UiT The Arctic University of Norway

Faculty of Science and Technology

Department of Physics and Technology

Unsupervised Change Detection in Heterogeneous Remote Sensing Imagery

Luigi Tommaso Luppino

A dissertation for the degree of Philosophiae Doctor – March 2020

It don't matter what you say or do
It just seems to work out if you want it to
Let out all the slack, take it off your back
Carry on, carry on

J.J. Cale, *Carry on*,
Shades, Shelter Records, 1981.

Abstract

Change detection is a thriving and challenging topic in remote sensing for Earth observation. The goal is to identify changes that happen on the Earth by comparing two or more satellite or aerial images acquired at different times. Traditional methods rely on homogeneous data, that is, images acquired by the same sensor, under the same geometry, seasonal conditions, and recording configurations.

However, the assumption of homogeneity does not hold true for many practical examples and applications, and in particular when different sensors are involved. This represents a significant limitation, both in terms of response time to sudden events and in terms of temporal resolution when monitoring long-term trends.

The alternative is to combine heterogeneous data, which on one hand allows to fully exploit the capabilities of all the available sensors, but on the other hand raises additional technical challenges. Indeed, heterogeneous sources imply different data domains, diverse statistical distributions and inconsistent surface signatures across the various image acquisitions.

This thesis tries to explore the kinds of techniques meant to cope with these issues, which are referred to as *heterogeneous* change detection methods. Specifically, the effort is dedicated to unsupervised learning, the branch of machine learning which does not rely on any prior knowledge about the data. This problem setting is as challenging as important, in order to tackle the task in the most automatic way without relying on any user interaction.

The main novelty driving this study is that the comparison of affinity matrices can be used to define crossdomain similarities based on pixel relations rather than the direct comparison of radiometry values. Starting from this fundamental idea, the research endeavours presented in this thesis result in the formulation of three methodologies that prove themselves reliable and perform favourably when compared to the state-of-the-art. These methods leverage this affinity matrix comparison and incorporate both conventional machine learning techniques and more contemporary deep learning architectures to tackle the problem of unsupervised heterogeneous change detection.

Acknowledgments

Since these words are very likely to be the most read of the whole thesis I will make the most out of them, so please bear with me! Who knows me best should remember that I used to refer to this Ph.D. as the labours of Sisyphus. For the ones unfamiliar with Greek mythology, Sisyphus was the king of Ephyra (now known as Corinth), whose craftiness and intelligence were matched only by his arrogance. He hubristically believed that he could even outsmart Gods, who punished his overconfidence by forcing him to eternally repeat a task as laborious as futile: push an immense boulder up a hill only for it to roll down when it nears the top. Truly, my endeavours to achieve my goals felt useless, my attempts seemed clumsy, and my failures had the bitter taste of unending frustration. In fact, I must admit that I was ready to quit my quest. Eventually I did not, and I am now aware that research does not mean reaching one peak. Instead, it is an endless journey in which the accomplishments of today are just the first steps of tomorrow.

If I am here bothering you with this cheap philosophy, I owe it mainly to you, *Stian*, who helped me going through difficult paths, encouraged me not to give up, and held me up on this slippery ladder. Without your providential guidance, wise advise, and generous help, I would have lost my way a long ago. In an equal manner, my most sincere gratitude goes to you, *Gabriele*, for being the most enthusiastic of my supporters and the most strict of my reviewers. From the moment I decided to do research, you two are the ones I have been looking up to.

I would also like to thank my co-supervisors *Robert* and *Bruno*, who provided me with fruitful discussions, precious suggestions, and thorough revisions. In addition, I am grateful to my opponents *Francesca* and *Yann* who have spent their time and efforts to evaluate my thesis.

Filippo and *Michael*, both of you deserve a special mention, because your invaluable support and contributions throughout this project have been priceless to me. You managed to bring the best out of my ideas and succeeded in the miracle to translate these into concrete results. I want you to know that working by your side is a great pleasure.

Another person I shall give a lot of credits is *Thomas*. The endless job you do to maintain the servers of our group goes way beyond your duties,

and the fact that I did not (entirely) lose my mind in the period October - December 2019 is especially because of you. *Sigurd*, you too are equally worth mentioning: I always appreciate the incredible patience you apply with me, coming up with all my bothering questions and stupid doubts! *Karl Øyvind* as well, I *"bet"* you know by now how much I esteem your opinion, whether we are discussing about non-parametric regression, skiing equipment, political matters or football games.

I will be Stian's first graduated Ph.D. candidate: what a shame! But no worries, the other two members of the Team Satellite, *Sara* and *Jørgen*, will restore his reputation soon! Jokes aside, what I want to attest is the great regard I have for both of you, whose potential is in my opinion glaring! I also would like to extend my gratitude to the whole Machine Learning Group. The working environment is the best I could ask for, where there is always time for a laugh. Imagine *Jonas* whispering Italian swearwords to *Michael* (like "*crucco del cazzo*") while *Kristoffer* shouts something in pseudo-Spanish hoping for *Miguel* to understand his gibberish. All this while *Changkyu* is trying to teach us some Korean! What an amazing tower of Babel!

Latins used to say *"mens sana in corpore sano"*, so if the writing of this thesis did not drive me insane is also thanks to the *Tromsø Studentenes Idrettslag* (*TSI*) volleyball team. For me, this team is more than that: becoming part of *TSI* meant gaining a whole new family branch, because this is how it feels hanging out with you guys. Till the end, *"To, Tre, Børre!"*

If I consider Tromsø my new home, it is also because I felt *at home* wherever I have been living with you, my brother *Bilal*, and with you, *Glenn*. Thank you, for the incredible memories we share, the cosy dinners, the amazing parties, the dreadful hangovers, and even a published paper! An unbelievable feeling strikes me when I think of all the highs and the lows of this incredible journey, which would have been much less thrilling without having you alongside me. When I mention my flatmates I should also include you, *Dorota*, given how much time you spent as part of our gang. Thank you, for your eruptive energy and your incurable optimism, which matches only with your total craziness. For the three of you, I will always be your *diva*. Another person who jollied up my winters in Tromsø is *Karoline*: although I am not your favourite Italian anymore, I am sure I am still making you proud with the continuous developments of my skiing skills! About Italians, I would like to acknowledge how important my fellow countrymen have been to contain my homesickness

for our motherland: *Enrico*, *Pietro*, *"er Messi de Torbellamonaca" Filippo*, *Umberto* and all the others, I thank you all for the wonderful time spent reminding ourselves what we miss (and what we do not) about Italy.

For sure, something I miss about Italy is my friends, from my hometown Ventimiglia and from Genova, for whom I would need pages and pages to name them all. I will limit myself to the two pillars who sustained me the most, especially in the darkest winters: *Michele* and *Silvia*. To me, you are the most clear examples of friendship, wisdom, empathy, and the living proof that true friends can be apart for months and meet up at the bar as if time never went by.

I would not be the son I am without a father like you, *Papo*, who taught me to stand strong in front of the difficulties and who always showed me unconditional support. *"From the moment I could talk I was ordered to listen"*: these words by Cat Steven are not a fair description of our relationship, because you always treat me as an adult at your same level, expecting me to behave like one and considering my opinion as important as yours. I would not be the brother I am without a sister like you, *Irene*, who eased my way by setting an example, who spurred me to become better because I had to be better! Or simply, who guides me like a lighthouse in the dark, shining so bright that I can still see you from up here so far in the north. I cannot wait to embarrass you with my speech at your wedding. Finally, I would not be the son I am without a mother like you, *Mamma*, who raised me with discipline and love, who keeps on redefining my platonic ideas of inner strenght and power of will with continuous examples of unbreakable determination, and who more than anyone encouraged me to chase my dreams and ambitions. No spoken language can express how much I love you.

Now, I left the sugar for last, as they say in Croatian. The fact that you, Tena, bore with me during these stressful months has proven once more that you deserve to be made a saint. Nonetheless, the halo does not suit you because you are a girl with her feet firmly on the ground, which is why I keep on saying that *"sei un fiore che è cresciuto sull'asfalto e sul cemento."* You scared away my ghosts, and for that I will always be grateful to you. Thank you, Bubu, for being right here when I need you, and for reminding me that it is OK to take a break, before I resume pushing that fucking stupid rock up the Goddamn hill.

Gigi, February 2020

Contents

List of Figures

List of Abbreviations

ACE-Net Adversarial Cyclic Encoder Network.

AE Autoencoder.

CD Change Detection.

cGAN Conditional Generative Adversarial Network.

CNN Convolutional Neural Network.

CT Computerised Tomography.

DKAE Deep Kernelised Autoencoder.

DL Deep Learning.

DM Dissimilarity Measure.

GAN Generative Adversarial Network.

GP Gaussian Process.

GPU Graphics Processing Unit.

I2I Image-to-Image.

MRI Magnetic Resonance Imaging.

NN Neural Network.

PET Positron-Emission Tomography.

RBF Radial Basis Function.

ReLU Rectified Linear Unit.

RF Random Forest.

RGB Red, Green, and Blue.

SAR Synthetic Aperture Radar.

SDAE Stacked Denoising Autoencoder.

SM Similarity Measure.

SVM Support Vector Machine.

Chapter 1

Introduction

Change detection (CD) is a well known task in pattern recognition and image analysis: the goal is to recognise changes by the comparison of imagery acquired over the same scene but at different times. CD applications encompass, to name a few, medical diagnosis and treatment [1], surveillance [2], civil infrastructures condition assessment [3], underwater monitoring [4], and Earth observation, which is the unique focus of this thesis.

1.1 Brief overview

The flourishing of Earth observation platforms in the new millennium has led to a large plethora of available products [5, 6]. There is a myriad of satellite, airborne, and unmanned aircraft missions, and all the combinations of acquisition settings and modalities are innumerable [7]. Thanks to the open access policies applied nowadays by the space agencies, the end users have access to a tremendous amount of free data stored in databases which are growing by the day. Data fusion methodologies [8, 9] are then necessary to exploit the totality of this goldmine.

CD is one of the methodological approaches that are thriving thanks to the growth of the remote sensing industry. This is because of its undeniable importance for society. Changes on the Earth surface are the result of natural and human processes, and can be abrupt, due to sudden events, or subtle, caused by slow trends difficult to perceive at the human time scale [9]. De-

tecting them with certainty, assessing them adequately, and responding to them promptly can save resources, potentially lives, or guide the planning of future strategies and politics. For example, time is of the essence when it comes to containing the damages of a forest fire or an oil spill, so it is crucial to intervene as soon as possible. In the same way, becoming aware of the unexpected growth of a city over a long period might lead to reconsider the appropriateness of its infrastructures, in order to prevent long term consequences.

Surely, the analyses can be carried out better if there is an abundance of images that can be used for comparison. However, conventional CD methods come with a great limitation, since they are designed to operate with homogeneous data. The latter refers to imagery recorded by the same payloads, under the same geometries and seasonal or weather conditions, and using the same configurations and settings. Truly, this means that once a sensor acquisition is selected as reference, most of the other available images do not fulfil these requirements, and cannot be considered to perform CD in its traditional fashion.

The latest breakthroughs in computational technology and the advances in machine learning methodologies have eventually led the CD community to elaborate new approaches able to combine data collected by different sources. These techniques, for which the hypothesis of homogeneity across the images does not necessarily need to hold true, are called *heterogeneous* CD methods. Clearly, their strongest advantage lies in the ability to make use of any sort of data, regardless of the circumstances under which these data have been produced. On the other hand, relaxing (or even lifting) the restrictions of homogeneity imposed on the acquisitions imply the raise of additional issues. In fact, these represent the main drawback: dealing with heterogeneous sources can imply incompatible data, for which the direct comparison is pointless, if not even unfeasible. There might be a mismatch between the data probability distributions, which may lie in unrelated domains where the investigated objects can have inconsistent representations. Heterogeneous CD methods are apt to meet these challenges, and they face them in many diverse ways, among others by means of similarity measures [10, 11], local descriptors [12], data transformation [13, 14], segmentation [15], classification [16, 17], and clustering [18].

The study conducted in this work dedicates most of its attention to the ap-

proaches tackling the problem by finding meaningful transformations able to map data across the different domains and, therefore, allowing data comparisons which would be impossible otherwise. Most importantly, the focus is set on the case of unsupervised learning. Unsupervised frameworks do not require any information about the data to be provided in advance, and can therefore be more appealing than the supervised counterparts in many practical settings. Although it might be argued that the supply of training data by manual selection does not represent a strong requirement [19, 20], it still prompts meticulous user interaction which can be costly, time-consuming, sometimes incompatible with the time requirements of the applications, and possibly even inaccurate, especially when images are difficult to interpret visually [21].

This thesis presents a selection of unsupervised methodologies for heterogeneous CD proposed by the author, which are enclosed in the form of the papers hereby listed:

(I) Luigi T. Luppino, Filippo M. Bianchi, Gabriele Moser and Stian N. Anfinsen, "**Unsupervised image regression for heterogeneous change detection**," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 12, pp. 9960-9975, Dec. 2019.

(II) Luigi T. Luppino, Michael Kampffmeyer, Filippo M. Bianchi, Gabriele Moser, Sebastiano B. Serpico, Robert Jenssen, and Stian N. Anfinsen, "**Deep image translation with an affinity-based change prior for unsupervised multimodal change detection**," *IEEE Transactions on Geoscience and Remote Sensing*, submitted.

(III) Luigi T. Luppino, Mads A. Hansen, Michael Kampffmeyer, Filippo M. Bianchi, Gabriele Moser, Robert Jenssen, and Stian N. Anfinsen, "**Code-aligned autoencoders for unsupervised change detection in multimodal satellite images**," *IEEE Transactions on Neural Networks and Learning Systems*, submitted.

This dissertation is built upon the key idea that the comparison of affinity matrices across multimodal images is a fruitful analysis useful to extract preliminary information about where the changes have happened. The effectiveness of this approach is demonstrated within all the aforementioned proposed methods. In Paper I, this knowledge serves the purpose of selecting training data samples automatically from unchanged areas, which are then

used for the tuning of regression functions mapping data across domains. Instead, Paper II leverages this prior information to train two deep learning frameworks designed to perform image translation. Paper III achieves the same goal by assuring that the inferred crossmodal similarities evaluated across the input domains are embedded also in a common latent space.

1.2 Reading guide

In the following, a summary of the content of this thesis is provided, including background theory, proposed methodologies, resulting publications, and final remarks.

Chapter 2 introduces the problem, putting particular stress on motivations, challenges, and solutions.

Chapter 3 covers the paradigm of data transformation, which is central for all the topics included in this manuscript.

Chapter 4 presents the most advanced techniques and models related to deep learning, which inspired the design of the proposed CD frameworks featuring state-of-the-art architectures.

Chapter 5 describes the most important contributions to the field of study, mainly associated with local information extraction through affinity matrix comparison.

Chapter 6 summarises the achievements accomplished with the research endeavours.

Chapters 7 to 9 report the enclosed papers.

Chapter 10 concludes this work with some take-away messages and proposes a number of possible future developments.

Chapter 2

Heterogeneous change detection in remote sensing

This chapter offers an overview of the main motivations behind this project, namely the variety of the available sensors and the limitations of the traditional CD techniques, and the challenges faced by heterogeneous CD methods. Finally, a possible taxonomy of the latter is provided, and the methods presented in the enclosed papers are framed within this picture.

2.1 The variety of remote sensing data

Several books and surveys reporting the basics and the last advances in remote sensing can be found in the literature [5, 22, 23, 24]. Toth and Jóźków [7] list the main remote sensing platforms, providing a compact but yet comprehensive review of applications, specifics and technical details. What can be noticed in all these sources is that optical and synthetic aperture radar (SAR) sensors are the most important for Earth observation applications, and those that are dominantly used for CD in remote sensing [25, 26]. Nonetheless, the number of possible different configurations for both of these type of image sources is remarkably large.

2.1.1 Optical and SAR sensors

Optical and SAR payloads are often seen as *complementary*, because of the physical processes and properties they record. Optical systems consist of passive sensors that measure radiance in multispectral bands covering visible, near-infrared and thermal infrared wavelengths of the electromagnetic spectrum. SAR systems carry active sensors: they transmit pulses of microwaves and receive the backscattered echoes resulting from these pulses bouncing off the Earth surface. Clearly, 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.

That said, the advantages of optical data with respect to SAR are in fact considerable. The optical images take real values affected by a modest additive Gaussian noise (mainly due to atmospheric disturbance and thermal noise inside the sensor) [22, 24], whose effect can be easily accounted for. In addition, improved receiver gains can enhance the power-to-noise ratio [23]. On the contrary, the working principle of SAR systems is also the intrinsic cause of their main issue: SAR pixels 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 [27]. This is

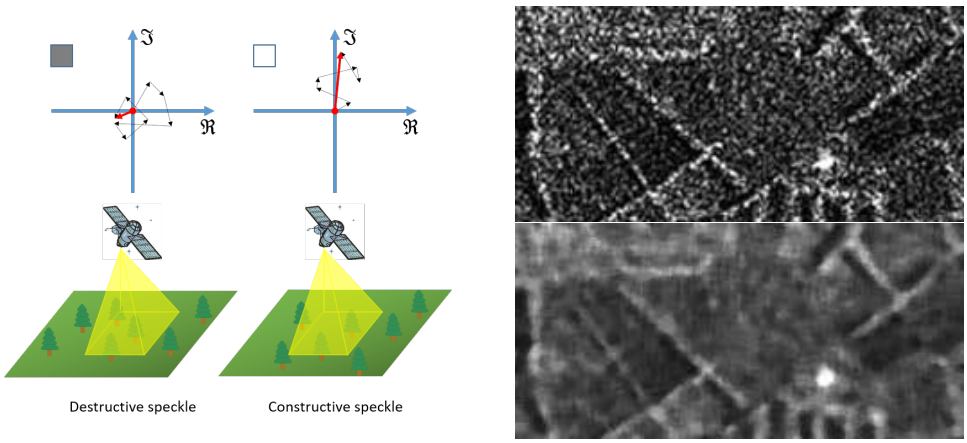


Figure 2.1: **(Left)** The surface roughness might cause destructive (constructive) speckle, for which the received echoes cancel (sum up) in the complex plane, resulting in the dark (bright) pixel intensity shown in the corner. **(Right)** Spatial filtering can smooth the images, at the cost of a lower resolution. Images from [28].

the so-called *speckle*, a multiplicative effect notoriously more difficult to mitigate. Figure 2.1 on the left helps to visualise the problem, and shows that increasing the intensity of the sent pulses to increase the power-to-noise ratio at the receiver is not beneficial [23]. Two possible solutions are *multilooking* [28], a noncoherent local averaging of the input during data acquisition, and postprocessing spatial filtering [28]. Both techniques smooth the image, but reduce the spatial resolution (see Figure 2.1 on the right).

Optical instruments suffer from panoramic distortions that worsen as the look angle increases, referring to the angle between the azimuth direction and the observed objects [6]. Also shadowing effects can arise with wider angles: when the scene contains high objects (e.g. mountains), one of their sides might be invisible to the sensor. Both these problems are illustrated in Figure 2.2. The same can be said for SAR systems for which, in fact, the problem is more complicated due to the side-looking viewing geometry, and the fact that the radar is fundamentally a distance measuring device (i.e. measuring range). Truly, the height and the steepness of the observed objects have an impact at any range, causing additional artifacts [6]. Foreshortening indicates the case in which the slope of a surface facing the sensor is such that it looks shorter in the SAR image. When the slope is steep enough, the pulses bouncing off the top of an object are received earlier than the ones at the bottom, causing so-called layovers (see Figure 2.3).

Geocoding is applied in order to solve these issues. That is, digital elevation models are used to compensate the effects of the terrain geometry. Nonetheless, these are not useful at higher image resolutions, when even the building shapes and dispositions matter, as illustrated in Figure 2.4. The examples in this figure offer also an overview of the heterogeneity between SAR and optical data. Apart from the obvious differences between the surface signatures, the latter are in general more user-friendly and clear, and they do not depend so much on the geometry of the acquisition as the former, which require more expertise for visual interpretation.

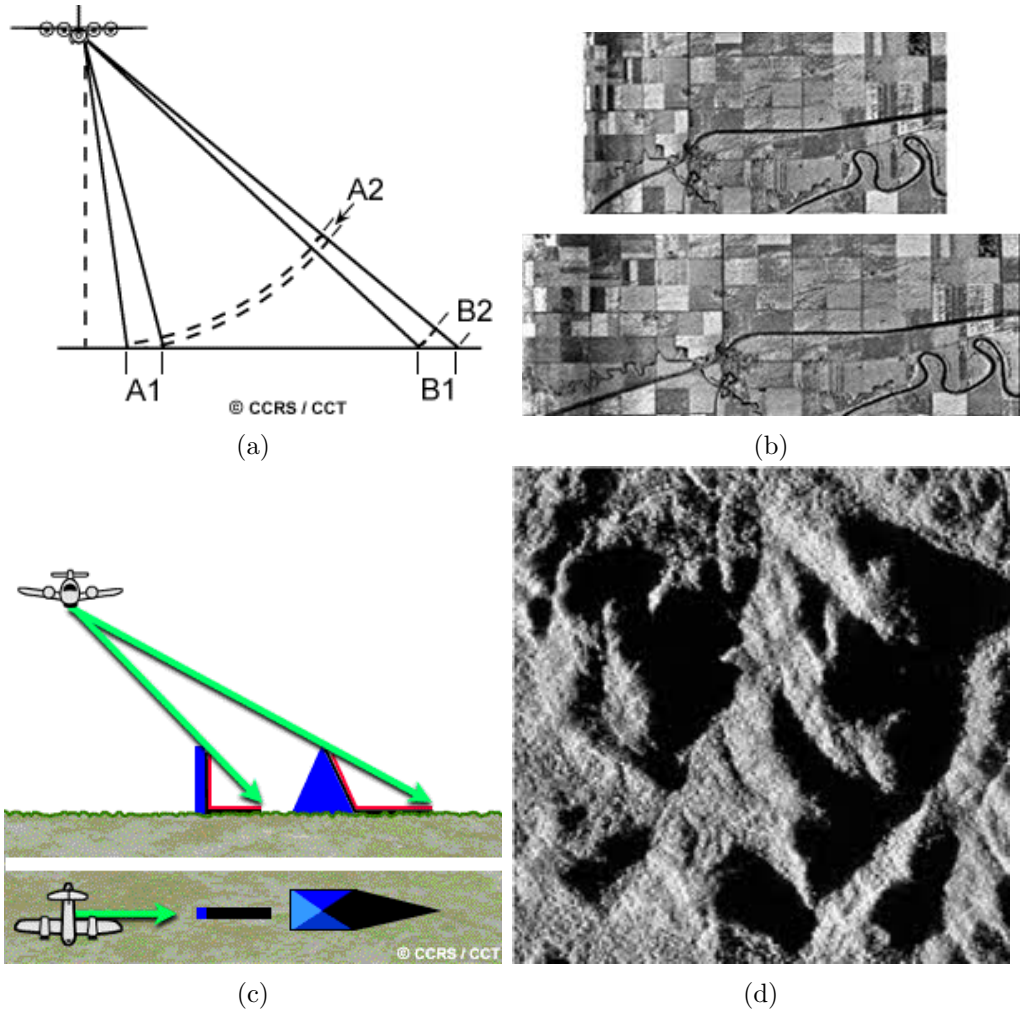


Figure 2.2: Panoramic distortion and shadowing. (a) The area of A1 and B1 are the same on the ground, but A2 is smaller than B2 on the image plane; (b) This slant distortions can be easily corrected thanks to basic trigonometry; (c) shadowing due to tall objects cannot be corrected. The red surfaces are invisible to the sensor, resulting in black areas in the image which contain no information; (d) Example of shadowing in a SAR image. Images from [28].

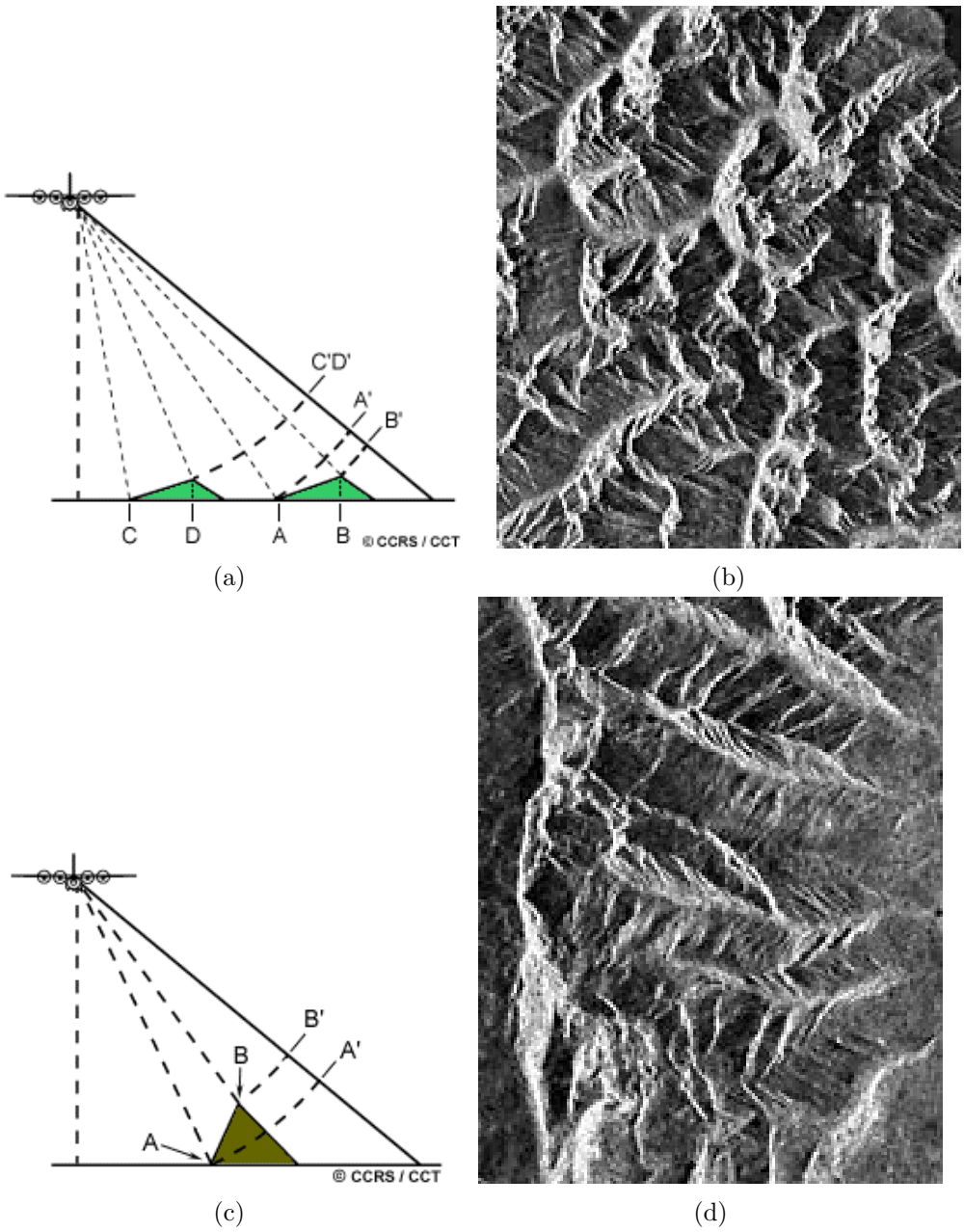


Figure 2.3: Terrain effects on a SAR image. (a-b) Foreshortening: the slopes appear compressed ($AB \rightarrow A'B'$) or even reduced to zero ($CD \rightarrow C'D'$); (c-d) Layover: the return signal from the top of an object is received before the signal from the bottom, flipping its representation upside-down. Images from [28].

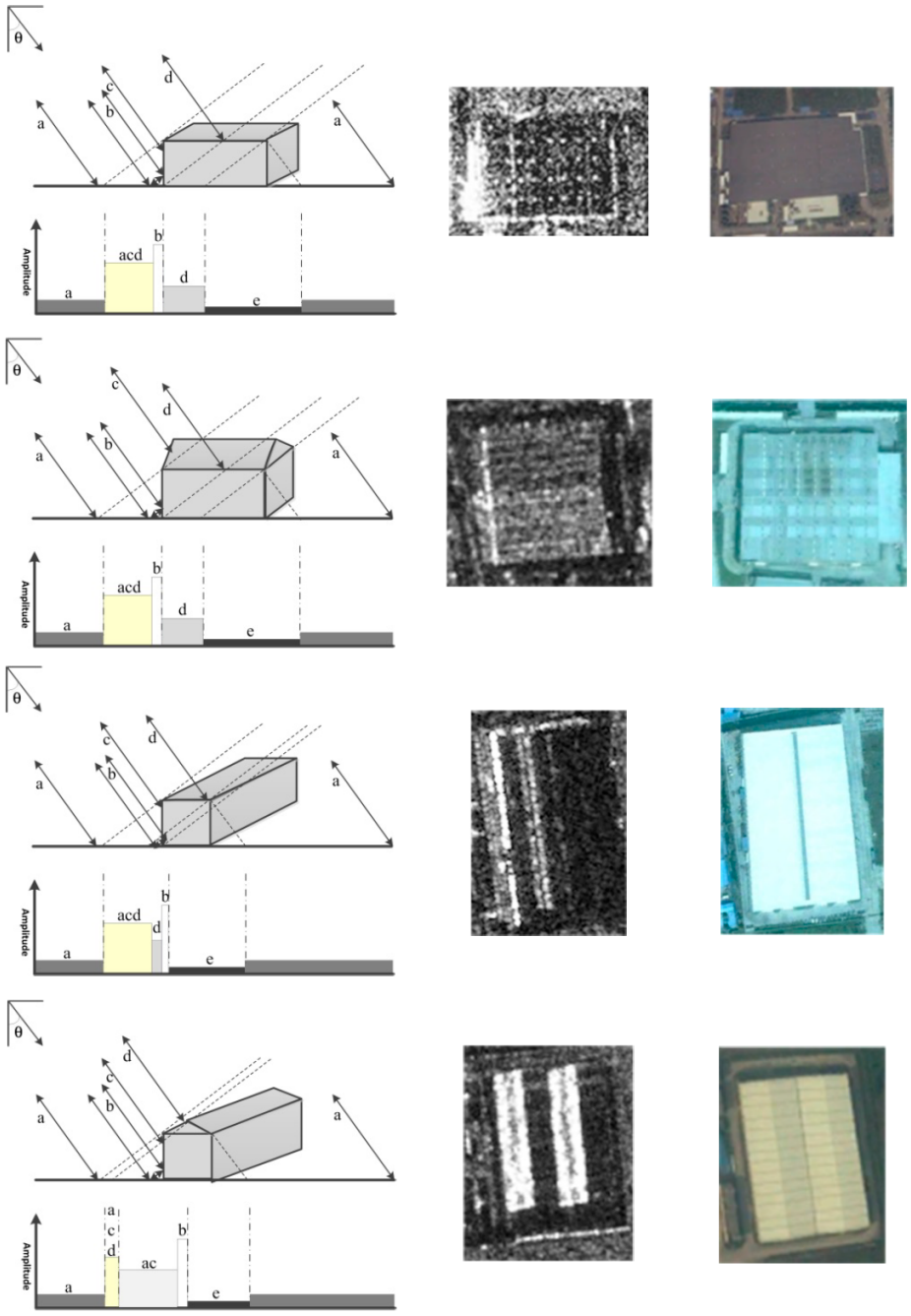


Figure 2.4: Examples of building scattering models and corresponding SAR and optical images. *a*: returns from the ground; *b*: double bounce vertical wall-ground; *c*: single front wall backscattering; *d*: returns from the roof; *e*: shadow area in ground range; *acd*: layover area where ground, front wall, and roof contributions are superimposed. Images from [29]

2.1.2 Temporal, spatial, and spectral resolution

For satellite systems, the **temporal resolution** is dictated by the satellite revisit period. It ranges from tenths of minutes for geostationary satellites to few days or a couple of weeks for polar-orbiting satellites, and it sets the minimum time interval between two image acquisitions over a certain area. The same does not apply to aerial imaging, which does not follow a fixed schedule. Within a single flight, the same scene can be observed several times, but these data collection campaigns happen at a much lower frequency, even in the order of magnitude of years [7]. In any case, a high temporal resolution is desirable when detecting changes in a time series of two or more images.

The **spatial resolution** defines the size of the smallest object that can be discriminated in the image. It is upper bounded by the size of Earth surface portion corresponding to one pixel, whose dimensions usually go from less than a meter to several hundred or thousand meters. It has strong ties with the swath width, which instead indicates the width of the area covered by the image along the axis perpendicular to the platform flying trajectory. The trade-off between them implies that a higher resolution comes with a narrower swath width [6]. The principle is the same as the zoom of a camera: zooming in reduces the field of view, but allows to appreciate finer details [9]. Clearly, the various levels of granularity are more suitable for some kinds of applications than others, depending on the scale of the region of interest and the size of the objects under investigation.

The **spectral resolution** refers to the range of frequencies (or, equivalently, wavelengths) covered by each of the sensors' channels. Figure 2.5 illustrates how the light spectrum can be divided: multispectral (optical) images can be composed of about a dozen channels over the bands from the deep blue to the short-wavelength infrared or thermal infrared, hyperspectral images can have up to a couple of hundreds. Also in this case there is a link with the spatial resolution, because narrower channel bandwidths imply poorer pixel resolutions [9]. For example, a panchromatic channel covering the frequencies of the visible light usually has a resolution 4 to 5 times higher than the corresponding multispectral channels [7]. The false-colour composite in the left panel of Figure 2.6 shows how different bands can highlight some ground covers rather than others. The natural colours for the human perception are shown in the red, green, and blue (RGB) panel on the right.

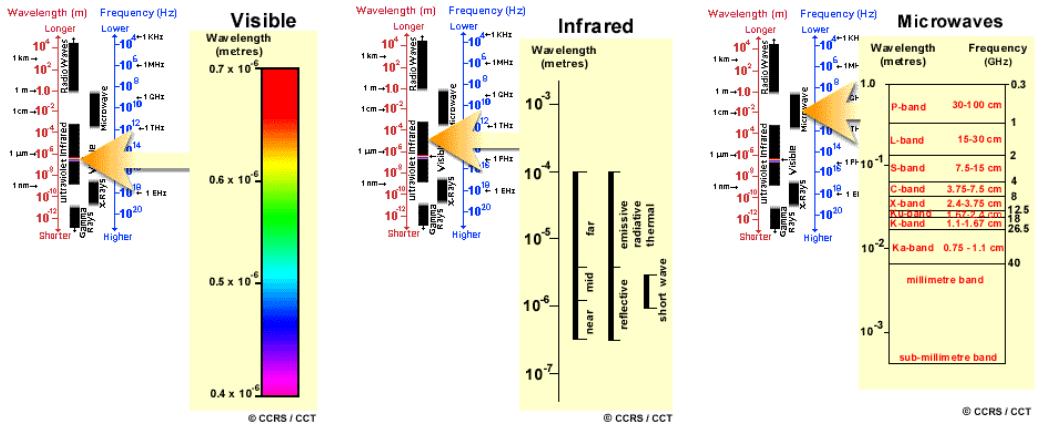


Figure 2.5: The electromagnetic spectrum parted in different ways. The visible and infrared bands are covered by optical and thermal sensors. SAR systems work with microwaves. Images from [28].

Figure 2.6: Different channel combinations highlight different characteristics of the scene. In this example, a Landsat 5 acquisition over Grand Forks, North Dakota, USA. **Left**: infrared channels; **Right**: RGB channels. Credit: NASA Earth Observatory.

SAR sensors commonly cover a single frequency band in the microwave range of the electromagnetic spectrum. Still, the SAR pulses can be sent and received with vertical (V) or horizontal (H) polarisation, depending on the electric field orientation with respect to the direction of propagation of the electromagnetic wave. *Polarimetric* SAR is the most advanced, because it is able to work with more than one mode: dual-pol SAR can record a like-polarised image and a crosspolarised image (e.g., VV and HV); quad-pol SAR can work with any polarisation: VV, HH, HV, and VH [23].

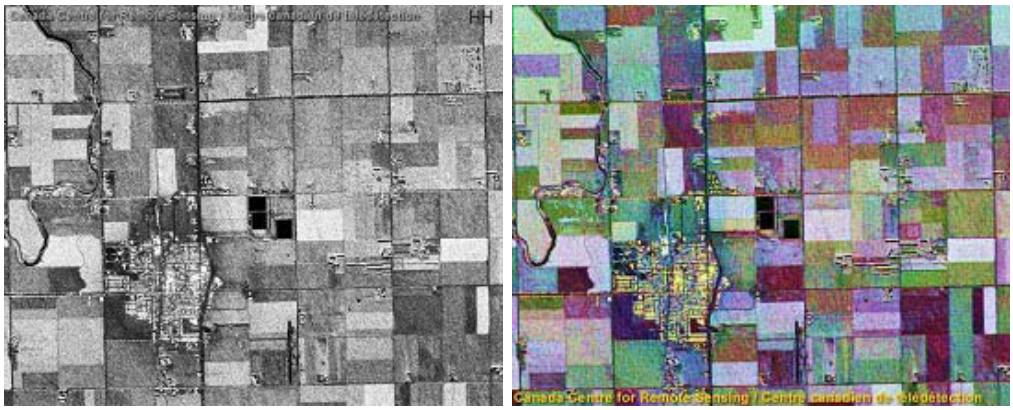


Figure 2.7: SAR images of the same scene recorded in single polarisation **(left)** and quad-polarisation **(right)**. Images from [28].

The added information content of polarimetric SAR can be appreciated in Figure 2.7.

2.2 Change detection

The introduction of the concept of CD for time series of remote sensing images dates back to the 1960s [30]. Even from its early definitions, it has always been referring to the detection or the assessment of both natural and human-caused phenomena affecting the Earth surface [4]. Singh [30] calls CD *the process of identifying differences in the state of an object or phenomenon by observing it at different times.*

2.2.1 What do we consider as a change?

On the contrary, defining which events should be highlighted as changes is still debatable, and the question whether a CD algorithm should also detect differences due to, e.g., weather conditions, seasonal trends or phenological processes is still open and highly application-dependent. Nonetheless, this ambiguity must be solved before proposing a CD framework, in order to evaluate its performance objectively. Arguably, a good definition should be flexible and adaptive, that is, when a change stands out over minor ones, the main event should be of major interest and the others should be ignored. For example, the growth stage of plants is an important aspect when monitoring

agricultural productions, but it should be irrelevant when assessing a forest fire aftermath. On the other hand, one may think of a more complex framework able to detect and distinguish all the diverse changes without discarding any [17, 21, 31, 32, 33, 34, 35].

2.2.2 Change detection methods pipeline

Traditionally, a CD framework usually consists of the three main phases listed below, as described in [6, 23, 24, 36]. Postclassification methods constitute a notable exception [16, 17, 31]. Although these methods may be fit for specific-purpose applications, they are generally considered as inferior due to the accumulation of error from the underlying classifications, approximated as the product of the overall accuracies of the individual classifications [23, 36].

Image preprocessing

The image rectification and restoration aims to correct distorted or degraded image data to create a more faithful representation of the original scene. This typically involves the initial processing of raw image data to correct for geometric distortions, to calibrate the data radiometrically, and to eliminate noise present in the data. Thus, the nature of any particular restoration process is highly dependent upon the characteristics of the instrument itself. These procedures are termed preprocessing operations since they precede further image manipulation and data analysis.

Geometric distortions are both systematic and random: some are well understood and mathematically modelled effects due to for example the previously mentioned panoramic distortion, the Earth's curvature, and the Earth's rotation; others are caused by a wrong positioning and inclination of the sensor (most frequently happening to airborne and drone systems). To geocode and georeference an image means to take care of these problems and make sure that each pixel represents a well-defined position on the Earth. Coregistration is another fundamental preprocessing step: in order to perform meaningful analyses, one must bring all the images to a common spatial grid where a pixel represents the exact same area of the Earth in all of them. Depending on the spatial resolution, this operation might require more than simple geometric transformations such as translations and rotations.

For optical data, also the **radiometry degradation** sources can be distin-

guished between systematic and random. The corrections of these account for Earth-sun distance and sun elevation to normalise the reflectance with respect to the seasonal position of the sun, but also for unpredictable atmospheric distortions. Finally, **noise** removal includes the restoration of missing lines (*destriping*), median filtering, *multilooking* and other techniques to improve the quality of the data before it is actually processed.

Change extraction

Once the images are ready for inspection, the next step is the extraction of change features: after a meaningful comparison of the images, the changes stand out from the background. Traditional CD methods are based on the comparison of homogeneous images, i.e. two or more images acquired by the same kind of sensor. Hence, the most logical and straightforward feature to consider when dealing with optical data affected by additive noise is the image difference, and the image ratio when dealing with SAR data and their multiplicative signal model. Clearly, the idea is to highlight the changes across images while removing the noise at the same time. For the bitemporal case, the result generally reduces to a difference image with a single value per pixel that represents to which degree (or probability) the pixel is likely to belong to changed areas. For a time series of N images, each pixel can be associated to $N-1$ values corresponding to the difference images between consecutive acquisitions.

Before proceeding with the next phase, a very common postprocessing step is filtering. Local, nonlocal, or global information can be used to smooth the difference image and further eliminate outliers caused by input noise or other issues. Without this procedure these pixels could turn into false positives or false negatives at the end of the CD pipeline. Examples range from simple local median filtering [37] to rather complex algorithms such as the Gaussian filtering that exploits fully connected conditional random field models [38].

Change image thresholding

Finally, the last operation required to distinguish changed parts from unchanged parts is thresholding the difference images or alternative test statistics. By splitting their histogram into two, thresholding allows to classify their pixels into changes (foreground) and no changes (background). The optimal thresholds can be set either manually after visual inspection or au-

tomatically by exploiting an algorithm such as [39, 40, 41, 42], or by using them in an ensemble fashion by a majority vote [43].

2.3 Heterogeneous change detection

So far in this thesis, the problem of CD in a time series of remote sensing images has been discussed without assuming any relationship between the images themselves. In the following, a clear distinction between the definitions of homogeneous and heterogeneous data is set, to show the limitations imposed by using the former and the challenges faced when dealing with the latter.

2.3.1 Motivation

When describing the ideal scenario for CD, Campbell *et al.* [23] refer to the case in which the images are captured by the same or well intercalibrated sensors, at the same time of day, using the same field of view and look angle, and so on. Working under these assumptions assures that spurious and irrelevant discrepancies between the acquisition schemes are kept to the minimum and the change extraction is optimised to detect only what truly has changed within the area under investigation. Far from this ideal scenario, the reality is in fact much harder to face: even when the images are acquired by the same sensors, unpredictable bias and distortions might be too strong to be corrected, or the data might even be corrupted or missing due to instrument errors (or cloud coverage in the case of optical data). Also, being limited to the use of one sensor can be unpractical, if not problematic.

Imagine the timeline depicted in Figure 2.8: a particular area is covered by three satellites, each revisiting this same location every 12 days. A forest fire flares up at time t_0, and the most logical thing to do would be to compare the two images from Sensor 3 at time $t_0 - 3$ *days* and Sensor 1 at time $t_0 + 3$ *days*. Instead, detecting this event with a homogeneous CD method requires the use of the image acquired at time $t_0 - 9$ *days*. In the same way, one may think to monitor the development and the velocity of spread of this fire, however they would not be able to do so with images acquired every three days, but only by comparing data collected 12 days apart.

Undoubtedly, the limitations imposed by the assumptions of homogeneity are

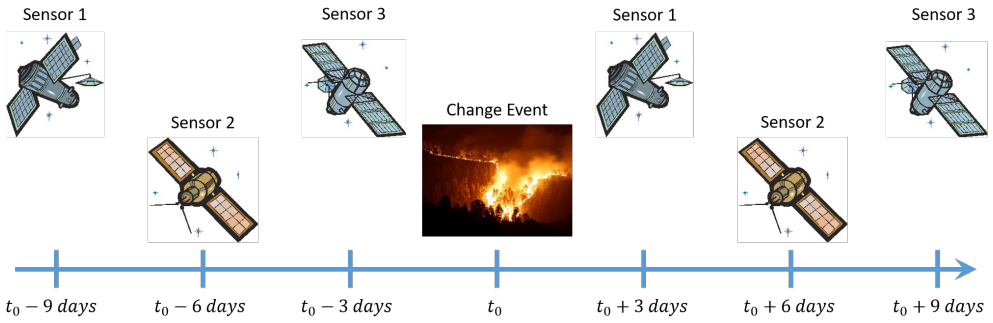


Figure 2.8: Combining heterogeneous data sources allows to increase the time resolution for detecting changes promptly and monitor their development more frequently.

too strict. The variety of available data and the methodological and computational evolution of the last decade have eventually led the remote sensing community to develop CD algorithms that overcome these restrictions and are able to fully exploit all the available sources. These are called *heterogeneous* CD methods, whose input data is also named multisource [21], multisensor [10], cross-sensor [14], multimodal [15] and information unbalanced data [44]. The last two can also be seen as more general, since they cover both the multisensor case and the case when we have data from the same sensor, but with differences that can be attributed to sensor modes, sensor parametres and environmental parameters.

2.3.2 Challenges and solutions

When invalidating the assumptions of homogeneity, conventional homogeneous CD techniques are unsuitable, and additional pre- or postprocessing steps are required [18, 20]. Indeed, heterogeneous data imply different domains, diverse statistical distributions, and inconsistent surface signatures across the images, especially when different sensors are involved that are not measuring the same physical quantities. Coping with these issues is much more complex than simply adding a preprocessing or cocalibration step to the CD pipeline described previously. In other words, a direct comparison is meaningless or even unfeasible without severe manipulations of the data [45]. Nonetheless, an assumption which must necessarily hold true is *class separability*, where the term class can refer to land covers, land uses, or single objects, depending on the specific applications and the spatial resolutions

used. If the representations of two or more classes of data produced by a sensor cannot be distinguished from one another, the resulting ambiguities cannot be coped with. Classes would mistakenly be thought as merging or splitting from one time to the next, and false or missed alarms could arise. Therefore, there must be a one-to-one correspondence across domains for the class signatures involved in the changes. Moreover, the concept of class separability must be extended further. If a change alters a target's physical property, which is not among the ones quantified by a specific sensor, then this change is inevitably invisible to the latter. Clearly, this requires that the correct sensor systems are used in order to detect a specific change process or change event [35, 46].

The taxonomy of heterogeneous CD methodologies is not trivial nor well-defined. The approaches to these problems are multiple and very diverse, and one can find several possible ways to categorise them [47, 48]. A first distinction can be made between supervised and unsupervised methods. Supervision in heterogeneous CD refers to the fact that training data is available, where some pixels are labelled as changed and others as unchanged. The labels can be obtained e.g. as a result of a visual inspection and a manual selection or of a ground campaign. These labels can be used as targets during training of a change detector, or to exclude change pixels from the training set when learning an image regression function. Unsupervised methods do not have access to training data and cannot rely on any such labels.

This thesis uses the term *self-supervised* to mean that labels of changed and unchanged pixels have not been provided by an external source, but have been inferred from the data by the algorithm itself. This kind of automatic selection of training data points has already been referred to as self-supervision in other research fields, such as as robotics [49, 50]. There are also a few examples of using this term in remote sensing [51, 52], although it has not taken root in the heterogeneous CD literature prior to this work. In any case, it should be made clear that a self-supervised method is unsupervised.

Another proposed classification of heterogeneous CD methods is the following: some are using similarity measures [10, 11, 53] or scale-invariant local descriptors [12, 54] with assumed invariant properties across the acquisitions. Data transformation methods instead include those procedures based on the projection of the heterogeneous images into a common domain or feature

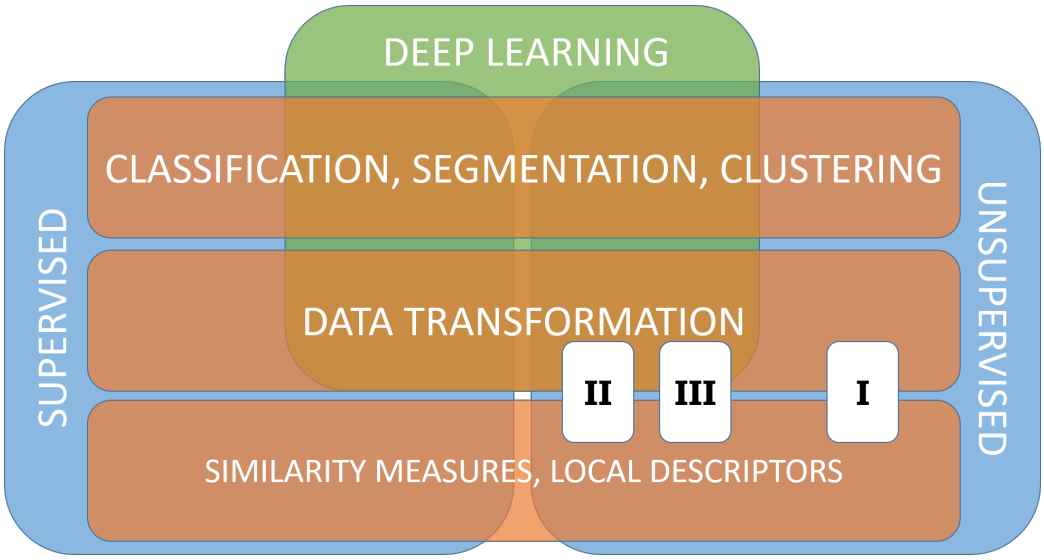


Figure 2.9: Proposed taxonomy for the topic of heterogenous CD. The papers included in Chapters 7 to 9 are placed accordingly.

space, where they share the same statistics and for which classical CD methods can be applied [13, 14, 19, 20, 21, 48, 53, 55]. In the same spirit, superpixel segmentation [15], classification [16, 17, 31], or clustering [18] allow the mapping to a semantic space where it is easier to detect changes. Figure 2.9 depicts a combination of these classifications, and show where the presented papers fit in this overviewing picture.

An alternative subdivision into two groups sees parametric methods being contrasted against nonparametric ones. The former make use of a mixture of multivariate (or meta-Gaussian) distributions to model the dependencies between the two imaging modalities, or the joint statistics, or the different types of multisensor data [13, 56, 57, 58]. Instead, the latter come with the advantage of not explicitly assuming a specific parametric distribution for the data [19, 20, 21, 25, 44, 47, 59, 60, 61]. Among these, the most recently developed for heterogeneous CD are deep learning methodologies, which are also the most popular given the trend of the last few years, not only in remote sensing, but in many other fields of research in general.

2.4 Main focus of the Ph.D. activity

The analysis in this thesis concentrates on the use of heterogeneous satellite data, and more specifically, on the scenario where the changes must be detected from satellite images with high to medium spatial resolution (10 to 30 meters). At these resolutions, a common and reasonable assumption is that the images can be easily coregistered with sufficient precision by applying simple image transformations such as translation, rotation, and resampling [20, 21, 61, 62]. These resolutions allow 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.).

Working with these resolutions, multitemporal CD examples comprise land usage planning of urban and agricultural areas [63, 64], or the monitoring of trends such as deforestation [65], lakes or glaciers reduction [66, 67], urbanisation [68], and desertification [69]. Instead, bitemporal applications mainly consist of the detection and assessment of natural disasters, like earthquakes [53], floods [48], forest fires [14], and oil spills [70]. This work focuses on the latter case, in particular on finding unsupervised solutions to the problem of data transformation and mapping for heterogeneous change detection in bitemporal images.

Chapter 3

Data transformation

In this chapter, firstly the notation used throughout the thesis is introduced. Then, a general idea of regression is presented, followed by a selection of regression methods. From now on, the discussion is restricted to the bitemporal case, but most of the analysis conducted below can be extended to the multitemporal case as well.

3.1 Definitions and notation

Let $\mathcal{X}$ and $\mathcal{Y}$ be the domains where the single-pixel measurements of two different sensors (or sensor modes) lie. These domains could be e.g. $\mathbb{R}_{\geq 0}$ (nonnegative real numbers) for the intensities of a single-channel SAR sensor, $\mathbb{R}_{\geq 0}^{C}$ for a multispectral radiometer with C bands, or $\mathbb{C}_{\succeq 0}^{C\times C}$ for a polarimetric SAR system with C polarisations that records a complex and semipositive definite covariance matrix for each pixel. In this thesis, $\mathcal{X}$ and $\mathcal{Y}$ are assumed to be $\mathbb{R}_{\geq 0}^{|\mathcal{X}|}$ and $\mathbb{R}_{\geq 0}^{|\mathcal{Y}|}$ respectively, whose dimensions $|\mathcal{X}|$ and $|\mathcal{Y}|$ are in general not the same.

Further on, $\mathcal{I}_{\mathcal{X}} \in \mathcal{X}^{H\times W}$ denotes a $H \times W$ image acquired at time t_1 by the first sensor. Similarly, $\mathcal{I}_{\mathcal{Y}} \in \mathcal{Y}^{H\times W}$ is the corresponding $H \times W$ image collected over the same area at time $t_2 > t_1$ by the other sensor. Their common dimensions H and W have been obtained through resampling and coregistration, however they will have different numbers of channels, $|\mathcal{X}|$ and $|\mathcal{Y}|$ respectively. Assume that a limited part of the area covered by the images

has changed between time t_1 and t_2.

These two images can be thought of as realisations of stochastic processes that generate data tensors from domain $\mathcal{X}$ and $\mathcal{Y}$. Therefore, $\boldsymbol{X} \in \mathcal{X}^{h \times w}$ and $\boldsymbol{Y} \in \mathcal{Y}^{h \times w}$ indicate subtensors holding colocated patches of size $h \times w$ extracted from the full images $\mathcal{I}_\mathcal{X}$ and $\mathcal{I}_\mathcal{Y}$. Their pixels are represented by the vectors $\boldsymbol{x}_{i,j} \in \mathcal{X}$ and $\boldsymbol{y}_{i,j} \in \mathcal{Y}$, with $i \in \{1, \dots, h\}$ and $j \in \{1, \dots, w\}$. Alternatively, $\boldsymbol{X} \in \mathcal{X}^n$ and $\boldsymbol{Y} \in \mathcal{X}^n$ refer to subsets of n (not necessarily adjacent) pixels selected from the images. In this case, the vectors $\boldsymbol{x}_i \in \mathcal{X}$ and $\boldsymbol{y}_i \in \mathcal{Y}$ with $i \in \{1, \dots, n\}$ are single elements of $\boldsymbol{X}$ and $\boldsymbol{Y}$ respectively.

3.2 Regression

What makes heterogeneous CD difficult to tackle, is that data collected from different sensors lie in distinct, diverse and unrelated domains. To a certain extent, this resembles the topic of domain adaptation [15], however the problem faced here is in fact more complex. These domains do not share any common characteristics, they represent realities which are not corresponding entirely because of the changes, and the relationships between their stochastic sources are nontrivial to formalise.

Among the possible solutions listed in Chapter 2 there is data transformation. In particular, one can define two convenient regression functions f and g that are able to translate data from one domain to another and vice versa, where it is possible to compare entities which would be incompatible otherwise. Hence,

$$\hat{\boldsymbol{Y}} = f(\boldsymbol{X}) \quad \text{and} \quad \hat{\boldsymbol{X}} = g(\boldsymbol{Y}) \tag{3.1}$$

represent the mappings of $\boldsymbol{X}$ into $\mathcal{Y}^{H \times W}$ and of $\boldsymbol{Y}$ into $\mathcal{X}^{H \times W}$. As a special case where $h = w = 1$, the patches can reduce to single pixels, with mappings $\hat{\boldsymbol{y}} = f(\boldsymbol{x})$ and $\hat{\boldsymbol{x}} = g(\boldsymbol{y})$. Traditional regression functions correspond to single-pixel mappings, whereas convolutional neural networks work on patches and incorporate contextual information.

If a suitable training set is available, these regression functions can be learned directly from examples that provide a clear one-to-one correspondence between land surfaces across the two domains. In an ideal situation, all the ground covers are encompassed by the training set. The training set should not include pixel pairs from the changed areas, which would promote a wrong

data transformation. Once the training is over, the images can be translated into the other domain, where they are compared against their counterpart to highlight the changes. This approach is also referred to as *image regression* in the CD literature, a term which has on some occasions been used when translating between two more or less heterogeneous image domains [30, 71, 72].

It must be stressed that when $|\mathcal{X}| \gg |\mathcal{Y}|$, $f(\boldsymbol{X})$ is a many-to-few mapping and a compression function, which is usually not problematic. However, the other side of the coin is that the inverse few-to-many mapping of $g(\boldsymbol{Y})$ can be easily ill-posed, even though the contextual information of the patch may alleviate the problem to some degree. Obviously, the vice versa applies to the case in which $|\mathcal{Y}| \gg |\mathcal{X}|$.

Linear regression, basically the most simple approach one can consider, is clearly too far from being satisfactory [14]:

$$\hat{\boldsymbol{y}} = \boldsymbol{W_f}\, \boldsymbol{x} + \boldsymbol{b_f} \tag{3.2}$$

where each feature of the transformed pixel $\hat{\boldsymbol{y}}$ is a linear combination of those of $\boldsymbol{x}$ weighted by each row of $\boldsymbol{W_f} \in \mathbb{R}^{|\mathcal{Y}|\times|\mathcal{X}|}$ plus a bias $\boldsymbol{b_f} \in \mathbb{R}^{|\mathcal{Y}|}$. The same equation can be written for the function g, $\boldsymbol{W_g} \in \mathbb{R}^{|\mathcal{X}|\times|\mathcal{Y}|}$ and $\boldsymbol{b_g} \in \mathbb{R}^{|\mathcal{X}|}$. Despite the advantage of being fast to train, linear regression lacks flexibility due to a limited number of parameters. It describes the relationship between explanatory and response variables ($\boldsymbol{x}$ and $\boldsymbol{y}$ respectively) by drawing hyperplanes, which are often too rigid to fit the data. Therefore, more complex techniques must be exploited.

Nonlinear regression is more appropriate in our case, because it is inclined to more correctly match the shapes of the functions it approximates. A natural extension of linear regression to the nonlinear case is polynomial regression, which includes polynomial terms of higher order than just the first. Still, as with other parametric models, it is more convenient when the shapes of the functional relationships between the independent and dependent variables are predetermined, so the right order r of the polynomial can be chosen. If these relationships are totally unknown, one may think of increasing r to increase the flexibility. However, the number of parameters grows very quickly as a function of r [73], and these higher-order polynomials show undesired nonlocal effects [74].

3.3 Nonlinear nonparametric regression

Nonparametric regression is in this sense preferable, especially because it can also be adjusted more easily to capture unusual or unexpected features of the data. In the following, a selection of nonlinear nonparametric regression methods are presented. For brevity, only the derivations for $f(\boldsymbol{x})$ are reported, whilst the ones for $g(\boldsymbol{y})$ are omitted because they are analogue.

3.3.1 Gaussian process regression

Let $\{\boldsymbol{x}_i\}_{i=1}^{n}$ be a collection of random variables. If any finite subset of these has a joint multivariate Gaussian distribution, then $\boldsymbol{x}_i$ can be seen as a realisation of the Gaussian process (GP) specified completely by its mean function $\boldsymbol{m}(\boldsymbol{x})$ and covariance (kernel) function $k_{\boldsymbol{x}_i,\boldsymbol{x}_j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. For regression purposes, a zero mean GP is most often used [75].

Consider the training set of n input vectors $\boldsymbol{X} \in \mathcal{X}^n$ and the corresponding target vectors $\boldsymbol{Y} \in \mathcal{Y}^n$, a set of n_{test} new observed vectors $\boldsymbol{X}_* \in \mathcal{X}^{n_{\text{test}}}$ and the sought vectors $\hat{\boldsymbol{Y}} \in \mathcal{Y}^{n_{\text{test}}}$. The joint distribution of $\boldsymbol{Y}$ and $\hat{\boldsymbol{Y}}$ conditioned on $\boldsymbol{X}$ and $\boldsymbol{X}_*$ is

$$\left[\boldsymbol{Y}, \hat{\boldsymbol{Y}}\right] | \boldsymbol{X}, \boldsymbol{X}_* \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}_*} \\ \boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}_*} \end{bmatrix}\right), \tag{3.3}$$

where the element (i, j) of the matrix $\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}_*}$ is the covariance between the ith vector in $\boldsymbol{X}$ and the jth vector in $\boldsymbol{X}_*$. The same applies to $\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}}$, $\boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}_*}$, and $\boldsymbol{K}_{\boldsymbol{X}_*\boldsymbol{X}} = \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}_*}^T$. Starting from Equation (3.3), the following posterior distribution is derived [75]:

$$\begin{aligned} \hat{\boldsymbol{Y}} \,|\, \boldsymbol{X}_*, \boldsymbol{X}, \boldsymbol{Y} \sim \mathcal{N}\big(&\boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}} \cdot \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}}^{-1} \cdot \boldsymbol{Y}, \\ &\boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}_*} - \boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}} \cdot \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}}^{-1} \cdot \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}_*}\big) \end{aligned} \tag{3.4}$$

Hence, the corresponding conditional mean is the maximum prediction

$$\hat{\boldsymbol{Y}} = \boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}} \cdot \boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}}^{-1} \cdot \boldsymbol{Y}. \tag{3.5}$$

The quality of the regression is affected by two key factors: which kernel function is applied and how its hyperparameters are tuned. The radial basis

function (RBF) is a very common choice [75]:

$$k_{\boldsymbol{x}_i,\boldsymbol{x}_j} = \sigma_f^2 \exp\left(-\frac{1}{2}\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^T L \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)\right), \tag{3.6}$$

where the set of hyperparameters $\boldsymbol{\theta} = \left\{L, \sigma_f^2\right\}$ contains the signal variance σ_f^2 and $L = \ell^{-2} I$, if the length-scale parameter ℓ is a scalar (isotropic kernel), or $L = \operatorname{diag}\left(\boldsymbol{\ell}^{-2}\right)$, if $\boldsymbol{\ell}$ is a vector (anisotropic kernel) [75]. The optimisation of $\boldsymbol{\theta}$ is carried out by a gradient ascent maximisation of the marginal likelihood $P\left(\boldsymbol{Y} \mid \boldsymbol{X}, \boldsymbol{\theta}\right)$. The optimisation might lead to a local maximum instead of the global one, so iterating the procedure several times starting from random positions in the hyperparameter space $\Omega_{\boldsymbol{\theta}}$ is recommendable. The Achilles' heel of GPs is the evaluation of large matrices multiplications and inversions, which can become prohibitive as n increases. It might require long computational times and memory issues may also arise.

3.3.2 Random forest regression

Random forest (RF) regression is a tree-based regression method that has become very popular in recent years due to its strong performance, ease of implementation and low computational cost. It is an ensemble learning technique developed by Leo Breiman [76], which is based on the construction of a multitude of regression trees. Each tree is trained by using a bootstrap sample extracted from the whole training set $\boldsymbol{X}$. This sample is successively split in two by a combination of threshold tests, where each compares a subset of r randomly selected features of $\boldsymbol{x}$ to a set of random thresholds (e.g., $\text{feat}_1 > \text{thr}_1$ & $\ldots$ & $\text{feat}_r > \text{thr}_r$). Each split produces two branches with corresponding child nodes, where a new test can be defined. The process of dividing the input training data over branches is iterated until the terminal nodes of the tree, referred to as leaf nodes, contain one or more data points from $\boldsymbol{X}$. These have their corresponding output training data points from $\boldsymbol{Y}$, which are combined (for example averaged) to yield the final value associated to each leaf. Once the tree is fully formed, a validation data point can traverse it following a particular path, reaching one of the leafs that gives as the output its associated value. The latter is in fact the output $\boldsymbol{y}_t$ of that tree for that specific data point. Bootstrap samples allow to generalise better and to use the rest of the training set as validation set to perform *out-of-bag* estimation [76]: if the output of the tree for this set leads to a sufficient R^2

score, then the tree is validated, or discarded otherwise. The training stops when the forest reaches the size (number of regression trees) T specified a priori by the user. Finally, for each element $\boldsymbol{x}_*$ of the test set $\boldsymbol{X}_*$, the forest of regression trees produces an ensemble of regression values, from which the final regression value $\hat{\boldsymbol{y}}$ can be determined, e.g. by averaging:

$$\hat{\boldsymbol{y}} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{y}_t. \tag{3.7}$$

The randomness introduced both by feature selection and threshold determination has been shown to result in attractive properties such as a controlled variance, resistance to overtraining, and robustness to outliers as well as irrelevant variables. Moreover, RF regression inherently provides estimates of generalisation error and measures of variable importance [77, 78]. The structure of the forest and hence the regression behaviour can be controlled by several hyperparameters, but the main ones are:

r : the number of features considered in each node.

T : the number of trees in the forest.

N_s : elements in a node required to perform a split

N_l : elements required to create a new node

L : the maximum depth up to which a tree can grow

In [76], empirical results suggest to set the number of features considered at every node as $r = \lfloor \log_2 |\mathcal{X}| \rfloor$ or $r = \lfloor |\mathcal{X}|/3 \rfloor$, $|\mathcal{X}|$ being the dimensionality of the vectors $\boldsymbol{x}$. The number of trees T is not as critical as the rest of the hyperparameters. However, increasing it has two main effects: first, the computational load increases, and second, an initial increase in the accuracy of the regression is observed, before reaching a saturation point [45], after which improvements are limited by a strong correlation between the trees [76]. Therefore, a compromise between gained accuracy and computational load must be found. Allowing the branches to grow in depth without a limit leads to a large number of leaves carrying one single data point. This can cause overfitting, that is, the model learns to reproduce very good regressions when it is fed with data similar to its training sample, but it fails to achieve the same accuracy with new data. Pruning, i.e. limiting the node splits, was not part of the first formalisations of the RF in [76], but it is supposed to reduce

overfitting by tuning the remaining hyperparameters, namely L, N_s and N_l. These drive different pruning criteria but they lead to similar effects on the structure of the trees. Setting L allows the branches to grow up to L levels while pruning the rest of the nodes. Instead, N_s defines the minimum amount of data points a parent node must carry in order to perform a split. N_l defines the minimum number of samples that both child nodes must receive from the parent. Consequently, the latter is more restrictive, since it may prevent a split allowed by the former, so it is reasonable to set $N_l \ll N_s$.

3.3.3 Support vector regression

Support vector machines (SVMs) represent a very powerful paradigm useful for both classification and regression. In classification, they seek the best curve separating the classes by minimising a cost function that accounts for misclassification. In regression, the curve is brought as close as possible to the approximated function by minimising the reconstruction error. In their latest formalisations, the SVM loss function includes a sensitivity term defining the width of a soft margin around such a curve, which allows to reduce the effects of noisy data and outliers. By solving the so-called dual problem that involves the method of Lagrange multipliers [73, 79], this sought curve is found and the training points defining the margin are highlighted from the rest of the training set. These are called the support vectors, which the method is named after.

Tuia *et al.* [80] proposed a multiple-input-multiple-output SVM regression method to cope with a multiple-output problem (i.e. the regression of a multivariate variable) all at once, instead of training a dedicated SVM for each dependent variable. Thus, it overcomes the limitations of the standard SVM regression implementations, designed to predict a single output feature and ignoring the potentially nonlinear relations across the target features [80].

The sought regression function is in the form

$$\hat{\boldsymbol{y}} = \boldsymbol{W}\phi(\boldsymbol{x}) - \boldsymbol{b}. \tag{3.8}$$

Here, $\boldsymbol{W} = \left[\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{|\mathcal{Y}|}\right]^T$ with column vectors $\boldsymbol{w}_q \in \mathbb{R}^{|\mathcal{X}|'}$ is the weight matrix and $\boldsymbol{b} = \left[b_1, \ldots, b_{|\mathcal{Y}|}\right]^T$ are the biases in the linear combination of the data points $\boldsymbol{x}_i$ transferred into a finite-dimensional space by the kernel function $\phi : \mathbb{R}^{|\mathcal{X}|} \to \mathbb{R}^{|\mathcal{X}|'}$. The extension to a (possibly infinitely-dimensional)

separable Hilbert space is straightforward. The loss function minimised during the training phase is

$$L_{\text{SVM}}\left(\boldsymbol{W},\boldsymbol{b}\right)=\frac{1}{2}\sum_{q=1}^{|\mathcal{Y}|}\parallel\boldsymbol{w}_q\parallel^2+\lambda\sum_{i=1}^{n}L\left(\mu_i\right) \tag{3.9}$$

where

$$L\left(\mu_i\right)=\begin{cases}0 & \mu_i<\epsilon\\ \mu_i^2-2\mu_i\epsilon+\epsilon^2 & \mu_i\geq\epsilon\end{cases}, \tag{3.10}$$

$$\mu_i=\parallel\boldsymbol{e}_i\parallel=\sqrt{\boldsymbol{e}_i^T\boldsymbol{e}_i}\,, \tag{3.11}$$

$$\boldsymbol{e}_i=\boldsymbol{y}_i-\boldsymbol{W}\phi(\boldsymbol{x}_i)-\boldsymbol{b}. \tag{3.12}$$

The parameter ϵ is half the width of the insensitivity zone. This zone delimits a "tube" around the approximated function and the training data points within this insensitivity zone do not contribute to the cost function (see Equation (3.10)). For too small values of ϵ, too many data points will be considered as support vectors (overfitting), the generalisation performance will be affected and the variance of the fitted curve will be too large. On the contrary, a too large ϵ will cause underfitting and the overall accuracy will be low. The penalty factor λ in Equation (3.9) sets the trade-off between the regularisation term that keeps $\boldsymbol{W}$ sparse and the sum of the error terms $L\left(\mu_i\right)$. If λ is too large, nonseparable points would highly penalise the cost function and too many data points will turn into support vectors, favoring overfitting. Vice versa, a small λ may lead to underfitting. Finally, the kernel function ϕ might include other critical hyperparameters $\boldsymbol{\sigma}_\phi$. To select the right combination of hyperparameters $\boldsymbol{\theta}=\{\lambda,\epsilon,\boldsymbol{\sigma}_\phi\}$, a grid search for the smallest crossvalidation error or the minimization of an error bound can be applied. Once the optimal parameters $\{\boldsymbol{W}_{opt},\boldsymbol{b}_{opt}\}$ are found, they are plugged into Equation (3.8) and the sought regression is achieved.

3.3.4 Feed-forward neural networks

Artificial neural networks, or simply neural networks (NNs), were first thought as a paradigm able to emulate the behaviour of the human brain [81, 82]. Their atomic unit, the *perceptron* [83], is modelled after the human neurons. In Figure 3.1, the stimuli $\boldsymbol{x}=\left[x_1,x_2,\ldots,x_P\right]^T$ from the P input features

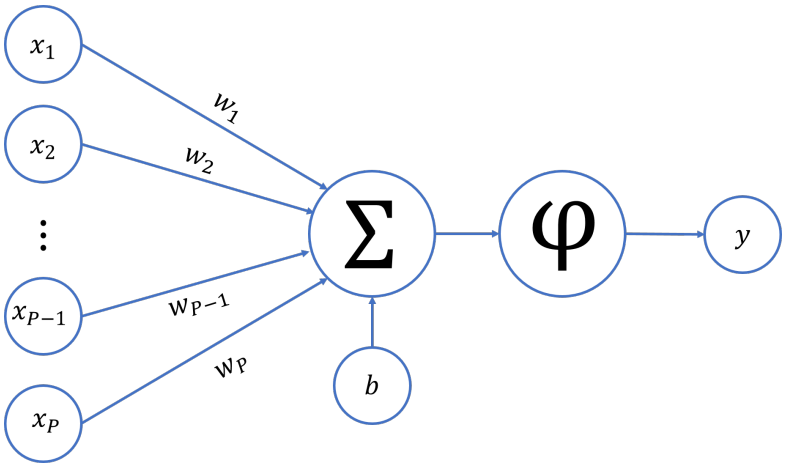


Figure 3.1: The perceptron: a weighted sum of input features (plus a bias) stimulates more or less intensively a nonlinear function φ, producing a new stimulus in output.

are modulated by the synaptic weights $\boldsymbol{w} = [w_1, w_2, \ldots, w_P]$ associated with their connection to the neuron. The perceptron unit aggregates them (plus a bias b) in a weighted sum which activates more or less intensively a nonlinear function φ that fires a new signal y as its output. In the following, the analysis does not include the case of recurrent NNs and feedback connections, considered out of scope, but is instead limited to the case of *feed-forward* NNs, for which the information flows only in one direction.

If the stimuli are connected to more than one neuron, each with its own weights and bias, then a layer ℓ of perceptrons is obtained. Starting from $P^{(\ell)}$ features in input, the layer generates as many features in output as the number $Q^{(\ell)}$ of units belonging to ℓ. This model is entirely captured by Equation (3.13), where $\boldsymbol{\vartheta}^{(\ell)} \triangleq \left\{\boldsymbol{W}^{(\ell)}, \boldsymbol{b}^{(\ell)}\right\} \in \mathbb{R}^{Q^{(\ell)} \times (P^{(\ell)}+1)}$ is the set of parameters that controls the layer's behaviour:

$$\boldsymbol{y}^{(\ell)} = f^{(\ell)}\left(\boldsymbol{x}^{(\ell)}, \boldsymbol{\vartheta}^{(\ell)}\right) = \varphi\left(\boldsymbol{W}^{(\ell)} \cdot \boldsymbol{x}^{(\ell)} + \boldsymbol{b}^{(\ell)}\right). \tag{3.13}$$

The true potential of the perceptron is fully exploited when several layers are connected one after another, forming a *multilayer perceptron*, often referred to as a *fully connected* NN. In such a scheme, the output of one layer is the input of the next. Therefore, $\boldsymbol{x}^{(\ell)} \equiv \boldsymbol{y}^{(\ell-1)}$, and the resulting function for L layers is the sought regression function

$$\hat{\boldsymbol{y}} = f\left(\boldsymbol{x}, \boldsymbol{\vartheta}\right) = \left(f^{(L)} \circ f^{(L-1)} \circ \cdots \circ f^{(2)} \circ f^{(1)}\right)(\boldsymbol{x}) \tag{3.14}$$

where $\circ$ denotes the function composition, e.g. $\left(f^{(2)} \circ f^{(1)}\right)(\boldsymbol{x}) = f^{(2)}\left(f^{(1)}(\boldsymbol{x})\right)$, and $\boldsymbol{\vartheta} = \left\{\boldsymbol{\vartheta}^{(\ell)}\right\}_{\ell=1}^{L}$ contains the parameters of all the layers, i.e. of the multilayer perceptron. Notice that $P^{(1)} = |\mathcal{X}|$, $P^{(\ell)} \equiv Q^{(\ell-1)}$, and $Q^{(L)} = |\mathcal{Y}|$.

A single layer injects a certain amount of nonlinearity thanks to $\varphi^{(\ell)}$, and all the layers concatenated contribute to the expressive power of the NN. If $\varphi^{(\ell)}$ is set as the identity function, i.e. if a layer ℓ has no activation function, this is equivalent to a linear combination. Thus, it is straightforward to show that a multilayer perceptron without activation functions is basically a concatenation of linear transformations, whose result is itself linear. Hence, some may think that the ability of NNs to approximate a highly nonlinear function depends on the choice of $\varphi^{(\ell)}$, and also on the number of layers L. In fact, the *universal approximation theorem* contradicts them:

Theorem 1 (Universal Approximation Theorem [82, 84, 85, 86]) *A feed-forward NN with a linear output layer and at least one hidden layer with any nonlinear activation function can approximate any Borel measurable*[1] *function from one finite-dimensional space to another with any desired nonzero amount of error, provided that the network is given enough hidden units. The derivatives of the feed-forward NN can also approximate the derivatives of the function arbitrarily well.*

Moreover, Hornik *et al.* [84] showed that it is the multilayer feed-forward architecture itself rather than the specific choice of the activation functions which gives NNs the potential of being universal approximators. However, it must be stressed that using only one layer is highly inefficient, especially because the number of neurons required to approximate a function grows exponentially as the complexity of the function increases, whereas adding more layers to match the same level of complexity is less critical [87].

During the inference phase NNs are very fast, the information in input flows through the nodes and the output is almost instantly computed. On the other hand, training a network requires an enormous computational load, and the optimisation of $\boldsymbol{\vartheta}$ is not trivial at all. In fact, before the introduction of the *backpropagation* algorithm [88], the first NNs were not learning much [89]. This algorithm exploits the chain rule of differentiation [73], allowing the inexpensive computation of the gradients of the loss function with

[1] Any continuous function on a closed and bounded subset of $\mathbb{R}^C$ is Borel measurable.

respect to the parameters of the network, which are updated accordingly to minimise the errors in output. As said, the optimisation of NNs are based on the stochastic gradient descent [90, 91] which, in addition, saw many improvements introduced throughout the years [92], for example decaying strategies [93], momentum [94], and adaptive learning rate [95, 96, 97, 98].

These methodological advances, combined with the technological achievements in terms of computational power, are the main historical reasons why the popularity of NNs exploded only by the end of the last century [89]. In particular, this created the fertile ground on which deep learning could burgeon, as discussed in detail in the next chapter.

Chapter 4

Deep learning

In this chapter, data transformation is discussed from the perspective of deep learning (DL), presenting the main paradigms that inspired the design of the methodologies developed throughout this thesis.

The distinction between *deep* and *shallow* NNs is still vague. Schmidhuber in his compact yet extensive, meticulous and painstaking historical survey on DL could not set a sharp boundary between them [89]:

> *"At which problem depth does shallow learning end, and deep learning begin? Discussions with deep learning experts have not yet yielded a conclusive response to this question."*

The book *Deep Learning* by Goodfellow, Bengio, and Courville [87], which can be considered a manual of the topic, summarises all the capabilities of DL without providing a brief definition of it. On one hand, one can say that DL generally refers to architectures featuring a large number of hidden layers [87, 89]. On the other hand, bearing in mind the universal approximation theorem (Theorem 1, Section 3.3.4), it is reasonable to suggest that any network having more than two layers can be considered a DL model. This definition is surely debatable: according to it, almost every NN would be classified as deep. For sure, DL as a keyword has become very popular lately, and it is a thriving topic also for remote sensing applications. Figure 4.1 shows the exponential increase of publications related to remote sensing using the keyword *Deep Learning*, which is in line with other fields of research.

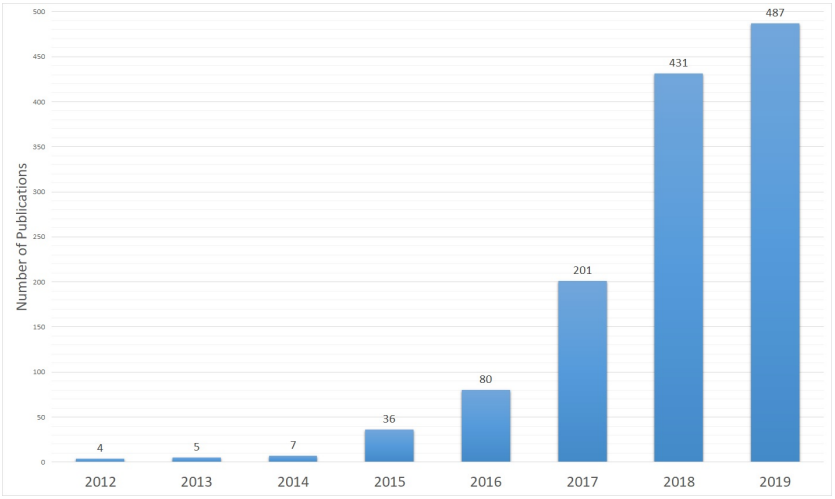


Figure 4.1: Number of publications related to remote sensing and associated with the keyword *deep learning*, over the past 8 years. Source: Clarivate Analytics, January 2020.

4.1 Convolutional neural network

One of the drawbacks of fully-connected NNs is that their size, i.e. the number of parameters $|\boldsymbol{\vartheta}|$, grows very quickly as the number of layers and/or the number of nodes per layer increase. Take the simplistic example in which Q, the number of nodes in a layer, is the same for all the L layers from the input to the output: in that case $|\boldsymbol{\vartheta}| \sim (Q+1)^{2L}$, showing an evident deficiency of scalability. Apart from the obvious memory issues that may arise, learning all these parameters requires also a large training set [99].

The implications are multiple and possibly more severe when dealing with entire images in input. In fact, they are typically large in terms of number of variables (each channel of each pixel is a variable). Therefore, even a small patch with, e.g., 32×32 greyscale pixels fed to a fully-connected layer with a hundred nodes implies already hundreds of thousands of weights. Clearly, the amount of parameters becomes quickly unfeasible even for few layers. In addition, multilayer perceptrons are not able to capture the structures and the topological properties of the images, and they lack robustness with respect to rotations, translations, scaling factors, and local distortions [99]. This means that two very similar images given in input to the same fully-connected NN might yield two very different results.

Convolutional neural networks (CNNs) overcome the issues related to spatial information extraction, network size and translations, because they are specialised in processing data with a grid-like topology and they leverage sparse interaction, parameter sharing, and equivariant representations [87]. They do not have intrinsic equivariance to rotations and scaling, but they can certainly achieve it [87]. They became more popular at the beginning of the new millennium, especially after the achievements of LeCun *et al.* in 1998 with their network called LeNet-5 [99]. However, they raised tremendous interest only after Krizhevsky *et al.* proposed the AlexNet in 2012 [100], whose fame is mostly due to the efficient implementation of the parallelised optimisation over graphic processing units (GPUs). Also, the combined deployment of the rectified linear unit (ReLU) [101], dropout [102], as well as the use of extensive data augmentation allowed to attain large performance gains with respect to previous state-of-the-art methods. These developments opened the doors to very famous DL architectures such as VGG [103] and U-Net [104]. In this section, most of the attention is given to the use of CNNs on image data, but they have proven to be a powerful tool also when dealing with time series and $4D$ input such as videos and volumetric data [87].

A CNN deploys at least one layer in the following form:

$$\boldsymbol{y}^{(\ell)} = f^{(\ell)}\left(\boldsymbol{X}^{(\ell)}, \boldsymbol{\vartheta}^{(\ell)}\right) = \varphi\left(\boldsymbol{K}^{(\ell)} * \boldsymbol{X}^{(\ell)} + \boldsymbol{b}^{(\ell)}\right). \tag{4.1}$$

where $\boldsymbol{K}^{(\ell)} \in \mathbb{R}^{h_f^{(\ell)} \times w_f^{(\ell)} \times P^{(\ell)} \times Q^{(\ell)}}$ is the convolution kernel of layer ℓ. The kernel sizes $h_f^{(\ell)}$, $w_f^{(\ell)}$ and the number of filters $Q^{(\ell)}$ are specified for each layer ℓ during the design of the network. Focusing on the interaction between the convolution kernel and the input, denoted as $\boldsymbol{S}^{(\ell)} = \boldsymbol{K}^{(\ell)} * \boldsymbol{X}^{(\ell)}$, each location i, j on the grid of $\boldsymbol{S}$ takes $Q^{(\ell)}$ values, one for each filter q in the layer:

$$S_{i,j,q}^{(\ell)} = \sum_m \sum_n \sum_p K_{m,n,p,q}^{(\ell)} \cdot X_{i+m,j+n,p}^{(\ell)} \tag{4.2}$$

where

$$\begin{aligned} m &\in \left\{-\lfloor h_f^{(\ell)}/2 \rfloor, \ldots, \lfloor h_f^{(\ell)}/2 \rfloor\right\}, \\ n &\in \left\{-\lfloor w_f^{(\ell)}/2 \rfloor, \ldots, \lfloor w_f^{(\ell)}/2 \rfloor\right\}, \\ p &\in \left\{1, \ldots, P^{(\ell)}\right\}. \end{aligned} \tag{4.3}$$

Figure 4.2 helps to visualise the convolution mechanism, and it also allows to introduce the concept of padding: in order to keep the dimensions of

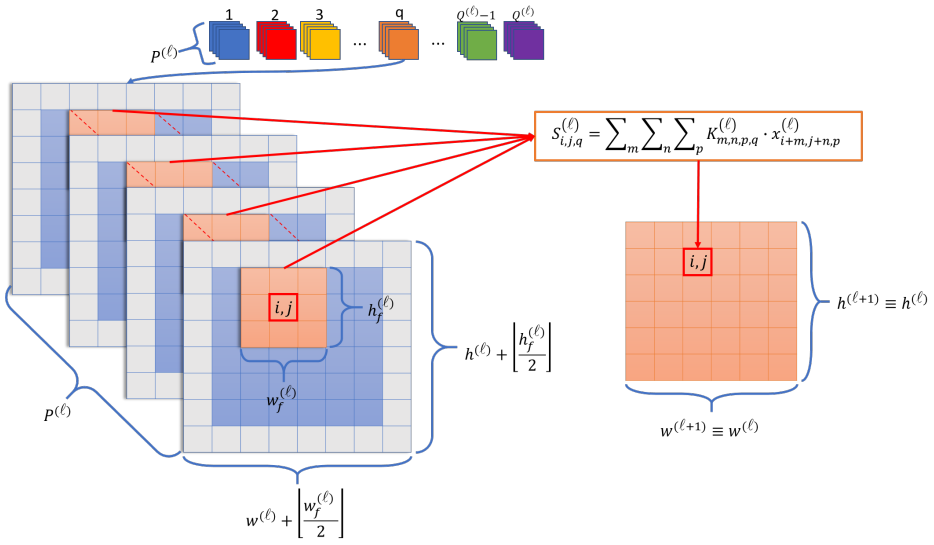


Figure 4.2: Illustration of a convolutional layer: the interaction between the P channels of the input $\boldsymbol{X}^{(\ell)}$ (in blue) padded with zeros (in grey) and the q^{th} filter (in orange) of the convolution kernel $\boldsymbol{K}^{(\ell)}$ results in $S^{(\ell)}_{i,j,q}$ (orange on the right), for $i \in \{1, \dots, h^{(\ell)}\}$ and $j \in \{1, \dots, w^{(\ell)}\}$.

the input throughout the network, one can add a frame of fictitious pixels (usually zeros), so that the convolution is feasible also for those pixels around the borders.

The first important thing to be noticed in Equation (4.2) is that i and j are not related to m and n, which means that the input height and width do not influence the number of parameters in $\boldsymbol{K}^{(\ell)}$. Anther two fundamental accomplishments are sparse interaction and parameter sharing. By having a kernel which is much smaller in size than the input, meaningful spatial features can be extracted from a small neighbourhood rather than considering the whole image. Moreover, reusing the same parameters for all the input locations reduces considerably both memory and training requirements. Instead, in fully-connected layers every output unit interacts with every input unit, implying that all the parameters describing each of these interconnections must be stored in massive matrices which, most of the times, would be sparse [87]. Lastly, demonstrating the robustness of CNNs with respect to translation is straightforward, given that their filters are agnostic to the positions in the image.

To a large extent, deep architectures and CNNs resort to supervised data for their training. Still, there are alternative solutions exploiting strategies and expedients to overcome the need of labelled sets, as presented below.

4.2 Autoencoders

The autoencoder (AE) is a powerful deep learning architecture which has proven capable of solving problems like feature extraction, dimensionality reduction, and clustering [105]. This construct is composed of an encoder-decoder pair (U,V) taught to map the data from the input domain $\mathcal{X}$ into a latent space $\mathcal{Z}$, named *code* space, and vice versa:

$$\begin{aligned} U &: \mathcal{X} \to \mathcal{Z} \\ V &: \mathcal{Z} \to \mathcal{X}. \end{aligned} \tag{4.4}$$

To do so, the optimisation is carried out by minimising the loss function $\mathcal{L}$, usually quantified by the mean squared error between the input $\boldsymbol{x}$ and its reconstruction $\tilde{\boldsymbol{x}}$ in output:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{x}} \left[\|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|_2^2 \right] = \mathbb{E}_{\boldsymbol{x}} \left[\|\boldsymbol{x} - V(U(\boldsymbol{x}))\|_2^2 \right] \tag{4.5}$$

The risk of reaching the trivial solution of two identity mappings, and therefore having a nonmeaningful mapping $\boldsymbol{Z} = U(\boldsymbol{X}) \equiv \boldsymbol{X}$, is definitely plausible. To avoid this, the expressive power of the AE is limited by, for example, placing a bottleneck that reduces the spatial dimensionality of the code space (e.g. by pooling [87]), or by L_1 regularisation of the weights to ensure a sparse representation [87].

Another way to regularise the network without necessarily requiring a bottleneck is to inject random noise into the input, so that the goal of the AE is not only to reconstruct the latter but also to denoise it. This is the case of denoising AEs, which are trained to reconstruct an input signal that has been artificially corrupted by noise. Denoising helps the model to generalise better [106], which means that the denoising AE can perform well on new inputs, not just on the training data. Their most advanced variant, the stacked denoising AE (SDAE) [107], is probably the most used model to infer spatial information from data and learn new representations and features. These learn the ability of denoising in a layerwise manner, because during training

the noise is injected into one layer at the time, starting from the outermost layer and moving on toward the innermost one. The most commonly adopted noising strategy is dropout [102], which consist in setting to zero some input nodes randomly selected following a Bernoulli distribution with a certain dropout rate. Dropout also prevents overfitting: masking out a portion of randomly selected neurons during training avoids complex coadaptations in which a neuron is only useful in the context of several other specific neurons [108]. Finally, the deep kernelized AE (DKAE) [109] is an architecture based on the intuition that meaningful representations should incorporate similarities between the data points. Ergo, it is regularised by aligning inner products between codes with respect to a kernel matrix computed in input space. The ability of the DKAE to learn effective data representations is enhanced because it learns similarity-preserving embeddings of input data, where the notion of similarity is explicitly controlled by the user and encoded in a positive semidefinite kernel matrix.

4.3 Generative adversarial networks

One of the most revolutionary paradigms of the DL era is the generative adversarial network (GAN). Proposed by Goodfellow *et al.* in 2014 [86], its training principle takes inspiration from game theory: two networks with conflicting goals compete against each other, and both become better by trying to overcome their opponent throughout the simultaneous training, benefiting from this competition. Figure 4.3 illustrates the scheme of a GAN: on one side, the *generator* G takes samples drawn from a random noise variable $\boldsymbol{z} \sim P_z(\boldsymbol{z})$ and aims at reproducing samples from a specific target distribution $\boldsymbol{x} \sim P_x(\boldsymbol{x})$. On the other, a discriminator D has the goal to detect *fake* data $\hat{\boldsymbol{x}}$ produced by the generator and discriminate it from *real* data drawn from the target distribution.

D produces in output a scalar, representing for a given input the probability of it being drawn from the real distribution. The loss function to be optimised is thus formulated as a two-player minmax game summarised by the equation

$$\begin{aligned} \min_G \max_D \mathcal{L}(G, D) = &\mathbb{E}_{\boldsymbol{x}\sim P_x(\boldsymbol{x})} \left[\log\left(D(\boldsymbol{x})\right)\right] \\ &+ \mathbb{E}_{\boldsymbol{z}\sim P_z(\boldsymbol{z})} \left[\log\left(1 - D\left(G(\boldsymbol{x})\right)\right)\right]. \end{aligned} \tag{4.6}$$

A drawback of these NNs is the difficulty in balancing the strength of the

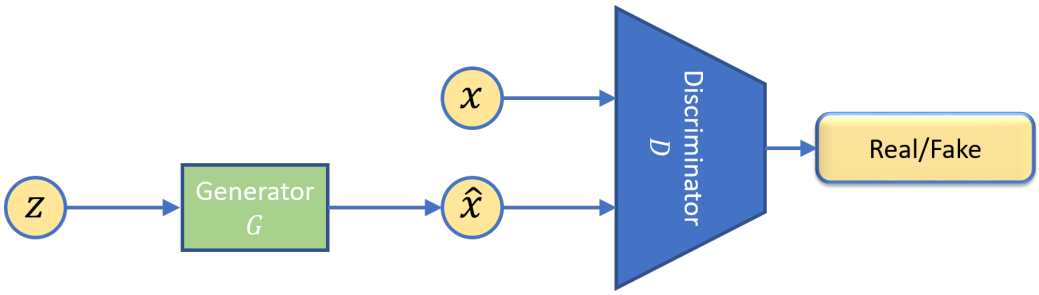


Figure 4.3: Illustration of a Generative Adversarial Network. The discriminator D must detect the fake data $\hat{\boldsymbol{x}}$, which was produced starting from $\boldsymbol{z}$ by the generator G to emulate the distribution of $\boldsymbol{x}$.

two counterparts. Their efforts have to be equal, otherwise one will start to dominate the other, hindering the simultaneous improvement of both. Moreover, training is prone to mode collapse and the convergence is difficult to evaluate due to oscillating and unstable behaviour of the loss function [110].

In more recent years, variations on the theme have been proposed: The Wasserstein GAN named after the Wasserstein distance adopted to account for differences between completely disjoint probability distributions; The Least Square GAN [111], which replaces the log-likelihood terms with squared errors; Conditional GANs (cGANs) whose generator samples from a distribution conditioned on the input data, instead of sampling from random noise.

4.4 Image-to-image translation

Image-to-image (I2I) translation indicates the concept of transferring image contents from one style to another (e.g. drawings or paintings into real pictures, winter landscapes into summer ones, maps of cities into aerial images), and it dates back to the image analogies of Hertzmann *et al.* from 2001 [112]. The *pix2pix* model designed by Isola *et al.* [113] is probably the most notorious example of I2I translation network. It consists of a tandem of cGANs whose generators map data from one domain to the other and vice versa, and the two discriminators try to detect fake data in both their respective domains. The principle of *cycle-consistency* is included in the training strategy of this framework: a composite translation of data from one domain to the other, and then back to the original domain (say $\mathcal{X} \to \mathcal{Y} \to \mathcal{X}$) represents a

full translation cycle which should give in output a high-fidelity reproduction of the original input [113].

The first I2I translation networks relied on a training set of aligned image pairs, for which one image would be the input of one transformation and the other image would be its target, vice versa for the other transformation. However, the need of a paired training set is a major drawback for these models, because it requires that each element from one domain must have its corresponding counterpart in the other domain, which is not always available. Unpaired I2I translation goes beyond this limitation, as it aims at mapping the distributions from the two domains rather than single instances across them. The most famous example is the *Cycle*GAN proposed by Zhu *et al.* [114], which achieves extraordinary results on many tasks including collection style transfer, object transfiguration, season transfer, and photo enhancement. Figure 4.4 shows the capabilities of the *Cycle*GAN which is able to transform Monet paintings into photos and vice versa.

Figure 4.4: Examples of results obtained with the CycleGAN on the Monet to photo dataset provided by [114]. Credits: Sigurd Løkse.

Chapter 5

Self-supervision with affinity matrix comparison

The previous two chapters showed that there are several regression methods to map data across domains. Some are more complex than others, but all share the need of prior information (or a training set), whose gathering can be either manual or automatic. The main contributions of this thesis are built upon the analysis carried out in this chapter: the affinity matrix comparison across domains is shown to be useful for the extraction of local information in the form of preliminary estimates of where the changes have happened. This is exploited for the self-supervised learning of transformations across domains and to perform heterogeneous CD in a fully automatic manner.

After defining distances and similarities, the affinity matrices and their relation to graph theory are presented. Then, the information that can be inferred from their comparison is discussed, highlighting potentials and limitations of this approach.

5.1 Proximity measures

In the following, the notation adopted in [73] is used. A **dissimilarity measure** (DM) d between the elements $\boldsymbol{x}_i$ of a dataset $\boldsymbol{X}$ is a symmetric function that has a lower bound d_0 obtained for equal vectors. That is,

$$d : \boldsymbol{X} \times \boldsymbol{X} \to \mathbb{R} \tag{5.1}$$

such that

$$\exists d_0 \in \mathbb{R} : d_0 \leq d(\boldsymbol{x}_i, \boldsymbol{x}_j), \quad \forall \boldsymbol{x}_i, \boldsymbol{x}_j \in \boldsymbol{X} \tag{5.2}$$

$$d(\boldsymbol{x}_i, \boldsymbol{x}_i) = d_0, \quad \forall \boldsymbol{x}_i \in \boldsymbol{X} \tag{5.3}$$

and

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = d(\boldsymbol{x}_j, \boldsymbol{x}_i), \quad \forall \boldsymbol{x}_i, \boldsymbol{x}_j \in \boldsymbol{X}. \tag{5.4}$$

Moreover, d is a *metric* DM if d_0 can be achieved *only* for equal vectors, and if the so-called *triangual inequality* holds:

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = d_0 \iff \boldsymbol{x}_i = \boldsymbol{x}_j \tag{5.5}$$

and

$$d(\boldsymbol{x}_i, \boldsymbol{x}_k) \leq d(\boldsymbol{x}_i, \boldsymbol{x}_j) + d(\boldsymbol{x}_j, \boldsymbol{x}_k), \quad \forall \boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k \in \boldsymbol{X}. \tag{5.6}$$

A DM can be called a *distance* if and only if $d_0 = 0$.

Moving on, the definition of a **similarity measure** (SM) s on $\boldsymbol{X}$ is analogous:

$$s : \boldsymbol{X} \times \boldsymbol{X} \to \mathbb{R} \tag{5.7}$$

such that

$$\exists s_0 \in \mathbb{R} : s(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq s_0, \quad \forall \boldsymbol{x}_i, \boldsymbol{x}_j \in \boldsymbol{X} \tag{5.8}$$

$$s(\boldsymbol{x}_i, \boldsymbol{x}_i) = s_0, \quad \forall \boldsymbol{x}_i \in \boldsymbol{X} \tag{5.9}$$

and

$$s(\boldsymbol{x}_i, \boldsymbol{x}_j) = s(\boldsymbol{x}_j, \boldsymbol{x}_i), \quad \forall \boldsymbol{x}_i, \boldsymbol{x}_j \in \boldsymbol{X}. \tag{5.10}$$

Additionally, if

$$s(\boldsymbol{x}_i, \boldsymbol{x}_j) = s_0 \iff \boldsymbol{x}_i = \boldsymbol{x}_j \tag{5.11}$$

and

$$s(\boldsymbol{x}_i, \boldsymbol{x}_j)s(\boldsymbol{x}_j, \boldsymbol{x}_k) \leq \left[s(\boldsymbol{x}_i, \boldsymbol{x}_j) + s(\boldsymbol{x}_j, \boldsymbol{x}_k)\right] s(\boldsymbol{x}_i, \boldsymbol{x}_k) \quad \forall \boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k \in \boldsymbol{X} \tag{5.12}$$

then s is a metric SM.

This last equation is less intuitive than its counterpart in Equation (5.6), but it is quite obvious that DMs and SMs have opposite definitions. In fact, it can be proven that (metric) DMs can be turned into (metric) SMs by applying a nonincreasing function, and do the vice versa with a nondecreasing function [73].

5.2 Affinities and graphs

An affinity value describes how close, or similar, two points are in some feature space [115]. Sometimes this incorporates a topological proximity to include spatial structure, forming a composite SM [116]. Affinity matrices have strong ties to proximity graphs. In fact, their use is well-known from computer vision [117, 118, 119], and most importantly from spectral clustering [115, 120] and graph methods [121].

5.2.1 Affinity matrices

For a set $\boldsymbol{X}$ of n data points from domain $\mathcal{X}$, the affinities $A_{i,j}$ between all pairs of points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are enclosed in the affinity matrix $\boldsymbol{A}^{\mathcal{X}} \in \mathbb{R}^{n\times n}$. Affinities are symmetric, i.e. $A_{i,j} = A_{j,i}$, and it is very common that $A_{i,j} \in [0, 1]$. The choice of which measure should be used to quantify $A_{i,j}$ is not unique.

The most commonly used is the Gaussian metric SM [73, 115], obtained by applying the RBF kernel to the Euclidean distance:

$$A_{i,j} = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{\sigma^2}\right), \quad i, j \in \{1, \ldots, n\}. \tag{5.13}$$

Selecting the kernel width σ is a pivotal issue that strongly depends on the tackled problem, the kind of data, and the desired properties [73, 115]. There are approaches that can select σ by estimating it from the data [122], such as Silverman's rule of thumb [123]. Alternatively, one can set it equal to the average distance to the k^{th} nearest neighbour of all data points in $\boldsymbol{X}$, with k being a reasonable number. This heuristic allows to capture a characteristic distance within the samples and it is robust with respect to outliers: the neighbourhood of $\boldsymbol{x}_i$ presents values of $A_{i,j}$ within a reasonable interval, whereas the rest gradually decays to 0 [124].

Another example of SM that could be used to build the affinity matrix is the *cosine similarity* [73]

$$A_{i,j} = \frac{\boldsymbol{x}_i \cdot \boldsymbol{x}_j}{\|\boldsymbol{x}_i\| \cdot \|\boldsymbol{x}_j\|}, \quad i, j \in \{1, \ldots, n\} \tag{5.14}$$

in which there are no parameters, but on the other hand the amplitudes of the vectors are normalised and do not play any role in the measurement [125].

5.2.2 Graphs

A graph is composed of vertices (or nodes) and edges [126]. The former are a set of integers $\mathcal{V} = \{1, \ldots, n\}$, indexing the elements of the system (or dataset) represented by the graph. The latter are a set of connections $\mathcal{E} = \{\{v_i, v_j\} : v_i, v_j \in \mathcal{V}\}$ pairing two nodes in the graph. A graph can be either *undirected*, which means that these links are bidirectional and symmetric and $\{v_i, v_j\} \in \mathcal{E} \iff \{v_j, v_i\} \in \mathcal{E}$, or *directed*, for which the connections can go one-way only, or be asymmetric.

In a *weighted* graph a weight is assigned to each edge, indicating its importance or strength. Normally these weights are nonnegative, and a zero weight means that there is no edge, that is $w_{i,j} = 0 \iff \{v_i, v_j\} \notin \mathcal{E}$. In a *fully-connected* graph $w_{i,j} > 0 \, \forall i, j$, and in an undirected graph $w_{i,j} = w_{j,i} \, \forall i, j$. A nonweighted graph is just the particular case in which the weights assume binary values, namely $w_{i,j} = \{0, 1\} \, \forall i, j$. An important value associated with each node is its *degree* d_i, calculated by summing up the weights of all its connections towards the other vertices.

If $w_{i,j}$ encodes some SM between the vertices v_i and v_j, it can be the entry of an affinity matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ [115]. From the opposite perspective, a dataset $\boldsymbol{X}$ whose pairwise relationships are described by an affinity matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ can be represented by a weighted undirected graph with n nodes and up to n^2 edges. In that case, the degree d_i is computed as the sum over the rows of $\boldsymbol{A}$, or columns, since $\boldsymbol{A}$ is symmetric. Notice that when Equation (5.13) is used, the graph associated with $\boldsymbol{A}$ is also fully-connected, since $0 < A_{i,j} \leq 1 \, \forall i, j$. Sometimes, the graph is regularised by making $\boldsymbol{A}$ sparser, which results in a lighter pruned graph. For example, a threshold ϵ for the Euclidean distances in Equation (5.13) can be introduced so that

$$A_{i,j} = \begin{cases} \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{\sigma^2}\right) & \text{, if } \|\boldsymbol{x}_i - \boldsymbol{x}_j\| \leq \epsilon \\ 0 & \text{, otherwise} \end{cases}, \quad i, j \in \{1, \ldots, n\} \tag{5.15}$$

From now on, the affinity values are assumed to be computed by means of Equation (5.13), but the analysis in the following applies in general to other metrics as well.

5.3 Affinity matrix comparison

Ideally, affinity matrices should describe the relationships between samples without depending on which domain these data lie in. For similar spatial structures in different domains, the corresponding affinity matrices should be similar as well. Take the simple example in which a scene contains three classes of data, and it is observed by sensors $\mathcal{X}$ and $\mathcal{Y}$ at the same time. Figure 5.1 illustrates each pixel from $\boldsymbol{X} \in \mathbb{R}^{2\times 25}$ (that is, a data matrix holding 25 instances of two-dimensional vectors) in the feature space of $\mathcal{X}$ (left) and the ones from $\boldsymbol{Y} \in \mathbb{R}^{2\times 25}$ in the feature space of $\mathcal{Y}$ (right).

Three separate clusters can be clearly distinguished from one another. However, their signatures differ across the two domains, because the sensors measure different physical properties, encoded in quantities whose magnitude can be on completely different scales. Obviously, the clusters do not present the exact same structure in the two spaces because of the intrinsic numerical fluctuations in the measurements, as discussed in Section 2.1.

Focusing on the pixel highlighted by a red diamond, its affinity to other data points decays exponentially with the squared Euclidean distance measured

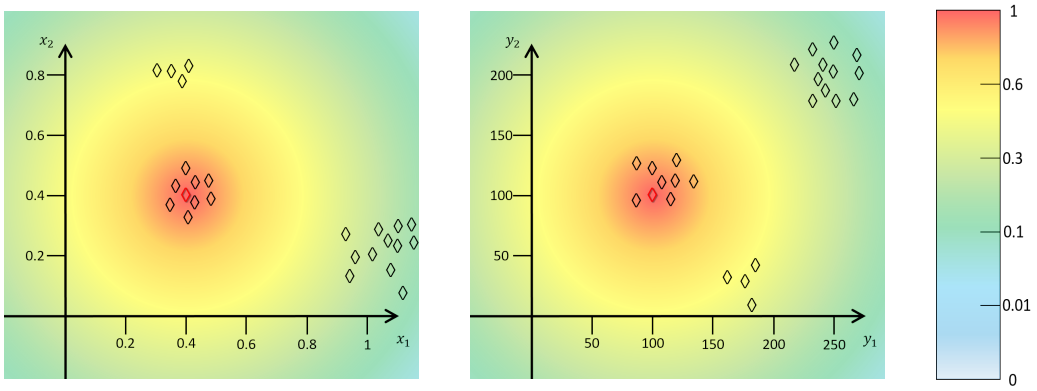
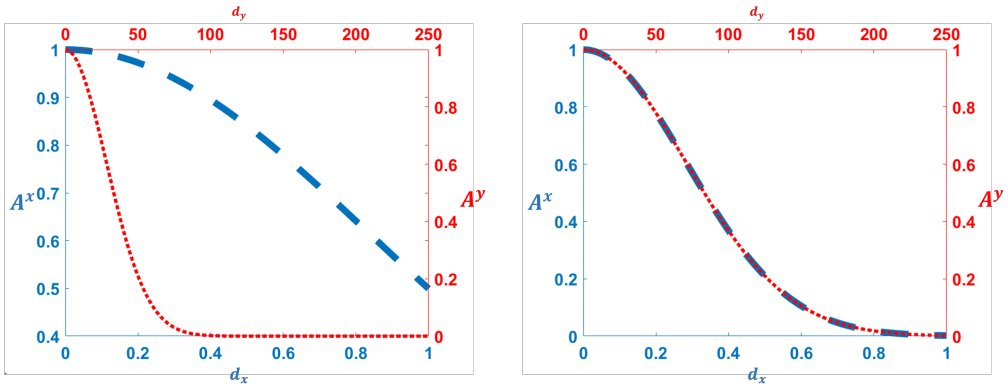


Figure 5.1: The acquisitions by two different sensors $\mathcal{X}$ and $\mathcal{Y}$ of the same scene can differ considerably in signature and in relations between classes. Thus, there can be a large discrepancy between the feature spaces where their respective data $\boldsymbol{X}$ and $\boldsymbol{Y}$ lie. Still, the affinity matrices computed separately in each of the domains are required to be congruent to one another. That is, affinities should not depend on the scale of the distances from which they are evaluated, as in this example, where the affinity heat maps for the selected pixel (red diamond) are comparable in the two domains.



(a) (b)

Figure 5.2: Alignment of the affinity profiles. a) The affinities of $\boldsymbol{A}^{\mathcal{X}}$ ($\boldsymbol{A}^{\mathcal{Y}}$) decay too slow (fast) due to a large $\sigma_{\mathcal{X}}$ (small $\sigma_{\mathcal{Y}}$); b) the proper selection of $\sigma_{\mathcal{X}}$ and $\sigma_{\mathcal{Y}}$ leads to congruous behaviours for $\boldsymbol{A}^{\mathcal{X}}$ and $\boldsymbol{A}^{\mathcal{Y}}$.

from the pixel. The heat map shows the radial behaviour of the affinity. In a perfect scenario, this affinity profile should be agnostic to the system of reference, which means that affinities between pixels in $\mathcal{X}$ should be similar to the affinities computed in $\mathcal{Y}$. In a nutshell, $\boldsymbol{A}^{\mathcal{X}}$ and $\boldsymbol{A}^{\mathcal{Y}}$, respectively obtained from the distances $d_{\mathcal{X}}$ in $\boldsymbol{X}$ and the distances $d_{\mathcal{Y}}$ in $\boldsymbol{Y}$, should be very close to each other. For the RBF kernel, this can be achieved by a correct selection of the kernel bandwidths $\sigma_{\mathcal{X}}$ and $\sigma_{\mathcal{Y}}$, thus aligning the shapes of the functions $\boldsymbol{A}^{\mathcal{X}}$ and $\boldsymbol{A}^{\mathcal{Y}}$, as depicted in the example of Figure 5.2.

The bandwidths $\sigma_{\mathcal{X}}$ and $\sigma_{\mathcal{Y}}$ are very unlikely to be set to the same value, because each must be tuned according to the data (and the distances) related to its corresponding domain. A reasonable approach is to use the same rationale but applied separately, once for $\sigma_{\mathcal{X}}$ and once for $\sigma_{\mathcal{Y}}$. Recalling one of the suggestions from Section 5.2.1, one can set $\sigma_{\mathcal{X}}$ equal to the mean distance to the k^{th} nearest neighbour in $\boldsymbol{X}$, and similarly for $\sigma_{\mathcal{Y}}$ and $\boldsymbol{Y}$. This approach is apparently heuristic, but has ties to the k-nearest neighbour kernel density estimator proposed by Mack and Rosenblatt [127] already in 1979, and has been used as a practical method for bandwidth selection since then.

5.4 Change graph

Assuming that the affinities in $\boldsymbol{A}^{\mathcal{X}}$ and $\boldsymbol{A}^{\mathcal{Y}}$ have been appropriately aligned, then these two matrices are ready to be compared. In the previous example, the main assumption is that sensors observing the same reality should produce similar graphs. Extending it further, this should hold for heterogeneous images acquired at different times over an unchanged scene. From the opposite perspective, dissimilar affinity matrices mean that the spatial relationships within the images are different, suggesting that changes have occurred. This means that the *change graph* associated with the affinity values

$$A_{i,j}^{\text{change}} = |A_{i,j}^{\mathcal{X}} - A_{i,j}^{\mathcal{Y}}| \in [0,1], \quad i,j \in \{1,\dots,n\} \tag{5.16}$$

should capture strategic information about where and how strong the perturbations in the affinities are, and consequently give an indication of change in the scene. This core hypothesis is the fundamental base for the methodologies developed in Paper I and Paper II presented in chapters 7 and 8, as explained below.

5.4.1 Frobenius norm of $\boldsymbol{A}^{\text{change}}$

The intuition behind this choice is quite simple: the more drastic, intense, and widespread a change is, the more different $\boldsymbol{A}^{\mathcal{X}}$ and $\boldsymbol{A}^{\mathcal{Y}}$ are, and the larger the Frobenius norm of $\boldsymbol{A}^{\text{change}}$ is. Theoretically, for the patches $\boldsymbol{X}$ and $\boldsymbol{Y}$ of size $h \times w$ rearranged in sets of $n = h \cdot w$ vectors,

$$0 \leq \|\boldsymbol{A}^{\text{change}}\|_F \leq \sqrt{n^2 - n}, \tag{5.17}$$

where the term n^2 in the square root stems from the number of elements in $A_{i,j}^{\text{change}} \leq 1 \; \forall i,j \in \{1,\dots,n\}$ and the subtraction of n is due to the fact that the elements on the diagonal of $\boldsymbol{A}^{\text{change}}$ cannot be different from 0.

These bounds are in fact far from being useful in practice, because it is very unlikely that the $\boldsymbol{A}_{i,j}^{\mathcal{X}}$ and $\boldsymbol{A}_{i,j}^{\mathcal{Y}}$ are exactly the same (lower bound) or complementary, being zero when the other is one and vice versa (upper bound). Thus, there is no real reference to say whether a patch contains changes or not just by looking at $\|\boldsymbol{A}^{\text{change}}\|_F$ on its own. Moreover, this does not indicate *where* the changes have occurred inside the patch, because it returns a single value for the whole covered area. In addition, the number of elements

Algorithm 1 Possibilities of change for each pixel

for all $\boldsymbol{X} \in \mathcal{X}^{h \times w} \subset \mathcal{I}_{\mathcal{X}}$ and $\boldsymbol{Y} \in \mathcal{Y}^{h \times w} \subset \mathcal{I}_{\mathcal{Y}}$ **do**
  Compute $d_{\mathcal{X}}$ between all pixel pairs in $\boldsymbol{X}$
  Compute $d_{\mathcal{Y}}$ between all pixel pairs in $\boldsymbol{Y}$
  Determine $\sigma_{\mathcal{X}}$ and $\sigma_{\mathcal{Y}}$
  Compute $\boldsymbol{A}^{\mathcal{X}}$ and $\boldsymbol{A}^{\mathcal{Y}}$
  Compute $A^{\text{change}}_{i,j} = |A^{\mathcal{X}}_{i,j} - A^{\mathcal{Y}}_{i,j}| \in [0,1], \quad i,j \in \{1,\dots,n\}$
  Compute $f = \|\boldsymbol{A}^{\text{change}}\|_F$
  Add f to $\mathcal{S}^F_i \; \forall i \in \{1,\dots,n\}$
end for
for all $i = 1,\dots,N$ **do**
  Compute the mean over $\mathcal{S}^F_i$
end for

in the affinity matrices is equal to n^2, which becomes quickly unfeasible in terms of computations and memory consumption, so it is possible to apply this strategy only for $h \ll H$ and $w \ll W$, being H and W the sizes of the whole images counting $N = H \cdot W$ pixels.

For these reasons, the most suitable solution to exploit the Frobenius norm of $\boldsymbol{A}^{\text{change}}$ is to compute it for small patches, and associate each pixel in the image grid to the average over the set $\mathcal{S}^F$ of matrix norms evaluated for all the patches covering such pixel. Intuitively, a small average is associated with a small possibility of change, and vice versa for a large average. Algorithm 1 summarises the whole procedure, whose output is an image with size $H \times W$ and one value per pixel. In a nutshell, a sliding window starts from the upper left corner, and after the evaluation on that patch is done, it is shifted by one pixel and the procedure is iterated for the whole set of overlapping patches. This is the main drawback of this technique, since the loop over all these patches can be computationally heavy. Shifting the sliding window by a factor larger than one would speed up the algorithm, but with a much poorer result: intuitively, the final outcome would exhibit an unnatural tile pattern.

The Frobenius norm of A^{change} was exploited in Paper I of this thesis, but

was replaced with the improved algorithms described in the next sections for Paper II and Paper III.

5.4.2 Vertex degrees of the change graph

An alternative way to infer information from the change graph is to draw the attention to its vertices. The main idea is that if some pixels have been affected by changes, many of their affinities with the other pixels will have changed between $\boldsymbol{A}^{\mathcal{X}}$ and $\boldsymbol{A}^{\mathcal{Y}}$, apart from the exceptional cases in which two pixels experience the same change from one class to another. Instead, unchanged pixels have only their affinity with those changed pixels affected. From the perspective of the change graph, this means that changed pixels have most of their edges associated with a large weight, whereas unchanged pixels have strong connections only towards changed ones. Ergo, the vertex degrees of the change graph can be considered a score that expresses the chance of a pixel being affected by a change. By introducing an adequate scaling factor $n = h \cdot w$ denoting the number of data points in the patches, the score

$$\alpha_i = \frac{1}{n-1} \sum_{j=1}^{n} A_{i,j}^{\text{change}} \in [0,1], \quad i \in \{1, \dots, n\} \tag{5.18}$$

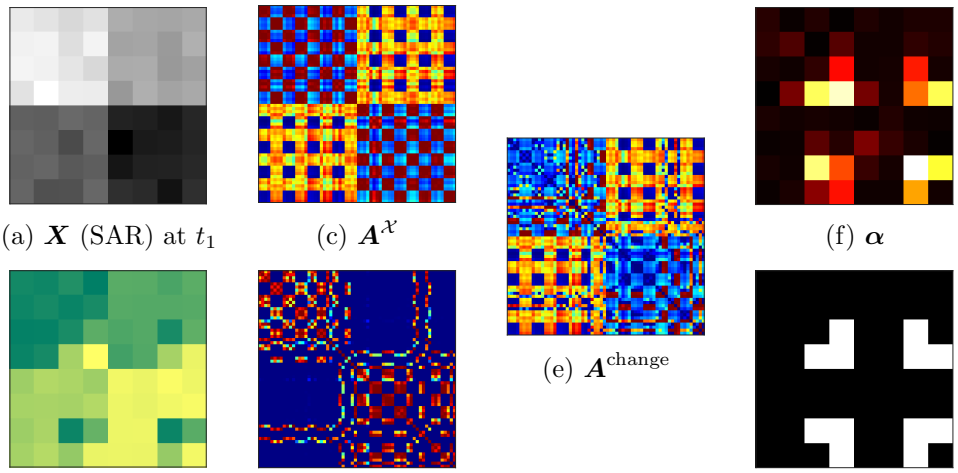
can be interpreted as the probability of change for every pixel i inside the patches.

First of all, the main advantage of using this approach rather than the Frobenius norm is that these values have an absolute reference, they can be seen as probabilities bounded between 0 and 1. Moreover, each α_i relates only to pixel i, instead of being one value assigned to all the involved pixels. In theory, this comparison could be applied directly to the whole images $\mathcal{I}_{\mathcal{X}}$ and $\mathcal{I}_{\mathcal{Y}}$, if it were not for the computation and memory constraints. In practice, it still must be applied patchwise, but on the other hand it is not necessary to consider the whole set of all the overlapping patches. The sliding window can be moved with shift factor greater than one, selecting a significantly smaller subset of patches. This reduces greatly the computational load without affecting substantially the final result. The method is summarised in Algorithm 2, where $\mathcal{P}$ is used to indicate the subset of selected patches. The output is again one value per each of the $N = H \cdot W$ pixels.

The toy example in Fig. 5.3 helps to explain the effectiveness of the approach.

Algorithm 2 Probability of change for each pixel

for all patches in the subset $\mathcal{P}$ **do**
  Compute distances between all pixel pairs in $\boldsymbol{X}$
  Compute distances between all pixel pairs in $\boldsymbol{Y}$
  Determine $\sigma_{\mathcal{X}}$ and $\sigma_{\mathcal{Y}}$
  Compute $\boldsymbol{A}^{\mathcal{X}}$ and $\boldsymbol{A}^{\mathcal{Y}}$
  Compute $A^{\text{change}}_{i,j} = |A^{\mathcal{X}}_{i,j} - A^{\mathcal{Y}}_{i,j}|, \quad i,j \in \{1,\dots,n\}$
  Compute $\alpha_i = \frac{1}{n}\sum_{j=1}^{n} A^{\text{change}}_{i,j}, \quad i \in \{1,\dots,n\}$
  Add α_i to $\mathcal{S}^{\alpha}_i \; \forall i \in \{1,\dots,n\}$
end for
for all $i = 1,\dots,N$ **do**
  Compute the mean over $\mathcal{S}^{\alpha}_i$
end for



(a) $\boldsymbol{X}$ (SAR) at t_1 (c) $\boldsymbol{A}^{\mathcal{X}}$ (f) $\boldsymbol{\alpha}$

(e) $\boldsymbol{A}^{\text{change}}$

(b) $\boldsymbol{Y}$ (optical) at t_2 (d) $\boldsymbol{A}^{\mathcal{Y}}$ (g) CD map

Figure 5.3: 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) $\boldsymbol{\alpha}$ obtained by applying Equation (5.18); g) CD map obtained by thresholding $\boldsymbol{\alpha}$.

Figure 5.3a simulates a patch $\boldsymbol{X}$ of 8×8 pixels extracted from a SAR image captured at t_1. It consists of four blocks representing four different classes. The corresponding patch $\boldsymbol{Y}$ extracted from an optical image at t_2 is depicted

in Figure 5.3b, where the classes are arranged in the same way. Changes are introduced by placing 4 pixels representing each class in the bottom right quadrant of each block of $\boldsymbol{Y}$. In this way, all the possible transitions between one class and the others occur between t_1 and t_2. The 64×64 affinity matrices $\boldsymbol{A}^{\mathcal{X}}$ and $\boldsymbol{A}^{\mathcal{Y}}$ are depicted in Figure 5.3c and Figure 5.3d. They both show a regular squared pattern with high affinities in red and low affinities in blue, but the latter presents clear irregularities and perturbations due to the changed pixels which are breaking the block pattern in Fig. 5.3b. Once $\boldsymbol{A}^{\text{change}}$ is evaluated (Figure 5.3e), Equation (5.18) yields the 8×8 result of Figure 5.3f, denoted as $\boldsymbol{\alpha}$, where a darker (brighter) pixel means a smaller (larger) α_i. Finally, one may retrieve a CD map by thresholding $\boldsymbol{\alpha}$, as shown in Figure 5.3g.

5.5 Affinities as new high-dimensional representations

In this section, the comparison of the affinity matrices is approached from a different angle. First extract row i of affinity matrix $\boldsymbol{A}^{\mathcal{X}}$ and row j of affinity matrix $\boldsymbol{A}^{\mathcal{Y}}$ as:

$$\begin{aligned} \boldsymbol{A}_i^{\mathcal{X}} &= \left[A_{i,1}^{\mathcal{X}}, \ldots, A_{i,n}^{\mathcal{X}}\right], \\ \boldsymbol{A}_j^{\mathcal{Y}} &= \left[A_{j,1}^{\mathcal{Y}}, \ldots, A_{j,n}^{\mathcal{Y}}\right]. \end{aligned} \tag{5.19}$$

Let these vectors be representations of pixel i from patch $\boldsymbol{X}$ and pixel j from patch $\boldsymbol{Y}$, respectively, in a new affinity space with n features. Then, a novel crossmodal distance between these pixels can be defined as

$$D_{i,j} = \frac{1}{\sqrt{n}} \|\boldsymbol{A}_i^{\mathcal{X}} - \boldsymbol{A}_j^{\mathcal{Y}}\|_2 \in [0,1], \quad i,j \in \{1,\ldots,n\}, \tag{5.20}$$

noting that since the affinities are normalised to the range $[0,1]$, so is $D_{i,j}$. This crossmodal distance allows to compare data across the two domains directly from their input space features. It further allows us to distinguish pixels that have consistent relations to other pixels in both domains from those that do not. Hence, two transformations $\boldsymbol{Z}_{\mathcal{X}}(\boldsymbol{X}) : \mathcal{X}^{h\times w} \to \mathcal{Z}^{h\times w}$ and $\boldsymbol{Z}_{\mathcal{Y}}(\boldsymbol{Y}) : \mathcal{Y}^{h\times w} \to \mathcal{Z}^{h\times w}$ can be defined and trained, so that the data are mapped patchwise into a new common space $\mathcal{Z}$ where these relations hold

true. This is accomplished by enforcing that

$$R\left(\boldsymbol{z}_i^{\mathcal{X}}, \boldsymbol{z}_j^{\mathcal{Y}}\right) \simeq S_{i,j}, \quad i,j \in \{1,\dots,n\}, \tag{5.21}$$

where $S_{i,j}$ are elements of a similarity matrix $\boldsymbol{S} \triangleq 1 - \boldsymbol{D}$ and $R\left(\boldsymbol{z}_i^{\mathcal{X}}, \boldsymbol{z}_j^{\mathcal{Y}}\right)$ is a correlation that measures the similarity between the new representations of $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$ produced by the two transformations. Note that $S_{i,i}$ represents the similarity between $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$, so $S_{i,i}$ can be different from 1. Following the same line of thoughts, $\boldsymbol{S}$ is not symmetric, because the similarity between $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$ is not necessarily the same as between $\boldsymbol{x}_j$ and $\boldsymbol{y}_i$.

5.6 Limitations

It is undeniable that the affinity matrix comparison for CD is not flawless, and it can be ineffective in some specific cases. The most obvious case occurs when a change does not affect the structures within the area under investigation. Truly, this method relies on detecting changes in the image shapes rather than in the classes per se. If only the latter are changing, but not the former, the affinity matrices would not be perturbed and their comparison would be inevitably helpless.

Another issue related to this kind of analysis is summarised in the examples depicted in Figure 5.4. For these, the affinity matrices are assumed to be associated with a nonweighted graph, so $A_{i,j} = 1$ if pixels i and j are identical, 0 otherwise. Only the probability score α from Section 5.4.2 is calculated because it highlights the problem unequivocally, but also the other two algorithms are affected nonetheless. The simple 2×2 patch $\boldsymbol{X}$ contains two classes. To show different scenarios, three cases of $\boldsymbol{Y}$ are presented. The first leads to a correct evaluation, with α_1 suggesting that pixel I is very likely the changed one. Instead, the output in the second case is wrong. Since the shapes in $\boldsymbol{Y}$ are identical to the previous case (but not the classes), so are the affinity matrix $\boldsymbol{A}^{\mathcal{Y}}$, $\boldsymbol{A}^{\text{change}}$ and the final output $\boldsymbol{\alpha}$. As a result, pixel I is mistakenly pointed out as changed, and vice versa for the other three. Thus, in case of ambiguous situations in which a permutation in the spatial structures between $\boldsymbol{X}$ and $\boldsymbol{Y}$ is attributable to either few or many pixels changing, the comparison of the affinity matrices tends to conclude with the former situation rather than with the latter. The third case is the most extreme: two pixels have changed, namely pixels II and IV, but the

information encapsulated by $\boldsymbol{A}^{\mathcal{Y}}$ would be the same if pixels I and III had changed (classes arranged in the same vertical shapes, but swapped). Given that the two events are equiprobable, the uncertainty is such that the pixel shares the same score in $\boldsymbol{\alpha}$, i.e. they all present 50% probability of change, leading to a meaningless result.

These examples are meant to emphasise that comparing affinity matrices relies on changes in the shapes of $\boldsymbol{X}$ and $\boldsymbol{Y}$, but without any ancillary information or prior knowledge about the data and the classes involved, there

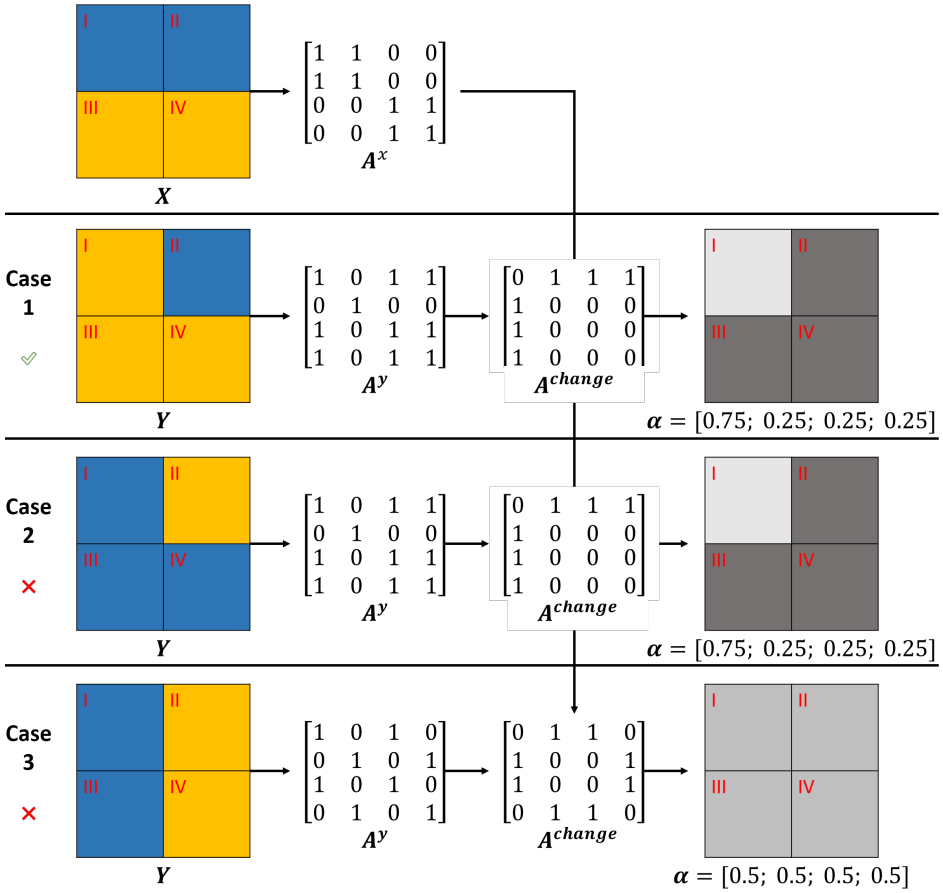


Figure 5.4: Limitations of the affinity matrix comparison: the patch $\boldsymbol{X}$ and its corresponding affinity matrix $\boldsymbol{A}^{\mathcal{X}}$ on top are compared against three different cases of $\boldsymbol{Y}$ and $\boldsymbol{A}^{\mathcal{Y}}$. The first case leads to a correct result (green check mark), the other two lead to a wrong one (red cross mark).

could be residual ambiguities which are too serious to be resolved. Choosing large patch sizes can mitigate these issues, because it becomes less likely to encounter situations similar to the ones described in Figure 5.4.

Finally, the analysis carried out in Section 5.3 evidences that the kernel parameter selection can be critical. In fact, it shows that the affinity matrices are sensible to different parameter settings, which can lead to totally meaningless outcomes when they are not ideal.

Chapter 6

Research publications

This chapter offers an executive summary of the publications enclosed in this thesis and a list of the excluded works.

6.1 Paper summaries

Paper I - Unsupervised image regression for heterogeneous change detection

In this paper, the problem of unsupervised heterogeneous CD was tackled by means of the pixel-based regression methods presented in Chapter 3. In particular, RFs, GPs, and SVMs were employed for the image translations, along with a state-of-the-art transformation method based on kernel regression [20, 123] used as a reference. The comparison between these approaches was not strictly meant to determine the best candidate, but rather to show their advantages and disadvantages, and to support the idea that the selection of a proper training set is the most important aspect.

Concerning the self-supervised delineation of unchanged pixels, Algorithm 1 from Section 5.4.1 was applied, showing its effectiveness in enclosing the vast majority of all the data classes while excluding changed pixels at the same time. The results suggest that the method is suitably robust to the patch size and the training set size, although suggesting for both that the larger, the better.

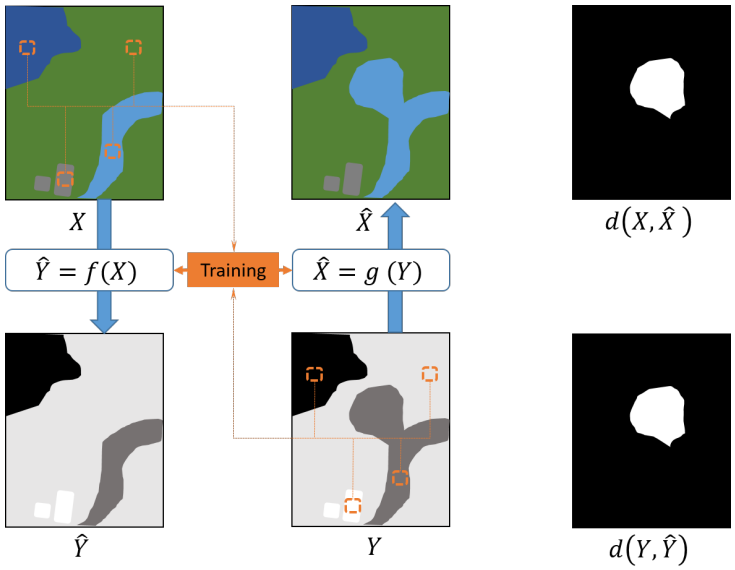


Figure 6.1: Illustration of the methodology proposed in paper I. The regression functions are tuned to fit the training data. This must not contain changed areas, so that the proper mappings are achieved. Afterwards, the transformed images are compared against the original ones to highlight changes.

Figure 6.1 illustrates the main idea behind this work: two transformations are trained to map data across two domains, so the images become comparable to one another and the changes can be detected by a simple image difference.

Contributions by the author

- The approach was first conceived by my supervisor Stian N. Anfinsen and me, then all the coauthors contributed equally to its development.
- I created the images and the ground truth composing the *California* dataset.
- The MATLAB implementation of the method and the experiments were carried out by me.
- I wrote the first draft of the manuscript and oversaw the subsequent editing process.

Paper II - Deep image translation with an affinity-based change prior for unsupervised multimodal change detection

This paper has two distinct contributions. The first regards the formulation of Algorithm 2 from Section 5.4.2, which infers the same self-supervised prior information about the changed pixels as its predecessor, but it can be reputed a direct improvement. Truly, the computational load is highly reduced while at the same time the output quality is increased.

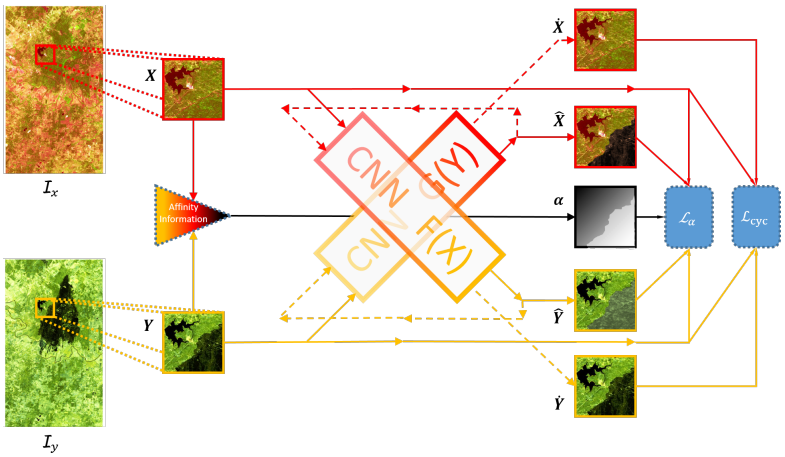
The second contribution consists in the definition of two new deep learning frameworks for heterogeneous CD: crossdomain weighted translation network (X-Net) and the adversarial cyclic encoder network (ACE-Net). These are able to exploit sinergically Algorithm 2 and the concepts of I2I and adversarial learning, discussed in Chapter 4, to fulfil image transformation in an unsupervised manner without being affected by the changed areas. The results indicate that both the X-Net and the ACE-Net perform favourably compared to the state-of-the-art, with the former producing stable and consistent performance, and the latter achieving the best results, at the cost of higher complexity and a more diligent training.

In Figure 6.2, the schematics of the two networks are depicted. For simplicity, the arrows in Figure 6.2b represent the data flow involving only the loss terms related to $\boldsymbol{X}$. The reader is referred to Paper II for further details.

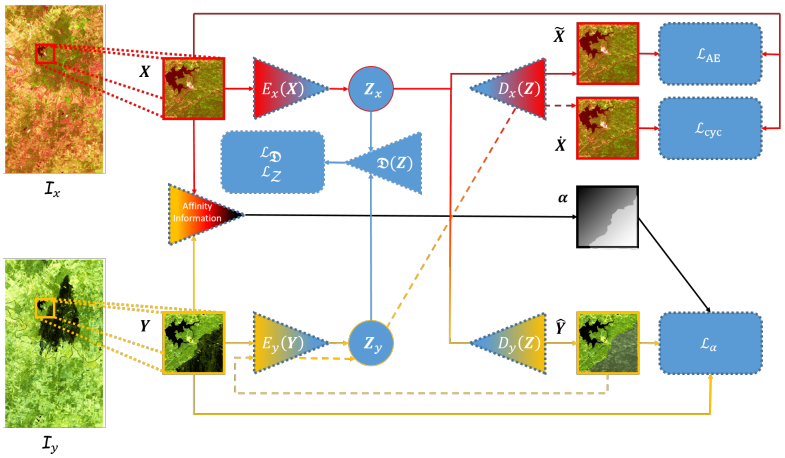
Contributions by the author

The main breakthroughs that led to the work accomplishment came during the author's stay at the DITEN Department, University of Genoa, Italy.

- Algorithm 2 was developed by me in close collaboration with the other authors, who equally participated to the design of the proposed DL architectures and their loss functions.
- I implemented the proposed framework in TensorFlow 1.4, together with two reference DL methods representing the state-of-the-art. I also conducted the experiments.
- The manuscript draft was written by me and edited in collaboration with the coauthors.



(a) X-Net



(b) ACE-Net

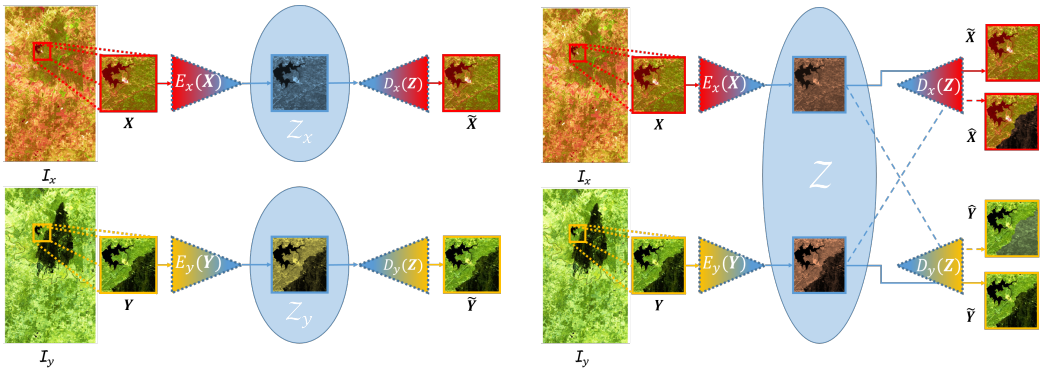
Figure 6.2: Data flows of the architectures proposed in Paper II. a) The X-Net transforms $\boldsymbol{X}$ into $\mathcal{Y}$ obtaining $\hat{\boldsymbol{Y}}$, whose comparison with $\boldsymbol{Y}$ is weighted by $\boldsymbol{\alpha}$. Then, $\hat{\boldsymbol{Y}}$ is translated back to $\mathcal{X}$, and the resulting $\dot{\boldsymbol{X}}$ is expected to match the input $\boldsymbol{X}$. The same rationale applies to the flow of $\boldsymbol{Y}$. b) The ACE-Net seeks the alignment of its code spaces $\mathcal{Z}_\mathcal{X}$ and $\mathcal{Z}_\mathcal{Y}$ also thanks to the adversarial training against a discriminator $D(\boldsymbol{Z})$, which tries to discriminate codes produced by one encoder from the ones produced by the other. Then, this network enforces the same principles as the X-Net, but in addition $\boldsymbol{X}$ is supposed to be reconstructed starting from the code $\boldsymbol{Z}^\mathcal{X}$, which means that the reconstructed data $\tilde{\boldsymbol{X}}$ should be a reproduction of the input. The flow related to $\boldsymbol{Y}$ is omitted for an easier visual interpretation, but it follows the same schematics.

Paper III - Code-aligned autoencoders for multimodal change detection in remote sensing images

The concepts presented in Section 5.5, together with the ones of I2I and cycle-consistency, is the fundamental core upon which this paper is built. The latter exploits these ideas to align the code spaces of two autoencoders and treat them as a common latent space, so that the output of one encoder can be the input of both decoders, leading in one case to reconstruction of data in their original domain, and in the other case to their transformation into the other domain.

To a certain extent, this work represent a further development of the previous paper. The proposed architecture resembles the ACE-Net, but is in fact lighter, simpler, and easier to train, especially because it does not need the adversarial training. Moreover, the information retrieved from the affinity matrices is more advanced, since it relates all possible pixel pairs across the dual-domain patches, not only those that are colocated.

The illustration in Figure 6.3 shows how the architecture per se is very simple, since the real accomplishments lie on the definition of the code alignment and its incorporation in the loss function.



(a) Not aligned code spaces (b) Aligned code spaces

Figure 6.3: Illustration of the methodology proposed in Paper III. a) Two autoencoders are trained to reconstruct their input images after being mapped into two latent spaces. The latter, without any further constraints, cannot be assumed to be ovelapped and matching. b) By introducing loss terms that enforce the alignment of the two code space into a common latent domain, the images can be either reconstructed in their original domain or transformed into the other domain.

Contributions by the author

- The ideas in Section 5.5 were conceived and formulated by supervisor Stian N. Anfinsen and me.
- Mads Adrian Hansen and I worked with equal efforts to release a heterogeneous CD framework written in TensorFlow 2.0, which is publicly available and easily adaptable to develop new methodologies upon it.
- The experiments enclosed in this paper were conducted by me.
- I carried out the writing of the manuscript draft.

6.2 Other publications

The following papers and works were not included in the thesis:

- Luigi T. Luppino, Stian N. Anfinsen, Gabriele Moser, Robert Jenssen, Filippo M. Bianchi, Sebastiano B. Serpico, and Grégoire Mercier, "**A clustering approach to heterogeneous change detection**," *Scandinavian Conference on Image Analysis (SCIA)*. Tromsø, 2017, pp. 181–192
- Luigi T. Luppino, Filippo M. Bianchi, Gabriele Moser and Sstian N. Anfinsen, "**Remote sensing image regression for heterogeneous change detection**," *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, Aalborg, 2018. pp. 1-6.
- Luigi T. Luppino, Michael Kampffmeyer, Filippo M. Bianchi, Gabriele Moser, Robert Jenssen, and Stian N. Anfinsen, "**An adversarial autoencoder network for heterogeneous change detection**," *Northern Lights Deep Learning Workshop (NLDL)*, Tromsø, 2019.
- Luigi T. Luppino, Michael Kampffmeyer, Filippo M. Bianchi, Gabriele Moser, Robert Jenssen, and Stian N. Anfinsen, "**Adversarial cyclic encoder networks for heterogeneous change detection**," *European Space Agency Living Planet Symposium (LPS)*, Milan, 2019.
- Julian Fagir, Luigi T. Luppino, Max Frioud, and Daniel Henke, "**Change detection between high-resolution airborne SAR and oblique optical data by projection on a point cloud**," *IEEE Journal of*

Selected Topics in Applied Earth Observations and Remote Sensing, submitted.

- Luigi T. Luppino, Michael Kampffmeyer, Filippo M. Bianchi, Gabriele Moser, Robert Jenssen, and Stian N. Anfinsen, "**Code-aligned autoencoders for multimodal change detection in remote sensing Images**," *Northern Lights Deep Learning Workshop (NLDL)*, Tromsø, 2020.
- Bilal Babar, Luigi T. Luppino, Stian N. Anfinsen, and Tobias Boström, "**Random forest regression for improved mapping of solar radiation at high latitudes**," *Solar Energy*, vol. 198, pp 81-92, 2020.
- Federico Figari Tomenotti, Luigi T. Luppino, Mads A. Hansen, Gabriele Moser, and Stian N. Anfinsen, "**Heterogeneous change detection with self-supervised deep canonically correlated autoencoders**," *IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2020)*, submitted.
- Gabriele Moser, Stian N. Anfinsen, Luigi T. Luppino, and Sebastiano B. Serpico, "**Change detection with heterogeneous remote sensing data: from semi-parametric regression to deep learning**," *IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2020)*, submitted.

Paper I

Paper II

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 two 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 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].

*L.T. Luppino, M. Kampffmeyer, R. Jenssen and S.N. Anfinsen are with the Machine Learning Group, Department of Physics and Technology, UiT The Arctic University of Norway, e-mail: luigi.t.luppino@uit.no.

F.M. Bianchi is with NORCE Norwegian Research Center, Norway.

G. Moser and S.B. Serpico are with DITEN Department, University of Genoa, Italy.

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. The alternative is to combine heterogeneous data, which on one hand allows to exploit the capabilities of 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], [13]. In other 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 [14], [15], 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], [16]. 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 [17], nonlinear regression [15], dictionary learning [14], manifold learning [18], or copula theory [19] are worth mentioning.

C. Proposed method

We propose a deep image translation approach to perform unsupervised CD based on heterogeneous remote sensing data. Most importantly, a comparison of domain-specific affinity matrices allows us to retrieve in a self-supervised manner the *a priori* change indicator driving our training process, referred to as the prior. 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.

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 two 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.

The implementations of our architectures are available at this link: https://github.com/llu025/Heterogeneous_CD, together with the re-implementation of the two reference methods and the two 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 two 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 [20]. 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 [21]. In the following, some examples from the heterogeneous change detection literature are presented.

2) Applications: Su *et al.* [22] used change vector analysis to distinguish between three classes: unchanged areas, positive changes and negative changes, as defined in [23]. 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.* [24] 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.* [16], 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 [25] 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 [26], 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 [27] 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 [28] 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 [29]. 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 [30], 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 [13]. 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

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

to map image patches $\boldsymbol{X} \in \mathcal{X}^{h\times w} \subseteq \mathcal{I}_\mathcal{X}$ and $\boldsymbol{Y} \in \mathcal{Y}^{h\times w} \subseteq \mathcal{I}_\mathcal{Y}$ between the image domains by using the entire images $\mathcal{I}_\mathcal{X}$ and $\mathcal{I}_\mathcal{Y}$ 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 pixels based on affinity matrices, we present an improved version of the original method proposed in our previous work [15]. A $k \times k$ sliding window covers an area p of both $\mathcal{I}_\mathcal{X}$ and $\mathcal{I}_\mathcal{Y}$, from which a pair of corresponding patches $\boldsymbol{X}$ and $\boldsymbol{Y}$ are extracted. $\boldsymbol{X}_i$ and $\boldsymbol{Y}_j$ stand for pixel i and j of patch $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively, with $i, j \in \{1, \dots, k^2\}$. The distance between a pixel pair (i, j) is defined as $d^m_{i,j}$, where the modality $m \in \{\mathcal{X}, \mathcal{Y}\}$ depends on whether the pixels are taken from $\boldsymbol{X}$ or $\boldsymbol{Y}$. 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 [23], [31]. Concerning SAR intensity data, a logarithmic transformation is sufficient to bring it to near-Gaussianity [2], [16]. We use the computationally efficient Euclidean distance, as it is suitable for (nearly) Gaussian data.

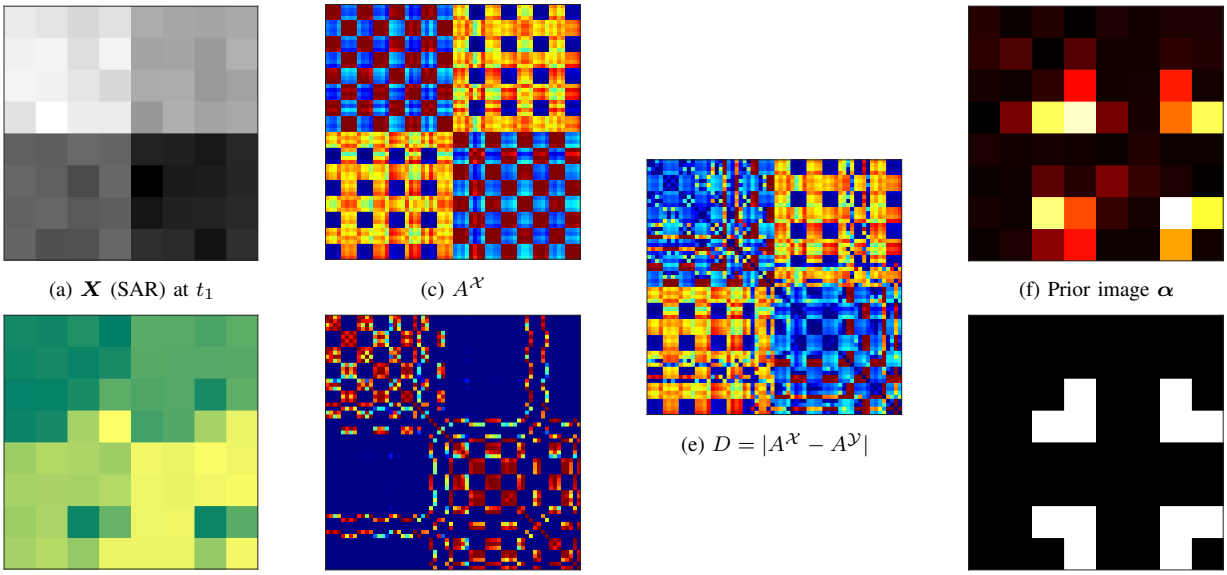
Once computed, the distances between all pixel pairs can be converted to affinities, for instance by the Gaussian kernel:

$$A^m_{i,j} = \exp\left\{-\frac{\left(d^m_{i,j}\right)^2}{h^2_m}\right\} \in (0, 1], \quad i, j \in \{1, \dots, k^2\}. \quad (1)$$

$A^m_{i,j}$ are the entries of the affinity matrix $A^m \in \mathbb{R}^{k^2\times k^2}$ for the given patch and modality m. 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 K^{th} nearest neighbour for all data points in the relevant patch ($\boldsymbol{X}$ or $\boldsymbol{Y}$), 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 [32]. Silverman's rule of thumb [33] 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 $D_{i,j} = |A^\mathcal{X}_{i,j} - A^\mathcal{Y}_{i,j}|$ can be obtained.

Our previous algorithm [15] 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

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



(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) CD 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 Eq. (3); g) CD map obtained by thresholding $\boldsymbol{\alpha}$. Best viewed in colour.

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):

$$\alpha_i = \frac{1}{k^2} \sum_{j=1}^{k^2} |A^{\mathcal{X}}_{i,j} - A^{\mathcal{Y}}_{i,j}|, \quad i \in \{1, \dots, k^2\} \tag{3}$$

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 $\boldsymbol{X}$ 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 $\boldsymbol{Y}$ 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 $\boldsymbol{Y}$. In this way, all the possible transitions between one class and the others occur between t_1 and t_2. The 64×64 affinity matrices $A^{\mathcal{X}}$ and $A^{\mathcal{Y}}$ computed from $\boldsymbol{X}$ and $\boldsymbol{Y}$ 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 $\boldsymbol{X}$ and $\boldsymbol{Y}$. 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 Eq. (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 $\boldsymbol{\alpha}$, as shown in Fig. 1g.

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

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

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

$p_\ell \in \mathcal{P}$ covering pixel i. If Δ is a factor of k, this average is calculated over $(k/\Delta)^2$ 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 $d^m_{i,j}$ and then $A^m_{i,j}$ 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

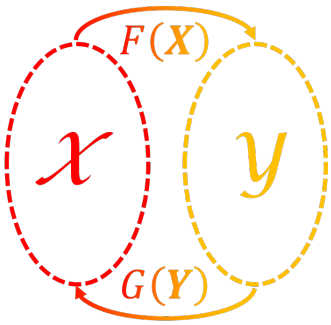


Fig. 2: First proposed framework, where two domains and two transformations which can translate data across them.

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.

In the loss terms we will need to compute distances between patches, where input patches are compared with translated ones. We therefore define a general weighted distance between two equal-sized $h \times w$ patches $\boldsymbol{A}$ and $\boldsymbol{B}$ as $\delta(\boldsymbol{A}, \boldsymbol{B}|\boldsymbol{\pi})$, where $\boldsymbol{\pi}$ is a vector of weights, each associated with a pixel $i \in \{1, \dots, n\}$ of the patches, with $n = h \cdot w$. In this work we use the mean squared L_2 norm as a particular choice of $\delta(\cdot)$, which is allowed since the pixel measurements $\boldsymbol{a}_i \in \boldsymbol{A}$ and $\boldsymbol{b}_i \in \boldsymbol{B}$ in our datasets are vectors. This means that

$$\delta(\boldsymbol{A}, \boldsymbol{B}|\boldsymbol{\pi}) = \frac{1}{n}\sum_{i=1}^{n} \pi_i \|\boldsymbol{a}_i - \boldsymbol{b}_i\|_2^2 \,. \tag{4}$$

When no weights are applied ($\boldsymbol{\pi} = \mathbf{1}$, where $\mathbf{1}$ denotes a vector of ones), the patch distance is written as $\delta(\boldsymbol{A}, \boldsymbol{B}|\mathbf{1}) = \delta(\boldsymbol{A}, \boldsymbol{B})$.

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

$$\begin{aligned} \hat{\boldsymbol{Y}} &= F(\boldsymbol{X}) \simeq \boldsymbol{Y} \,, \\ \hat{\boldsymbol{X}} &= G(\boldsymbol{Y}) \simeq \boldsymbol{X} \,, \end{aligned} \tag{5}$$

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. As outlined above, every pixel pair $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}$ will be associated with a precomputed prior, α_i, that indicates its probability of being changed. Hence, the weighted translation loss is defined as:

$$\mathcal{L}_\alpha(\vartheta) = \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[\delta(\hat{\boldsymbol{X}}, \boldsymbol{X}|\boldsymbol{\Pi}))\right] + \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[\delta(\hat{\boldsymbol{Y}}, \boldsymbol{Y}|\boldsymbol{\Pi})\right] \,, \tag{6}$$

where $\boldsymbol{\Pi} = [\Pi(\alpha_1), \dots, \Pi(\alpha_n)]^T$, and $\Pi(\alpha) : [0, 1] \to [0, 1]$ is a monotonically decreasing function that maps the $\{\alpha_i\}$, indicating probability of change, into $\{\Pi(\alpha_i)\}$, that are used to weight the pixels' contribution to the loss function. In this way, we use the precomputed priors obtained from Section III-A to drive the learning process and penalise the contribution of pixels most likely to be affected by changes. 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 [29], 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

$$\begin{aligned} \dot{\boldsymbol{X}} &= G(\hat{\boldsymbol{Y}}) = G\left(F(\boldsymbol{X})\right) \simeq \boldsymbol{X} \,, \\ \dot{\boldsymbol{Y}} &= F(\hat{\boldsymbol{X}}) = F\left(G(\boldsymbol{Y})\right) \simeq \boldsymbol{Y} \,, \end{aligned} \tag{7}$$

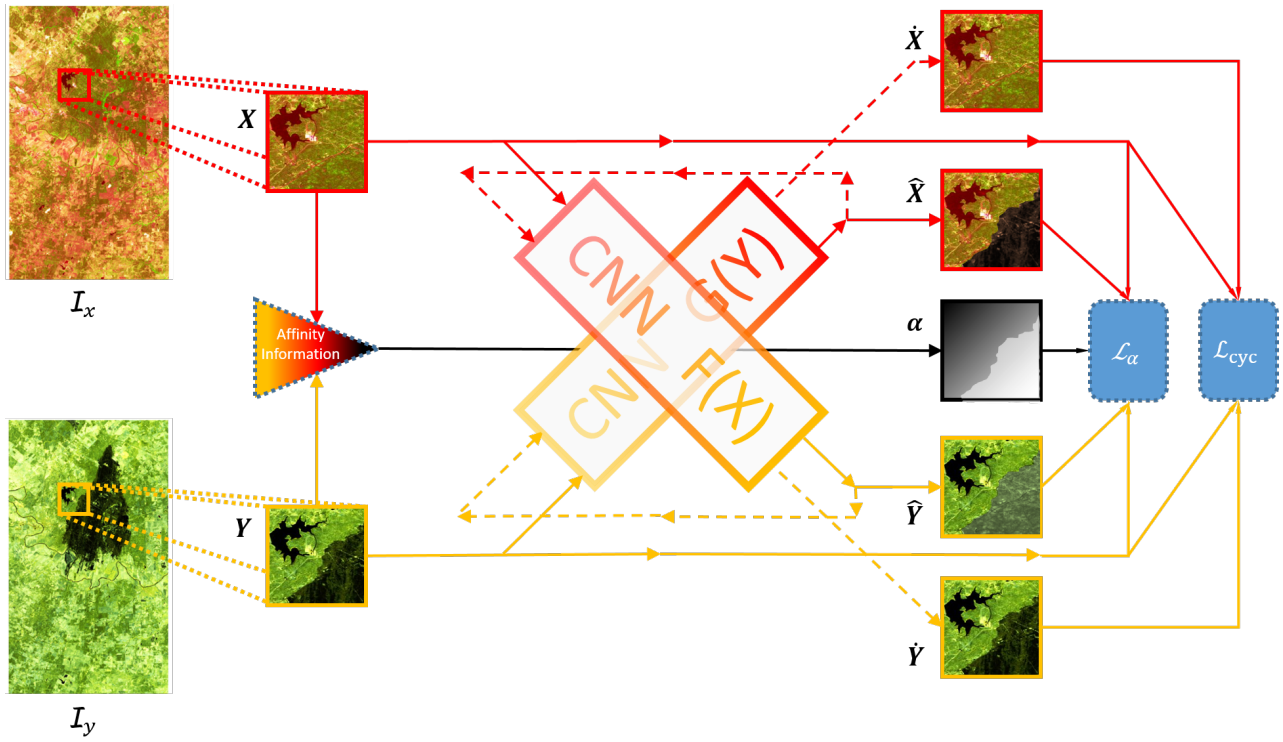


Fig. 3: Data flow of the X-Net. Two CNNs transform data from the domain of $\boldsymbol{X}$ to the domain of $\boldsymbol{Y}$ and vice versa. Solid lines going through them indicate data transferred from one domain to the other, dashed lines indicate data re-transformed back to their original domain.

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:

$$\mathcal{L}_{\text{cyc}}(\vartheta) = \mathbb{E}_{\boldsymbol{X}}\left[\delta(\dot{\boldsymbol{X}}, \boldsymbol{X})\right] + \mathbb{E}_{\boldsymbol{Y}}\left[\delta(\dot{\boldsymbol{Y}}, \boldsymbol{Y})\right], \quad (8)$$

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

$$\mathcal{L}(\vartheta) = \left\{ w_{\text{cyc}}\mathcal{L}_{\text{cyc}}(\vartheta) + w_\alpha \mathcal{L}_\alpha(\vartheta) + w_\vartheta \left\|\vartheta\right\|_2^2 \right\}. \quad (9)$$

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.* [30], 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

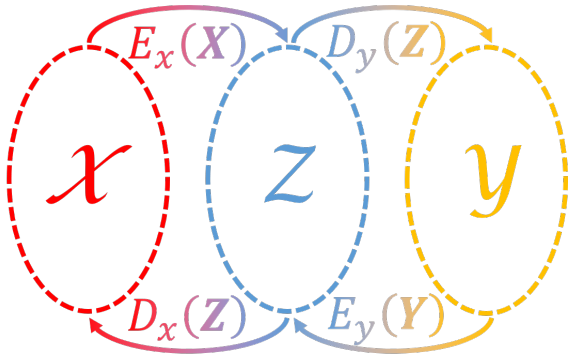


Fig. 4: Second proposed framework: a latent space $\mathcal{Z}$ is introduced between domains $\mathcal{X}$ and $\mathcal{Y}$, and four regression functions mapping data across them. In this case, $F(\boldsymbol{X}) = D_{\mathcal{Y}}\left(E_{\mathcal{X}}\left(\boldsymbol{X}\right)\right)$ and $G(\boldsymbol{Y}) = D_{\mathcal{X}}\left(E_{\mathcal{Y}}\left(\boldsymbol{Y}\right)\right)$.

consists of five CNNs. The first four networks are image regression functions (see Fig. 4): Encoders $E_{\mathcal{X}}\left(\boldsymbol{X}\right) : \mathcal{X}^{h\times w}$ and $E_{\mathcal{Y}}\left(\boldsymbol{Y}\right) : \mathcal{Y}^{h\times w}$ 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 $D_{\mathcal{X}}\left(\boldsymbol{Z}\right) : \mathcal{Z}^{h\times w} \rightarrow \mathcal{X}^{h\times w}$ and $D_{\mathcal{Y}}\left(\boldsymbol{Z}\right) : \mathcal{Z}^{h\times w} \rightarrow \mathcal{Y}^{h\times w}$ map latent space data back into their original domains. The fifth network is a discriminator, which is described later.

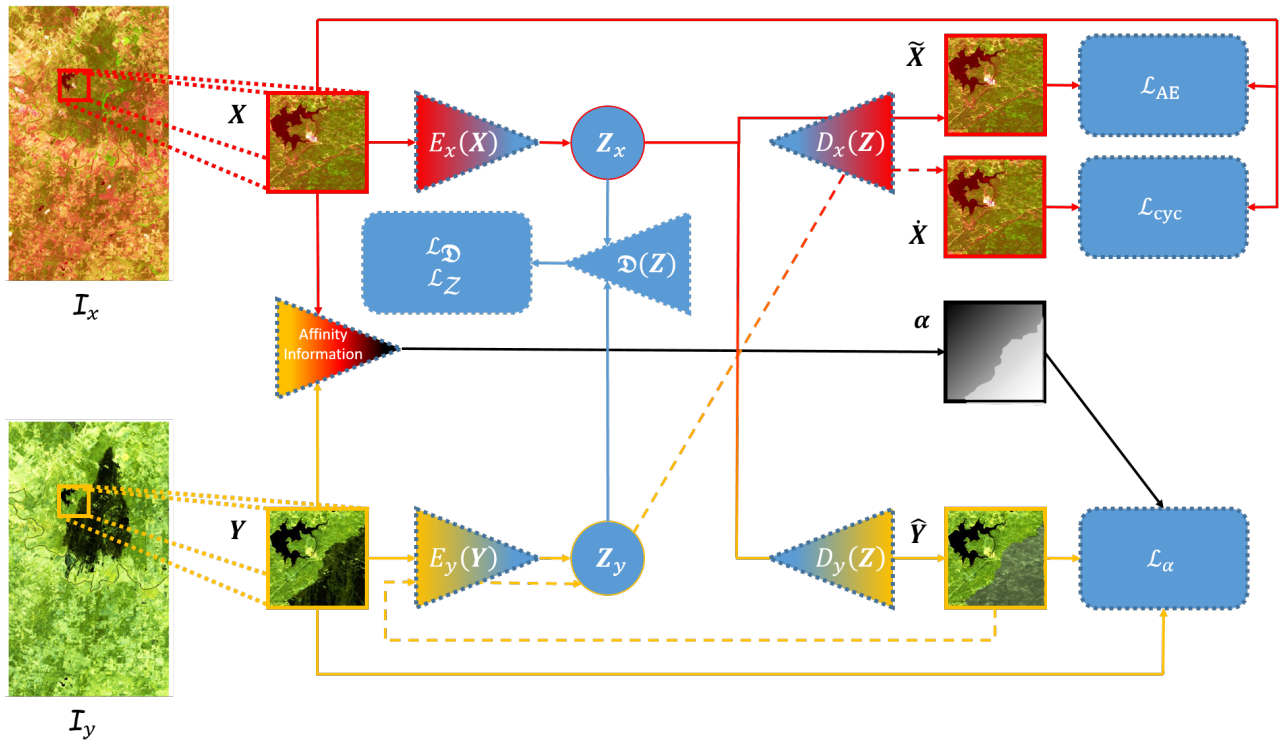


Fig. 5: Data flow of the ACE-Net. The encoders $E_{\mathcal{X}}(\boldsymbol{X})$ and $E_{\mathcal{Y}}(\boldsymbol{Y})$ transform incompatible data into two code spaces, which are aligned by adversarial training against the discriminator $\mathfrak{D}(\boldsymbol{Z})$. The decoders $D_{\mathcal{X}}(\boldsymbol{Z})$ 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.

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,

$$\begin{aligned}
\hat{\boldsymbol{X}} &= G(\boldsymbol{Y}) = D_{\mathcal{X}}(E_{\mathcal{Y}}(\boldsymbol{Y}))\,, \\
\hat{\boldsymbol{Y}} &= F(\boldsymbol{X}) = D_{\mathcal{Y}}(E_{\mathcal{X}}(\boldsymbol{X}))\,, \\
\dot{\boldsymbol{X}} &= G(\hat{\boldsymbol{Y}}) = D_{\mathcal{X}}(E_{\mathcal{Y}}(D_{\mathcal{Y}}(E_{\mathcal{X}}(\boldsymbol{X}))))\,, \\
\dot{\boldsymbol{Y}} &= F(\hat{\boldsymbol{X}}) = D_{\mathcal{Y}}(E_{\mathcal{X}}(D_{\mathcal{X}}(E_{\mathcal{Y}}(\boldsymbol{Y}))))\,.
\end{aligned} \tag{10}$$

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:

$$\begin{aligned}
\tilde{\boldsymbol{X}} &= D_{\mathcal{X}}(E_{\mathcal{X}}(\boldsymbol{X})) \simeq \boldsymbol{X}\,, \\
\tilde{\boldsymbol{Y}} &= D_{\mathcal{Y}}(E_{\mathcal{Y}}(\boldsymbol{Y})) \simeq \boldsymbol{Y}\,.
\end{aligned} \tag{11}$$

Consequently, we introduce the reconstruction loss term:

$$\mathcal{L}_{\mathrm{AE}}(\vartheta_{\mathrm{AE}}) = \mathbb{E}_{\boldsymbol{X}}\left[\delta(\tilde{\boldsymbol{X}}, \boldsymbol{X})\right] + \mathbb{E}_{\boldsymbol{Y}}\left[\delta(\tilde{\boldsymbol{Y}}, \boldsymbol{Y})\right]\,, \tag{12}$$

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 $\mathfrak{D}(\boldsymbol{Z}) : \mathcal{Z}^{h\times w} \to [0, 1]$ 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:

$$\mathcal{L}_{\mathfrak{D}}(\vartheta_{\mathfrak{D}}) = \mathbb{E}_{\boldsymbol{X}}\Big[\big(\mathfrak{D}(E_{\mathcal{X}}(\boldsymbol{X}))-1\big)^2\Big] + \mathbb{E}_{\boldsymbol{Y}}\Big[\mathfrak{D}(E_{\mathcal{Y}}(\boldsymbol{Y}))^2\Big] \tag{13}$$

$$\mathcal{L}_{\mathcal{Z}}(\vartheta_E) = \mathbb{E}_{\boldsymbol{X}}\Big[\mathfrak{D}(E_{\mathcal{X}}(\boldsymbol{X}))^2\Big] + \mathbb{E}_{\boldsymbol{Y}}\Big[\big(\mathfrak{D}(E_{\mathcal{Y}}(\boldsymbol{Y}))-1\big)^2\Big] \tag{14}$$

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 Eq. (13) and (14), the two generators and the discriminator aim at the opposite goal and, therefore, have opposite loss terms. As in [29], 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 [13]. 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:

$$\begin{aligned}\mathcal{L}(\vartheta) =& w_{\text{adv}} \left[\mathcal{L}_{\mathcal{Z}}\left(\vartheta_E\right) + \mathcal{L}_{\mathfrak{D}}\left(\vartheta_{\mathfrak{D}}\right)\right] + \\ & w_{\text{AE}}\mathcal{L}_{\text{AE}}\left(\vartheta_{\text{AE}}\right) + w_{\text{cyc}}\mathcal{L}_{\text{cyc}}(\vartheta_{\text{AE}}) + \\ & w_{\alpha}\mathcal{L}_{\alpha}(\vartheta_{\text{AE}}) + w_{\vartheta}\left\|\vartheta\right\|_2^2 .\end{aligned} \tag{15}$$

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 $\mathfrak{D}\left(\boldsymbol{Z}\right)$ takes as input $E_{\mathcal{X}}\left(\boldsymbol{X}\right)$ and $E_{\mathcal{Y}}\left(\boldsymbol{Y}\right)$ and tries to tell them apart.

D. Change extraction

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 $i \in \{1, \dots, N\}$, 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 [34] 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 [15]: 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 [35], [36], [37], [38] are able to set the threshold automatically. Among these, we selected the well known Otsu's method [35].

IV. Experimental results

In Section IV-A we first provide the details of our implementation of the various methods. The two datasets used in this work are presented in Section IV-B. 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 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. Along with the mean elapsed times, this section reports Cohen's Kappa Coefficient κ [39].

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. Network configurations

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 [40] 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 [41]. Batch normalisation [42] turned out to be unnecessary and was discarded, as it did not improve the optimisation and it actually slowed down our experiments.

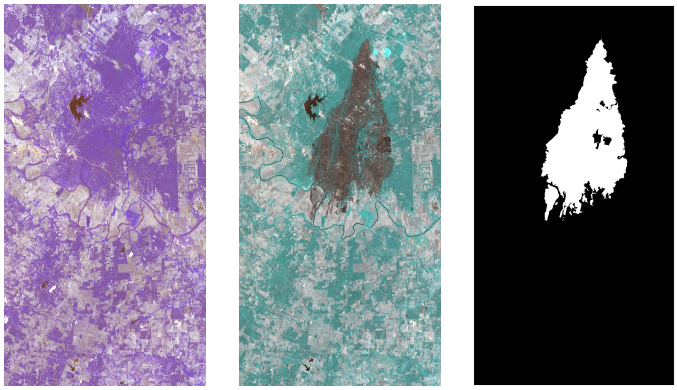
After each layer, dropout is applied with a dropout rate of 20% during the training phase to enhance the robustness of the framework against overfitting and input noise [43]. Also, data augmentation helps increasing the size of the training sample by introducing some more variety in the data: Before feeding the patches to the network, these were randomly flipped and rotated.

The weights in $\boldsymbol{\vartheta}$ were initialised with a truncated normal distribution according to [44] 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 [45] 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.

2) SCCN and CAN: 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, 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.

B. 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.* [46] provided the ground truth shown in Fig. 6c.

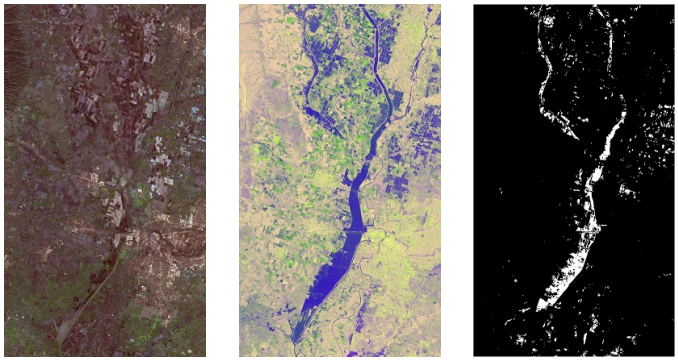


(a) Landsat 5 (t_1) (b) EO-1 ALI (t_2) (c) Ground Truth

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

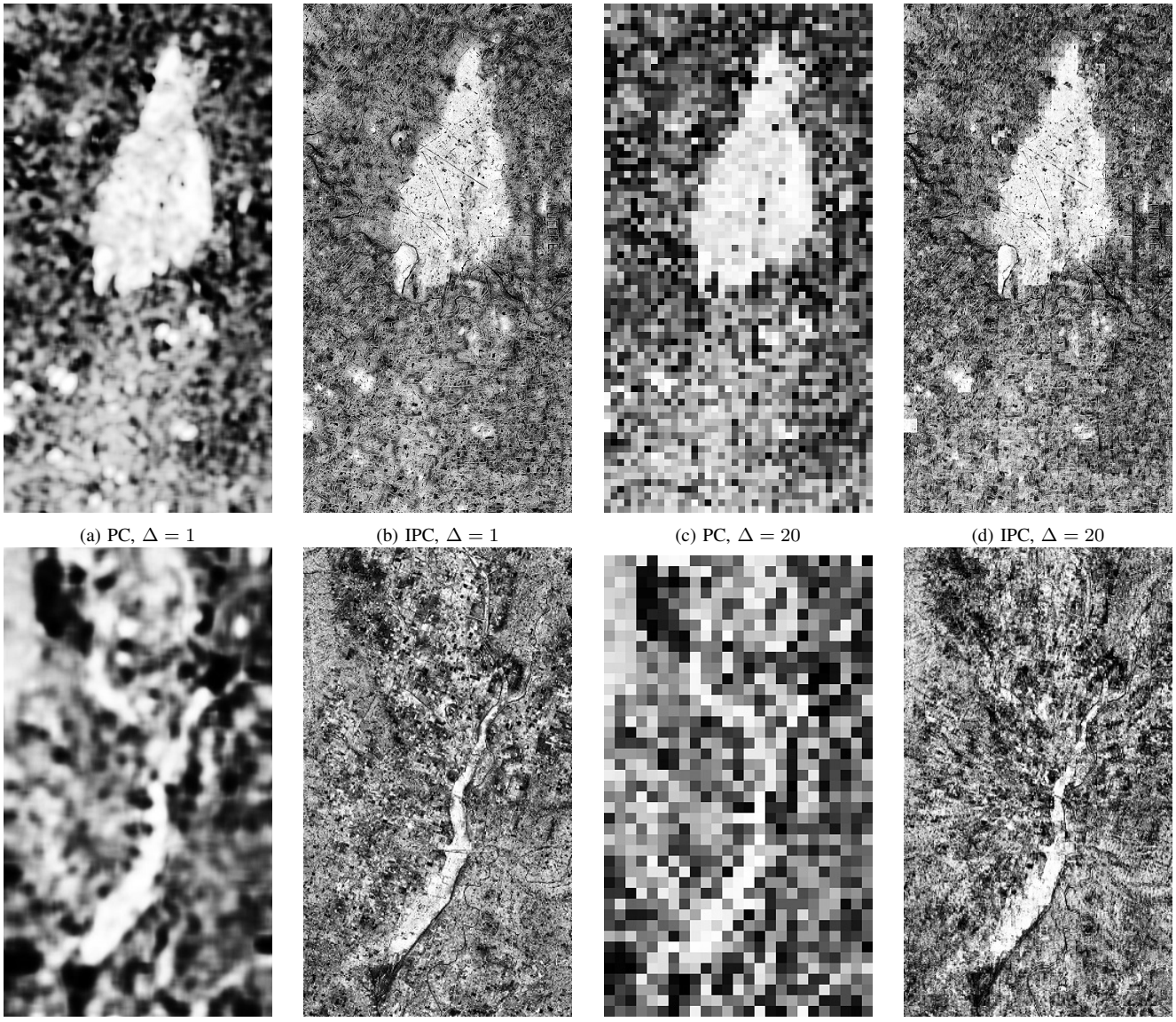
2) Flood in California: Fig. 7a displays the RGB channels of a Landsat 8 acquisition[1] covering Sacramento County, Yuba County and Sutter County, California, on 5 January 2017. The OLI and TIRS sensors on Landsat 8 together acquire data in 11 channels, from deep blue up to thermal infrared. The same 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 to the networks. The ground truth in Fig. 7c is provided by Luppino *et al.* [15]. Originally of 3500×2000 pixels, these images were resampled to 850×500 pixels to reduce the computation time.

[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.



(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, $\Delta = 20$

Fig. 8: Results on the two datasets for the PC and the IPC, for $\Delta = 1$ and for $\Delta = 20$.

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 [15], a patch size of $k = 20$ was selected for all the experiments. Fig. 8 shows the outcomes for the two 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 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. 8.

Table I reports an approximate total number of patches $|\mathcal{P}|$ and the computation time spent by the two methods on the two 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,

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

	$\Delta = 1$		$\Delta = 20$	
	Texas	California	Texas	California
$\lvert\mathcal{P}\rvert$	1.2×10^6	4×10^5	3×10^3	1×10^3
PC	45 min	15 min	2:37 min	0:37 min
IPC	76 min	24 min	6 min	1:45 min

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 and 13 min for the Texas and California datasets, respectively.

D. Results

Each of the four architectures was initialised randomly and trained for 100 independent runs, and their metrics are reported in the form of boxplots. These plots represent the behaviour of κ for the compared methods: a box covers the values from the 25^{th} percentile to the 75^{th} with an orange line showing the median, while whiskers indicate the span between the 5^{th} and the 95^{th} percentile. Outliers beyond the whiskers are marked as circles. As a reference, the κ achieved by directly filtering and thresholding the prior $\boldsymbol{\alpha}$ is indicated by a red horizontal line.

Table II contains the average times spent to train the four methods on the two datasets. 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.

In Fig. 9, the results of the four methods on the Texas dataset are compared. The X-Net and the CAN show stable and consistent performance. However, only the former achieves better results than the filtered and segmented IPC, which produces $\kappa = 0.65$. The ACE-Net and the SCCN sometimes reach higher values of κ than the X-net, but their median κ is lower and the variance is high. When compared to the IPC reference, the ACE-Net exceeds it performance in 75% of the test runs, and the SCCN only in 50%.

A different scenario was found for the California dataset, as depicted in Fig. 10. The methods perform similarly and their metrics reach consistently above the reference $\kappa = 0.2$, which is the reference value produced by the IPC. The ACE-Net outperforms the X-Net and the CAN in terms of median 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 the following section.

Fig. 11 and Fig. 12 show examples of the best output delivered by each of the four methods on the two 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 $\hat{\boldsymbol{Y}}$ and a approximated image $\tilde{\boldsymbol{Y}}$ in the $\mathcal{Y}$ domain. For the SCCN algorithm, these are code images $\boldsymbol{Z}_{\mathcal{X}}$ and $\boldsymbol{Z}_{\mathcal{Y}}$ from a common latent space.

V. Discussion

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

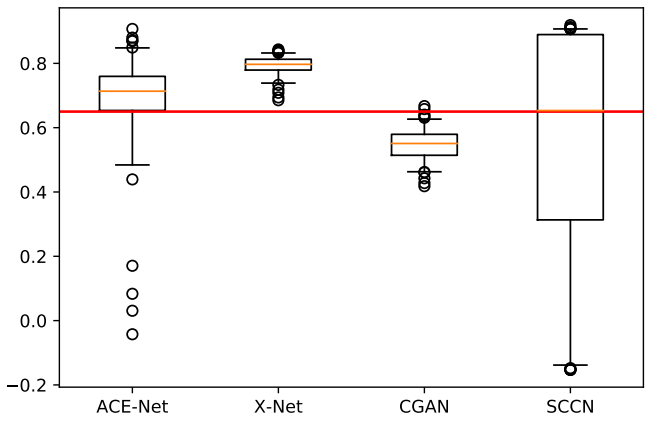


Fig. 9: Boxplots of the κ coefficient for the four methods applied to the Texas dataset. The red horizontal line shows the κ achieved with the affinity matrices comparison.

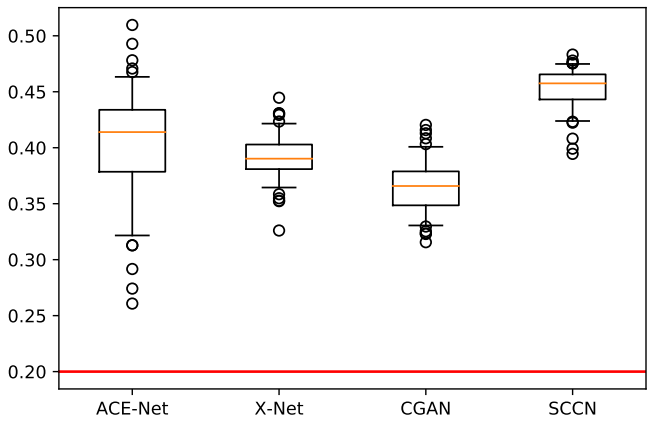


Fig. 10: Boxplots of the κ coefficient for the four methods applied to the California dataset. The red horizontal line shows the κ achieved with the affinity matrices comparison.

TABLE II: Average training time of the four methods on the two datasets.

	ACE-Net	X-Net	CAN	SCCN
Texas	12:52 min	6:46 min	1:09:05 h	15:39 min
California	12:12 min	5:42 min	21:26 min	14:34 min

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. 11s and Fig. 12s.

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 two 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 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 Fig. 9). The very good results displayed in Fig. 10 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 $\boldsymbol{Y}$ 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. 12e that the encoder is not able to capture more than the background average colour of Fig. 12j, which can be characterized as degenerate behaviour. Basically, the difference image in Fig. 12o 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. 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 California dataset, 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.

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 two 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.

VII. Acknowledgement

The project and the first author was funded by the Research Council of Norway under research grant no. 251327. We gratefully acknowledge the support of NVIDIA Corporation by the donation of the GPU used for this research. The authors thank Devis Tuia for valuable discussions.

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.

[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.

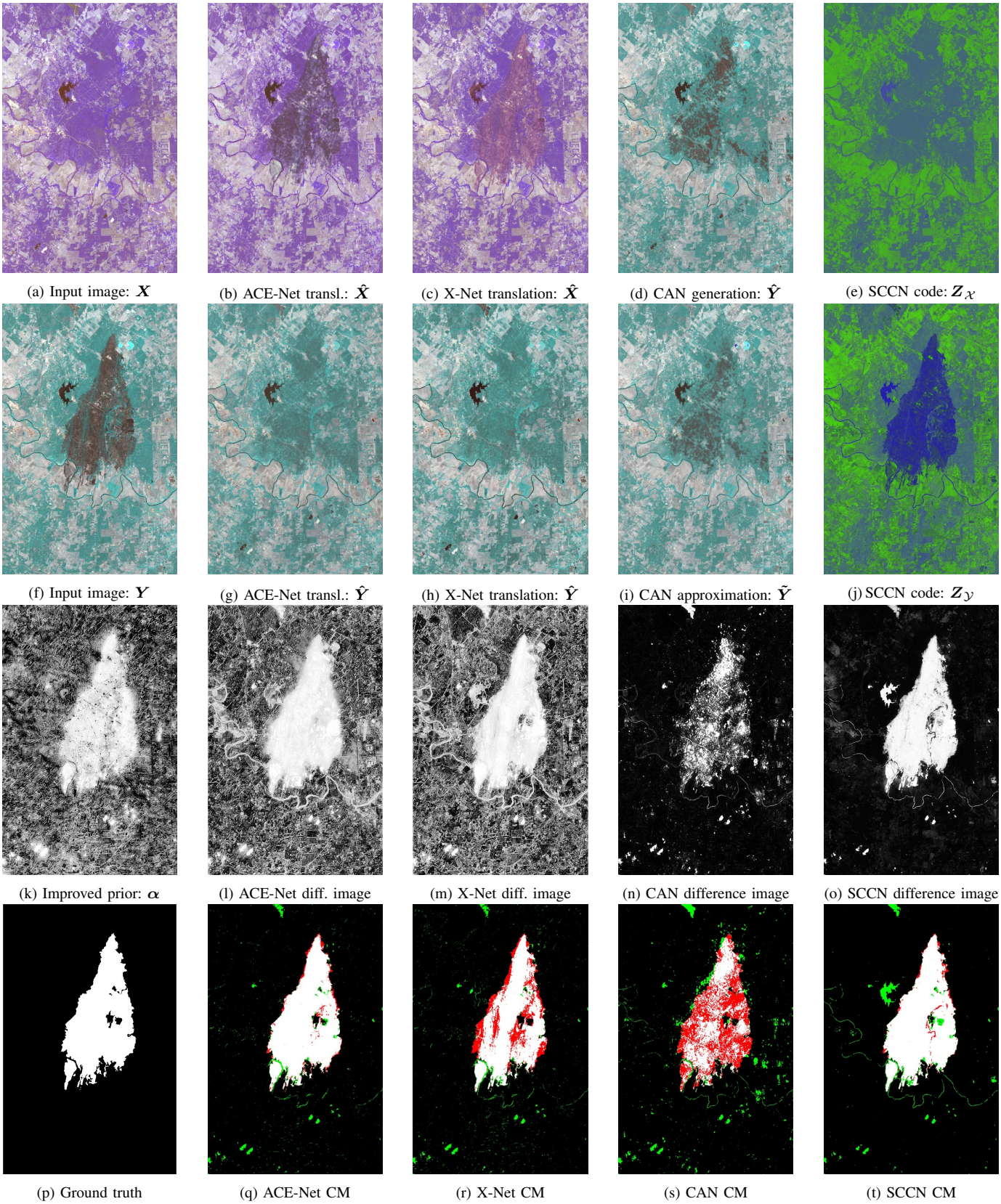
[9] M. Gong, J. Zhao, J. Liu, Q. Miao, and L. Jiao, "Change detection in synthetic aperture radar images based on deep neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 1, pp. 125–138, 2016.

[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 Sensing*, 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. 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.



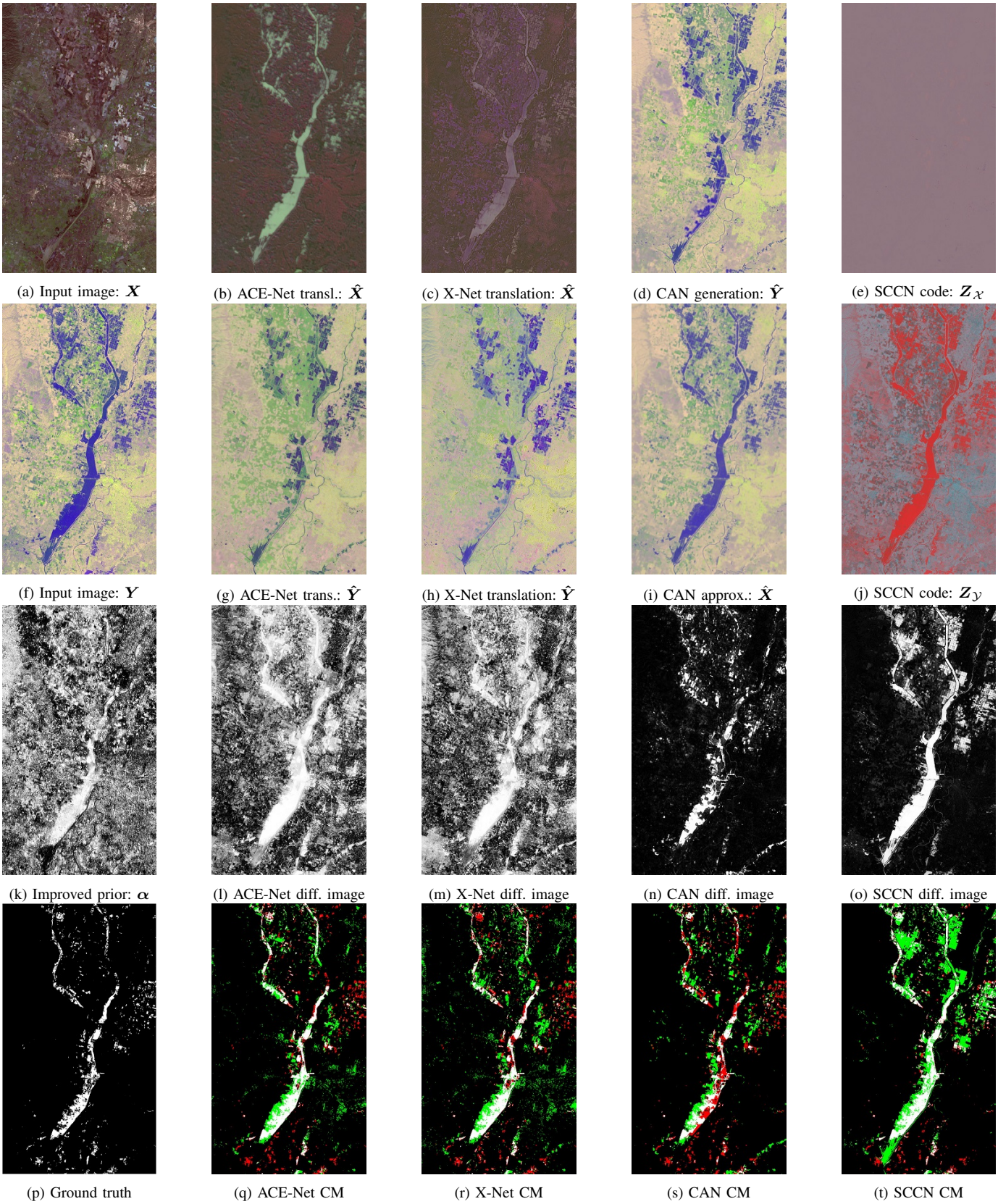
(a) Input image: $\boldsymbol{X}$ (b) ACE-Net transl.: $\hat{\boldsymbol{X}}$ (c) X-Net translation: $\hat{\boldsymbol{X}}$ (d) CAN generation: $\hat{\boldsymbol{Y}}$ (e) SCCN code: $\boldsymbol{Z}_{\mathcal{X}}$

(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. 11: 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. 12: 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).

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

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

[21] 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. Dec, pp. 3371–3408, 2010.

[22] 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.

[23] 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.

[24] 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, 06 2016.

[25] 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.*, 2017.

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

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

[28] 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.

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

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

[31] 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.

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

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

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

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

[36] 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.

[37] 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.

[38] 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.

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

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

[41] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 9–48.

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

[43] 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.

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

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

[46] 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.

Paper III

Code-Aligned Autoencoders for Unsupervised Change Detection in Multimodal Remote Sensing Images

Luigi T. Luppino, Mads A. Hansen, Michael Kampffmeyer,
Filippo M. Bianchi, Gabriele Moser, Robert Jenssen, and Stian Normann Anfinsen

***Abstract*—Image translation with convolutional autoencoders has recently been used as an approach to multimodal change detection in bitemporal satellite images. A main challenge is the alignment of the code spaces by reducing the contribution of change pixels to the learning of the translation function. Many existing approaches train the networks by exploiting supervised information of the change areas, which, however, is not always available. We propose to extract relational pixel information captured by domain-specific affinity matrices at the input and use this to enforce alignment of the code spaces and reduce the impact of change pixels on the learning objective. A change prior is derived in an unsupervised fashion from pixel pair affinities that are comparable across domains. To achieve code space alignment we enforce that pixel with similar affinity relations in the input domains should be correlated also in code space. We demonstrate the utility of this procedure in combination with cycle consistency. The proposed approach are compared with state-of-the-art deep learning algorithms. Experiments conducted on four real datasets show the effectiveness of our methodology.**

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

I. Introduction

CHANGE detection (CD) methods in remote sensing aim at identifying changes happening on the Earth by comparing two or more images acquired at different times [1]. Multitemporal analyses with satellite data include land use mapping of urban and agricultural areas [2], [3], and monitoring of large scale changes such as deforestation [4], lake and glacier reduction [5], [6], urbanisation [7], etc. Bitemporal applications mainly concerned with the detection and assessment of natural disasters and sudden events, like earthquakes [8], floods [9], forest fires [10], and so forth.

Traditional CD methods rely on homogeneous data, namely a set of images acquired by the same sensor, under the same geometry, seasonal conditions, and recording settings. However, these restrictions are too strong for many practical examples. First of all, the satellite revisit period sets the upper limit to the temporal resolution when monitoring long-term trends, and the lower limit to the response time when assessing the damages of sudden events. Moreover, even when two images are collected with the same configurations, they might be not homogeneous because of other factors, for example light conditions for optical data or humidity and precipitation for synthetic aperture radar (SAR).

Heterogeneous CD methods overcome these limitations, but at the cost of having to handle more complicated issues; Heterogeneous data imply different domains, diverse statistical distributions and inconsistent class signatures across the two images, especially when different sensors are involved, which makes a direct comparison infeasible [11]. These problems have been tackled by use of many different techniques: copula theory [1], marginal densities transformations [12], evidence theory [13], [14], graph theory [15], manifold learning [16], kernelised or deep canonical correlation analysis [10], [17], [18], dictionary learning [19], scale-invariant local descriptors [20], [21], superpixel segmentation [22], clustering [23], minimum energy [24], multidimensional scaling [25], nonlinear regression [26], [9], and deep learning (especially autoencoders) [27], [28], [29], [30], [31], [32].

A common solution in heterogeneous CD is to apply highly nonlinear transformations to transfer the data from one domain to the other and vice versa [30], [33], [34]. Alternatively, all the data are mapped to a common domain where they can be compared [12], [27], [28], [32]. Nonetheless, this crucial step often requires iterative fine-tuning of the transformation functions starting from unreliable preliminary results, e.g. random initialisation [28], [32] and clustering [30], or from manually selected training samples [1], [10], [16] that are not always available.

One contemporary way to map data across two domains is image-to-image (I2I) translation using a conditional generative adversarial network (cGAN) [35], which was extended by enforcing cyclic consistency in the cycleGAN architecture [36]. These approaches have inspired many recent heterogeneous CD methods [33], [34], [37]. A notable difference between the cGAN and the cycleGAN is that training of the former requires paired images that contain the same objects imaged with different styles or sensor modes, whereas the cycleGAN does not. Paired I2I translation can only be applied in heterogeneous CD if change pixels are censored, as these will otherwise distort the training process and promote a transformation between different objects.

When generative adversarial frameworks are used in heterogeneous CD, the translated (or cyclically translated) images take the role as fake or generated data, and the network is

L.T. Luppino, M.A. Hansen, M. Kampffmeyer, R. Jenssen and S.N. Anfinsen are with the Machine Learning Group, Department of Physics and Technology, UiT The Arctic University of Norway, e-mail: luigi.t.luppino@uit.no.
F.M. Bianchi is with NORCE Norwegian Research Center, Norway.
G. Moser is with DITEN Department, University of Genoa, Italy.
Manuscript received -; revised -.

trained to make them indistinguishable from true images from the relevant domain. The cGAN and cycleGAN may succeed to align the distributions of translated data and true data, but they are also seen to suffer from inherent drawbacks: They rely on large training sets, the iterative training of generator and discriminator must be judiciously balanced, training is prone to mode collapse, and reasonable values of the hyperparameters can be difficult to find due to oscillating and unstable behaviour of the loss function. We therefore seek alternative training strategies to the adversarial ones.

In this work, we propose a simple unsupervised, heterogeneous CD method, inspired by the paradigm of I2I translation. The idea is to align the code layers of two autoencoders and treat them as a common latent space, so that the output of one encoder can be the input of both decoders, leading in one case to reconstruction of data in their original domain, and in the other case to their transformation into the other domain. Local information extracted directly from the input images is exploited to drive the code alignment in an unsupervised manner. Specifically, affinity matrices of the training patches are computed and compared, and the extracted information is used to ensure that pixel pairs that are similar in both input domains also have a high correlation in the common latent space. The implementation of this principle is inspired by the deep kernelised autoencoder of Kampffmeyer et al. [38], [39], where the inner product between the codes produced by two datapoints is forced to match their precomputed affinity.

To summarise, the contributions of this work are the following:

- We propose a simple, yet effective loss term, able to align the latent spaces of two autoencoders in an unsupervised manner.
- We implement a deep neural network for heterogeneous CD that incorporates this loss term.
- The well-documented TensorFlow 2.0 framework that we provide can be easily used for the development of other CD methods and for direct comparison with ours. Source code is made available at https://github.com/llu025/Heterogeneous_CD.

The remainder of this paper is organised as follows: The core ideas and the main contribution are presented in Sec. II; Experiments were conducted on four different real datasets, and Sec. III shows the results of the proposed approach against several state-of-the-art methods; Sec. IV concludes the paper.

II. Methodology

Assume that we have two different sensors (or sensor modes) whose single-pixel measurements lie in the domains $\mathcal{X}$ and $\mathcal{Y}$. These could be e.g. $\mathbb{R}_{\geq 0}$ (nonnegative real numbers) for a single-channel SAR sensor, $\mathbb{R}^C_{\geq 0}$ for a multispectral radiometer with C bands, or $\mathbb{C}^{C\times C}_{\succeq 0}$ for a polarimetric SAR sensor with C polarisations that records a complex and semipositive definite covariance matrix for each pixel.

Further assume that these sensors are scanning the same geographical region at separate times and we obtain an image $\mathcal{I}_\mathcal{X} \in \mathcal{X}^{H\times W}$ recorded at time t_1 and an image $\mathcal{I}_\mathcal{Y} \in \mathcal{Y}^{H\times W}$ recorded at $t_2 > t_1$. The images and their domains have common dimensions, the shared height H and width W, which are obtained after coregistration and resampling. They will in general have different numbers of channels, denoted as $|\mathcal{X}|$ and $|\mathcal{Y}|$. The two images can be thought of as realisations of stochastic processes that generate data tensors from domain $\mathcal{X}$ and $\mathcal{Y}$.

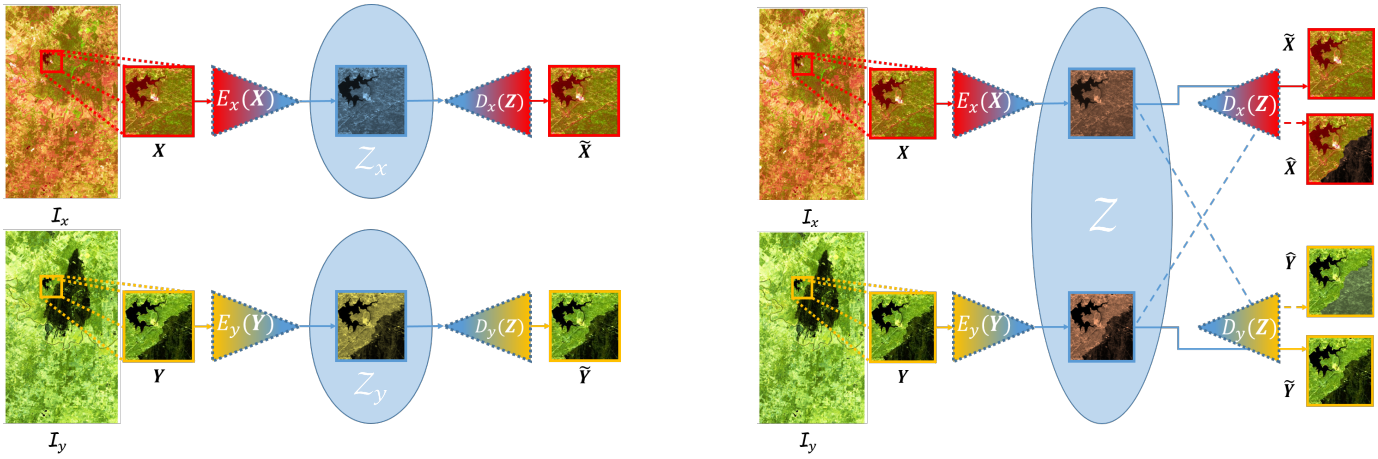
An underlying assumption is that a limited part of the image has changed between t_1 and t_2. The final goal is to detect all changes in the scene. However, given the heterogeneity of $\mathcal{X}$ and $\mathcal{Y}$, direct comparison is meaningless, if not unfeasible, without any preprocessing step. Let $\boldsymbol{X} \in \mathcal{X}^{h\times w}$ and $\boldsymbol{Y} \in \mathcal{Y}^{h\times w}$ be data tensors holding size $h \times w$ patches of the full images $\mathcal{I}_\mathcal{X}$ and $\mathcal{I}_\mathcal{Y}$. We are interested in implementing the two transformations: $\hat{\boldsymbol{Y}} = F(\boldsymbol{X})$ and $\hat{\boldsymbol{X}} = G(\boldsymbol{Y})$, defined as $F : \mathcal{X}^{h\times w} \to \mathcal{Y}^{h\times w}$ and $G : \mathcal{Y}^{h\times w} \to \mathcal{X}^{h\times w}$, to map data between the image domains. In this way, the input images can be transferred to the opposite domain, and the changes can be detected by computing the difference image as the weighted average:

$$\boldsymbol{\Delta} = W_\mathcal{X} \cdot d^\mathcal{X}(\boldsymbol{X}, \hat{\boldsymbol{X}}) + W_\mathcal{Y} \cdot d^\mathcal{Y}(\boldsymbol{Y}, \hat{\boldsymbol{Y}}), \tag{1}$$

where $d^\mathcal{X}(\cdot,\cdot)$ and $d^\mathcal{Y}(\cdot,\cdot)$ are sensor-specific distances, chosen according to the statistical distribution of the data, which operate pixel-wise. The generic weights $W_\mathcal{X}$ and $W_\mathcal{X}$ can be used to balance the contribution of the domain-specific distances. We may want to use $W_\mathcal{X} = 1/|\mathcal{X}|$ and $W_\mathcal{Y} = 1/|\mathcal{Y}|$ in order to remove undue influence of the number of channels if $d^\mathcal{X}$ and $d^\mathcal{Y}$ involve summations on the corresponding channels. Alternatively, it may be appropriate to compensate for different noise levels of the sensors that affect the magnitude of the distances, for instance by boosting the contribution of optical data with respect to highly speckled radar data. The weights can be set heuristically or according to empirical optimisation and theoretical considerations. We prefer to use L_2 distances to limit the computational cost.

To implement $F(\boldsymbol{X})$ and $G(\boldsymbol{Y})$, we use a framework that consists of two autoencoders, each associated with one of the two image domains $\mathcal{X}$ and $\mathcal{Y}$ (We will from now suppress the superscripting with image patch dimensions $h \times w$). Specifically, they consist of two encoder-decoder pairs implemented as deep neural networks: the encoder $E_\mathcal{X}(\boldsymbol{X}) : \mathcal{X} \to \mathcal{Z}_\mathcal{X}$ and decoder $D_\mathcal{X}(\boldsymbol{Z}) : \mathcal{Z}_\mathcal{X} \to \mathcal{X}$; the encoder $E_\mathcal{Y}(\boldsymbol{Y}) : \mathcal{Y} \to \mathcal{Z}_\mathcal{Y}$ and decoder $D_\mathcal{Y}(\boldsymbol{Z}) : \mathcal{Z}_\mathcal{Y} \to \mathcal{X}$. Here, $\mathcal{Z}_\mathcal{X}$ and $\mathcal{Z}_\mathcal{Y}$ denote the code layer or latent space domains of the respective autoencoders. These are implemented with common dimensions, such that the code layer representation $\boldsymbol{Z}$ (also known as the *code*) can denote data tensors in both $\mathcal{Z}_\mathcal{X}$ and $\mathcal{Z}_\mathcal{Y}$. When we need to specify which input space the codes originate from, they will be written as $\boldsymbol{Z}^\mathcal{X}$ and $\boldsymbol{Z}^\mathcal{Y}$.

When trained separately and under the appropriate regularisation, the autoencoders will learn to encode their inputs and reconstruct them with high fidelity in output. Without any external forcing, the distributions of the codes in $\mathcal{Z}_\mathcal{X}$ and $\mathcal{Z}_\mathcal{Y}$ will in general not be close (see Fig. 1a for a visual example). However, we will introduce loss terms that enforce their alignment, both in distribution and in the location of land



(a) Not aligned code spaces

(b) Aligned code spaces

Fig. 1: Two autoencoders without (a) and with (b) code space alignment.

covers within the distributions[1]. If the code distributions in $\mathcal{Z}_\mathcal{X}$ and $\mathcal{Z}_\mathcal{Y}$ align successfully, the encoders can be cascaded with the adjacent decoders to map the latent domain codes back to their original domains, or with the opposite decoders to map data across domains, leading to the sought transformations:

$$\begin{aligned} \hat{\boldsymbol{Y}} &= F(\boldsymbol{X}) = D_\mathcal{Y}(\boldsymbol{Z}^\mathcal{X}) = D_\mathcal{Y}\left(E_\mathcal{X}\left(\boldsymbol{X}\right)\right) , \\ \hat{\boldsymbol{X}} &= G(\boldsymbol{Y}) = D_\mathcal{X}(\boldsymbol{Z}^\mathcal{Y}) = D_\mathcal{X}\left(E_\mathcal{Y}\left(\boldsymbol{Y}\right)\right) , \end{aligned} \tag{2}$$

as depicted in Fig. 1b.

Autoencoders require regularisation in order to avoid learning an identity mapping. This is commonly implemented as sparsity constraints or compression at the code layer by dimensionality reduction, with the latter measure known as a bottleneck. In our implementation, we retain the image patch dimensions (h and w) throughout the hidden layers of the autoencoder and do not resort to bottlenecking, as this is seen to produce the best results. The additional constraints associated with code alignment and crossdomain mapping are seen to enforce the required regularisation.

In the following, we define the terms of the loss function $\mathcal{L}\left(\boldsymbol{\vartheta}\right)$. The loss function is minimised with respect to the parameters of the networks, $\boldsymbol{\vartheta}$, to train the two autoencoders with the goal of obtaining the desired $F(\boldsymbol{X})$ and $G(\boldsymbol{Y})$. In order to compare input patches and translated ones, a weighted distance between patches is defined. Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be two equal-sized $h \times w$ patches, then $\delta(\boldsymbol{A}, \boldsymbol{B}|\boldsymbol{\pi})$ denotes a general weighted distance between patches, where $\boldsymbol{\pi}$ is a vector of weights, each associated with a pixel $i \in \{1, \ldots, n\}$ of the patches, with $n = h \cdot w$. In particular, $\delta(\boldsymbol{A}, \boldsymbol{B}|\mathbf{1}) = \delta(\boldsymbol{A}, \boldsymbol{B})$, being $\mathbf{1}$ a vector of ones. When the pixel measurements $\boldsymbol{a}_i \in \boldsymbol{A}$ and $\boldsymbol{b}_i \in \boldsymbol{B}$ are vectors, the mean squared L_2 norm can be used:

$$\delta(\boldsymbol{A}, \boldsymbol{B}|\boldsymbol{\pi}) = \frac{1}{n}\sum_{i=1}^{n} \pi_i \|\boldsymbol{a}_i - \boldsymbol{b}_i\|_2^2 . \tag{3}$$

[1]Alignment in distribution is not sufficient, since the arrangement of land covers within the distributions may have changed, for instance by mode swapping.

A. Reconstruction Loss

Consider two training patches of $h \times w$ pixels extracted at the same location from $\mathcal{I}_\mathcal{X}$ and $\mathcal{I}_\mathcal{Y}$. The first requirement for the autoencoders is to reproduce their input as faithfully as possible in output, which means that for the reconstructed image patches $\tilde{\boldsymbol{X}}$ and $\tilde{\boldsymbol{Y}}$,

$$\begin{aligned} \tilde{\boldsymbol{X}} &= D_\mathcal{X}\left(E_\mathcal{X}\left(\boldsymbol{X}\right)\right) \simeq \boldsymbol{X} \\ \tilde{\boldsymbol{Y}} &= D_\mathcal{Y}\left(E_\mathcal{Y}\left(\boldsymbol{Y}\right)\right) \simeq \boldsymbol{Y} \end{aligned} \tag{4}$$

must hold true. We introduce the mean squared error between the desired and the predicted output as the reconstruction loss term:

$$\mathcal{L}_\mathrm{r}(\boldsymbol{\vartheta}) = \mathbb{E}_{\boldsymbol{X}}\left[\delta(\tilde{\boldsymbol{X}}, \boldsymbol{X})\right] + \mathbb{E}_{\boldsymbol{Y}}\left[\delta(\tilde{\boldsymbol{Y}}, \boldsymbol{Y})\right] . \tag{5}$$

B. Cycle-consistency Loss

Cycle-consistency implies that data transformed from $\mathcal{X}$ to $\mathcal{Y}$ and back to $\mathcal{X}$ should match exactly the input data we started from. The same applies to the transformations from $\mathcal{Y}$ to $\mathcal{X}$ and back. If $F(\boldsymbol{X})$ and $G(\boldsymbol{Y})$ are close to be perfectly adapted, it must hold true that

$$\begin{aligned} \dot{\boldsymbol{X}} &= G(\hat{\boldsymbol{Y}}) = G\left(F(\boldsymbol{X})\right) \simeq \boldsymbol{X} , \\ \dot{\boldsymbol{Y}} &= F(\hat{\boldsymbol{X}}) = F\left(G(\boldsymbol{Y})\right) \simeq \boldsymbol{Y} , \end{aligned} \tag{6}$$

where $\dot{\boldsymbol{X}} = G(\hat{\boldsymbol{Y}})$ and $\dot{\boldsymbol{Y}} = F(\hat{\boldsymbol{X}})$ indicate the data cyclically transformed to the original domains. Hence, we define the cycle-consistency loss term as:

$$\mathcal{L}_\mathrm{c}(\boldsymbol{\vartheta}) = \mathbb{E}_{\boldsymbol{X}}\left[\delta(\dot{\boldsymbol{X}}, \boldsymbol{X})\right] + \mathbb{E}_{\boldsymbol{Y}}\left[\delta(\dot{\boldsymbol{Y}}, \boldsymbol{Y})\right] . \tag{7}$$

We note that cycle-consistency, like reconstruction, can be evaluated with unpaired data, since $\tilde{\boldsymbol{X}}$ and $\dot{\boldsymbol{X}}$ are computed from $\boldsymbol{X}$ while $\tilde{\boldsymbol{Y}}$ and $\dot{\boldsymbol{Y}}$ are computed from $\boldsymbol{Y}$.

C. Weighted Translation Loss

For those pixels not affected by changes, we require

$$\begin{aligned} \hat{\boldsymbol{Y}} &= F(\boldsymbol{X}) \simeq \boldsymbol{Y} \\ \hat{\boldsymbol{X}} &= G(\boldsymbol{Y}) \simeq \boldsymbol{X} . \end{aligned} \tag{8}$$

From the opposite perspective, pixels that are likely to be changed shall not fulfil these same requirements. Thus, the weighted translation loss term is defined as follows:

$$\mathcal{L}_{\mathrm{t}}(\boldsymbol{\vartheta}) = \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[\delta(\hat{\boldsymbol{X}}, \boldsymbol{X}|\boldsymbol{\pi})\right] + \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[\delta(\hat{\boldsymbol{Y}}, \boldsymbol{Y}|\boldsymbol{\pi})\right], \quad (9)$$

where the contribution to the translation loss of the pixels is weighted by the prior $\boldsymbol{\pi}$, whose elements $\{\pi_i\}_{i=1}^n$ can be interpreted as the probability of pixel $i \in \{1, \ldots, n\}$ not being changed. The π_i for the entire image are stored in a matrix $\boldsymbol{\Pi} \in [0,1]^{H \times W}$, from which the patch corresponding to $\boldsymbol{X}$ and $\boldsymbol{Y}$ is extracted and flattened into the vector $\boldsymbol{\pi}$. These probabilities are not available at the beginning of training, so all entries of $\boldsymbol{\Pi}$ are initialised as 0. After several training epochs, a preliminary evaluation of the difference image $\boldsymbol{\Delta}$ is computed and scaled to fall into the range $[0, 1]$, so that the prior can be updated as $\boldsymbol{\Pi} = 1 - \boldsymbol{\Delta}$. In this way, pixels associated with a large $\boldsymbol{\Delta}$ entry are penalised by a small weight, whereas the opposite happens to pixels more likely to be unchanged. The $\boldsymbol{\Pi}$ is updated iteratively at a rate that we can tune to accommodate both performance and computational cost. This form of self-supervision paradigm has already proven robust in other tasks such as deep clustering [40] and deep image recovery [41].

The translation loss must be evaluated with paired data, since $\hat{\boldsymbol{X}}$ is computed from $\boldsymbol{Y}$ and compared with $\boldsymbol{X}$, while $\hat{\boldsymbol{Y}}$ is computed from $\boldsymbol{X}$ and compared with $\boldsymbol{Y}$. The code correlation loss, presented in the next section, also requires paired data.

D. Code Correlation Loss

The main contribution of this work lies in the way the codes are aligned. It therefore rests on the design and definition of the specific loss term associated with code alignment, referred to as the code correlation loss.

The distances in the input spaces between all pixel pairs (i, j) in the co-located training patches are computed as $d_{i,j}^{\mathcal{X}} = d^{\mathcal{X}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $d_{i,j}^{\mathcal{Y}} = d^{\mathcal{Y}}(\boldsymbol{y}_i, \boldsymbol{y}_j)$ for $i, j \in \{1, \ldots, n\}$, where $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$ denote the feature vectors of pixel $i \in \boldsymbol{X}$ and pixel $j \in \boldsymbol{Y}$, respectively. The appropriate choice of distance measure depends on the underlying data distribution, but should also consider complexity. The hypothesis of normality for imagery acquired by optical sensors is commonly assumed [42], [43]. Concerning SAR intensity data, a logarithmic transformation is sufficient to bring it to near-Gaussianity [23], [31]. This qualifies the use of the computationally efficient Euclidean distance for both these data sources.

Once computed, the distances between all pixel pairs can be converted to the affinities

$$A_{i,j}^{\ell} = \exp\left\{-\frac{\left(d_{i,j}^{\ell}\right)^2}{\sigma_{\ell}^2}\right\} \in (0, 1], \quad i, j \in \{1, \ldots, n\}. \quad (10)$$

Here, $A_{i,j}^{\ell}$ are the entries of the affinity matrix $\boldsymbol{A}^{\ell} \in \mathbb{R}^{n \times n}$ for a given patch and modality $\ell \in \{\mathcal{X}, \mathcal{Y}\}$, and σ_{ℓ} is the kernel width, which must be automatically determined. Our choice is to set it equal to the average distance to the k^{th} nearest neighbour for all data points in the patch of modality ℓ, with $k = \frac{3}{4}n$. This heuristic, which can be traced back to [44], captures the scale of local affinities within the patch and is robust with respect to outliers. Other common approaches to determine the kernel width, such as the Silverman's rule of thumb [45], were discarded because they have not proven themselves as effective.

At this point, one can consider the rows

$$A_i^{\mathcal{X}} = \left[A_{i,1}^{\mathcal{X}}, \ldots, A_{i,n}^{\mathcal{X}}\right] \text{ and } A_j^{\mathcal{Y}} = \left[A_{j,1}^{\mathcal{Y}}, \ldots, A_{j,n}^{\mathcal{Y}}\right]$$

as representations of pixel i from patch $\boldsymbol{X}$ and pixel j from patch $\boldsymbol{Y}$, respectively, in a new affinity space with n features. Moreover, we can define a novel crossmodal distance between these pixels as

$$D_{i,j} = \frac{1}{\sqrt{n}}\|A_i^{\mathcal{X}} - A_j^{\mathcal{Y}}\|_2 \in [0, 1], \; i, j \in \{1, \ldots, n\}, \quad (11)$$

noting that since the affinities are normalised to the range $[0, 1]$, then so is $D_{i,j}$. This crossmodal distance allows to compare data across the two domains directly from their input space features. It further allows us to distinguish pixels that have consistent relations to other pixels in both domains from those that do not. This information can be interpreted in terms of probability of change.

The crossmodal input space distances $D_{i,j}$ for $i, j \in \{1, \ldots, n\}$ are stored in $\boldsymbol{D}$. We next want to make sure that these are maintained in the code layer. We do this by defining similarities $S_{ij} = 1 - D_{ij}$ and forcing them to be as similar as possible to correlations between the codes of corresponding pixels. Let $\boldsymbol{z}_i^{\mathcal{X}}$ and $\boldsymbol{z}_j^{\mathcal{Y}}$ denote the entry of code patch $\boldsymbol{Z}^{\mathcal{X}}$ corresponding to pixel i and the entry of code patch $\boldsymbol{Z}^{\mathcal{Y}}$ corresponding to pixel j, respectively. In mathematical terms, we enforce that

$$R_{i,j} \triangleq \frac{\left(\boldsymbol{z}_i^{\mathcal{X}}\right)^T \boldsymbol{z}_j^{\mathcal{Y}} + |\mathcal{Z}|}{2\,|\mathcal{Z}|} \simeq S_{i,j}, \quad i, j \in \{1, \ldots, n\}, \quad (12)$$

where the $S_{i,j}$ are elements of $\boldsymbol{S} = 1 - \boldsymbol{D}$. The normalisation of the codes, $\boldsymbol{z}_i^{\mathcal{X}}, \boldsymbol{z}_j^{\mathcal{Y}} \in [-1, 1]^{|\mathcal{Z}|}$, and their dimensionality $|\mathcal{Z}|$ is such that the code correlations $R_{i,j}$ falls in the range $[0, 1]$. Note that the elements on the diagonal of $\boldsymbol{S}$ represent the similarity between $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$, that are not identical, so $S_{i,i}$ can be different from 1. Also observe that $\boldsymbol{S}$ is not symmetric, because the similarity between $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$ is not necessarily the same as between $\boldsymbol{x}_j$ and $\boldsymbol{y}_i$.

Based on the above definitions and considerations, the code correlation loss term is defined as

$$\mathcal{L}_z(\boldsymbol{\vartheta}) = \mathbb{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[\delta(\boldsymbol{R}, \boldsymbol{S})\right], \quad (13)$$

where the code correlation matrix $\boldsymbol{R}$ stores the $R_{i,j}$ from the left-hand side of Eq. (12). Note that only encoder parameters are adjusted with this loss term.

E. Total Loss Function

Finally, the loss function minimised in this framework is the following weighted sum:

$$\mathcal{L}(\boldsymbol{\vartheta}) = \lambda_{\mathrm{r}}\,\mathcal{L}_{\mathrm{r}}(\boldsymbol{\vartheta}) + \lambda_{\mathrm{c}}\,\mathcal{L}_{\mathrm{c}}(\boldsymbol{\vartheta}) + \lambda_{\mathrm{t}}\,\mathcal{L}_{\mathrm{t}}(\boldsymbol{\vartheta}) + \lambda_z\,\mathcal{L}_z(\boldsymbol{\vartheta}), \quad (14)$$

where the weights λ_{r}, λ_{c}, λ_{t} and λ_{z} are used to balance the loss terms and their impact on the optimisation result. Together, the cycle-consistency and the code correlation let us achieve the sought alignment, while at the same time the other two terms keep focus on a correct reconstruction and transformation of the input.

After the training and the computation of $\mathbf{\Delta}$, the CD workflow includes an optional step and a mandatory step. The former consists of spatial filtering of $\mathbf{\Delta}$ to reduce errors, based on the simple idea that spurious changed (unchanged) pixels surrounded by unchanged (changed) ones are most likely outliers that have been erroneously classified. For our method we selected the Gaussian filtering presented in [46], which uses spatial context to regularise $\mathbf{\Delta}$. The last step of a CD pipeline is to obtain the actual change map by thresholding $\mathbf{\Delta}$, and so all the pixels whose value is below the threshold are considered unchanged, vice versa for those with a larger value. The optimal threshold can be found by visual inspection or automatically by exploiting an algorithm such as [47], [48], [49]. We opted for the classical Otsu's method [50].

III. Results

A. Implementation details

For the proposed framework we deploy fully convolutional neural networks designed as follows: Conv($3 \times 3 \times 100$)–ReLU–Conv($3\times3\times100$)–ReLU–Conv($3\times3\times C$)–Tanh. Conv($3\times3\times C$) indicates a convolutional layer with C filters of size 3×3, being $C = 3$ for the encoders, $C = |\mathcal{X}|$ for $D_{\mathcal{X}}$ and $C = |\mathcal{Y}|$ for $D_{\mathcal{Y}}$. All the layers are non-strided and we apply padding to preserve the input size. Leaky-ReLU [51] with slope of $\beta = 0.3$ for negative arguments is used. Tanh indicates the hyperbolic tangent [51], which normalises data between -1 and 1, as this has shown to speed up convergence [52]. Dropout [53] with a 20% rate is applied. A low number of features in the latent space allows to achieve the sought alignment more easily, whereas the number of layers and filters has been set to find a balance between flexibility of the network representations and the limited trainability of the networks, due to a small amount of training data. Concerning the latter, at every epoch 10 batches containing 10 random patches of 100×100 pixels are extracted and randomly augmented (90 degrees rotations and upside-down flips). As specified, the code correlation loss term $\mathcal{L}_z$ requires computation of a size $N \times N$ crossmodal distance matrix $\boldsymbol{D}$ when the training patch is $h \times w$. Due to memory constraints, only the inner 20×20 pixels of the training patches have been used to compute $\boldsymbol{D}$. For normalisation of the matrix $\boldsymbol{D}$ between 0 and 1, the framework responded better when applying contrast stretching between the empirical batch minimum and maximum values of $\boldsymbol{D}$. The four λ values controlling the weighted sum of $\mathcal{L}$ were all set to 1.

The Adam optimiser [54] was selected to perform the minimisation of $\mathcal{L}$ for 100 epochs with a learning rate of 10^{-4}, which experienced a stair-cased exponential decay with rate 0.96. Actually, we found it beneficial to reduce the learning rate associated with $\mathcal{L}_z$ more aggressively with rate 0.9. This was implemented because it turned out most beneficial to correlate the code spaces at the beginning, when the autoencoder just started to learn a meaningful representation of the latent spaces and a reasonable transformation of the data. After some updates of $\mathbf{\Pi}$, $\mathcal{L}_z$ was experienced to function more as a regulariser, whereas the translation loss $\mathcal{L}_{\mathrm{t}}$ came more into play. These updates were made every 25 epochs, so at epoch 25, 50, and 75.

B. Evaluation criteria

The performance of the proposed approach is measured in terms of two metrics. The overall accuracy, $OA \in [0, 1]$, is the ratio between correctly classified pixels and the total amount of pixels. Cohen's kappa coefficient, $\kappa \in [-1, 1]$, indicates the agreement between two classifiers [55]. $\kappa = 1$ means total agreement, $\kappa = -1$ means total disagreement, $\kappa = 0$ means no correlation (random guess). When comparing against a ground truth dataset, Cohen's kappa is expressed as

$$\kappa = \frac{p_o - p_e}{1 - p_e}. \tag{15}$$

Here, p_o stands for the observed agreement between predictions and labels, i.e. the OA, while p_e is the probability of random agreement, which is estimated from the observed true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) as:

$$\begin{aligned} p_e = &\left(\frac{\mathrm{TP}+\mathrm{FP}}{N} \cdot \frac{\mathrm{FN}+\mathrm{TN}}{N}\right) \\ &+ \left(\frac{\mathrm{TP}+\mathrm{FN}}{N} \cdot \frac{\mathrm{FP}+\mathrm{TN}}{N}\right). \end{aligned} \tag{16}$$

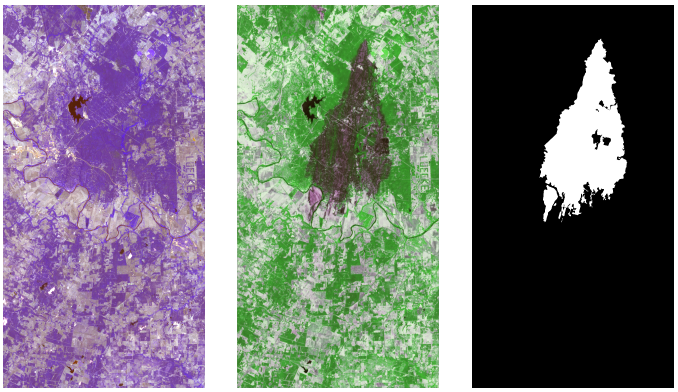
In general, a high κ implies a high OA, but not vice versa. In any case, the papers presenting state-of-the-art methods do not always report both, so we compare algorithm performance dataset by dataset in terms of the available metrics.

C. Methods compared

We will in the following present four datasets that are used to test the proposed method and reference algorithms. On the first two datasets, the proposed method is compared to four similar deep learning approaches. The first two are the conditional adversarial network (CAN) of Niu et al. [33] and the symmetric convolutional coupling network (SCCN) of Liu et al. [28], which represent seminal work on unsupervised multimodal change detection with convolutional neural networks. The final two are are the ACE-Net and the X-Net recently proposed by the current authors in [37]. To be aware of the characteristics of the training strategies employed by these methods, it should be noted that the CAN and the ACE-Net apply adversarial training, the ACE-Net and the SCCN exploit code alignment, while the ACE-Net and the X-Net use similar weighted image-to-image translation schemes as the proposed method. The final two datasets have been used extensively by others in testing of methods whose source code we do not have access to. For these datasets we compare our results with the performance reported in Zhang et al. [27] for post-classification comparison (PCC) and a deep learning model based on stacked denoising autoencoders (SDAE). We also compare with several methods proposed by Touati et al.,

namely a method that obtains its result my filtering a textural gradient-based similarity map (TGSM) [56], a method using energy-based encoding of nonlocal pairwise pixel interactions (EENPPI) [24], a method based on modality invariant multidimensional scaling (MIMDS) [25], and a Markov model for multimodal change detection (M3CD) [57]. Finally, we compare with results obtained with the manifold learning-based statistical model (MLSM) of Prendes et al. [16], [58].

D. First dataset: Forest fire in Texas



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

Fig. 2: Forest fire in Texas: Landsat 5 ($t1$), (b) EO-1 ALI ($t2$), (c) ground truth. RGB false color composites are shown for both images.

A Landsat 5 Thematic Mapper (TM) multispectral image (Fig. 2a) was acquired before a forest fire that took place in Bastrop County, Texas, during September-October 2011[2]. An Earth Observing-1 Advanced Land Imager (EO-1 ALI) multispectral acquisition after the event completes the dataset (Fig. 2b)[1]. Both images are optical, with 1534×808 pixels, and 7 and 10 channels respectively. The ground truth of the event (see Fig. 2c) is provided by Volpi *et al.* [10].

Fig. 3 displays the results obtained on this dataset by the proposed framework as compared to the reference methods. As one can notice, the proposed network produces consistently higher accuracy than the competitors and also maintains a low variance. We also report that Volpi *et al.* [10] and Luppino *et al.* [9] achieved a κ of 0.65 and 0.91 respectively with respect to the same ground truth. Concerning the training times, their averages are listed in Table I. These are comparable because the computation of the affinity matrices is time-consuming, but the proposed method is implemented with relatively small networks and trained for fewer iterations.

TABLE I: Average training time of the five methods on the Texas dataset.

CAN	SCCN	ACE-Net	X-Net	Proposed
70 min	16 min	13 min	7 min	11 min

[2]Distributed by LP DAAC, http://lpdaac.usgs.gov

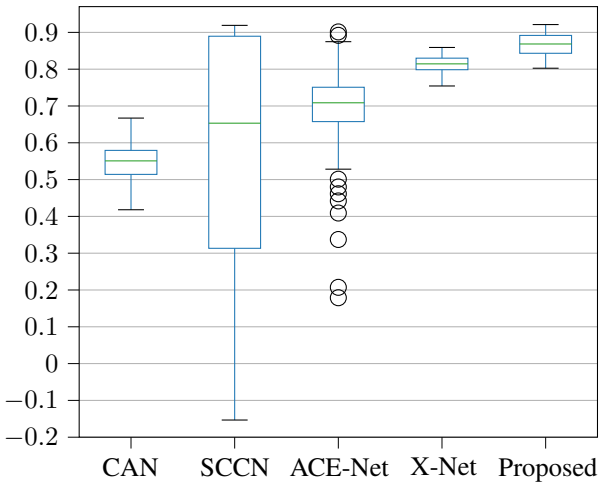
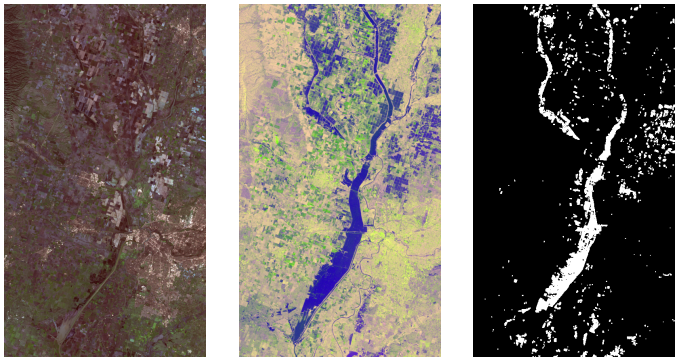


Fig. 3: κ obtained on the Texas dataset by the proposed approach and several state-of-the-art methods.

E. Second dataset: Flood in California



(a) Landsat 8 (t_1) (b) Sentinel1-A (t_2) (c) Ground Truth

Fig. 4: Flood in California: Landsat 8 ($t1$), (b) Sentinel1-A ($t2$), (c) ground truth. RGB false color composites are shown for both images.

Fig. 4a shows the RGB channels of the Landsat 8 acquisition[1] covering Sacramento County, Yuba County and Sutter County, California, on 5 January 2017. In addition, the multispectral sensors mounted on Landsat 8 provides another 8 channels, going from deep blue to long-wave infrared. The same area was affected by a flood, as it can be noticed in Fig. 4b. This is a Sentinel-1A[3] acquisition, recorded in polarisations VV and VH on 18 February 2017 and augmented with the ratio between the two intensities as the third channel. The ground truth in Fig. 4c is provided by Luppino *et al.* [9]. Originally of 3500×2000 pixels, these images were resampled to 850×500 pixels as in [37] to compare the results.

The metrics obtained on this dataset are summarised in Fig. 5. Also in this case, the proposed framework outperforms the state-of-the-art counterparts, both in terms of high quality and low variance. For this dataset, $\kappa = 0.46$ was achieved in [9]. Table II contains the average training times on this dataset.

[3]Data processed by ESA, http://www.copernicus.eu/

TABLE II: Average training time of the five methods on the California dataset.

CAN	SCCN	ACE-Net	X-Net	Proposed
21 min	15 min	12 min	6 min	8 min

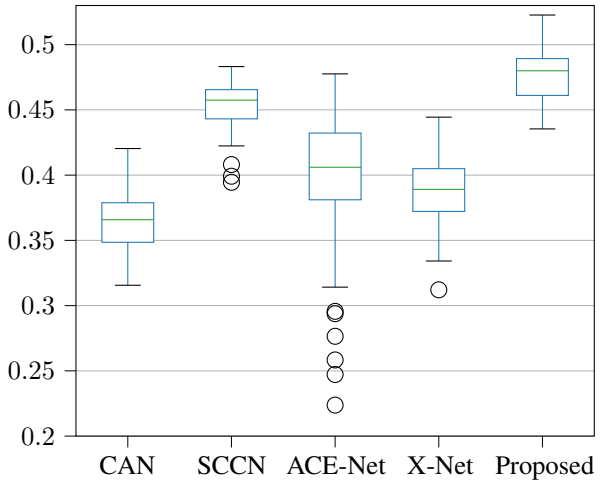


Fig. 5: κ obtained on the California dataset by the proposed approach and several state-of-the-art methods

Again, the proposed approach required a training time which is in line with the state-of-the-art algorithms.

F. Third dataset: Lake overflow in Italy

(a) Landsat 5 NIR(t_1) (b) Landsat 5 RGB (t_2) (c) Ground Truth

Fig. 6: Lake overflow in Italy: Landsat 5 Near InfraRed (NIR) band ($t1$), (b) Landsat 5 red, green, and blue (RGB) bands ($t2$), (c) ground truth.

The next two datasets were provided by Touati *et al.* [57]. In Fig. 6a and Fig. 6b are two Landsat 5 images of 412×300 pixels: the first is the Near InfraRed (NIR) band of an image acquired in September 1995, the second represents the red, green, and blue (RGB) bands sensed on the same area in July 1996. These images were recorded before and after a lake overflow in Italy, whose profile is highlighted as ground truth in Fig. 6c. Table III presents the average overall accuracy for several methods. For the proposed method, the standard deviation is provided as well, and one may see that the results are very stable and close to the state-of-the-art. The small amount of data in terms of the number of pixels does not in general favour deep learning approaches, and the relative performance could potentially change with larger training samples. In this respect, Zhang *et al.* [27] proposed a method that seems to be an exception, as this deep learning approach produces the best

TABLE III: Average accuracy of several methods on the lake overflow dataset. Best on top, proposed method in bold.

Lake overflow dataset	OA
SDAE [27]	0.975
M3CD [57]	0.964
MIMDS [25]	0.942
Proposed	$\mathbf{0.922 \pm 0.007}$
PCC [27]	0.882

performance on this dataset. However, it must be pointed out that, unlike us, they adapt their architectures to the dataset, which is infeasible in a completely unsupervised setting. The average training time for the proposed framework on this dataset was a few seconds below 7 minutes.

G. Fourth dataset: Construction site in France

(a) Pleiades (t_1) (b) WorldView 2 (t_2) (c) Ground Truth

Fig. 7: Constructions in France: Pleiades ($t1$), (b) WorldView 2 ($t2$), (c) ground truth.

The last dataset includes two RGB images captured by Pleiades (Fig. 7a) and WorldView 2 (Fig. 7b), showing the work progress of road constructions in Toulouse, France, during May 2012 and July 2013. The ground truth in Fig. 7c depicts such progress. For computational reasons, the images were reduced from 2000×2000 pixels to 500×500 as in [57], leading to an average training time of 7 minutes. The average accuracy obtained by several methods on this dataset is listed in Table IV. Again, the accuracy of the proposed method comes with a standard deviation, and also in this case it is very stable and close to the state-of-the-art.

Finally, in Fig. 8 we present a visual example of the transformations obtained with the proposed method on the datasets used in this section. As it can be seen, the data

TABLE IV: Average accuracy of several methods on the constructions dataset. Best on top, proposed method in bold.

Constructions dataset	OA
MIMDS [25]	0.877
TGSM [56]	0.870
M3CD [57]	0.862
Proposed	$\mathbf{0.859 \pm 0.003}$
EENPPI [24]	0.853
MLSM [16]	0.844

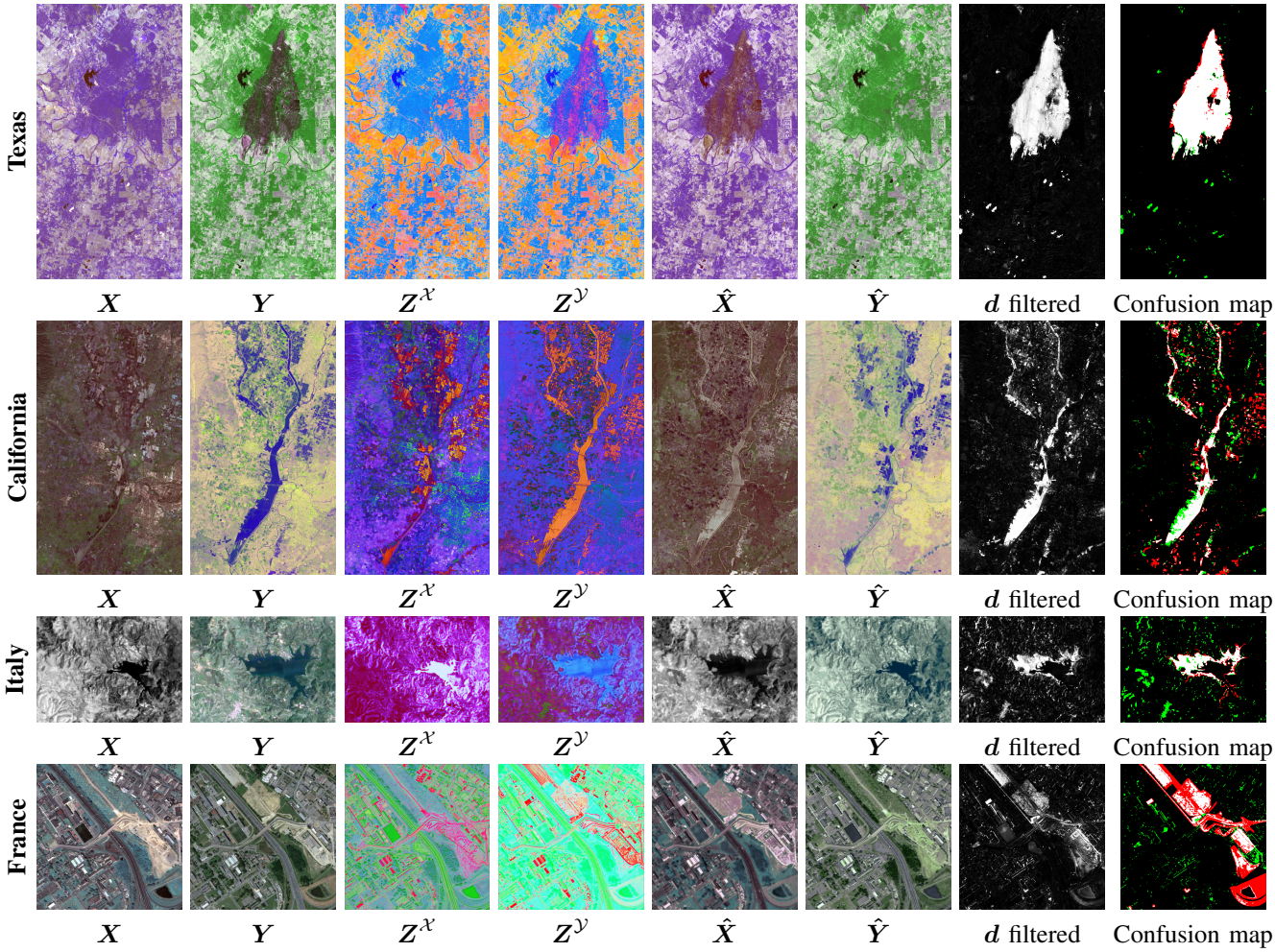


Fig. 8: Examples of final results, organized in one row for each dataset. Col. 1: input image $\boldsymbol{X}$; Col. 2: input image $\boldsymbol{Y}$; Col. 3: transformations of $\boldsymbol{X}$ into the code space $\boldsymbol{Z}^{\mathcal{X}} = E_{\mathcal{X}}(\boldsymbol{X})$; Col. 4: transformations of $\boldsymbol{Y}$ into the code space $\boldsymbol{Z}^{\mathcal{Y}} = E_{\mathcal{Y}}(\boldsymbol{Y})$; Col. 5 transformations $\hat{\boldsymbol{Y}} = F(\boldsymbol{X})$; Col. 6: transformations $\hat{\boldsymbol{X}} = G(\boldsymbol{Y})$; Col. 7: $\boldsymbol{d}$ filtered; Col. 8: Confusion map (TP: white; TN: black; FP: green; FN: red) (g)

from one input domain are transformed into the other in a meaningful way, and the resemblance between the styles of the fake images and the original images is clear. In the two last datasets, one could speculate that the low amount of data and features (few pixels consisting of few channels) did not allow to achieve a proper alignments of the code spaces. This endorses the choice to compute $\boldsymbol{d}$ as a weighted sum of the difference images in the input spaces rather than just the difference image in the latent space, although it still remains a valid option.

IV. Conclusions

In this work, we presented a novel unsupervised methodology to align the code spaces of two autoencoders based on affinity information extracted from the input data. In particular, this is part of a heterogeneous CD framework that allows to achieve this latent space entanglement even when the input images contain changes, whose misleading contribution to the training is considerably reduced. The method proved to perform consistently on par with or better than the state-of-the-art across four different datasets. Its performance worsen when handling a limited amount of features in input, especially when only one channel is available in one of the images, implying a regression from one variable to many, which is an ill-posed problem. On the other hand, it deals properly with multispectral and multipolarisation images, by being able to map data appropriately across domains in a meaningful manner.

V. Acknowledgement

The project and the first author was funded by the Research Council of Norway under research grant no. 251327. We gratefully acknowledge the support of NVIDIA Corporation by the donation of the GPU used for this research.

References

[1] 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.

[2] X. Li and A. G.-O. Yeh, "Analyzing spatial restructuring of land use patterns in a fast growing region using remote sensing and GIS," *Landscape and Urban planning*, vol. 69, no. 4, pp. 335–354, 2004.

[3] M. Herold, J. Scepan, and K. C. Clarke, "The use of remote sensing and landscape metrics to describe structures and changes in urban land uses," *Environment and Planning A*, vol. 34, no. 8, pp. 1443–1458, 2002.

[4] 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.

[5] A. A. Alesheikh, A. Ghorbanali, and N. Nouri, "Coastline change detection using remote sensing," *Int. J. Environmental Sci. Tech.*, vol. 4, no. 1, pp. 61–66, 2007.

[6] E. Berthier, Y. Arnaud, R. Kumar, S. Ahmad, P. Wagnon, and P. Chevallier, "Remote sensing estimates of glacier mass balances in the Himachal Pradesh (Western Himalaya, India)," *Remote Sens. Environ.*, vol. 108, no. 3, pp. 327–338, 2007.

[7] P. Griffiths, P. Hostert, O. Gruebner, and S. van der Linden, "Mapping megacity growth with multi-sensor data," *Remote Sens. Environ.*, vol. 114, no. 2, pp. 426–439, 2010.

[8] D. Brunner, G. Lemoine, and L. Bruzzone, "Earthquake damage assessment of buildings using VHR optical and SAR imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 5, pp. 2403–2420, 2010.

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

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

[11] L. T. Luppino, F. M. Bianchi, G. Moser, and S. N. Anfinsen, "Remote sensing image regression for heterogeneous change detection," in *Proc. IEEE Int. Workshop Mach. Learn. Signal Process. (MLSP)*, 2018, pp. 1–6.

[12] B. Storvik, G. Storvik, and R. Fjortoft, "On the combination of multisensor data using meta-Gaussian distributions," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 7, pp. 2372–2379, 2009.

[13] Z.-G. Liu, J. Dezert, G. Mercier, and Q. Pan, "Dynamic evidential reasoning for change detection in remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 5, pp. 1955–1967, 2011.

[14] Z.-G. Liu, G. Mercier, J. Dezert, and Q. Pan, "Change detection in heterogeneous remote sensing images based on multidimensional evidential reasoning," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 1, pp. 168–172, 2014.

[15] D. Tuia, J. Munoz-Mari, L. Gomez-Chova, and J. Malo, "Graph matching for adaptation in remote sensing," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 1, pp. 329–341, Jan 2013.

[16] 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.

[17] Y. Zhou, H. Liu, D. Li, H. Cao, J. Yang, and Z. Li, "Cross-sensor image change detection based on deep canonically correlated autoencoders," in *Proc. Int. Conf. Artif. Intell. Commun. Netw.*, 2019, pp. 251–257.

[18] J. Yang, Y. Zhou, Y. Cao, and L. Feng, "Heterogeneous image change detection using deep canonical correlation analysis," in *Proc. Int. Conf. Pattern Recogn. (ICPR)*, 2018, pp. 2917–2922.

[19] 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.

[20] G. Liu, J. Delon, Y. Gousseau, and F. Tupin, "Unsupervised change detection between multi-sensor high resolution satellite images," in *Proc. Eur. Signal Process. Conf. (EUSIPCO)*. IEEE, 2016, pp. 2435–2439.

[21] G. Liu, Y. Gousseau, and F. Tupin, "A contrario comparison of local descriptors for change detection in very high spatial resolution satellite images of urban areas," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 6, pp. 3904–3918, 2019.

[22] D. Marcos, R. Hamid, and D. Tuia, "Geospatial correspondences for multimodal registration," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn. (CVPR)*, 2016, pp. 5091–5100.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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, 06 2016.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] X. Niu, M. Gong, T. Zhan, and Y. Yang, "A conditional adversarial network for change detection in heterogeneous images," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 1, pp. 45–49, 2018.

[34] 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.

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

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

[37] L. T. Luppino, M. Kampffmeyer, F. M. Bianchi, G. Moser, S. B. Serpico, R. Jenssen, and S. N. Anfinsen, "Deep image translation with an affinity-based change prior for unsupervised multimodal change detection," `arXiv:2001.04271 [cs.LG]`, 2020.

[38] M. Kampffmeyer, S. Løkse, F. M. Bianchi, R. Jenssen, and L. Livi, "The deep kernelized autoencoder," *Applied Soft Computing*, vol. 71, pp. 816–825, 2018.

[39] F. M. Bianchi, L. Livi, K. Ø. Mikalsen, M. Kampffmeyer, and R. Jenssen, "Learning representations of multivariate time series with missing data," *Pattern Recognition*, vol. 96, no. 106973, 2019.

[40] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 132–149.

[41] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proc. IEEE Comp. Soc. Conf. Comput. Vis. Pattern Recogn. (CVPR)*, 2018, pp. 9446–9454.

[42] 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.

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

[44] Y. Mack and M. Rosenblatt, "Multivariate k-nearest neighbor density estimates," *J. Multivar. Anal.*, vol. 9, no. 1, pp. 1–15, 1979.

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

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

[47] 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.

[48] 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.

[49] 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.
[50] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst. Man Cybern.*, vol. 9, no. 1, pp. 62–66, 1979.
[51] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 30, no. 1, 2013, p. 3.
[52] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 9–48.
[53] 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.
[54] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of ADAM and beyond," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.
[55] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37–46, 1960.
[56] R. Touati, M. Mignotte, and M. Dahmane, "A new change detector in heterogeneous remote sensing imagery," in *Proc. IEEE Int. Conf. Image Process. Theory Tools Applic. (IPTA)*, 2017, pp. 1–6.
[57] ——, "Multimodal change detection in remote sensing images using an unsupervised pixel pairwise-based Markov random field model," *IEEE Trans. Image Proc.*, vol. 29, pp. 757–767, 2019.
[58] J. Prendes, "New statistical modeling of multi-sensor images with application to change detection," Ph.D. dissertation, Université Paris-Saclay, 2015.

Chapter 10

Concluding remarks

At the beginning of this project, half a decade ago, the topic of unsupervised heterogeneous CD was still not explored in depth by the remote sensing community. Not so many examples could be found in the literature [19, 47], and they were very few when compared to the supervised ones. Throughout the years this trend gradually changed, especially thanks to the research endeavours of two research groups which led to valuable contributions to the field. Specifically, Touati *et al.* from the Department of Computer Science and Operations Research, Université de Montréal, Montréal, Canada, leveraged the strength of traditional pattern recognition methods based on pixel relations [11, 55, 128, 129, 130], promoting approaches of image analysis and keeping the focus on mathematical rigorousness. Very recently, they steered their attention to DL as well [131]. Concerning the latter, Gong *et al.* from the Key Laboratory of Intelligent Perception and Image Understanding, Xidian University, Xi'an, China adopted DL much earlier [19, 25, 21, 26, 44, 61, 132, 133], and deserve credit for exploring extensively the effectiveness and the power of these techniques in tackling the problem of unsupervised heterogeneous CD.

The outcomes of this Ph.D. activity can be set somewhere in between. At first, more conventional machine learning methods were developed, and these were combined later on with DL techniques, which became more and more central. In addition to Paper II and III, this converged also to the work presented in [134]. The idea is to incorporate the affinity matrices directly into the optimisation of the X-Net presented in Paper II. By introducing a

loss term to minimise the differences between the affinity matrices evaluated on the input images and on the transformed images, the performance has shown to improve both in terms of accuracy and robustness. This work is yet to be published in the journal literature.

10.1 Outlook

This thesis presented the problem of heterogeneous CD, emphasising the reasons why it is an important topic of remote sensing. Also, it proposed a selection of solutions apt to meet the challenges of this task, such as unpaired sensor domains, inconsistent class signatures, unrelated distributions and so on. Most of the focus was dedicated to the concepts of unsupervised data transformation and domain mapping, in the particular case of bitemporal CD in mid-resolution satellite images.

This work gravitated around a core hypothesis: the comparison of affinity matrices across the two images yields a preliminary information about the changes on a local scale. This assumption was validated through the development of three heterogeneous CD methods, and effectively three proofs of concept exploited the affinity matrices in different ways to infer information at different levels: patchwise, pixelwise for colocated pixels in the two domains, and pixelwise across different locations in the two domains.

This prior knowledge was exploited to achieve the optimisation of three paradigms of image translation, one based on more conventional pixel-based regression functions and two in the form of deep convolutional NNs that exploit the power of contextual information. In the first case, this information allowed the automatic selection of training samples. In the second case, it highlighted the changed areas to be penalised during the unsupervised training. In the last case, it led to the definition of a crossmodal similarity, indicating whether data points generated from different input domains should be aligned in a common latent space.

The dissertation covered also the weaknesses of these techniques, highlighting their limitations in terms of actual capabilities and their sensibility to parameter selection. The former are intrinsically related to the underlying assumptions which are required to perform heterogeneous CD in an unsupervised manner, the latter require cautious parameter tuning.

10.2 Future developments

A natural extension of this work is represented by multitemporal applications that go beyond the bitemporal case. Even though the proposed frameworks must then be modified to include the time variable in the affinity matrices comparison, this redesign has no evidence of being unfeasible. Notice that applying bitemporal CD methods to each and every couple of consequent images of a multitemporal dataset is possible, although perhaps not so elegant, because this would mean to ignore completely the undoubted correlations over time between the data.

The performance of the proposed methods must be evaluated at higher resolutions, for which very often the assumption of precise coregistration does not hold so firmly. Even though the paper presented in Chapter 9 includes an experiment on a dataset of roughly 2.5 meters resolution, a more thorough investigation is needed to prove the robustness of these approaches. The main issue would be the gradual degradation of the information associated with the affinity matrices as the coregistration becomes poorer and poorer.

The analysis in this manuscript was concentrated on the problem of heterogeneous CD in remote sensing. Nevertheless, the proposed methodologies are not strictly limited to the field of Earth observation. In fact, another potential application is heterogeneous CD in biomedical images. The interest in the field is enormous, especially with the increasing variety of scanning systems such as magnetic resonance imaging (MRI), computerised tomography (CT), and positron-emission tomography (PET). The ability to compare multimodal biomedical images is undoubtedly important to support doctors and medical staff in decision making and diagnosing. There is also a strong potential in extending the methods to perform multimodal image registration, which is a vital and challenging task in medical imaging. It would be a natural research goal to automate this process in a robust manner based on the current results presented in this thesis.

Bibliography

[1] Marcel Bosc, Fabrice Heitz, Jean-Paul Armspach, Izzie Namer, Daniel Gounot, and Lucien Rumbach. Automatic change detection in multimodal serial MRI: application to multiple sclerosis lesion evolution. *NeuroImage*, 20(2):643–656, 2003.

[2] Stefan Huwer and Heinrich Niemann. Adaptive change detection for real-time surveillance applications. In *Third IEEE International Workshop on Visual Surveillance*, pages 37–46. IEEE, 2000.

[3] Christian Koch, Kristina Georgieva, Varun Kasireddy, Burcu Akinci, and Paul Fieguth. A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure. *Advanced Engineering Informatics*, 29(2):196–210, 2015.

[4] Richard J. Radke, Srinivas Andra, Omar Al-Kofahi, and Badrinath Roysam. Image change detection algorithms: A systematic survey. *IEEE Transactions on Image Processing*, 14(3):294–307, 2005.

[5] Assefa M Melesse, Qihao Weng, Prasad S Thenkabail, and Gabriel B Senay. Remote sensing sensors and applications in environmental resources mapping and modelling. *Sensors*, 7(12):3209–3241, 2007.

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

[7] Charles Toth and Grzegorz Jóźków. Remote sensing platforms and sensors: A survey. *ISPRS Journal of Photogrammetry and Remote Sensing*, 115:22–36, 2016.

[8] Francesca Bovolo and Lorenzo Bruzzone. The time variable in data fusion: A change detection perspective. *IEEE Geoscience and Remote Sensing Magazine*, 3(3):8–26, 2015.

[9] Pedram Ghamisi, Behnood Rasti, Naoto Yokoya, Qunming Wang, Bernhard Hofle, Lorenzo Bruzzone, Francesca Bovolo, Mingmin Chi, Katharina Anders, and Richard Gloaguen. Multisource and multitemporal data fusion in remote sensing: A comprehensive review of the state of the art. *IEEE Geoscience and Remote Sensing Magazine*, 7(1):6–39, 2019.

[10] Vito Alberga. Similarity measures of remotely sensed multi-sensor images for change detection applications. *Remote Sensing*, 1(3):122–143, 2009.

[11] Redha Touati, Max Mignotte, and Mohamed Dahmane. A new change detector in heterogeneous remote sensing imagery. In *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6. IEEE, 2017.

[12] Gang Liu, Julie Delon, Yann Gousseau, and Florence Tupin. Unsupervised change detection between multi-sensor high resolution satellite images. In *Signal Processing Conference (EUSIPCO), 2016 24th European*, pages 2435–2439. IEEE, 2016.

[13] Grégoire Mercier, Gabriele Moser, and Sebastiano B. Serpico. Conditional copulas for change detection in heterogeneous remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 46(5):1428–1441, May 2008.

[14] Michele Volpi, Gustau Camps-Valls, and Devis Tuia. Spectral alignment of multi-temporal cross-sensor images with automated kernel canonical correlation analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 107:50–63, 2015.

[15] Diego Marcos, Raffay Hamid, and Devis Tuia. Geospatial correspondences for multimodal registration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5091–5100, 2016.

[16] Zhun-ga Liu, Jean Dezert, Grégoire Mercier, and Quan Pan. Dynamic evidential reasoning for change detection in remote sensing images.

IEEE Transactions on Geoscience and Remote Sensing, 50(5):1955–1967, 2011.

[17] Zhun-ga Liu, Grégoire Mercier, Jean Dezert, and Quan Pan. Change detection in heterogeneous remote sensing images based on multidimensional evidential reasoning. *IEEE Geoscience and Remote Sensing Letters*, 11(1):168–172, 2014.

[18] Luigi T. Luppino, Stian N. Anfinsen, Gabriele Moser, Robert Jenssen, Filippo M. Bianchi, Sebastiano B. Serpico, and Grégoire Mercier. A clustering approach to heterogeneous change detection. In *Scandinavian Conference on Image Analysis*, pages 181–192. Springer, 2017.

[19] Puzhao Zhang, Maoguo Gong, Linzhi Su, Jia Liu, and Zhizhou Li. Change detection based on deep feature representation and mapping transformation for multi-spatial-resolution remote sensing images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 116:24–41, June 2016.

[20] Zhun-ga Liu, Gang Li, Grégoire Mercier, You He, and Quan Pan. Change detection in heterogenous remote sensing images via homogeneous pixel transformation. *IEEE Transactions on Image Processing*, 27(4):1822–1834, 2018.

[21] Tao Zhan, Maoguo Gong, Jia Liu, and Puzhao Zhang. Iterative feature mapping network for detecting multiple changes in multi-source remote sensing images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 146:38–51, 2018.

[22] David A. Landgrebe. *Signal Theory Methods in Multispectral Remote Sensing*, volume 29. John Wiley & Sons, 2005.

[23] James B. Campbell and Randolph H. Wynne. *Introduction to Remote Sensing*. Guilford Press, 2011.

[24] Thomas Lillesand, Ralph W. Kiefer, and Jonathan Chipman. *Remote Sensing and Image Interpretation*. John Wiley & Sons, 2015.

[25] Wei Zhao, Zhirui Wang, Maoguo Gong, and Jia Liu. Discriminative feature learning for unsupervised change detection in heterogeneous

images based on a coupled neural network. *IEEE Transactions on Geoscience and Remote Sensing*, 55(12):7066–7080, 2017.

[26] Tao Zhan, Maoguo Gong, Xiangming Jiang, and Shuwei Li. Log-based transformation feature learning for change detection in heterogeneous images. *IEEE Geoscience and Remote Sensing Letters*, 15(9):1352–1356, 2018.

[27] Alberto Moreira, Pau Prats-Iraola, Marwan Younis, Gerhard Krieger, Irena Hajnsek, and Konstantinos P. Papathanassiou. A tutorial on synthetic aperture radar. *IEEE Geoscience and Remote Sensing Magazine*, 1(1):6–43, 2013.

[28] The Canada Centre for Mapping and Earth Observation. Remote sensing tutorial. `https://www.nrcan.gc.ca/maps-tools-publications/satellite-imagery-air-photos/tutorial-fundamentals-remote-sensing/9309`, 2019. Accessed: 1 February 2020.

[29] Rong Gui, Xin Xu, Hao Dong, Chao Song, and Fangling Pu. Individual building extraction from TerraSAR-X images based on ontological semantic analysis. *Remote Sensing*, 8(9):708, 2016.

[30] Ashbindu Singh. Digital change detection techniques using remotely-sensed data. *International Journal of Remote Sensing*, 10(6):989–1003, 1989.

[31] Pere Serra, Xavier Pons, and David Sauri. Post-classification change detection with data from different sensors: some accuracy considerations. *International Journal of Remote Sensing*, 24(16):3311–3340, 2003.

[32] Yady T. Solano-Correa, Francesca Bovolo, and Lorenzo Bruzzone. An approach to multiple change detection in VHR optical images based on iterative clustering and adaptive thresholding. *IEEE Geoscience and Remote Sensing Letters*, 16(8):1334–1338, 2019.

[33] Rodrigo C. Daudt, Bertrand Le Saux, Alexandre Boulch, and Yann Gousseau. Multitask learning for large-scale semantic change detection. *Computer Vision and Image Understanding*, 187:102783, 2019.

[34] Sicong Liu, Lorenzo Bruzzone, Francesca Bovolo, and Peijun Du. Hierarchical unsupervised change detection in multitemporal hyperspectral images. *IEEE Transactions on Geoscience and Remote Sensing*, 53(1):244–260, 2014.

[35] Francesca Bovolo, Silvia Marchesi, and Lorenzo Bruzzone. A framework for automatic and unsupervised detection of multiple changes in multitemporal images. *IEEE Transactions on Geoscience and Remote Sensing*, 50(6):2196–2212, 2011.

[36] Pol Coppin, Inge Jonckheere, Kristiaan Nackaerts, Bart Muys, and Eric Lambin. Digital change detection methods in ecosystem monitoring: A review. *International Journal of Remote Sensing*, 25(9):1565–1596, 2004.

[37] Thomas S. Huang, George J. Yang, and Greory Y. Tang. A fast two-dimensional median filtering algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 27(1):13–18, 1979.

[38] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *Advances in Neural Information Processing Systems*, pages 109–117, 2011.

[39] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[40] Jagat N. Kapur, Prasanna K. Sahoo, and Andrew K. Wong. A new method for gray-level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics, and Image Processing*, 29(3):273–285, 1985.

[41] Abhijit G. Shanbhag. Utilization of information measure as a means of image thresholding. *CVGIP: Graphical Models and Image Processing*, 56(5):414–419, 1994.

[42] Jui-Cheng Yen, Fu-Juay Chang, and Shyang Chang. A new criterion for automatic multilevel thresholding. *IEEE Transactions on Image Processing*, 4(3):370–378, 1995.

[43] Farid Melgani and Yakoub Bazi. Robust unsupervised change detection with markov random fields. In *IEEE International Symposium on Geoscience and Remote Sensing (IGARSS 2006)*, pages 208–211. IEEE, 2006.

[44] Linzhi Su, Maoguo Gong, Puzhao Zhang, Mingyang Zhang, Jia Liu, and Hailun Yang. Deep learning and mapping based ternary change detection for information unbalanced images. *Pattern Recognition*, 66:213–228, 2017.

[45] Luigi T. Luppino, Filippo M. Bianchi, Gabriele Moser, and Stian N. Anfinsen. Remote sensing image regression for heterogeneous change detection. In *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2018.

[46] Francesca Bovolo and Lorenzo Bruzzone. A theoretical framework for unsupervised change detection based on change vector analysis in the polar domain. *IEEE Transactions on Geoscience and Remote Sensing*, 45(1):218–236, 2007.

[47] Maoguo Gong, Puzhao Zhang, Linzhi Su, and Jia Liu. Coupled dictionary learning for change detection from multisource data. *IEEE Transactions on Geoscience and Remote Sensing*, 54(12):7077–7091, 2016.

[48] Luigi T. Luppino, Filippo M. Bianchi, Gabriele Moser, and Stian N. Anfinsen. Unsupervised image regression for heterogeneous change detection. *IEEE Transactions on Geoscience and Remote Sensing*, 57(12):9960–9975, December 2019.

[49] Roman Katz, Juan Nieto, Eduardo Nebot, and Bertrand Douillard. Track-based self-supervised classification of dynamic obstacles. *Autonomous Robots*, 29(2):219–233, 2010.

[50] Christopher A. Brooks and Karl Iagnemma. Self-supervised terrain classification for planetary surface exploration rovers. *Journal of Field Robotics*, 29(3):445–468, 2012.

[51] Young-Woo Seo, Nathan Ratliff, and Chris Urmson. Self-supervised aerial images analysis for extracting parking lot structure. In *Twenty-First International Joint Conference on Artificial Intelligence*, 2009.

[52] Caglar Senaras and Fatoş T Yarman Vural. A self-supervised decision fusion framework for building detection. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(5):1780–1791, 2015.

[53] Dominik Brunner, Guido Lemoine, and Lorenzo Bruzzone. Earthquake damage assessment of buildings using VHR optical and SAR imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 48(5):2403–2420, 2010.

[54] Gang Liu, Yann Gousseau, and Florence Tupin. A contrario comparison of local descriptors for change detection in very high spatial resolution satellite images of urban areas. *IEEE Transactions on Geoscience and Remote Sensing*, 57(6):3904–3918, 2019.

[55] Redha Touati, Max Mignotte, and Mohamed Dahmane. Change detection in heterogeneous remote sensing images based on an imaging modality-invariant MDS representation. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3998–4002. IEEE, 2018.

[56] Florent Chatelain, Jean-Yves Tourneret, and Jordi Inglada. Change detection in multisensor SAR images using bivariate gamma distributions. *IEEE Transactions on Image Processing*, 17(3):249–258, 2008.

[57] Bård Storvik, Geir Storvik, and Roger Fjortoft. On the combination of multisensor data using meta-Gaussian distributions. *IEEE Transactions on Geoscience and Remote Sensing*, 47(7):2372–2379, 2009.

[58] Jorge Prendes, Marie Chabert, Frédéric Pascal, Alain Giros, and Jean-Yves Tourneret. A new multivariate statistical model for change detection in images acquired by homogeneous and heterogeneous sensors. *IEEE Transactions on Image Processing*, 24(3):799–812, 2015.

[59] Gustavo Camps-Valls, Luis Gómez-Chova, Jordi Muñoz-Marí, José L. Rojo-Álvarez, and Manel Martínez-Ramón. Kernel-based framework for multitemporal and multisource remote sensing data classification and change detection. *IEEE Transactions on Geoscience and Remote Sensing*, 46(6):1822–1835, 2008.

[60] Peijun Du, Sicong Liu, Junshi Xia, and Yindi Zhao. Information fusion techniques for change detection from multi-temporal remote sensing images. *Information Fusion*, 14(1):19–27, 2013.

[61] Jia Liu, Maoguo Gong, Kai Qin, and Puzhao Zhang. A deep convolutional coupling network for change detection based on heterogeneous optical and radar images. *IEEE Transactions on Neural Networks and Learning Systems*, 29(3):545–559, 2016.

[62] Xiao X. Zhu, Devis Tuia, Lichao Mou, Gui-Song Xia, Liangpei Zhang, Feng Xu, and Friedrich Fraundorfer. Deep learning in remote sensing: A comprehensive review and list of resources. *IEEE Geoscience and Remote Sensing Magazine*, 5(4):8–36, 2017.

[63] Xia Li and Anthony G. Yeh. Analyzing spatial restructuring of land use patterns in a fast growing region using remote sensing and GIS. *Landscape and Urban Planning*, 69(4):335–354, 2004.

[64] Martin Herold, Joseph Scepan, and Keith C. Clarke. The use of remote sensing and landscape metrics to describe structures and changes in urban land uses. *Environment and Planning A*, 34(8):1443–1458, 2002.

[65] Salman H. Khan, Xuming He, Fatih Porikli, and Mohammed Bennamoun. Forest change detection in incomplete satellite images with deep neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 55(9):5407–5423, 2017.

[66] Ali A. Alesheikh, Amir Ghorbanali, and Narjes Nouri. Coastline change detection using remote sensing. *International Journal of Environmental Science & Technology*, 4(1):61–66, 2007.

[67] Etienne Berthier, Yves Arnaud, Rajesh Kumar, Sarfaraz Ahmad, Patrick Wagnon, and Pierre Chevallier. Remote sensing estimates of glacier mass balances in the Himachal Pradesh (Western Himalaya, India). *Remote Sensing of Environment*, 108(3):327–338, 2007.

[68] Patrick Griffiths, Patrick Hostert, Oliver Gruebner, and Sebastian van der Linden. Mapping megacity growth with multi-sensor data. *Remote Sensing of Environment*, 114(2):426–439, 2010.

[69] Dengsheng Lu, Paul Mausel, Eduardo S. Brondizio, and Emilio Moran. Change detection techniques. *International Journal of Remote Sensing*, 25(12):2365–2401, 2004.

[70] Martine M. Espeseth, Stine Skrunes, Cathleen E. Jones, Camilla Brekke, Benjamin Holt, and Anthony P. Doulgeris. Analysis of evolving oil spills in full-polarimetric and hybrid-polarity SAR. *IEEE Transactions on Geoscience and Remote Sensing*, 55(7):4190–4210, 2017.

[71] Ross S. Lunetta and Christopher D. Elvidge. *Remote Sensing Change Detection*, volume 310. Taylor & Francis, 1999.

[72] Jean F. Mas. Monitoring land-cover changes: a comparison of change detection techniques. *International Journal of Remote Sensing*, 20(1):139–152, 1999.

[73] Konstantinos Koutroumbas and Sergios Theodoridis. *Pattern Recognition*. Elsevier Science, 2008.

[74] Lonnie Magee. Nonlocal behavior in polynomial regressions. *The American Statistician*, 52(1):20–22, 1998.

[75] Carl E. Rasmussen. Gaussian processes in machine learning. In *Advanced Lectures on Machine Learning*, pages 63–71. Springer, 2004.

[76] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[77] Tom Bylander. Estimating generalization error on two-class datasets using out-of-bag estimates. *Machine Learning*, 48(1-3):287–297, 2002.

[78] David S. Siroky. Navigating random forests and related advances in algorithmic modeling. *Statistics Surveys*, 3:147–163, 2009.

[79] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[80] Devis Tuia, Jochem Verrelst, Luis Alonso, Fernando Pérez-Cruz, and Gustavo Camps-Valls. Multioutput support vector regression for remote sensing biophysical parameter estimation. *IEEE Geoscience and Remote Sensing Letters*, 8(4):804–808, 2011.

[81] Robert Hecht-Nielsen. Neurocomputing: picking the human brain. *IEEE Spectrum*, 25(3):36–41, 1988.

[82] Balázs C. Csáji. Approximation with artificial neural networks. Master's thesis, Eötvös Loránd University, 2001.

[83] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386, 1958.

[84] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[85] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989.

[86] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[87] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[88] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[89] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[90] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.

[91] Jack Kiefer and Jacob Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, 1952.

[92] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[93] Christian Darken and John E. Moody. Note on learning rate schedules for stochastic optimization. In *Advances in Neural Information Processing Systems*, pages 832–838, 1991.

[94] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12(1):145–151, 1999.

[95] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(July):2121–2159, 2011.

[96] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[97] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[98] Timothy Dozat. Incorporating Nesterov momentum into Adam. In *6th International Conference on Learning Representations (ICLR), Workshop Track Posters*, 2016.

[99] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[100] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[101] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *International Conference on Machine Learning (ICML)*, volume 30, page 3, 2013.

[102] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

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

[104] Olaf Ronneberger, Paul Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015.

[105] Geoffrey E. Hinton and Ruslan R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[106] Jan Larsen and Lars K. Hansen. Generalization performance of regularized neural network models. In *IEEE Workshop on Neural Networks for Signal Processing*, pages 42–51. IEEE, 1994.

[107] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.

[108] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[109] Michael Kampffmeyer, Sigurd Løkse, Filippo M. Bianchi, Robert Jenssen, and Lorenzo Livi. The deep kernelized autoencoder. *Applied Soft Computing*, 71:816–825, 2018.

[110] Nina Merkle, Stefan Auer, Rupert Müller, and Peter Reinartz. Exploring the potential of conditional adversarial networks for optical and SAR image matching. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 11(6):1811–1820, 2018.

[111] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2794–2802, 2017.

[112] Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, and David H. Salesin. Image analogies. In *28th Annual Conference on Computer Graphics and Interactive Techniques*, pages 327–340, 2001.

[113] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[114] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2223–2232, 2017.

[115] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[116] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[117] Sifei Liu, Shalini De Mello, Jinwei Gu, Guangyu Zhong, Ming-Hsuan Yang, and Jan Kautz. Learning affinity via spatial propagation networks. In *Advances in Neural Information Processing Systems*, pages 1520–1530, 2017.

[118] Xinjing Cheng, Peng Wang, and Ruigang Yang. Depth estimation via affinity learned with convolutional spatial propagation network. In *European Conference on Computer Vision (ECCV)*, pages 103–119, 2018.

[119] Michael Maire, Takuya Narihira, and Stella X. Yu. Affinity cnn: Learning pixel-centric pairwise relations for figure/ground embedding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 174–182, 2016.

[120] Fan R. Chung and Fan C. Graham. *Spectral Graph Theory*. Number 92. American Mathematical Society, 1997.

[121] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT press, 2009.

[122] Simon J. Sheather and Michael C. Jones. A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the*

Royal Statistical Society: Series B (Methodological), 53(3):683–690, 1991.

[123] Matthew P. Wand and M. Chris Jones. *Kernel Smoothing*, volume 60 of *Monographs on Statistics and Applied Probability*. Chapman & Hall/CRC, 1995.

[124] Jonas N. Myhre and Robert Jenssen. Mixture weight influence on kernel entropy component analysis and semi-supervised learning using the lasso. In *Machine Learning for Signal Processing (MLSP), 2012 IEEE International Workshop on*, pages 1–6. IEEE, 2012.

[125] Arash Heidarian and Michael J. Dinneen. A hybrid geometric approach for measuring similarity level among documents and document clustering. In *2016 IEEE Second International Conference on Big Data Computing Service and Applications (BigDataService)*, pages 142–151. IEEE, 2016.

[126] Douglas B. West. *Introduction to Graph Theory*, volume 2. Prentice Hall Upper Saddle River, 2001.

[127] YP Mack and Murray Rosenblatt. Multivariate k-nearest neighbor density estimates. *Journal of Multivariate Analysis*, 9(1):1–15, 1979.

[128] Redha Touati and Max Mignotte. An energy-based model encoding nonlocal pairwise pixel interactions for multisensor change detection. *IEEE Transactions on Geoscience and Remote Sensing*, 56(2):1046–1058, 2018.

[129] Redha Touati, Max Mignotte, and Mohamed Dahmane. Multimodal change detection in remote sensing images using an unsupervised pixel pairwise-based Markov random field model. *IEEE Transactions on Image Processing*, 29:757–767, 2019.

[130] Redha Touati, Max Mignotte, and Mohamed Dahmane. A reliable mixed-norm-based multiresolution change detector in heterogeneous remote sensing images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(9):3588–3601, 2019.

[131] Redha Touati, Max Mignotte, and Mohamed Dahmane. Anomaly feature learning for unsupervised change detection in heterogeneous images: A deep sparse residual model. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:588–600, 2020.

[132] Maoguo Gong, Xudong Niu, Tao Zhan, and Mingyang Zhang. A coupling translation network for change detection in heterogeneous images. *International Journal of Remote Sensing*, 40(9):3647–3672, 2019.

[133] Xudong Niu, Maoguo Gong, Tao Zhan, and Yuelei Yang. A conditional adversarial network for change detection in heterogeneous images. *IEEE Geoscience and Remote Sensing Letters*, 16(1):45–49, 2018.

[134] Mads A. Hansen. Affinity-guided image-to-image translation for unsupervised heterogeneous change detection. Master's thesis, UiT The Arctic University of Norway, 2019.