymbol{B}) = \frac{1}{h \cdot w}\sum_{i=1}^{h \cdot w}\|\boldsymbol{a}_i - \boldsymbol{b}_i\|_2^2.\tag{6}$$

Here, $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$ denote the vectors associated with the i-th pixel in patches $\boldsymbol{A}$ and $\boldsymbol{B}$, respectively ($i = 1, 2, \ldots, h \cdot w$). Estimating the expectations in (5) is straightforward using a training set for H_0, as it has been done in [53]. However, a training set is unavailable in the fully unsupervised scenario that is considered here.

We prove in the Appendix that, under mild conditional independence assumptions, the conditional loss $\mathcal{L}_{H_0}(\vartheta)$ can be equivalently rewritten as:

$$\mathcal{L}_{H_0}(\vartheta) = \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[\delta(\boldsymbol{X},\hat{\boldsymbol{X}}|\boldsymbol{\Psi})\right] + \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[\delta(\boldsymbol{Y},\hat{\boldsymbol{Y}}|\boldsymbol{\Phi})\right],\tag{7}$$

where $\delta(\boldsymbol{A},\boldsymbol{B}|\boldsymbol{W})$ indicates a squared L_2 distance weighted on a vector of weights $\boldsymbol{W} = [W_1, W_2, \ldots, W_{h \cdot w}]^T$, i.e.:

$$\delta(\boldsymbol{A},\boldsymbol{B}|\boldsymbol{W}) = \frac{1}{h \cdot w}\sum_{i=1}^{h \cdot w} W_i\|\boldsymbol{a}_i - \boldsymbol{b}_i\|_2^2,\tag{8}$$

and where $\boldsymbol{\Psi}$ and $\boldsymbol{\Phi}$ are $(h \cdot w)$-dimensional weight vectors whose components are defined in terms of the joint probability distributions of $\boldsymbol{X}$ and $\boldsymbol{Y}$, given H_0 and H_1. Accordingly, the H_0-conditional MSE in (5) is equivalent to an unconditional but suitably weighted MSE. In particular, it is also proven in the appendix that the i-th component of $\boldsymbol{\Psi}$ takes values in the interval:

$$0 < \Psi_i \leq \frac{1}{P(H_0)},\tag{9}$$

where $P(H_0)$ is the prior probability of "no-change." This prior is strictly positive since we assumed at the beginning of this section that the changes affected a limited part of the image. According to a reasoning based on likelihood ratio testing (see Appendix), the lower end $\Psi_i \simeq 0$ suggests that the i-th pixel of the patch is likely changed, and the upper end $\Psi_i \simeq 1/P(H_0)$ suggests that it is likely unchanged ($i = 1, 2, \ldots, h \cdot w$). The same statement holds for the components of $\boldsymbol{\Phi}$ as well. This is consistent with the aforementioned interpretation of the equivalence between the H_0-conditional non-weighted MSE in (5) and the unconditional weighted MSE in (7) because the i-th pixel does not contribute to the loss in (7) when it is likely changed. Vice versa, it gives its maximum contribution when it is likely unchanged.

Without training samples, estimating the expectations in (7) is as difficult as estimating those in (5) because the weight
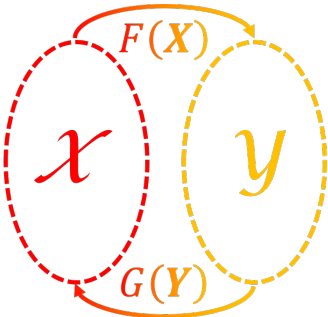

Fig. 2: First proposed framework, where two domains and two transformations which can translate data across them.
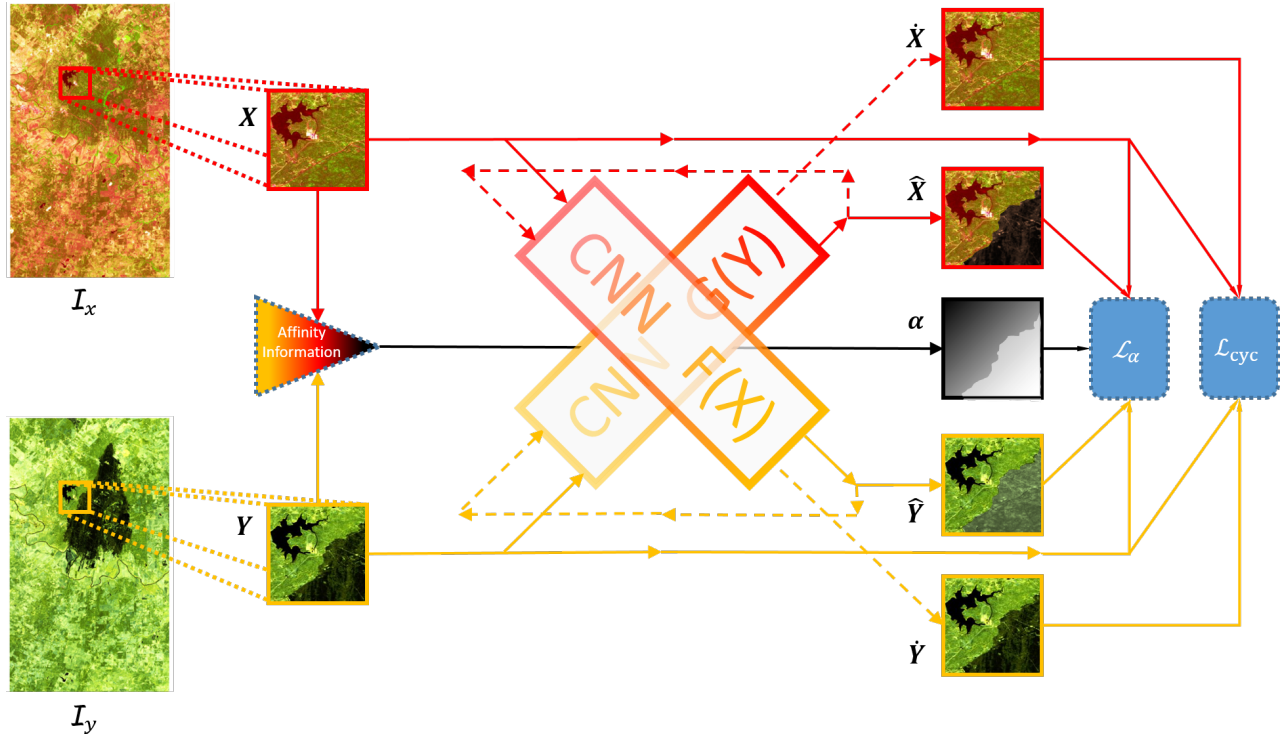
Fig. 3: Data flow of the X-Net. Two CNNs transform data from the domain of $\boldsymbol{X}$ to the domain of $\boldsymbol{Y}$ and vice versa. Solid lines going through them indicate data transferred from one domain to the other, dashed lines indicate data re-transformed back to their original domain.

vectors depend on the joint conditional distributions of $\boldsymbol{X}$ and $\boldsymbol{Y}$. Therefore, in the proposed method, we leverage on the reformulation as a weighted unconditional MSE in (7) to define an approximation of the H_0-conditional MSE in (5) by making use of the affinity prior defined in (3). As discussed in Section III-A, every pixel pair $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}$ will be associated with a precomputed prior, α_i, that measures through affinity reasoning its chances of being changed. We exploit this information to approximate (7) as follows:

$$\mathcal{L}_\alpha(\vartheta) = \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[\delta(\hat{\boldsymbol{X}}, \boldsymbol{X}|\boldsymbol{\Pi})\right] + \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[\delta(\hat{\boldsymbol{Y}}, \boldsymbol{Y}|\boldsymbol{\Pi})\right], \quad (10)$$

where $\boldsymbol{\Pi} = [\Pi(\alpha_1), \ldots, \Pi(\alpha_{h \cdot w})]^T$, and $\Pi(\alpha) : [0, 1] \rightarrow [0, 1]$ is a monotonically decreasing function that maps α_i, measuring the chances of change, into Π_i, that is used to weigh the contribution of the i-th pixel to the loss function ($i = 1, 2, \ldots, h \cdot w$). Specifically, $\Pi(\alpha_i)$ is supposed to be close to zero when the i-th pixel is likely changed (i.e., when $\alpha_i \simeq 1$) and close to one when it is likely unchanged (i.e., $\alpha_i \simeq 0$). Methodologically, the weighted translation loss $\mathcal{L}_\alpha(\vartheta)$ in (10) is meant as an approximation of (7) – and thus of the desired conditional non-weighted MSE in (5) –, up to a positive multiplicative constant equal to $1/P(H_0)$. We use the simple $\Pi(\alpha) = 1 - \alpha$, but other choices can be considered.

2) Cycle-consistency loss: In their seminal work on Cycle-GANs [41], Zhu *et. al* pointed out that domain translations should respect the principle of cycle-consistency: Ideally, if $F(\boldsymbol{X})$ and $G(\boldsymbol{Y})$ are perfectly tuned, it must hold true that

$$\dot{\boldsymbol{X}} = G(\hat{\boldsymbol{Y}}) = G\left(F(\boldsymbol{X})\right) \simeq \boldsymbol{X}, $$
$$\dot{\boldsymbol{Y}} = F(\hat{\boldsymbol{X}}) = F\left(G(\boldsymbol{Y})\right) \simeq \boldsymbol{Y}, \quad (11)$$

where $\dot{\boldsymbol{X}} = G(\hat{\boldsymbol{Y}})$ and $\dot{\boldsymbol{Y}} = F(\hat{\boldsymbol{X}})$ indicate the data re-transformed back to the original domains. Consequently, the cycle-consistency loss term is defined as:

$$\mathcal{L}_{\mathrm{cyc}}(\vartheta) = \mathbb{E}_{\boldsymbol{X}}\left[\delta(\dot{\boldsymbol{X}}, \boldsymbol{X})\right] + \mathbb{E}_{\boldsymbol{Y}}\left[\delta(\dot{\boldsymbol{Y}}, \boldsymbol{Y})\right]. \quad (12)$$

Note that training with the cycle-consistency principle does not require paired data.

3) Total Loss Function: The third and last term of the loss function is a weight decay regularisation term, which reduces overfitting by controlling the magnitude of the network parameters ϑ. The total loss function becomes

$$\mathcal{L}(\vartheta) = \left\{w_{\mathrm{cyc}}\mathcal{L}_{\mathrm{cyc}}(\vartheta) + w_\alpha\mathcal{L}_\alpha(\vartheta) + w_\vartheta \left\|\vartheta\right\|_2^2\right\}. \quad (13)$$

Optimisation is carried out by seeking its global minimum with respect to ϑ. The weights w_{cyc}, w_α and w_ϑ are set to balance the impact of the terms.

Fig. 3 shows the scheme of the X-Net: One CNN plays the role of $F(\boldsymbol{X})$, the other represents $G(\boldsymbol{Y})$. Solid lines going through them indicate data transferred from one domain to the other, dashed lines indicate data re-transformed back to their original domain. The patches from $\boldsymbol{X}$ and $\boldsymbol{Y}$ are used both as input and targets for the CNNs. Recall that the patch prior $\boldsymbol{\alpha}$ is computed in advance, as explained in Section III-A. For an easier representation, $\boldsymbol{\alpha}$ is deliberately depicted in Fig. 3 as computed on the fly.

C. ACE-Net: Adversarial Cyclic Encoder Network

Inspired by Murez *et al.* [42], we expand the X-Net framework by introducing a latent space $\mathcal{Z}$ between domain $\mathcal{X}$ and domain $\mathcal{Y}$. Differently from the X-Net, this architecture
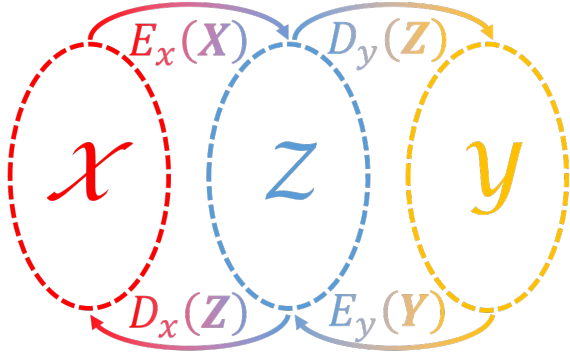
Fig. 4: Second proposed framework: a latent space $\mathcal{Z}$ is introduced between domains $\mathcal{X}$ and $\mathcal{Y}$, and four regression functions mapping data across them. In this case, $F(\boldsymbol{X}) = D_{\mathcal{Y}}(E_{\mathcal{X}}(\boldsymbol{X}))$ and $G(\boldsymbol{Y}) = D_{\mathcal{X}}(E_{\mathcal{Y}}(\boldsymbol{Y}))$.

consists of five CNNs. The first four networks are image regression functions (see Fig. 4): Encoders $E_{\mathcal{X}}(\boldsymbol{X}) : \mathcal{X}^{h \times w}$ and $E_{\mathcal{Y}}(\boldsymbol{Y}) : \mathcal{Y}^{h \times w}$ transform data from the original domains into the new common space and a representation referred to as the code: $\boldsymbol{Z} \in \mathcal{Z}^{h \times w}$. Note that the spatial dimensions of $\boldsymbol{Z}$, h and w, are equal to those of $\boldsymbol{X}$ and $\boldsymbol{Y}$. This is an empirical choice, as this is seen to produce best image translation and change detection performance. Bottlenecking (dimensionality reduction) at the code layer is not needed for regularisation, as with conventional autoencoders, due to the constraints imposed by loss functions associated with cross-domain mapping. The decoders $D_{\mathcal{X}}(\boldsymbol{Z}) : \mathcal{Z}^{h \times w} \rightarrow \mathcal{X}^{h \times w}$ and $D_{\mathcal{Y}}(\boldsymbol{Z}) : \mathcal{Z}^{h \times w} \rightarrow \mathcal{Y}^{h \times w}$ map latent space data back into their original domains. The fifth network is a discriminator, which is described later.

Despite the added complexity, is simple to notice an analogy between the two schemes, namely: $F(\boldsymbol{X}) = D_{\mathcal{Y}}(E_{\mathcal{X}}(\boldsymbol{X}))$ and $G(\boldsymbol{Y}) = D_{\mathcal{X}}(E_{\mathcal{Y}}(\boldsymbol{Y}))$. Therefore, we can include the same loss terms that the X-Net uses: weighted translation loss and cycle-consistency loss, in addition to the weight decay regularisation term. In this case,

$$
\begin{aligned}
\hat{\boldsymbol{X}} &= G(\boldsymbol{Y}) = D_{\mathcal{X}}(E_{\mathcal{Y}}(\boldsymbol{Y})) \,, \\
\hat{\boldsymbol{Y}} &= F(\boldsymbol{X}) = D_{\mathcal{Y}}(E_{\mathcal{X}}(\boldsymbol{X})) \,, \\
\dot{\boldsymbol{X}} &= G(\hat{\boldsymbol{Y}}) = D_{\mathcal{X}}(E_{\mathcal{Y}}(D_{\mathcal{Y}}(E_{\mathcal{X}}(\boldsymbol{X})))) \,, \\
\dot{\boldsymbol{Y}} &= F(\hat{\boldsymbol{X}}) = D_{\mathcal{Y}}(E_{\mathcal{X}}(D_{\mathcal{X}}(E_{\mathcal{Y}}(\boldsymbol{Y})))) \,.
\end{aligned} \tag{14}
$$

Nonetheless, the ACE-Net framework allows to define two additional loss terms.

1) Reconstruction Loss: The composite functions $D_{\mathcal{X}}(E_{\mathcal{X}}(\boldsymbol{X}))$ and $D_{\mathcal{Y}}(E_{\mathcal{Y}}(\boldsymbol{Y}))$ constitute autoencoders, whose goal is to reproduce their input as faithfully as possible in output. This means that the reconstructed images $\tilde{\boldsymbol{X}}$ and $\tilde{\boldsymbol{Y}}$ must satisfy:

$$
\begin{aligned}
\tilde{\boldsymbol{X}} &= D_{\mathcal{X}}(E_{\mathcal{X}}(\boldsymbol{X})) \simeq \boldsymbol{X} \,, \\
\tilde{\boldsymbol{Y}} &= D_{\mathcal{Y}}(E_{\mathcal{Y}}(\boldsymbol{Y})) \simeq \boldsymbol{Y} \,.
\end{aligned} \tag{15}
$$

Consequently, we introduce the reconstruction loss term:

$$
\mathcal{L}_{\mathrm{AE}}(\vartheta_{\mathrm{AE}}) = \mathbb{E}_{\boldsymbol{X}}\left[\delta(\tilde{\boldsymbol{X}}, \boldsymbol{X})\right] + \mathbb{E}_{\boldsymbol{Y}}\left[\delta(\tilde{\boldsymbol{Y}}, \boldsymbol{Y})\right], \tag{16}
$$

where ϑ_{AE} denotes all parameters in the autoencoders, consisting of $E_{\mathcal{X}}(\boldsymbol{X})$, $D_{\mathcal{Y}}(\boldsymbol{Z})$, $E_{\mathcal{Y}}(\boldsymbol{Y})$ and $D_{\mathcal{X}}(\boldsymbol{Z})$.

2) Adversarial Code Alignment Losses: Even after implementing the cycle-consistency loss and the weighted translation loss, there is no guarantee that the latent domain is the same for both AEs. Although the code layers might align in distribution, there is still a risk that class signatures do not correspond due to mode swapping or other perturbations in feature space. To ensure that they align both in distribution and in feature space location of classes, we apply adversarial training and feed a discriminator with a stack of the two codes. The discriminator $\mathfrak{D}(\boldsymbol{Z}) : \mathcal{Z}^{h \times w} \rightarrow [0, 1]$ is rewarded if it is able to distinguish the codes, whereas the generators (i.e. the encoders) are penalised when the discriminator succeeds. Let successful discrimination be defined as: $\mathfrak{D}(E_{\mathcal{X}}(\boldsymbol{X})) = 1$ and $\mathfrak{D}(E_{\mathcal{Y}}(\boldsymbol{Y})) = 0$. Thus, the last two loss terms become:

$$
\mathcal{L}_{\mathfrak{D}}(\vartheta_{\mathfrak{D}}) = \mathbb{E}_{\boldsymbol{X}}\left[(\mathfrak{D}(E_{\mathcal{X}}(\boldsymbol{X})) - 1)^2\right] + \mathbb{E}_{\boldsymbol{Y}}\left[\mathfrak{D}(E_{\mathcal{Y}}(\boldsymbol{Y}))^2\right] \tag{17}
$$

$$
\mathcal{L}_{\mathcal{Z}}(\vartheta_E) = \mathbb{E}_{\boldsymbol{X}}\left[\mathfrak{D}(E_{\mathcal{X}}(\boldsymbol{X}))^2\right] + \mathbb{E}_{\boldsymbol{Y}}\left[(\mathfrak{D}(E_{\mathcal{Y}}(\boldsymbol{Y})) - 1)^2\right] \tag{18}
$$

where the discrimination loss $\mathcal{L}_{\mathfrak{D}}$ is used to adjust the parameters $\vartheta_{\mathfrak{D}}$ of the discriminator. The code layer is used as generator, and the code loss $\mathcal{L}_{\mathcal{Z}}$ is used to train the parameters ϑ_E of the encoders $E_{\mathcal{X}}(X)$ and $E_{\mathcal{Y}}(Y)$ that generate the codes. The adversarial scheme is evident from (17) and (18), the two generators and the discriminator aim at the opposite goal and, therefore, have opposite loss terms. As in [41], we choose an adversarial objective function based on mean squared errors rather than a logarithmic one. Note that two discriminators could also have been placed after the decoders to distinguish transformed *fake* data from the reconstructed ones, as in [15]. However, to train two additional networks and find a good balance between all the involved parties is not trivial and require the correct design of each and every network in the architecture, on top of which fine-tuning of all the involved weights must be carried out. In conclusion, we decided to have a less complex framework with just one discriminator for the code space.

3) Total loss function: The total loss function $\mathcal{L}(\vartheta)$ in this case is composed of six terms:

$$
\begin{aligned}
\mathcal{L}(\vartheta) = &w_{\mathrm{adv}}\left[\mathcal{L}_{\mathcal{Z}}(\vartheta_E) + \mathcal{L}_{\mathfrak{D}}(\vartheta_{\mathfrak{D}})\right] + \\
&w_{\mathrm{AE}}\mathcal{L}_{\mathrm{AE}}(\vartheta_{\mathrm{AE}}) + w_{\mathrm{cyc}}\mathcal{L}_{\mathrm{cyc}}(\vartheta_{\mathrm{AE}}) + \\
&w_{\alpha}\mathcal{L}_{\alpha}(\vartheta_{\mathrm{AE}}) + w_{\vartheta}\|\vartheta\|_2^2 \,.
\end{aligned} \tag{19}
$$

The weights balancing the adversarial losses (w_{adv}), the reconstruction loss (w_{AE}), the cycle-consistency loss (w_{cyc}), the weighted translation loss (w_{α}), and the weight regularisation (w_{ϑ}) must be tuned.

Fig. 5 show the schematics of the ACE-Net. For simplicity, the arrows represent the data flow involving only the loss terms related to $\boldsymbol{X}$. $\boldsymbol{Y}$ in this image is used only to produce its code and as a target for translation from $\boldsymbol{X}$. The flow diagram for loss terms related to $\boldsymbol{Y}$ would be symmetric. Solid arrows represent images going through the encoder-decoder pairs only once (namely $\tilde{\boldsymbol{X}}$ and $\hat{\boldsymbol{Y}}$), dashed arrows are the second half of the cycle leading to $\dot{\boldsymbol{X}}$. The discriminator $\mathfrak{D}(\boldsymbol{Z})$ takes as input $E_{\mathcal{X}}(\boldsymbol{X})$ and $E_{\mathcal{Y}}(\boldsymbol{Y})$ and tries to tell them apart.
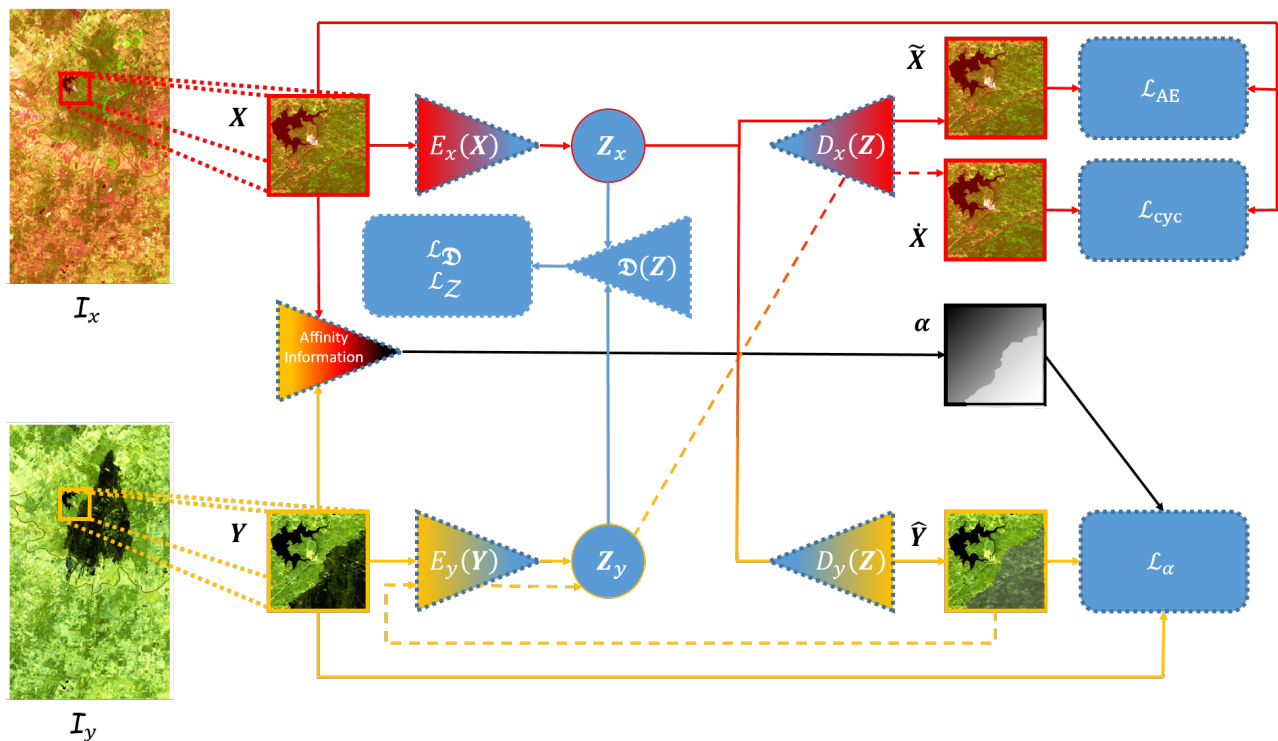
Fig. 5: Data flow of the ACE-Net. The encoders $E_{\mathcal{X}}(\boldsymbol{X})$ and $E_{\mathcal{Y}}(\boldsymbol{Y})$ transform incompatible data into two code spaces, which are aligned by adversarial training against the discriminator $\mathfrak{D}(\boldsymbol{Z})$. The decoders $D_{\mathcal{X}}(\boldsymbol{Z})$ and $D_{\mathcal{Y}}(\boldsymbol{Z})$ are taught to map data from the latent space back into the original spaces. For simplicity, only the loss terms related to $\boldsymbol{X}$ and their corresponding data flows are depicted. Dash lines refer to data which have been transformed already once, have gone through the framework again and have been transformed back into their original domain.

D. Change extraction

At this stage of the proposed methodology, any homogeneous change detection technique could be used to highlight changes. Among these, we must choose the most appropriate according to the characteristics of the data. However, the translated images go through severely nonlinear transformations, and defining an analytical model describing their statistics is not trivial. Moreover, the main objective of this work is to propose two translation methods, whose contribution might be concealed by a more complex homogeneous change detection approach. Therefore, image subtraction is the most appropriate operation: its requirement is that the original images and the translated ones are in the same domain, which is the final goal of the translation networks.

Once the X-Net and the ACE-Net are trained and the transformed images $\hat{\boldsymbol{X}}$ and $\hat{\boldsymbol{Y}}$ obtained, the elements of two distance images $d^{\mathcal{X}}$ and $d^{\mathcal{Y}}$ can be computed as the vector norms of the pixel-wise subtractions

$$d_i^{\mathcal{X}} = \|\hat{\boldsymbol{x}}_i - \boldsymbol{x}_i\|_2 \quad \text{and} \quad d_i^{\mathcal{Y}} = \|\hat{\boldsymbol{y}}_i - \boldsymbol{y}_i\|_2$$

for all pixels $i \in \{1, \ldots, N\}$, where $\boldsymbol{x}_i$, $\boldsymbol{y}_i$, $\hat{\boldsymbol{x}}_i$ and $\hat{\boldsymbol{y}}_i$ represent, respectively, pixels of $\boldsymbol{X}$, $\boldsymbol{Y}$, $\hat{\boldsymbol{X}}$ and $\hat{\boldsymbol{Y}}$. These difference images are normalised and combined together so that changes are highlighted, whereas false alarms that are present in only one of the two distance images are suppressed. Outliers might affect the two normalisations, so the distances in $d^{\mathcal{X}}$ and $d^{\mathcal{Y}}$ beyond three standard deviations of the mean values are clipped. We combine the normalised distance images with a simple average and obtain the final difference

image d. The latter is then filtered and thresholded to achieve a binary segmentation, which provides the final goal of a CD method: the change map.

Concerning filtering, the method proposed in [54] is used. It exploits spatial context to filter d with a fully connected conditional random field model. It defines pairwise edge potentials between all pairs of pixels in the image by a linear combination of Gaussian kernels in an arbitrary feature space. The main downside of the iterative optimisation of the random field is that it requires the propagation of all the potentials across the image. However, this highly efficient algorithm reduces the computational complexity from quadratic to linear in the number of pixels by approximating the random field with a mean field whose iterative update can be computed using Gaussian filtering in the feature space. The number of iterations and the kernel width of the Gaussian kernels are the only hyperparameters manually set, and we opted to tune them according to [21]: 5 iterations and a kernel width of 0.1.

Finally, it is fundamental to threshold the filtered difference image correctly: a low threshold yields unnecessary false alarms. Vice versa, a high threshold increases the number of missed changes. Methods such as [55], [56], [57], [58] are able to set the threshold automatically. Among these, we selected the well known Otsu's method [55].

IV. EXPERIMENTAL RESULTS

First, the three datasets used in this work are presented in Section IV-A. Section IV-B provides the details of our experimental setup. Then, the proposed prior computation is

compared against its previous version in Section IV-C. For simplicity, we refer to the latter as prior computation (PC) and to the former as improved PC (IPC). The improvements are demonstrated by qualitative comparisons and further reflected in reductions of the computation time. Finally, in Section IV-D the performance of the proposed networks is compared against the one obtained with several methods from the heterogeneous CD literature. Along with the mean elapsed times, this section reports the area under the curve (AUC), the overall accuracy (OA), the $F1$ score and Cohen's Kappa Coefficient κ [59].

The experiments were performed on a machine running Ubuntu 14 with a 8-core CPU @ 2.7 GHz. Moreover, 64 GB of RAM and an NVIDIA GeForce GTX TITAN X (Maxwell) allowed to reduce considerably the training times through parallel computation. The methods were all implemented in Python using TensorFlow 1.4.0.

A. Datasets

1) Forest fire in Texas: Bastrop County in Texas was struck by a forest fire during September-October, 2011. The Landsat 5 TM and the Earth Observing-1 Advanced Land Imager (EO-1 ALI) acquired two multispectral optical images before and after the event. The resulting co-registered and cropped images of size 1520×800 are displayed in false colour in Fig. 6a and Fig. 6b[1]. Some of the spectral bands of the instruments (7 and 10 in total, respectively) overlap, so the signatures of the land covers involved are partly similar. Volpi *et al.* [60] provided the ground truth shown in Fig. 6c.
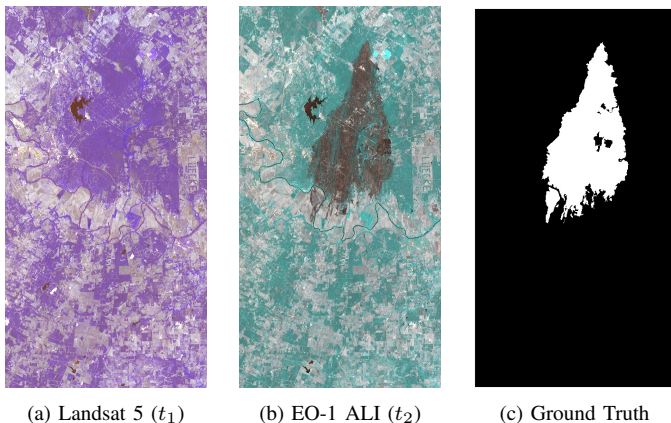

(a) Landsat 5 (t_1)   (b) EO-1 ALI (t_2)   (c) Ground Truth

Fig. 6: Forest fire in Texas. Landsat 5 ($t1$), (b) EO-1 ALI ($t2$), (c) ground truth.

2) Flood in California: Fig. 7a displays the RGB channels of a Landsat 8 acquisition[1] covering Sacramento County, Yuba County and Sutter County, California, on 5 January 2017. The OLI and TIRS sensors on Landsat 8 together acquire data in 11 channels, from deep blue up to thermal infrared. The same area was affected by a flood, as can be seen in Fig. 7b. This is a Sentinel-1A[2] acquisition, recorded in polarisations VV and VH on 18 February 2017. The ratio between the two intensities is included both as the blue component of the false colour composite in 7b and as the third channel provided as input
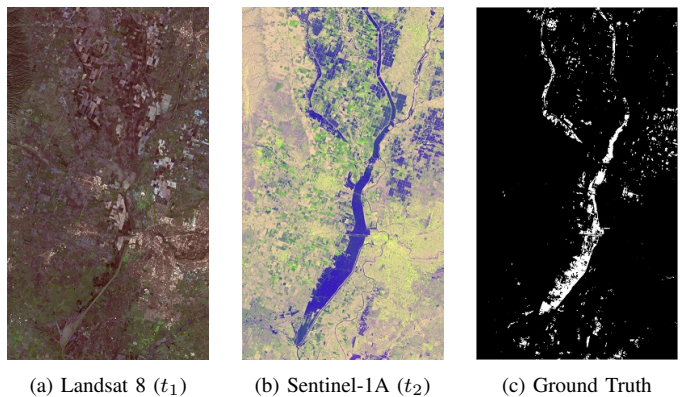
[1]Distributed by LP DAAC, http://lpdaac.usgs.gov
[2]Data processed by ESA, http://www.copernicus.eu/

(a) Landsat 8 (t_1)   (b) Sentinel-1A (t_2)   (c) Ground Truth

Fig. 7: Flood in California. (a) Landsat 8 (t_1), (b) Sentinel-1A (t_2), (c) ground truth.

to the networks. The ground truth in Fig. 7c is provided by Luppino *et al.* [21]. Originally of 3500×2000 pixels, these images were resampled to 850×500 pixels to reduce the computation time.

3) Constructions in China: The SAR image in Fig. 8a and the coregistered optical image in Fig. 8b were acquired in June 2008 and in September 2012 respectively over the Shuguang village next to Dongying City, China. Both images have 593×921 pixels with a spatial resolution of 8 meters, and the ground truth in Fig. 8c highlights the edification of buildings which took the place of some farmlands.

(a) SAR image at t_1 (b) Optical RGB image at t_2 (c) Ground Truth
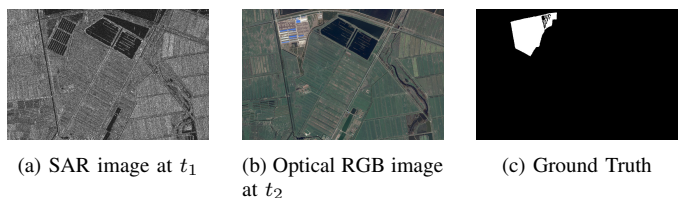
Fig. 8: Constructions in China. (a) RADARSAT-2 (t_1), (b) Quickbird / Landsat 7 (t_2), (c) ground truth.

B. Experimental setup

1) X-Net and ACE-Net: For the design of the proposed methods, we opted for CNNs with fully convolutional layers. One of the advantages is their flexibility with respect to the input size. At first, one can use batches of small patches extracted from the original images for the training, but once this stage is over, the banks of filters can be applied directly to the whole dataset at once.

Since the goal is to transform each pixel from one domain to another and regularisation of the autoencoders is efficiently handled by other network constraints, there is no need to have a bottleneck in the code layer of the ACE-Net, that is, to reduce the size of the input height and width to compress the data. Hence, 3×3 filters were applied without stride on the input patches, whose borders were padded with zeros. In the X-Net, both networks have four layers: The first three consist of 100, 50, and 20 filters; The last layer matches the number of channels of the translated data, with $|\mathcal{Y}|$ filters for $F(\boldsymbol{X})$ and $|\mathcal{X}|$ filters for $G(\boldsymbol{Y})$. The encoders of the ACE-Net have

three layers of 100, 50, and 20 filters, and these numbers are reversed for the decoders. The ACE-Net discriminator is the only network which, after three convolutional layers with 64, 32, and 16 filters, deploys a fully-connected layer with one output neuron.

Concerning the activation functions, a leaky ReLU [61] was chosen with the slope for negative arguments set equal to $\beta = 0.3$. The last layer of each network represents an exception: The sigmoid was selected for the discriminator, which must provide outputs between 0 and 1, whereas for every other network the hyperbolic tangent was chosen because our data was normalised between -1 and 1. With this range of data values the training was sped up as expected [62]. Batch normalisation [63] turned out to be unnecessary and was discarded, as it did not improve the optimisation and it actually slowed down our experiments.

After each layer, dropout is applied with a dropout rate of 20% during the training phase to enhance the robustness of the framework against overfitting and input noise [64]. Also, data augmentation helps increasing the size of the training sample by introducing some more variety in the data: Before feeding the patches to the network, these were randomly flipped and rotated.

The weights in ϑ were initialised with a truncated normal distribution according to [65] and the biases were initialised as zeros. For every epoch of the training 10 batches were used, each containing 10 patches of size 100×100. The Adam optimizer [66] minimised the loss function for 240 epochs at a learning rate of 10^{-5}. The weights of the loss functions in the ACE-Net are five: $w_{\text{adv}} = 1$; $w_{\text{AE}} = 0.2$; $w_{\text{cyc}} = 2$; $w_{\alpha} = 3$; and $w_{\vartheta} = 0.001$. The X-Net uses only three of these, namely w_{cyc}, w_{α} and w_{ϑ}, and the same values were used for these.

After several training epochs, a preliminary evaluation of the difference image d is computed and scaled to fall into the range $[0, 1]$, and the prior is updated as $\Pi = 1 - d$. In this way, pixels associated with a large d entry are penalised by a small weight, whereas the opposite happens to pixels more likely to be unchanged. The Π is updated at two milestones placed at one third and two thirds of the total epochs, namely at epoch 80 and epoch 160. This form of self-supervision paradigm has already proven robust in other tasks such as deep clustering [67] and deep image recovery [68].

2) SCCN and CAN: We implemented two methods as state-of-the-art competitors, namely SCCN [4] and the conditional adversarial network in [7], which is from now on referred to as CAN. A brief description of these methods can be found in the last paragraph of Section II-A2 and Section II-B2, respectively.

The most important aspect of the compared architectures is their ability to transform the data and, consequently, the quality of the obtained difference image d, whereas the postprocessing applied to d is not considered relevant in the present comparison. Therefore, although [4] and [7] deploy different filtering and thresholding techniques, the methods selected in this work are used on all the difference images for a fair comparison of the final change maps. The implementations of the SCCN and the CAN were as faithful as possible based on the details shared in [4] and [7]. However, to make the SCCN work we had to replace a fixed parameter described in the paper with

the output of Otsu's method to find an optimal threshold for the difference image in the iterative refinement of the change map. We also had to interpret the description in [4]: To avoid trivial solutions, we implemented their pretraining phase with decoders having one coupling layer (convolutional layer with filters of 1×1) and 250 epochs. This was empirically found to be the minimum amount of epochs needed to consistently obtain a meaningful representation of the data in the code space to be used as starting point for the training procedure. Also, in [4] Liu *et al.* selected a rigorous stopping criterion for the latter, but it was hardly reached during our experiments, so a maximum number of epochs was set to 500.

3) Comparisons with other methods: In order to better frame our architectures within the state-of-the-art of heterogeneous CD, we also present a comparison on the widely used benchmark dataset of the constructions in China. There are several versions of this dataset in terms of image sizes and ground truth, so we focused on the methods from the literature which used the same version, to ensure that all considered results are fully comparable.

Beside SCCN and CAN, we report for this dataset the results obtained by several methods. The mixed-norm-based (MNB) method by Touati *et al.* [69], the coupling translation network (CPTN) by Gong *et al.* [15], and the coupled dictionary learning (ICDL) method by Gong *et al.* [20] are unsupervised. Instead, the post-classification comparison (PCC) [70], the conditional copulas (CC) method by Mercier *et al.* [26], and the anomaly feature learning (AFL) method by Touati *et al.* [53] are supervised approaches. For the experimental setup and implementation details of these methods applied to this specific dataset, we refer to their original papers.

Although these methods are evaluated on the same dataset, the supervised ones make use of training samples (e.g., on the "no-change" class or on the thematic classes in the scene). In terms of change detection performance, this is a clear advantage over unsupervised method, however, it comes with the cost of manual annotation based on experts' knowledge or data collection on location. Therefore, the results must be interpreted fairly, since unsupervised methods do not make use of this kind of input but on the other hand they do not require any user prompt.

Finally, we stress another distinction: SCCN, CAN, CPTN and AFL deploy deep neural networks, so they present a similar methodological framework with respect to the proposed architectures, whereas PCC, CC, MNB, and ICDL rely on more traditional machine learning and pattern recognition techniques.

C. PC vs IPC

The effects of the proposed modifications to the affinity matrix analysis are evaluated by a visual comparison of the results obtained by both the PC and the IPC. Based on [21], a patch size of $k = 20$ was selected for all the experiments. Fig. 9 shows the outcomes for the three datasets in the two most extreme cases, namely with strides of $\Delta = 1$ and $\Delta = k$. In the first column, one can notice how the PC provides more blurry results where the areas highlighted by their α values

(a) PC, $\Delta = 1$ (b) IPC, $\Delta = 1$ (c) PC, $\Delta = 20$ (d) IPC, $\Delta = 20$

(e) PC, $\Delta = 1$ (f) IPC, $\Delta = 1$ (g) PC, $\Delta = 20$ (h) IPC, $\Delta = 20$

(i) PC, $\Delta = 1$ (j) IPC, $\Delta = 1$ (k) PC, $\Delta = 20$ (l) IPC, $\Delta = 20$
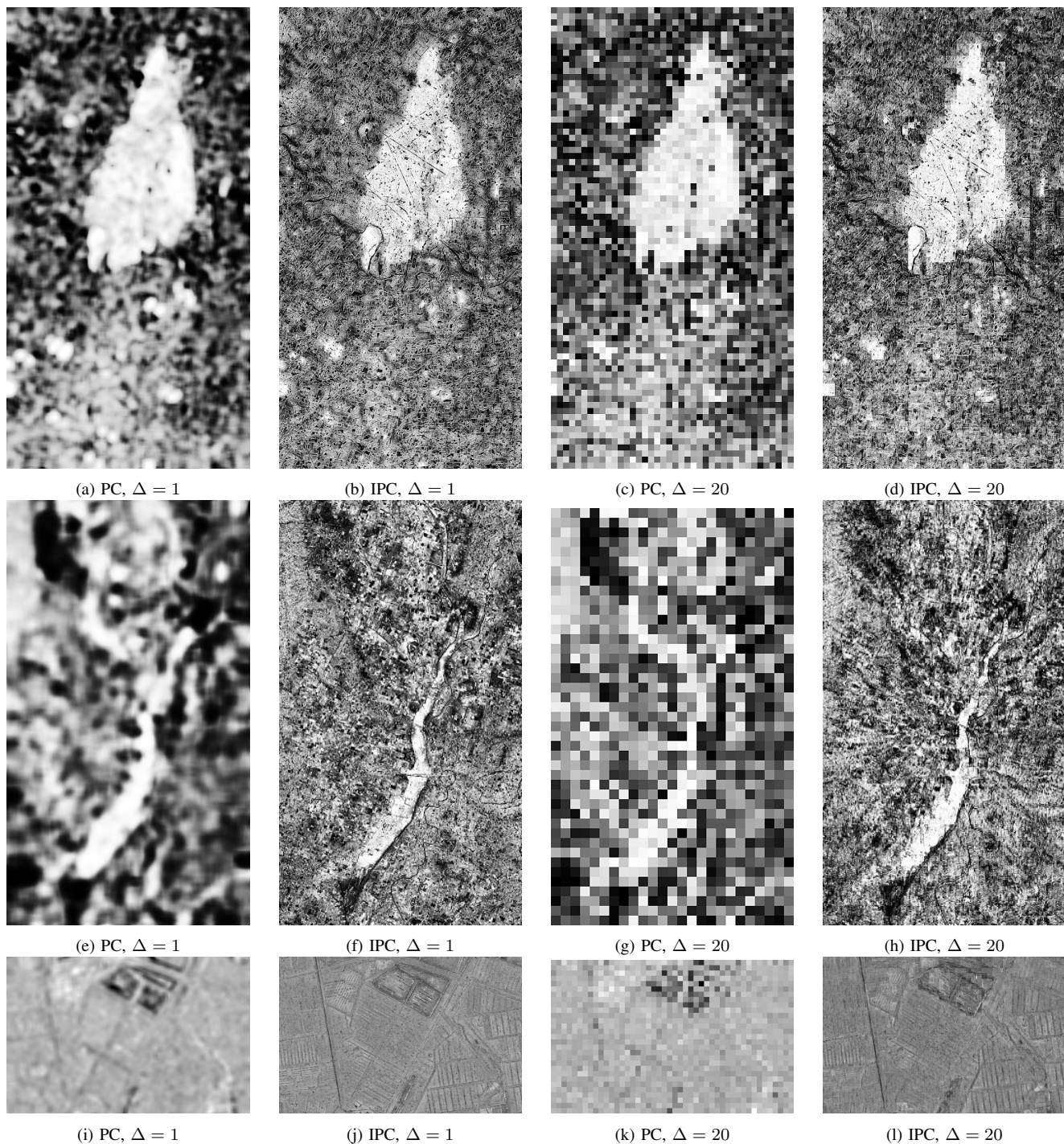
Fig. 9: Results on the three datasets for the PC and the IPC, for $\Delta = 1$ and for $\Delta = 20$.

have soft edges. In contrast, the images in the second column were obtained with the IPC and they unarguably represent a more precise result with sharp edges and smaller segments of highlighted pixels. The third column shows the strong impact that a large Δ has on the outcomes of PC. The PC method's assignment of one value to an entire patch leads to the tiled pattern mentioned in Section III-A. Instead, the IPC is not as affected by the stride applied to the patch shifts, as shown in the fourth column of Fig. 9.

Table I reports an approximate total number of patches $|\mathcal{P}|$

and the computation time spent by the two methods on the three datasets for the two considered cases. As it can be seen, the major drawback of setting $\Delta = 1$ is the large value of $|\mathcal{P}|$. Recall that we propose to apply the IPC three times: with $k_{small} = 10$ and $k = 20$ to the images at the original sizes, and with $k = 20$ to the images resampled at half the sizes.

Finally, for the training of the ACE-Net and the X-Net we opted for $k = 20$ and $\Delta = 5$, for which the proposed approach took approximately 42 min, 13 min, and 19 min for the Texas, California, and China datasets, respectively.

D. Results

For the first two datasets, the two proposed techniques and the previous SCCN and CAN methods were applied. Each of the four architectures was initialised randomly and trained for 100 independent runs. The average (standard deviation) of the evaluation metrics are reported in Table II and Table III, together with the average training times. As a reference, the results achieved by directly filtering and thresholding the prior α are also included. Recall that the X-Net and the ACE-Net require the computation of α as pre-training step: in practice, its computational time must be accounted for and added to the training time of the two architectures. The X-Net is the simplest framework, and this explains its fast training procedure. The ACE-Net and the SCCN have similar complexities, so they require similar times. By contrast, the CAN paper [7] defines one training epochs as using all 5×5 non-overlapping patches in the images, and the computational load of training grows accordingly with image size. One may suggest to train the networks on a subsample of patches randomly picked at every epoch, but there may be a trade-off between speed and performance. Focusing on Table II, The X-Net and the CAN show stable and consistent performance. The ACE-Net and the SCCN sometimes reach higher values of the evaluation metrics than the X-net, but the average is lower and the variance is high. The performance obtained starting from α suggests that technically the IPC algorithm can be used to perform heterogeneous CD autonomously, yet its application as a prior for the X-Net and the ACE-Net yields the best results. A different scenario was found for the California dataset in Table III. The ACE-Net outperforms the X-Net and the CAN in terms of average κ, but has more variability. The SCCN performs best on this dataset as measured by its κ, which reaches significantly higher values than the other algorithms, and with a low variability when compared to SCCN behaviour for the Texas dataset. However, upon closer inspection the transformations applied by this method on this dataset are not as intended and the performance is degenerate, which will be explained in Section V. In this case, the computation of α as heterogeneous CD does not seem as reliable, but it still boosts the performance of the ACE-Net and the X-Net.

Finally, the results obtained by the state-of-the-art methods on the China dataset, along with the ones obtained by ACE-Net and X-Net, are reported in Table IV. In addition, we note that the multidimensional scaling (MDS) method by Touati

et al. obtained an OA of 0.967 in [71]. Again, the result obtained by filtering and thresholding the prior α is included as reference, and the comments about the performance of α for the California dataset apply also here. The first part of Table IV consists of supervised methods, which are purposely separated from the rest: although they are evaluated on the same dataset, they require supervision and user prompt for sample selection, which makes the comparison with the other methods unfair. We also remark that our architectures are applied with the same hyperparameters for all the datasets, whereas the hyperparameters used in [7], [15] for the other methods were tuned on a case-by-case basis. Both the ACE-Net and the X-Net outperform the other unsupervised methods, with the latter reaching higher values. These results are discussed further in Section V.

Fig. 10, Fig. 11, and Fig. 12 show examples of output delivered by each of the four methods on the three datasets. False colour images of the original and transformed images are composed with a subset of three channels from those available. Translated images are shown for the X-Net and the ACE-Net, followed by the resulting difference image and a confusion map (CM), which allows to visualise the accuracy of the results: TN are depicted in black, TP in white, FN in red, and FP in green. For the CAN and SCCN algorithms, the translated images are replaced with the equivalent images used by these methods to compute the difference image. For the CAN algorithm, these are a generated image $\hat{Y}$ and a approximated image $\tilde{Y}$ in the $\mathcal{Y}$ domain. For the SCCN algorithm, these are code images $Z_\mathcal{X}$ and $Z_\mathcal{Y}$ from a common latent space.

V. Discussion

A. Comparison among X-Net, ACE-Net, SCCN, and CAN

Stability and consistency are the advantages of the X-Net and CAN algorithms. They both provide good results on the selected datasets, with the former performing better. The X-Net has other positive aspects, for example the simplicity of its architecture composed of only two CNNs of few layers each, yielding a total number of $|\vartheta| \sim 1.3 \times 10^5$ parameters, and fast convergence during training thanks to a limited number of terms in the loss function.

The same cannot be said about the CAN. The framework counts three fully connected networks with $|\vartheta| \sim 3.1 \times 10^5$, and the use of all possible 5×5 patches as input makes its training epochs time consuming, especially for bigger datasets like the Texas one. In addition, it shows a high tendency to miss some of the changes due to unwanted alignment of changed areas in the generated and the approximated images. This can be noticed by the high amount of FN in Fig. 10s and Fig. 11s.

The ACE-Net has a large amount of parameters ($|\vartheta| \sim 2.8 \times 10^5$), and together with its complex loss function they guarantee the flexibility that allows to achieve the best overall performance on the three datasets. However, the complexity is also the main drawback of this architecture, because it implies a difficult and possibly slow convergence, which also results in higher variability in performance. In conclusion, it

TABLE I: Approximate $|\mathcal{P}|$ and computation time of the two methods applied to the three datasets for $\Delta = 1$ and $\Delta = k$.

		$|\mathcal{P}|$	PC	IPC
Texas	$\Delta = 1$	1.2×10^6	45 min	76 min
	$\Delta = 20$	3×10^3	2:37 min	6 min
California	$\Delta = 1$	4×10^5	15 min	24 min
	$\Delta = 20$	1×10^3	0:37 min	1:45 min
China	$\Delta = 1$	5.2×10^5	35 min	62 min
	$\Delta = 20$	1×10^3	0:40 min	2:20 min

TABLE II: Mean and standard deviation of the evaluation metrics for the four methods applied to the Texas dataset. Best results are in bold.

	AUC	OA	F1	κ	t
α	0.956	0.922	0.695	0.652	42 min
ACE-Net	0.968 (0.12)	0.951 (0.008)	0.747 (0.047)	0.720 (0.051)	42 + 13 min
X-Net	**0.968 (0.007)**	**0.961 (0.006)**	**0.785 (0.049)**	**0.767 (0.028)**	42 + 7 min
CAN [7]	0.951 (0.009)	0.925 (0.006)	0.587 (0.048)	0.548 (0.050)	69 min
SCCN [4]	0.893 (0.14)	0.880 (0.010)	0.613 (0.309)	0.551 (0.362)	16 min

TABLE III: Mean and standard deviation of the evaluation metrics for the four methods applied to the California dataset. Best results are in bold.

	AUC	OA	F1	κ	t
α	0.803	0.788	0.281	0.204	13 min
ACE-Net	0.881 (0.008)	**0.915 (0.007)**	0.459 (0.027)	0.415 (0.030)	13 + 12 min
X-Net	0.892 (0.006)	0.911 (0.004)	0.447 (0.019)	0.402 (0.021)	13 + 6 min
CAN [7]	0.857 (0.008)	0.904 (0.005)	0.424 (0.020)	0.365 (0.023)	21 min
SCCN [4]	**0.920 (0.002)**	0.903 (0.007)	**0.500 (0.015)**	**0.454 (0.017)**	15 min

has the potential to outperform the other methods, but a costly optimisation of its parameters might be necessary.

The SCCN requires a thorough analysis. First of all, this network is very simple: it consists of two symmetric networks with four layers and the total amount of parameters is just $|\vartheta| \sim 6 \times 10^3$. Its parameters space is thus limited when compared to its contenders. This may explain why the method often fails to converge and provides very poor results on the first dataset (see Table II). The very good results displayed in Table III instead are explained by a visual inspection of the image translations it performs on the California dataset. After preliminary training of the two encoders, the one transforming Y is frozen, while the other is taught to align the codes of those pixels which are flagged as unchanged. However, it can be seen in Fig. 11e that the encoder is not able to capture more than the background average colour of Fig. 11j, which can be characterized as degenerate behaviour. Basically, the

TABLE IV: Evaluation metrics for the methods applied to the China dataset. The results indicated with † and ‡ are reported by [15] and [7] respectively. Best results are in bold.

	AUC	OA	F1	κ
α	0.848	0.699	0.248	0.171
AFL [53]	-	**0.980**	**0.732**	**0.722**
‡CC [26]	0.938	0.951	0.523	0.444
‡PCC	-	0.821	0.335	0.257
X-Net	**0.987**	**0.984**	0.731	0.696
ACE-Net	0.980	0.982	0.726	0.689
†SCCN [4]	0.959	0.976	0.728	0.679
CAN [7]	0.976	0.978	0.717	0.662
CPTN [15]	0.963	0.978	0.672	0.662
†ICDL [20]	0.921	0.951	0.469	0.444
MNB [69]	-	0.884	0.370	0.324

difference image in Fig. 11o is highlighting the water bodies of the SAR image in Fig. 7b, and this coincidentally results in high accuracy when detecting the flood. The same situation was faced when freezing the other encoder, and this issue was encountered similarly on the China dataset, as it can be seen in Fig. 12e. Note that high number of training epochs (500) in our customized implementation of the SCCN was beneficial for the Texas dataset, since it managed to converge more often to a meaningful solution, but it did not make much of a difference on the other two datasets, for which the method consistently brings the loss function to a local minimum that corresponds to a degenerate result within the first hundred of epochs, and then not being able to improve it further.

B. Discussion of the results on the benchmark dataset

The experiments on the China dataset offer a broader view in relation to the state-of-the-art. In general, one can appreciate that there is a trade-off between performance and interpretability: CC, PCC, ICDL, and MNB formalize well-defined intuitions – with different degrees of complexity –, and their results can be easily interpreted, but they do not achieve the same performance as the other methods. PCC is an intuitive and very simple approach, but is quite ineffective when the two separate classifications of the single images are not very accurate because it accumulates their errors. The concept of ICDL of using sparse dictionaries to map data into a common space is also intuitive, but its performance is less accurate in the application to this dataset as compared to the other considered methods. The same can be said for CC and MNB, whose approaches can be broadly interpreted as examples of feature engineering aimed at heterogeneous CD.

Another notable detail is the performance gap between the aforementioned CC, PCC, ICDL, and MNB, which deploy more traditional machine learning algorithms, and the deep learning methods, namely AFL, X-Net, ACE-Net, SCCN, CAN, and CPTN. Focusing on the κ coefficient and on the

considered benchmark dataset, the algorithms in the former group do not obtain higher values than 0.44, whereas the algorithms in the latter reach 0.66 and above.

More in particular, CPTN and ACE-Net are similar architectures with two AEs sharing their code spaces and adversarial losses. However, CPTN places two discriminators at the output of the AEs to set apart real reconstructed images from fake transformed images, whereas ACE-Net has only one discriminator acting in the code space. The results in Table IV suggest higher effectiveness of the proposed ACE-Net configuration in the application to the China dataset.

Finally, we recall that AFL also deploys AEs, but it uses a training set made of patches selected from unchanged areas to enforce the alignment of the code spaces. Still, the results obtained by the proposed unsupervised methodologies are in line with the ones of this supervised approach. This further suggests the effectiveness of the developed affinity prior in capturing information on unchanged areas in an unsupervised manner and of the proposed X-Net and ACE-Net architectures in taking benefit from this information.

C. Ablation study

In order to compare the contribution of each component of our networks, an ablation study was carried out on the *California* dataset. Alongside, we evaluated the impact of the proposed prior. Moreover, we investigated whether the overall performance can benefit from adding adversarial learning on the image contents, i.e. by adding two more discriminators, one for each input space, where the translated data are compared to the original data from the same domain.

The results in Table V were obtained with the X-Net and the ACE-Net applied with different configurations:

- **Discr Output:** with two added discriminators
- **No Alpha:** with a randomly initialised prior.
- **No Cycle:** without cycle-consistency.
- **No Milestone:** without the two milestone updates.
- **Proposed:** as proposed.
- **No Discr:** without the discriminator for the code contents (ACE-Net only).
- **No Recon:** without the reconstruction loss (ACE-Net only).

As it can be noticed, discarding any element from the total loss function of the two methods is not beneficial. Similarly, the gain of using the proposed prior rather than a random one is considerable. Also, the update of the prior at the two milestones is shown to improve the performance. Instead, the same cannot be said about adding the two discriminators for adversarial learning on the image contents.

VI. CONCLUSIONS

In this work we proposed two deep convolutional neural network architectures for heterogeneous change detection: the X-Net and the ACE-Net. In particular, we used an affinity-based change prior learnt from the input data to obtain an unsupervised algorithm. This prior was used to drive the training process of our architectures, and the experimental

results proved the effectiveness of our framework. Both outperformed consistently state-of-the-art methods, and each has its own advantages: the X-Net proved to produce very stable and consistent performance and reliable transformations of the data; the ACE-Net showed to be able to achieve the best results, at the cost of higher complexity and a more diligent training.

APPENDIX

In the following, we prove the equivalence between the H_0-conditional MSE in (5) and the unconditional weighted MSE in (7). Let us focus on the first term of the loss in (5), i.e.:

$$\mathcal{L}_1(\vartheta) = \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}} \left[\delta(\boldsymbol{X}, \hat{\boldsymbol{X}}) \,\middle|\, H_0 \right] = \\ = \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}} \left[\delta(\boldsymbol{X}, G(\boldsymbol{Y})) \mid H_0 \right]. \quad (20)$$

Plugging the expression of the L_2 squared distance into (20) leads to:

$$\mathcal{L}_1(\vartheta) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}} \left[\|G_i(\boldsymbol{Y}) - \boldsymbol{x}_i\|_2^2 \mid H_0 \right], \quad (21)$$

where $n = h \cdot w$ and $G_i(\boldsymbol{Y})$ is the vector corresponding in $G(\boldsymbol{Y})$ to the i-th pixel of the patch ($i = 1, 2, \ldots, n$). We assume that the sample pairs $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ associated with the pixels in the patch are mutually independent when conditioned to the "no-change" hypothesis H_0, and that all $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ vectors are continuous random vectors ($i = 1, 2, \ldots, n$). The former is a rather classical conditional independence assumption, which is frequently accepted in change detection studies [43], [26], [72], [73]. The latter is very common, and the reformulation in the case of discrete or mixed variables is straightforward. Accordingly:

$$\mathcal{L}_1(\vartheta) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{\boldsymbol{x}_i,\boldsymbol{Y}} \left[\|G_i(\boldsymbol{Y}) - \boldsymbol{x}_i\|_2^2 \mid H_0 \right] = \\ = \frac{1}{n} \sum_{i=1}^{n} \int \|G_i(\boldsymbol{Y}) - \boldsymbol{x}_i\|_2^2 \, p(\boldsymbol{x}_i, \boldsymbol{Y}|H_0) \, d\boldsymbol{x}_i d\boldsymbol{Y}, \quad (22)$$

where $p(\boldsymbol{x}_i, \boldsymbol{Y}|H_0)$ is the joint probability density function (PDF) of $\boldsymbol{x}_i$ and $\boldsymbol{Y}$ conditioned to H_0, and the Lebesgue integral is implicitly extended over the whole multidimensional space of all components of $\boldsymbol{x}_i$ and $\boldsymbol{Y}$. Thanks to the law of total probability ($i = 1, 2, \ldots, n$):

$$p(\boldsymbol{x}_i, \boldsymbol{Y}) = P(H_0)p(\boldsymbol{x}_i, \boldsymbol{Y}|H_0) + P(H_1)p(\boldsymbol{x}_i, \boldsymbol{Y}|H_1), \quad (23)$$

where $p(\boldsymbol{x}_i, \boldsymbol{Y})$ is the unconditional joint PDF of $\boldsymbol{x}_i$ and $\boldsymbol{Y}$, $p(\boldsymbol{x}_i, \boldsymbol{Y}|H_1)$ is their joint PDF conditioned to H_1, and $P(H_0)$

TABLE V: Ablation study on the California dataset: mean values (standard deviations) of metrics obtained before thresholding (AUC) and after thresholding (OA, F1, κ) with the two methodologies applied with different configurations. Best results are in bold.

		AUC	OA	F1	κ
X-Net	Discr Output	0.864(0.017)	0.891(0.013)	0.412(0.035)	0.359(0.040)
	No Alpha	0.876(0.004)	0.908(0.003)	0.439(0.012)	0.392(0.013)
	No Cycle	0.883(0.008)	0.910(0.005)	0.433(0.025)	0.387(0.027)
	No Milestones	0.873(0.006)	0.897(0.005)	0.423(0.015)	0.372(0.017)
	Proposed	**0.892 (0.006)**	**0.911 (0.004)**	**0.447 (0.019)**	**0.402 (0.021)**
ACE-Net	Discr Output	0.870(0.041)	0.884(0.079)	0.419(0.059)	0.367(0.070)
	No Alpha	0.865(0.030)	0.907(0.018)	0.434(0.041)	0.387(0.047)
	No Cycle	0.881(0.11)	0.908(0.006)	0.429(0.028)	0.382(0.032)
	No Milestones	0.866(0.030)	0.892(0.009)	0.402(0.041)	0.350(0.045)
	Proposed	**0.881 (0.008)**	**0.915 (0.007)**	**0.459 (0.027)**	**0.415 (0.030)**
	No Discr	0.872(0.008)	0.912(0.005)	0.455(0.025)	0.411(0.027)
	No Recon	0.875(0.016)	0.912(0.006)	0.447(0.029)	0.403(0.032)

and $P(H_1)$ are the prior probabilities of the two hypotheses. Straightforward algebraic manipulations allow proving that:

$$p(\boldsymbol{x}_i, \boldsymbol{Y}|H_0) = \psi(\boldsymbol{x}_i, \boldsymbol{Y})p(\boldsymbol{x}_i, \boldsymbol{Y}), \qquad (24)$$

where:

$$\psi(\boldsymbol{x}_i, \boldsymbol{Y}) = \frac{1}{P(H_0) + P(H_1)\Lambda(\boldsymbol{x}_i, \boldsymbol{Y})} \qquad (25)$$

and where:

$$\Lambda(\boldsymbol{x}_i, \boldsymbol{Y}) = \frac{p(\boldsymbol{x}_i, \boldsymbol{Y}|H_1)}{p(\boldsymbol{x}_i, \boldsymbol{Y}|H_0)} \qquad (26)$$

is the likelihood ratio associated with the two hypotheses [74]. Plugging (24) into (22) yields:

$$
\begin{aligned}
\mathcal{L}_1(\vartheta) &= \frac{1}{n}\sum_{i=1}^{n}\int ||G_i(\boldsymbol{Y}) - \boldsymbol{x}_i||_2^2\,\psi(\boldsymbol{x}_i, \boldsymbol{Y})p(\boldsymbol{x}_i, \boldsymbol{Y})\,d\boldsymbol{x}_i d\boldsymbol{Y} = \\
&= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{\boldsymbol{x}_i, \boldsymbol{Y}}\left[||G_i(\boldsymbol{Y}) - \boldsymbol{x}_i||_2^2\,\psi(\boldsymbol{x}_i, \boldsymbol{Y})\right] = \\
&= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{\boldsymbol{X}, \boldsymbol{Y}}\left[||G_i(\boldsymbol{Y}) - \boldsymbol{x}_i||_2^2\,\psi(\boldsymbol{x}_i, \boldsymbol{Y})\right],
\end{aligned}
$$
$$(27)$$

where in the last equality the conditional independence of the $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ pairs ($i = 1, 2, \ldots, n$) has been used again. Given the notation in (8) for the weighted L_2 squared loss, we can conclude that:

$$
\begin{aligned}
\mathcal{L}_1(\vartheta) &= \mathbb{E}_{\boldsymbol{X}, \boldsymbol{Y}}\left[\frac{1}{n}\sum_{i=1}^{n}||G_i(\boldsymbol{Y}) - \boldsymbol{x}_i||_2^2\,\psi(\boldsymbol{x}_i, \boldsymbol{Y})\right] = \\
&= \mathbb{E}_{\boldsymbol{X}, \boldsymbol{Y}}\left[\delta(\boldsymbol{X}, G(\boldsymbol{Y})|\boldsymbol{\Psi})\right] = \\
&= \mathbb{E}_{\boldsymbol{X}, \boldsymbol{Y}}\left[\delta(\boldsymbol{X}, \hat{\boldsymbol{X}}|\boldsymbol{\Psi})\right],
\end{aligned}
$$
$$(28)$$

where $\boldsymbol{\Psi}$ is the n-dimensional vector collecting all terms $\psi(\boldsymbol{x}_i, \boldsymbol{Y})$ that weigh the L_2 loss (i.e., $\Psi_i = \psi(\boldsymbol{x}_i, \boldsymbol{Y})$ for $i = 1, 2, \ldots, n$). This proves the equivalence, under the aforementioned conditional independence assumption, between the H_0-conditional MSE term involving $\boldsymbol{X}$ and $\hat{\boldsymbol{X}}$ in (5) and the corresponding term in the unconditional weighted MSE in (7). The same argument can be used to prove the equivalence

between the MSE terms involving $\boldsymbol{Y}$ and $\hat{\boldsymbol{Y}}$.

Furthermore, focusing on the i-th pixel of the patch ($i = 1, 2, \ldots, n$), we note that the likelihood ratio $\Lambda(\boldsymbol{x}_i, \boldsymbol{Y})$ takes values in $[0, +\infty)$. Hence, according to (25), $\psi(\boldsymbol{x}_i, \boldsymbol{Y})$ takes values in $(0, 1/P(H_0)]$. Specifically, small values of $\psi(\boldsymbol{x}_i, \boldsymbol{Y})$ are obtained in the case of large values of $\Lambda(\boldsymbol{x}_i, \boldsymbol{Y})$ (in the limit case, $\psi(\boldsymbol{x}_i, \boldsymbol{Y}) \longrightarrow 0^+$ if $\Lambda(\boldsymbol{x}_i, \boldsymbol{Y}) \longrightarrow +\infty$). Following the reasoning of a Bayesian likelihood ratio test, these comments suggest that, if $\psi(\boldsymbol{x}_i, \boldsymbol{Y})$ takes a small value, then the i-th pixel likely belongs to H_1 [74]. Vice versa, values of $\psi(\boldsymbol{x}_i, \boldsymbol{Y})$ close to the maximum $1/P(H_0)$ are achieved if $\Lambda(\boldsymbol{x}_i, \boldsymbol{Y})$ is small (in particular, $\psi(\boldsymbol{x}_i, \boldsymbol{Y}) = 1/P(H_0)$ if and only if $\Lambda(\boldsymbol{x}_i, \boldsymbol{Y}) = 0$) – a configuration that suggests that the i-th pixel likely belongs to H_0 [74]. This confirms the interpretation of small and large values of the components of $\boldsymbol{\Psi}$ in relation to membership to "change" or "no-change," respectively. Similar comments hold with regard to the components of $\boldsymbol{\Phi}$.

REFERENCES

[1] A. Singh, "Review article: Digital change detection techniques using remotely-sensed data," *Int. J. Remote Sens.*, vol. 10, no. 6, pp. 989–1003, 1989.

[2] L. T. Luppino, S. N. Anfinsen, G. Moser, R. Jenssen, F. M. Bianchi, S. Serpico, and G. Mercier, "A clustering approach to heterogeneous change detection," in *Proc. Scand. Conf. Image Anal. (SCIA)*, 2017, pp. 181–192.

[3] T. Zhan, M. Gong, J. Liu, and P. Zhang, "Iterative feature mapping network for detecting multiple changes in multi-source remote sensing images," *ISPRS J. Photogram. Remote Sens.*, vol. 146, pp. 38–51, 2018.

[4] J. Liu, M. Gong, K. Qin, and P. Zhang, "A deep convolutional coupling network for change detection based on heterogeneous optical and radar images," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 3, pp. 545–559, 2016.

[5] Z. Liu, G. Li, G. Mercier, Y. He, and Q. Pan, "Change detection in heterogenous remote sensing images via homogeneous pixel transformation," *IEEE Trans. Image Process.*, vol. 27, no. 4, pp. 1822–1834, 2018.

[6] X. X. Zhu, D. Tuia, L. Mou, G.-S. Xia, L. Zhang, F. Xu, and F. Fraundorfer, "Deep learning in remote sensing: A comprehensive review and list of resources," *IEEE Geosci. Remote Sens. Mag.*, vol. 5, no. 4, pp. 8–36, 2017.

[7] X. Niu, M. Gong, T. Zhan, and Y. Yang, "A conditional adversarial network for change detection in heterogeneous images," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 1, pp. 45–49, 2018.
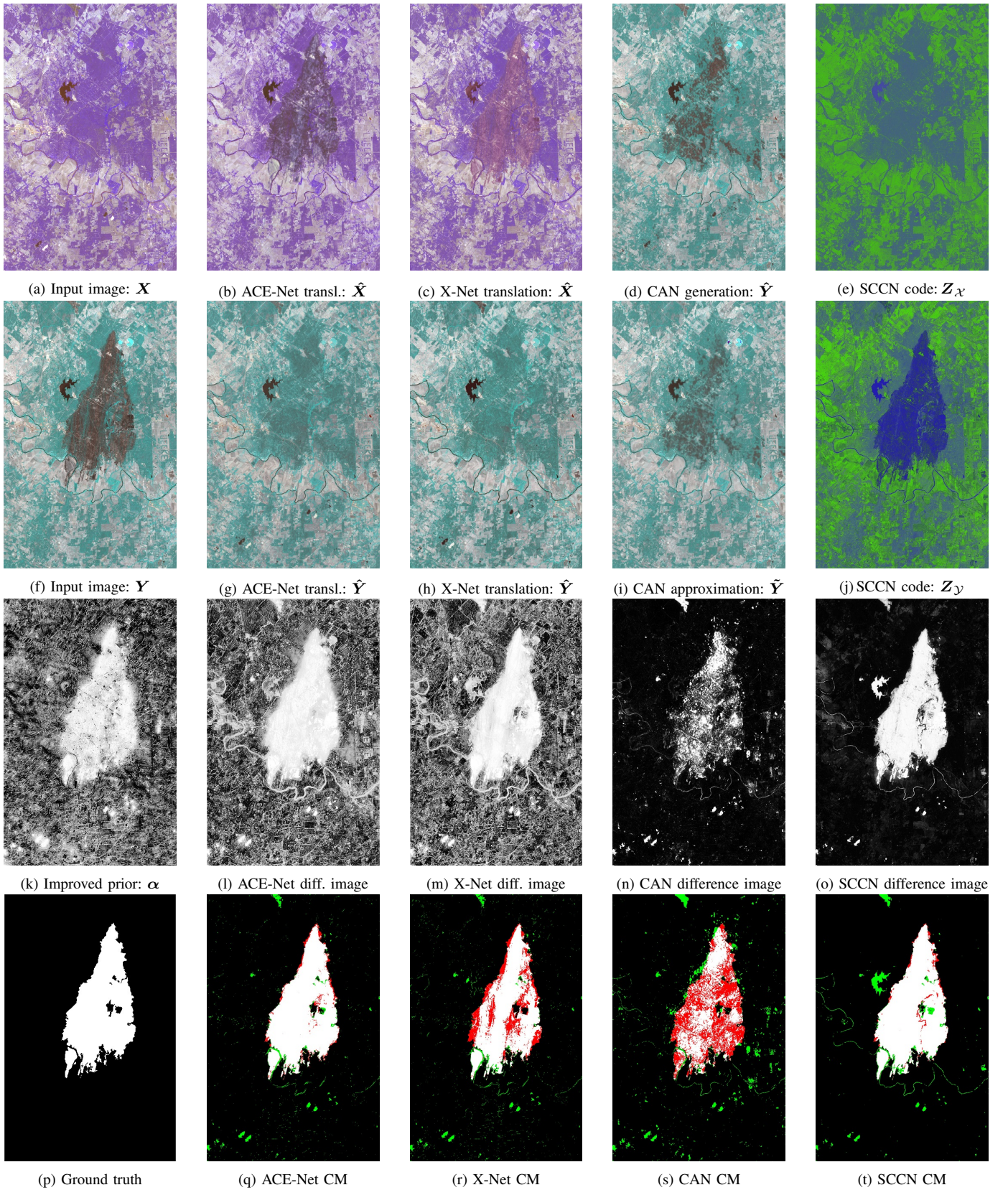
(a) Input image: $\boldsymbol{X}$

(b) ACE-Net transl.: $\hat{\boldsymbol{X}}$

(c) X-Net translation: $\hat{\boldsymbol{X}}$

(d) CAN generation: $\hat{\boldsymbol{Y}}$

(e) SCCN code: $\boldsymbol{Z}_{\mathcal{X}}$

(f) Input image: $\boldsymbol{Y}$

(g) ACE-Net transl.: $\hat{\boldsymbol{Y}}$

(h) X-Net translation: $\hat{\boldsymbol{Y}}$

(i) CAN approximation: $\tilde{\boldsymbol{Y}}$

(j) SCCN code: $\boldsymbol{Z}_{\mathcal{Y}}$

(k) Improved prior: $\boldsymbol{\alpha}$

(l) ACE-Net diff. image

(m) X-Net diff. image

(n) CAN difference image

(o) SCCN difference image

(p) Ground truth

(q) ACE-Net CM

(r) X-Net CM

(s) CAN CM

(t) SCCN CM

Fig. 10: Texas dataset. **First column**: Input images $\boldsymbol{X}$ (a) and $\boldsymbol{Y}$ (f), IPC output $\boldsymbol{\alpha}$ (k), and ground truth (p); **Second column**: Transformed images $\hat{\boldsymbol{X}}$ (b) and $\hat{\boldsymbol{Y}}$ (g) obtained with the ACE-Net, their difference image (l) and resulting confusion map (CM) (q); **Third column**: Transformed images $\hat{\boldsymbol{X}}$ (c) and $\hat{\boldsymbol{Y}}$ (h) obtained with the X-Net, their difference image (m) and resulting CM (r); **Fourth column**: Generated SAR image $\hat{\boldsymbol{Y}}$ (d) and approximated image $\tilde{\boldsymbol{Y}}$ (i) obtained with CAN, their image difference (n), and resulting CM (s); **Fifth column**: Code images $\boldsymbol{Z}_{\mathcal{X}}$ (e) and $\boldsymbol{Z}_{\mathcal{Y}}$ (j) obtained with SCCN, their image difference (o), and resulting confusion CM (t).

(a) Input image: $\boldsymbol{X}$

(b) ACE-Net transl.: $\hat{\boldsymbol{X}}$

(c) X-Net translation: $\hat{\boldsymbol{X}}$

(d) CAN generation: $\hat{\boldsymbol{Y}}$

(e) SCCN code: $\boldsymbol{Z}_{\mathcal{X}}$

(f) Input image: $\boldsymbol{Y}$

(g) ACE-Net trans.: $\hat{\boldsymbol{Y}}$

(h) X-Net translation: $\hat{\boldsymbol{Y}}$

(i) CAN approx.: $\hat{\boldsymbol{X}}$

(j) SCCN code: $\boldsymbol{Z}_{\mathcal{Y}}$

(k) Improved prior: $\boldsymbol{\alpha}$

(l) ACE-Net diff. image

(m) X-Net diff. image

(n) CAN diff. image

(o) SCCN diff. image

(p) Ground truth

(q) ACE-Net CM

(r) X-Net CM

(s) CAN CM

(t) SCCN CM

Fig. 11: California dataset. **First column**: Input images $\boldsymbol{X}$ (a) and $\boldsymbol{Y}$ (f), IPC output $\boldsymbol{\alpha}$ (k), and ground truth (p); **Second column**: Transformed images $\hat{\boldsymbol{X}}$ (b) and $\hat{\boldsymbol{Y}}$ (g) obtained with the ACE-Net, their difference image (l) and resulting confusion map (CM) (q); **Third column**: Transformed images $\hat{\boldsymbol{X}}$ (c) and $\hat{\boldsymbol{Y}}$ (h) obtained with the X-Net, their difference image (m) and resulting CM (r); **Fourth column**: Generated SAR image $\hat{\boldsymbol{Y}}$ (d) and approximated image $\tilde{\boldsymbol{Y}}$ (i) obtained with CAN, their image difference (n), and resulting CM (s); **Fifth column**: Code images $\boldsymbol{Z}_{\mathcal{X}}$ (e) and $\boldsymbol{Z}_{\mathcal{Y}}$ (j) obtained with SCCN, their image difference (o), and resulting confusion CM (t).
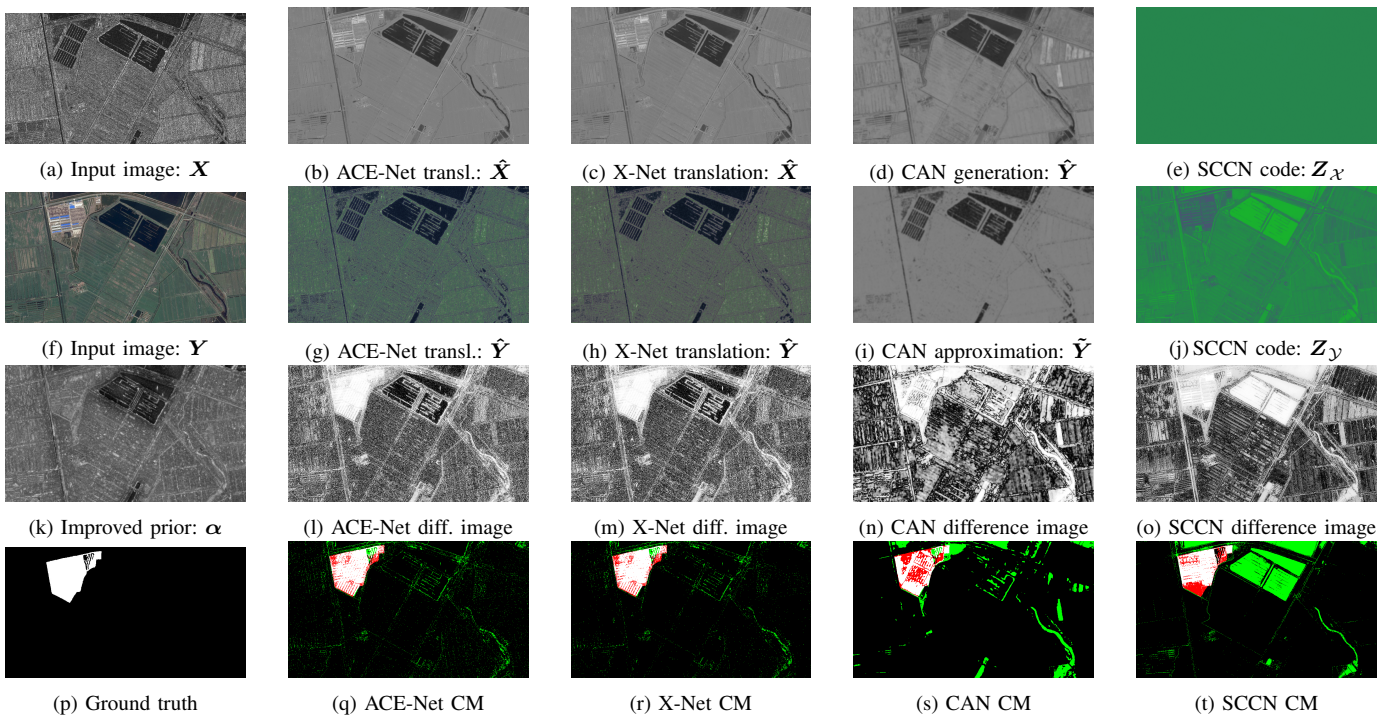
(a) Input image: $\boldsymbol{X}$ | (b) ACE-Net transl.: $\hat{\boldsymbol{X}}$ | (c) X-Net translation: $\hat{\boldsymbol{X}}$ | (d) CAN generation: $\hat{\boldsymbol{Y}}$ | (e) SCCN code: $\boldsymbol{Z}_{\mathcal{X}}$

(f) Input image: $\boldsymbol{Y}$ | (g) ACE-Net transl.: $\hat{\boldsymbol{Y}}$ | (h) X-Net translation: $\hat{\boldsymbol{Y}}$ | (i) CAN approximation: $\tilde{\boldsymbol{Y}}$ | (j) SCCN code: $\boldsymbol{Z}_{\mathcal{Y}}$

(k) Improved prior: $\boldsymbol{\alpha}$ | (l) ACE-Net diff. image | (m) X-Net diff. image | (n) CAN difference image | (o) SCCN difference image

(p) Ground truth | (q) ACE-Net CM | (r) X-Net CM | (s) CAN CM | (t) SCCN CM

Fig. 12: China dataset. **First column**: Input images $\boldsymbol{X}$ (a) and $\boldsymbol{Y}$ (f), IPC output $\boldsymbol{\alpha}$ (k), and ground truth (p); **Second column**: Transformed images $\hat{\boldsymbol{X}}$ (b) and $\hat{\boldsymbol{Y}}$ (g) obtained with the ACE-Net, their difference image (l) and resulting confusion map (CM) (q); **Third column**: Transformed images $\hat{\boldsymbol{X}}$ (c) and $\hat{\boldsymbol{Y}}$ (h) obtained with the X-Net, their difference image (m) and resulting CM (r); **Fourth column**: Generated SAR image $\hat{\boldsymbol{Y}}$ (d) and approximated image $\tilde{\boldsymbol{Y}}$ (i) obtained with CAN, their image difference (n), and resulting CM (s); **Fifth column**: Code images $\boldsymbol{Z}_{\mathcal{X}}$ (e) and $\boldsymbol{Z}_{\mathcal{Y}}$ (j) obtained with SCCN, their image difference (o), and resulting confusion CM (t).

[8] S. H. Khan, X. He, F. Porikli, and M. Bennamoun, "Forest change detection in incomplete satellite images with deep neural networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 9, pp. 5407–5423, 2017.

[9] M. Gong, J. Zhao, J. Liu, Q. Miao, and L. Jiao, "Change detection in synthetic aperture radar images based on deep neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 1, pp. 125–138, 2016.

[10] M. Gong, H. Yang, and P. Zhang, "Feature learning and change feature classification based on deep learning for ternary change detection in SAR images," *ISPRS J. Photogram. Remote Sens.*, vol. 129, pp. 212–225, 2017.

[11] H. Lyu, H. Lu, and L. Mou, "Learning a transferable change rule from a recurrent neural network for land cover change detection," *Remote Sens.*, vol. 8, no. 6, p. 506, 2016.

[12] L. Mou, L. Bruzzone, and X. X. Zhu, "Learning spectral-spatial-temporal features via a recurrent convolutional neural network for change detection in multispectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 924–935, 2018.

[13] M. Dalla Mura, S. Prasad, F. Pacifici, P. Gamba, J. Chanussot, and J. A. Benediktsson, "Challenges and opportunities of multimodality and data fusion in remote sensing," *Proc. IEEE*, vol. 103, no. 9, pp. 1585–1601, 2015.

[14] P. Ghamisi, B. Rasti, N. Yokoya, Q. Wang, B. Hofle, L. Bruzzone, F. Bovolo, M. Chi, K. Anders, and R. Gloaguen, "Multisource and multitemporal data fusion in remote sensing: A comprehensive review of the state of the art," *IEEE Geosci. Remote Sens. Mag.*, vol. 7, no. 1, pp. 6–39, 2019.

[15] M. Gong, X. Niu, T. Zhan, and M. Zhang, "A coupling translation network for change detection in heterogeneous images," *Int. J. Remote Sens.*, vol. 40, no. 9, pp. 3647–3672, 2019.

[16] W. Zhao, Z. Wang, M. Gong, and J. Liu, "Discriminative feature learning for unsupervised change detection in heterogeneous images based on a coupled neural network," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 12, pp. 7066–7080, 2017.

[17] T. Zhan, M. Gong, X. Jiang, and S. Li, "Log-based transformation feature learning for change detection in heterogeneous images," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 9, pp. 1352–1356, 2018.

[18] D. A. Landgrebe, *Signal Theory Methods in Multispectral Remote Sensing*. John Wiley & Sons, 2005, vol. 29.

[19] A. Moreira, P. Prats-Iraola, M. Younis, G. Krieger, I. Hajnsek, and K. P. Papathanassiou, "A tutorial on synthetic aperture radar," *IEEE Geosci. Remote Sens. Mag.*, vol. 1, no. 1, pp. 6–43, 2013.

[20] M. Gong, P. Zhang, L. Su, and J. Liu, "Coupled dictionary learning for change detection from multisource data," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 12, pp. 7077–7091, 2016.

[21] L. T. Luppino, F. M. Bianchi, G. Moser, and S. N. Anfinsen, "Unsupervised image regression for heterogeneous change detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 12, pp. 9960–9975, 2019.

[22] R. Touati and M. Mignotte, "An energy-based model encoding nonlocal pairwise pixel interactions for multisensor change detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 2, pp. 1046–1058, 2018.

[23] V. Ferraris, N. Dobigeon, Y. Cavalcanti, T. Oberlin, and M. Chabert, "Coupled dictionary learning for unsupervised change detection between multimodal remote sensing images," *Comput. Vis. Im. Und.*, vol. 189, p. 102817, 2019.

[24] J. Prendes, M. Chabert, F. Pascal, A. Giros, and J.-Y. Tourneret, "A new multivariate statistical model for change detection in images acquired by homogeneous and heterogeneous sensors," *IEEE Trans. Image Process.*, vol. 24, no. 3, pp. 799–812, 2015.

[25] M. Mignotte, "A fractal projection and markovian segmentation-based approach for multimodal change detection," *IEEE Trans. Geosci. Remote Sens.*, 2020.

[26] G. Mercier, G. Moser, and S. B. Serpico, "Conditional copulas for change detection in heterogeneous remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 5, pp. 1428–1441, May 2008.

[27] L. Gomez-Chova, D. Tuia, G. Moser, and G. Camps-Valls, "Multimodal classification of remote sensing images: A review and future directions," *Proceedings of the IEEE*, vol. 103, no. 9, 2015.

[28] D. Hong, L. Gao, N. Yokoya, J. Yao, J. Chanussot, Q. Du, and B. Zhang, "More diverse means better: Multimodal deep learning meets remote-sensing imagery classification," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–15, 2020.

[29] N. Audebert, B. Le Saux, and S. Lefèvre, "Beyond rgb: Very high resolution urban remote sensing with multimodal deep networks," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 140, pp. 20–32, 2018.

[30] P. Benedetti, D. Ienco, R. Gaetano, K. Ose, R. G. Pensa, and S. Dupuy, "M3 fusion: A deep learning architecture for multiscale multimodal multitemporal satellite data fusion," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 12, pp. 4939–4949, 2018.

[31] B. Rasti, D. Hong, R. Hang, P. Ghamisi, X. Kang, J. Chanussot, and J. A. Benediktsson, "Feature extraction for hyperspectral imagery: The evolution from shallow to deep (overview and toolbox)," *IEEE Geoscience and Remote Sensing Magazine*, pp. 0–0, 2020.

[32] N. Yokoya, P. Ghamisi, J. Xia, S. Sukhanov, R. Heremans, I. Tankoyeu, B. Bechtel, B. Le Saux, G. Moser, and D. Tuia, "Open data for global multimodal land use classification: Outcome of the 2017 ieee grss data fusion contest," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 5, pp. 1363–1377, 2018.

[33] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[34] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 110, pp. 3371–3408, 2010.

[35] L. Su, M. Gong, P. Zhang, M. Zhang, J. Liu, and H. Yang, "Deep learning and mapping based ternary change detection for information unbalanced images," *Pattern Recognition*, vol. 66, pp. 213–228, 2017.

[36] F. Bovolo and L. Bruzzone, "A theoretical framework for unsupervised change detection based on change vector analysis in the polar domain," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 1, pp. 218–236, 2007.

[37] P. Zhang, M. Gong, L. Su, J. Liu, and Z. Li, "Change detection based on deep feature representation and mapping transformation for multi-spatial-resolution remote sensing images," *ISPRS J. Photogram. Remote Sens.*, vol. 116, pp. 24–41, 2016.

[38] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 2672–2680, 2014.

[39] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn. (CVPR)*, pp. 5967 – 5976, 2017.

[40] N. Merkle, S. Auer, R. Müller, and P. Reinartz, "Exploring the potential of conditional adversarial networks for optical and sar image matching," *IEEE J. Select. Topics Appl. Earth Obs. Remote Sens.*, vol. 11, no. 6, pp. 1811–1820, 2018.

[41] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, pp. 2223–2232, 2017.

[42] Z. Murez, S. Kolouri, D. Kriegman, R. Ramamoorthi, and K. Kim, "Image to image translation for domain adaptation," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn. (CVPR)*, pp. 4500–4509, 2018.

[43] F. Bovolo and L. Bruzzone, "The time variable in data fusion: A change detection perspective," *IEEE Geosci. Remote Sens. Mag.*, vol. 3, no. 3, pp. 8–26, 2015.

[44] K. Koutroumbas and S. Theodoridis, *Pattern Recognition*. Elsevier Science, 2008. [Online]. Available: https://books.google.no/books?id=QgD-3Tcj8DkC

[45] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[46] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[47] S. Liu, S. De Mello, J. Gu, G. Zhong, M.-H. Yang, and J. Kautz, "Learning affinity via spatial propagation networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 1520–1530.

[48] X. Cheng, P. Wang, and R. Yang, "Depth estimation via affinity learned with convolutional spatial propagation network," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 103–119.

[49] M. Maire, T. Narihira, and S. X. Yu, "Affinity cnn: Learning pixel-centric pairwise relations for figure/ground embedding," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 174–182.

[50] F. R. Chung and F. C. Graham, *Spectral Graph Theory*. American Mathematical Society, 1997, no. 92.

[51] J. N. Myhre and R. Jenssen, "Mixture weight influence on kernel entropy component analysis and semi-supervised learning using the Lasso," *Proc. IEEE Int. Workshop Mach. Learn. Signal Process. (MLSP)*, pp. 1–6, 2012.

[52] M. P. Wand and M. C. Jones, *Kernel Smoothing*, ser. Monographs on Statistics and Applied Probability (Vol. 60). Chapman & Hall/CRC, 1995.

[53] R. Touati, M. Mignotte, and M. Dahmane, "Anomaly feature learning for unsupervised change detection in heterogeneous images: A deep sparse residual model," *IEEE J. Select. Topics Appl. Earth Obs. Remote Sens.*, vol. 13, pp. 588–600, 2020.

[54] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected CRFs with Gaussian edge potentials," *Proc. Adv. Neural Inf. Process. Syst.*, pp. 109–117, 2011.

[55] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst. Man Cybern.*, vol. 9, no. 1, pp. 62–66, 1979.

[56] J. N. Kapur, P. K. Sahoo, and A. K. Wong, "A new method for gray-level picture thresholding using the entropy of the histogram," *Computer Vision, Graphics, and Image Processing*, vol. 29, no. 3, pp. 273–285, 1985.

[57] A. G. Shanbhag, "Utilization of information measure as a means of image thresholding," *CVGIP: Graphical Models and Image Processing*, vol. 56, no. 5, pp. 414–419, 1994.

[58] J.-C. Yen, F.-J. Chang, and S. Chang, "A new criterion for automatic multilevel thresholding," *IEEE Trans. Image Process.*, vol. 4, no. 3, pp. 370–378, 1995.

[59] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37–46, 1960.

[60] M. Volpi, G. Camps-Valls, and D. Tuia, "Spectral alignment of multi-temporal cross-sensor images with automated kernel canonical correlation analysis," *ISPRS J. Photogram. Remote Sens.*, vol. 107, pp. 50–63, 2015.

[61] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 30, no. 1, p. 3, 2013.

[62] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, *Efficient backprop*. Springer, 2012, pp. 9–48.

[63] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," arXiv:1502.03167 [cs.LG], 2015.

[64] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[65] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *Proc. Int. Conf. Artificial Intell. Statist. (AISTATS)*, pp. 249–256, 2010.

[66] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[67] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," pp. 132–149, 2018.

[68] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," pp. 9446–9454, 2018.

[69] R. Touati, M. Mignotte, and M. Dahmane, "A reliable mixed-norm-based multiresolution change detector in heterogeneous remote sensing images," *IEEE J. Select. Topics Appl. Earth Obs. Remote Sens.*, vol. 12, no. 9, pp. 3588–3601, 2019.

[70] E. Chuvieco, *Fundamentals of Satellite Remote Sensing: An Environmental Approach*. CRC press, 2016.

[71] R. Touati, M. Mignotte, and M. Dahmane, "Change detection in heterogeneous remote sensing images based on an imaging modality-invariant MDS representation," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, 2018, pp. 3998–4002.

[72] V. Akbari, S. N. Anfinsen, A. P. Doulgeris, T. Eltoft, G. Moser, and S. B. Serpico, "Polarimetric sar change detection with the complex hotelling-lawley trace statistic," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 7, pp. 3953–3966, 2016.

[73] D. Solarna, G. Moser, and S. B. Serpico, "A markovian approach to unsupervised change detection with multiresolution and multimodality sar data," *Remote Sens.*, vol. 10, no. 11, 2018.

[74] H. L. V. Trees, *Detection, Estimation, and Modulation Theory*. John Wiley & Sons, 2001.

Luigi Tommaso Luppino received the B.Sc. in Telecommunication Engineering (2013), and the M.Sc. in Multimedia Signal Processing And Telecommunication Networks (2016) from University of Genoa, Italy. He received the Ph.D. degree from the Machine Learning Group at UiT The Arctic University of Norway in Tromsø, Norway, in 2020. Luigi's research is related to pattern recognition and machine learning applied to Earth observation and satellite remote sensing. His interests include image processing, clustering, classification, and deep learning.

Michael Kampffmeyer received the Ph.D. degree from the Machine Learning Group, UiT The Arctic University of Norway, Tromsø, Norway, in 2018. He holds a associate professor position at the Machine Learning Group, UiT The Arctic University of Norway. His research interests include the development of unsupervised deep learning methods for representation learning and clustering by utilizing ideas from kernel machines and information theoretic learning. His paper "Deep Kernelized Autoencoders" received the Best Student Paper Award at the Scandinavian Conference on Image Analysis in 2017. For more details, visit https://sites.google.com/view/michaelkampffmeyer/.

Filippo Maria Bianchi received the B.Sc. in Computer Engineering (2009), the M.Sc. in Artificial Intelligence and Robotics (2012) and the PhD in Machine Learning (2015) from "Sapienza" University, Rome. Filippo worked 2 years as research assistant at the Computer Science department at Ryerson University, Toronto. Filippo's research interests in machine learning and pattern recognition include graph and sequence matching, clustering, classification, reservoir computing, deep learning and data mining.

Gabriele Moser received the Laurea (M.Sc. equivalent) degree in telecommunications engineering and the Ph.D. degree in space sciences and engineering from the University of Genoa, Genoa, Italy, in 2001 and 2005, respectively. Since 2014, he has been an Associate Professor of Telecommunications with the University of Genoa. His research activity is focused on pattern recognition and image processing methodologies for remote sensing and energy applications. He is author or co-author or more than 140 papers in international scientific journals, conference proceedings, and edited books. Since 2001, he has been cooperated with the Image Processing and Pattern Recognition for Remote Sensing (IPRS) Lab, University of Genoa. He has been the head of the RIOS Lab since 2013.

Sebastiano Bruno Serpico received the Laurea (M.S.) degree in electronic engineering (1982) and the Ph.D. degree in telecommunications (1989) from the University of Genoa, Italy. He is currently a Full professor of Telecommunications with the Polytechnic School of the University of Genoa. He is also the Head of the IPRS Lab at DITEN. His current research interests include pattern recognition for remote sensing images and biomedical images. He is also the Chairman of the Institute of Advanced Studies in Information and Communication Technologies. He has been the Project Manager of numerous research projects and an Evaluator of project proposals for various programs of the European Union, the Italian Space Agency, and the Italian Ministry of Education and Research. He received the Best Paper Award at the 2010 IEEE Workshop on Hyperspectral Image and Signal Processing. He is an Associate Editor of the International Journal IEEE Transactions on Geoscience and Remote Sensing (TGRS). He was a Guest Editor of two special issues of TGARS regarding the analysis of hyperspectral image data (July 2001 issue) and recent advances in techniques for analysis of remotely sensed data (March 2005), and of a special issue of the Proceedings of the IEEE on remote sensing of natural disasters. From 1998 to 2002, he was the Chairman of the SPIE/EUROPTO series of conferences on signal and image processing for remote sensing. He was the Co-Chair of the IEEE International Geoscience and Remote Sensing Symposium in 2015 (Milan, Italy, 2015).

Robert Jenssen (M'02) received the PhD (Dr. Scient.) in Electrical Engineering from the University of Tromsø, in 2005. Currently, he is a Professor and Head of the Machine Learning Group at UiT The Arctic University of Norway, Tromsø, Norway. Jenssen received the 2005 IEEE ICASSP Outstanding Student Paper Award, the 2007 UiT Young Investigator Award, and the 2013 IEEE Geoscience and Remote Sensing Society Letters Best Paper Award. He serves on the IEEE Technical Committee on Machine Learning for Signal Processing, he is on the IAPR Governing Board, and he is an Associate Editor for the journal Pattern Recognition. Jenssen is the General Chair of the annual Northern Lights Deep Learning Workshop - NLDL.

Stian Normann Anfinsen received the M.Sc. degree in communications, control and digital signal processing from the University of Strathclyde in Glasgow, UK (1998) and the Cand.mag. (1997), Cand.scient. (2000) and Ph.D. degrees (2010) from UiT The Arctic University of Norway in Tromsø, Norway. Since 2014 he has been an Associate Professor with the Department of Physics and Technology at UiT, formerly with the Earth Observation Group and currently as a member of the Machine Learning Group. His interests are in statistical modelling, pattern recognition and machine learning algorithms for image analysis.