Masters Program in Geospatial Technologies

FEW-SHOT LEARNING FOR POST-EARTHQUAKE URBAN DAMAGE DETECTION

Eftychia Koukouraki

Dissertation submitted in partial fulfilment of the requirements for the Degree of *Master of Science in Geospatial Technologies*

NOVA IMS Information Management School
UNIVERSITAT JAUME·I
WESTFÄLISCHE WILHELMS-UNIVERSITÄT MÜNSTER

FEW-SHOT LEARNING FOR POST-EARTHQUAKE URBAN DAMAGE DETECTION

Dissertation supervised by:
Leonardo Vanneschi, PhD
NOVA Information Management School
Universidade Nova de Lisboa
Lisbon, Portugal

Co-supervised by:
Marco Octávio Trindade Painho, PhD
NOVA Information Management School
Universidade Nova de Lisboa
Lisbon, Portugal

Co-supervised by:
Filiberto Pla Bañón, PhD
GEOTEC
Universitat Jaume I
Castellón, Spain

February 21, 2021

Declaration of Authorship

I declare that the work described in this document is my own and not from someone else. All the assistance I have received from other people is duly acknowledged and all the sources (published or not published) are referenced.

This work has not been previously evaluated or submitted to NOVA Information Management School or elsewhere.

Lisbon, February 21, 2021
Eftychia KOUKOURAKI

[the signed original has been archived by the NOVA IMS services]

Acknowledgements

First and foremost, I would like to thank my supervisor, Professor Leonardo Vanneschi, for sharing his knowledge about a scientific field that I just became familiar with, and for being always available to resolve any questions that emerged during the elaboration of this thesis.

For his constructive feedback, orientation and support along the three semesters of the Master's in Geospatial Technologies, I would like to express my gratitude to Professor Marco Painho.

For the interesting conversations and their ideas, recommendations and enthusiasm, I especially thank my colleagues Georgios Lambropoulos, Georgios Epitropou, Pablo Henrique Alves Cruz, Yan-Liang Lin, Filip Loncar and Chukwuemeka Fortune Igwe.

As this journey comes to an end, I find myself more than grateful to all the people that assisted me in any way possible to get on board and to navigate during these turbulent times. To the ones that helped me apply for this course and to these people that accepted me as a scholarship holder. To all the people I met on the way and helped me grow and elaborate my academic, technical and soft skills. To all the new friends I made, and to the old ones that stayed with me all this time, thank you for broadening my horizons in ways that I could never have imagined.

Few Shot Learning for Post-Earthquake Urban Damage Detection

Abstract

Among natural disasters, earthquakes are recorded to have the highest rates in human loss in the past 20 years. Their unexpected nature has severe consequences on both human lives and material infrastructure and demands urgent action. For effective emergency relief, it is necessary to gain awareness about the level of damage in the affected areas. The use of remotely sensed imagery is popular in damage assessment applications, however it requires a considerable amount of labeled data, which are not always easy to obtain. Taking into consideration the recent developments in the fields of Machine Learning and Computer Vision, this thesis investigates and employs several Few-Shot Learning (FSL) strategies in order to address data insufficiency and imbalance in post-earthquake urban damage classification. The contribution of this work is double: we manage to prove that oversampling is the most suitable data balancing method for training Deep Convolutional Neural Networks (CNN) when compared to cost-sensitive learning and undersampling, and to demonstrate the feasibility of Prototypical Networks in a damage classification problem.

Keywords

Few-shot learning

Data balancing

Image classification

Remote sensing

Damage assessment

List of Abbreviations

ANN	**A**rtificial **N**eural **N**etwork
API	**A**pplication **P**rogramming **I**nterface
CNN	**C**onvolutional **N**eural **N**etwork
CRED	**C**entre for **R**esearch on the **E**pidemiology **D**isasters
FSL	**F**ew **S**hot **L**earning
GIS	**G**eographic **I**nformation **S**ystem
GSD	**G**round **S**ampling **D**istance
ML	**M**achine **L**earning
ProtoNets	**Proto**typical **Net**work**s**
RBFNN	**R**adial **B**asis **F**unction **N**eural **N**etwork
ReLu	**Re**ctified **L**inear **u**nit
RF	**R**andom **F**orest
UAV	**U**nmanned **A**erial **V**ehicle
UNDRR	**U**nited **N**ations Office for **D**isaster **R**isk **R**eduction
VHR	**V**ery **H**igh Spatial **R**esolution

Contents

List of Figures

List of Tables

1 Introduction

1.1 Context and Motivation

Earthquakes are events of episodic nature, that can have a grave impact on human life and cause immense property loss. UNDRR & CRED (2019) report on the cost of disasters for the period 2000-2019 declares earthquakes as the deadliest type of disaster for the first two decades of the 21st century and highlights their potential of massive damage to infrastructure. As they are unpredictable, both in terms of time and magnitude, responding appropriately is often critical to minimize the number of casualties. The success of emergency response operations relies in the efficient organizational management and the rapid reaction. A mandatory precondition to fulfill these requirements is to gain Situational Awareness: to know **what** has happened, **when** and **where**. This study intends to apply Machine Learning techniques on Very High Resolution (VHR) satellite imagery to detect earthquake-caused damage in the urban environment.

To identify the locations that require immediate relief, remotely sensed data are common to use, because they can provide an overview of a large region at once, and acquiring them does not pose the same risks as collecting ground-truth data (Gupta et al., 2019). In the context of emergency mapping, remotely sensed data usually refer to imagery acquired via satellite, Unmanned Aerial Vehicles (UAVs) or other aerial platforms (Lollino et al., 2015). As the availability of this kind of data and the processing power of modern computational systems have been constantly increasing, the possibility of automating the identification and assessment of post-disaster damage is also being explored (Kakooei and Baleghi, 2017). For this reason, methods encountered in the field of Computer Vision are employed, aiming to minimize the time overhead and the error that is introduced by the human factor (Xu et al., 2019).

The recent studies have proved that Machine Learning algorithms outperform traditional Remote Sensing techniques in assessing earthquake damage (Weber and Kané, 2019). The methodology to be followed by this study is based on Few-Shot Learning

(FSL), which is a type of Machine Learning problems, where the available data for the target classification task contain only a limited number of examples with supervised information (Wang et al., 2019). Since a destructive seismic event is rarely to happen, but the need of reacting quickly in such case is continuous, FSL is competent when it comes to extracting knowledge from a narrow amount of data and thus, the required effort for data gathering is also reduced. However, a FSL problem is not easy to solve. The lack of data requires a different approach than other Machine Learning problems, which rely on having a plethora of samples to train the model. The suggested solutions may vary in terms of algorithm, model parameters and data handling (Wang et al., 2019).

One of the main obstacles to overcome in the identification of earthquake damage is the small number of training samples (Ji, Liu, and Buchroithner, 2018). Machine Learning approaches have been employed to locate and measure post-earthquake urban damage, confirming the relevance of this field for such kind of applications. However, related research papers that have incorporated FSL for locating ravaged buildings, focus on binary classification (destroyed/non-destroyed), rather than further dividing buildings into different wreckage levels. Multi-class categorization, though, can emphasize or even create class imbalances within the data. The present study seeks to fill this gap, by leveraging FSL to tackle data deficiency for certain classes in a multi-class problem.

1.2 Objectives and Research Questions

There are several means of dealing with data deficiency and imbalance. Applications that track disaster-related damage with ML can benefit from the existence of pre-event data, but on some occasions, they may be impossible to acquire. For this reason, we examine how efficient can be a model that is based only on post-event data. The purpose of this study is to implement and evaluate the effect of different FSL approaches on an imbalanced dataset.

The Research Questions for which we seek to meet an answer are the following:

- How can the supervised classification of a highly imbalanced dataset be elaborated, where the instances of one class are multiple times higher than the other classes?

- In cases of imbalanced datasets, to what extent are the representatives of the majority and minority classes successfully detected?
- If overlaid with a map, is the visual interpretation of the predictions indicative of the severity of damage suffered by the region?

The answers will be achieved by accomplishing the following objectives' skeleton:

- Review the existing literature and gather applied practices in similar applications.
- Assess the relevant methods and implement the architectures that suit the needs of the thesis.
- Measure the performance and the training time of the implemented approaches and provide quantitative and qualitative evaluations.

1.3 Thesis Structure

This document presents our research in six Chapters. Chapter 1 introduces the reader to the topic, provides the motivation and contextual background of the work and lists the objectives and research questions to fulfill. Chapter 2 reviews related studies, tries to identify the major variables that affect the research approach and analyzes the theoretical background that is necessary to follow the present study. Chapter 3 explains the data and the methodological workflow of the thesis. Chapter 4 presents and discusses the results of the study and Chapter 5 summarizes the the main conclusions drawn.

2 Literature Review

2.1 Remote Sensing in Emergency Mapping

Boccardo and Tonolo (2015) have elaborated a systematic review concerning the role of Remote Sensing in post disaster damage mapping. Based on their paper, satellite data are overall preferred to material acquired by other platforms, as they provide an overview of large regions, even if they cannot be easily accessed. They also state that optical imagery is favored for damage estimation, as it permits finer spatial resolution and is semantically richer, which is crucial in operational conditions. The existence of pre-event data is mentioned to add up to the quality of collapsed building detection. The authors also name some limitations of optical satellite imagery, such as off-nadir acquisition angles causing low spatial resolution, spectral and geometric resolution correlation and short stopover time frames, and propose the use of emerging technologies, such as object detection, as part of the solution.

According to Cooner, Shao, and Campbell (2016), Machine Learning algorithms seem to gain popularity in damage assessment applications, due to outperforming traditional methods for change detection and image classification and due to being capable of handling non-linear datasets. The authors compared RF, ANN and RBFNN algorithms on panchromatic and multispectral VHR imagery obtained by the satellites WordView-1 and QuickBird-2. The study concluded that ANN demonstrates the lowest Error of Omission and the shortest training time, while the model could be produced only with panchromatic imagery.

Li et al. (2020a) did an extensive survey about object detection in optical Remote Sensing images. As stated in the paper, Deep Learning algorithms are currently the predominant approach for visual recognition tasks, including object detection related to the fields of Computer Vision and Earth Observation. Although in Computer Vision the employed methods can be region proposal-based or regression-based, Earth Observation applications favor the first approach (Li et al., 2020a). The study also proposed a new benchmark dataset, characterized by low inter-class and high intra-class

variability and tested it with different combinations of backbone and object prediction architectures, stating that deeper backbone networks, such as ResNet-101 and Hourglass-104, demonstrate higher overall accuracy.

2.2 Deep Learning for Urban Damage Assessment

Ji, Liu, and Buchroithner (2018) applied a CNN architecture called SqueezeNet on single-temporal post-earthquake VHR QuickBird imagery. The study divided the buildings in the city of Port-au-Prince after Haiti 2010 incident in two categories: collapsed and non-collapsed. As the non-collapsed buildings outnumbered the completely destroyed ones, the researchers used three different data balancing methods, namely random oversampling, random undersampling and cost-sensitive to improve the accuracy.

Ji et al. (2020) have shown the potential of pre-trained CNN models for post-earthquake damage identification. Two models were compared, one trained from scratch and one pre-trained on the benchmark dataset ImageNet, with the accuracy results favoring greatly the latter. Both models were fed with labeled bi-temporal VHR imagery and were responsible for binary classification of the buildings: collapsed and non-collapsed. The study also points out the risk of overfitting when the data are limited and makes use of data augmentation to figuratively expand the dataset.

Li et al. (2019) have also derived similar results when comparing a fine-tuned pre-trained model with one trained from scratch, for identifying post-hurricane structural damage. In contrast with Ji et al. (2020), this study used single post-event aerial imagery for classifying the damage in two different levels: damaged and debris.

Xu et al. (2019) did a comparative study of four different models using the Haiti 2010 dataset. Three of the models were built using both pre- and post-disaster labeled images and one was built using single post-earthquake data. The study tested the generalization ability of the best performing model and found it more competent as the number of earthquake incidents that are included in the training increases. The reason is the low variability that characterizes each individual incident dataset, a fact that can lead to overfitting (Xu et al., 2019).

The need for a common framework for building damage assessment after different kinds of natural disasters (earthquakes, floods, hurricanes, wildfires, volcanic eruptions, etc.) has been highlighted by Gupta et al. (2019). The study discusses the need

for having a benchmark dataset that is compliant with the requirements of ML algorithms, counting in the rarity of occurrences of a large-scale natural disaster and hence the relevant data shortage. The study's main contribution is a dataset that consists of bi-temporal VHR labeled satellite imagery from various disaster incidents.

Weber and Kané (2019) have exploited the aforementioned dataset (xBD) to estimate post-disaster building damage. The pre- and post-disaster images were input to a backbone based on Mask R-CNN and then the buildings-features were classified according to the damage level using semantic segmentation.

The transferability of pre-trained CNN models to new disaster occasions was also examined by Vetrivel et al. (2018). The study integrated 3D point cloud data on top of vertical and oblique aerial photography. Among the models that were employed, the authors recommend the use of a pre-trained CNN as a feature extractor with no further weight tuning, because it can achieve Overall Accuracy of the same levels, without being as costly in terms of data and computation. However, the inclusion of site-specific samples can impact positively the model's performance (Vetrivel et al., 2018).

Li et al. (2020b), taking into consideration the amount of time that labeled data needs to be produced, have employed an unsupervised domain adaptation model based on unlabeled post-hurricane imagery. The model, despite its complexity, as it consists of several Generative Adversarial Networks (GANs), a classifier and a self-attention module, was evaluated by the authors as successful with regards to the transfer learning tasks that were assigned to it.

2.3 Comparative Analysis

The aforementioned studies concern related applications, where post-disaster urban damage is tracked with Machine Learning. The approaches vary greatly with regards to the methods employed. The input imagery and the number of the predicted classes are parameters that affect the design of the learning pipeline. Table 2.1 summarizes the six parameters that were encountered in the literature and were evaluated as fundamental for the creation of the predictive model.

Different incident types affect urban structures differently (Gupta et al., 2019). However, we assume that earthquake and wind induced damages are comparable, because they impact on the structural materials in a similar way, causing lateral damage to the

Study	Incident type	Imagery	Dataset size	Image size	Temporality	Classes
Cooner et al. (2016)	Earthquake	Satellite	900		Bi-temporal	2
Ji et al. (2018)	Earthquake	Satellite	3928	96x96	Single post-event	2
Vetrivel et al. (2018)	Earthquake	Airborne (oblique)	12544	100x100	Bi-temporal	2
Xu et al. (2019)	Earthquake	Satellite	80000	161x161	Single post-event	2
Li et al. (2019)	Hurricane	Airborne	500	512x512	Single post-event	3
Weber and Kané (2019)	All (xBD)	Satellite	700000 (xBD)	512x512	Bi-temporal	5
Gupta et al. (2019)	All (xBD)	Satellite	700000 (xBD)	128x128	Single post-event	4
Ji et al. (2020)	Earthquake	Satellite	1789	96x96	Bi-temporal	2
Li et al. (2020b)	Hurricane	Airborne	34000	200x200	Single post-event	3

TABLE 2.1: Data parameters that affect the architecture design. Dataset size refers to building instances. Image size refers to pixels and concerns the input to the classification module.

constructions (Taher, 2010; Fannela and Munshi, 1998). For this reason, hurricane-related studies are also included in the literature review.

The most common incident among earthquake-related studies is the Haiti 2010 earthquake and has often been the unique data source of the analysis (Cooner, Shao, and Campbell, 2016; Ji, Liu, and Buchroithner, 2018; Ji et al., 2020). Given that the models built on very specific data have a poor generalization ability (Xu et al., 2019), newer studies are incorporating more earthquake incidents (Vetrivel et al., 2018; Xu et al., 2019) or do not distinguish between the damage cause, especially the ones utilizing the dedicated xBD dataset (Gupta et al., 2019; Weber and Kané, 2019).

The preferred imagery type in the relevant Machine Learning applications is of VHR, acquired either by satellite (usually WorldView and QuickBird) or by aerial platforms. Additional data sources, such as 3D point cloud features (Vetrivel et al., 2018), can be used collaterally, but the basis for the learning process remains optical imagery, which is also meaningful to human vision.

To overcome possible data shortage and accelerate the creation of a competent predictive model, a plethora of approaches have been put to use. Pre-trained models can transfer knowledge and save time and computational resources (Vetrivel et al., 2018; Ji et al., 2020). Data augmentation is necessary for small datasets (Li et al., 2019) and data balancing for non-linear datasets (Ji, Liu, and Buchroithner, 2018). Unsupervised classification is also gaining popularity (Li et al., 2020a), as it minimizes the effort for labeling the training data. All the aforementioned strategies are possible solutions of a FSL problem.

FSL has already been introduced as a means of dealing with emergency situations (Choi and Lee, 2019; Balamurugan and Zakhor, 2019; Liu et al., 2019). However, the related studies do not address post-disaster emergency mapping explicitly, but rather focus on video surveillance (Liu et al., 2019), tweet classification (Choi and Lee, 2019) or indoors safety (Balamurugan and Zakhor, 2019).

2.4 Few Shot Learning

The efforts to systematize FSL as a distinct branch of Machine Learning are very recent. The proposed definitions for FSL converge to it being a family of methods for solving Machine Learning problems that are characterized by a small quantity of available labeled data. Since the human brain is capable of learning only from a few examples,

FSL can be seen as a way for Artificial Intelligence to approximate human learning even more (Wang et al., 2019).

According to the taxonomy proposed by Kadam and Vaidya (2020), the coping strategies can be divided in two categories: data-bound and learning-bound. Data-bound strategies focus on attaining more data, so that the sample is big enough to leverage standard Deep Learning network architectures. This can be achieved by transforming the existing data, by creating artificial new data or by incorporating similar datasets. Data augmentation is the most common example of a data-bound strategy.

Wang et al. (2019) further divides the learning-bound methods depending on how the error rate of the learning process is minimized, into model- and algorithm-based. The model-based approaches aim to narrow down the hypothesis space, so that new unlabeled data can be identified based on similarity. This is also called metric-learning (Kadam and Vaidya, 2020). Algorithm-based approaches make use of knowledge acquired by similar learning problems and adjust it accordingly. Pre-trained models are relevant examples that were encountered in the literature.

2.5 Data Balancing

As concluded in the previous Sections, data augmentation is common practice when it comes to datasets that do not contain enough labeled examples to train a Deep Learning model. Although state-of-the-art CNN architectures can achieve very high levels of accuracy, they are usually benchmarked with datasets that contain a massive amount of distinct class instances, such as ImageNet (Mikołajczyk and Grochowski, 2018). However, in real-life scenarios, a plethora of labeled image data is not always the case and furthermore, the number of representatives for every class can be highly imbalanced. Hence, data augmentation and data balancing modules are common in ML applications.

According to Branco, Torgo, and Ribeiro (2015), Data Balancing approaches can be categorized as following: 1) Data Pre-Processing, 2) Special-purpose Learning Methods, 3) Prediction Post-processing and 4) Hybrid. Data Pre-Processing methods involve solutions that transform the data distribution in the pre-processing stage and their main advantage is that they can be integrated to any existing architecture.

The Data Pre-Processing methods can be further divided into *resampling*, *active learning* and *weighting the data space* (Branco, Torgo, and Ribeiro, 2015). In resampling, the

training examples are modified so that the classes' representatives are proportionate. Oversampling and undersampling are typical resampling strategies. Taking the concept of resampling one step further, active learning methods are able to choose which samples are more valuable for the learning process and use the rest to improve the performance. Weighting the data space methods, also known as cost-sensitive learning (Fernández et al., 2018), manipulate the misclassification costs so that the error is minimized. Cost-sensitive learning can also be part of the post-processing stage.

2.5.1 Resampling

Oversampling equalizes the classes by creating new, artificial examples for the minority or non-majority classes. On the other hand, undersampling balances the dataset by randomly eliminating examples of the majority or non-minority classes (Branco, Torgo, and Ribeiro, 2015). Both methods are uncomplicated and easy to implement, yet their simplicity can hold certain drawbacks. Undersampling may exclude very informative data, while oversampling may lead to overfitting (Fernández et al., 2018). Nevertheless, the relatively low computational overhead is also an asset for solving problems of strict time constraints.

2.5.2 Balance class weights

Cost-sensitive learning in the stage of pre-processing aims to modify the training set usually by weighting the classes according to a cost matrix. The cost matrix can either be provided by an expert or derived by estimations made on the training data. The importance of the majority and minority objects is tuned based on this matrix, so the heuristic approach that creates the matrix is decisive for the classification results (Fernández et al., 2018).

For the purposes of this study, the weight w_i of class i is normalized based on Equation 2.1:

$$w_i = \frac{|S|}{m \times |S_i|} \tag{2.1}$$

where m is the number of classes, $|S|$ is the number of examples in the dataset and $|S_i|$ is the number of representatives for this specific class. This formula follows the *Sklearn API documentation* for computing balanced class weights.

2.6 Convolutional Neural Networks

The term *Deep Learning* refers to ANN architectures with long chains of computational stages (Schmidhuber, 2014). One of the most widely used deep neural networks is the Convolutional Neural Network (CNN), which has been particularly successful in Computer Vision tasks (Albawi, Mohammed, and Al-Zawi, 2018), such as object detection and image classification.

A typical CNN architecture is principally composed of three types of layers: convolutional, pooling and fully-connected. Simply put, a CNN architecture is a stack that has been formed from the aforementioned layers (O'Shea and Nash, 2015), as can be observed in the example of Figure 2.1.

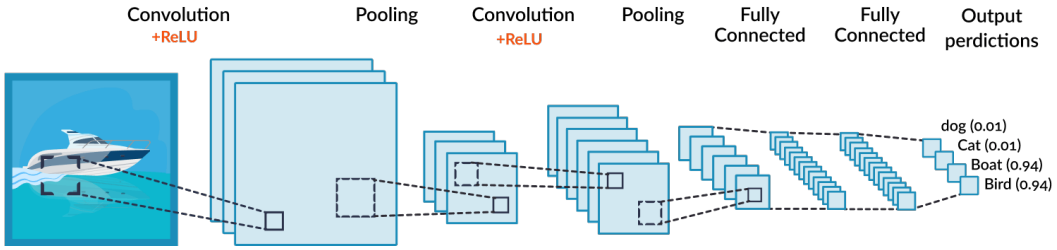

FIGURE 2.1: CNN architecture example (*Convolutional Neural Network Tutorial*).

- A **convolutional** layer applies a filter to the input image, in order to create a map that contains the features of interest. This is achieved by calculating the dot product between the input pixel subset and a matrix of weights and applying to it an activation function via the Rectified Linear unit (ReLu) (O'Shea and Nash, 2015).
- In **pooling** layers, the value of the feature map at a certain point is replaced with a summary statistic of its neighborhood (Goodfellow, Bengio, and Courville, 2016). This can be considered as a form of downscaling, since the input resolution is reduced, preserving only those characteristics that matter to the classification process (Albawi, Mohammed, and Al-Zawi, 2018).
- **Fully-connected** layers operate similarly to ANN nodes. Each node in a fully-connected layer is directly connected to every node in the two adjacent layers (Albawi, Mohammed, and Al-Zawi, 2018). Their responsibility is to produce class scores from the activations, to be used for classification (O'Shea and Nash, 2015).

In our case, the intention is to transform RGB vertical building images to a 4-level scale of damage. As such, subregions of the input image pervade the alternating convolution and pooling layers until the feature map of the input image is finalized. This is the feature extracting block of the CNN (Kim, 2017). The feature map is supposed to preserve only these imagery properties that are beneficial to our classification. The fully-connected layers at the end of the chain are responsible for calculating the probability of belonging to each of the 4 output classes.

2.6.1 ResNet

ResNet was introduced by He et al. (2016) as a proposal to minimize the momentum in error decrease induced by the several layers of backpropagation in deeper architectures. This problem has been alleviated with the introduction of a new layer, called the *Residual Block*. The number of layers (depth) in a ResNet is usually mentioned together with the name of the architecture. Examples of commonly used ResNet variants are ResNet36, ResNet50 and ResNet101.

By definition, a residual is the error in the result of a mathematical operation. In the context of neural networks, this means that if the residual is added to the prediction, the resulted value matches the actual.

Figure 2.2 illustrates the functionality of a Residual Block. Variable x stands for the prediction. If x is off by a margin from the actual value, function $F(x)$ will be triggered to produce the residual of the operation and correct the prediction to meet the true value. If x is already equal to the true value, $F(x)$ will produce 0. The identity function just copies x.

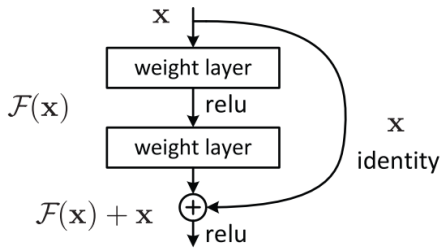

FIGURE 2.2: A residual block (He et al., 2016).

If we suppose $H(x)$ as the underlying mapping function to be learned by the block, we explicitly let these layers approximate a residual function $F(x) := H(x) - x$. Mathematically, the functionality of the residual block is described by equation 2.2:

$$y = F(x, \{W_i\}) + x \tag{2.2}$$

where x and y are the input and output vectors respectively of the referred layers and function $F(x, \{W_i\})$ represents the residual mapping to be fit.

2.7 Prototypical Networks

While deep networks like ResNet described previously can be predictive powerhouses when there are plenty of training data, opting for a deep architecture when the data are not enough may not be the most reasonable option. Among the state-of-the-art FSL-specific algorithms, Prototypical Networks have been proved to detect even new classes, that are not part of the training data.

Prototypical Networks combine elements from the fields of *Meta-learning* and *Metric learning*. Meta-learning is the subfield of ML that leverages experience acquired by similar tasks to accelerate the learning process of new tasks. For this reason, meta-learning is also referred to as *learning to learn* (Vanschoren, 2018). In Prototypical Networks, this is achieved by measuring distances between features, hence learning the metric space. The basic idea is to create a prototype (i.e. a vector mean) of each class and categorize the input feature based on the distance between the two. This distance is actually the "metric" in metric learning.

Hereinafter, we will present the theory behind Prototypical Networks, as introduced by Snell, Twitter, and Zemel (2017).

1. Let us suppose a set of N labeled examples $S = (x_1, y_1), ..., (x_N, y_N)$, where each $x_i \in R^D$ is the D-dimensional feature vector of an example and $y_i \in 1, ..., K$ is the corresponding label. S_k denotes the set of examples labeled with class k.
2. Training episodes are formed by randomly selecting a subset of classes from S, then choosing a subset of examples within each class to act as the **support** set and a subset of the remainder to serve as **query** points.

3. An M-dimensional representation $c_k \in R^M$ is computed for every class. This is the **prototype**. The classes' prototypes are calculated with the help of an embedding function $f_\phi : R^D \rightarrow R^M$, where ϕ represents the learnable parameters. The result is the mean vector of the embedded support points of the specific class:

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(x_i, y_i) \in S_k} f_\phi(x_i) \tag{2.3}$$

4. Assuming a distance function $d : R^M \times R^M \rightarrow [0, +\infty)$ and a query point x, a distribution based on a softmax over distances from the class prototypes in the embedding space is produced. As such, the probability of a query point belonging to a specific class is calculated as following:

$$p_\phi(y = k|\mathbf{x}) = \frac{exp(-d(f_\phi(\mathbf{x})\, \mathbf{c}_k))}{\sum_{k'} exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_{k'}))} \tag{2.4}$$

5. Consequently, the learning process aims to minimize the negative log-probability $J(\phi) = -log p_\phi(y = k|\mathbf{x})$ of the true class k through Stochastic Gradient Descent (SGD).

A graphical representation of the above procedure is provided in Figure 2.3, where class prototypes c_1, c_2, c_3 are computed and then their distance to query point x in the metric space is calculated.

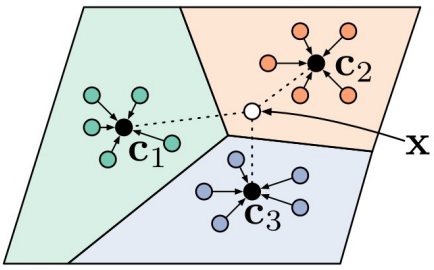

FIGURE 2.3: Prototypical Network functionality (Snell, Twitter, and Zemel, 2017).

2.8 Performance Metrics

The result image classification in Remote Sensing applications is usually evaluated with Overall Accuracy, which represents the fraction of correctly classified instances or pixels in a map (Waldner et al., 2019). However, Overall Accuracy is by definition biased against the minority/non-majority classes and relying entirely on this metric can lead to ambiguous conclusions (Branco, Torgo, and Ribeiro, 2015). For this reason, for the evaluation of the elaborated models we have incorporated *Precision*, *Recall* and *F-score* metrics. In order to clarify the definitions of these metrics, we suppose a confusion matrix for a binary classification problem (see Table 2.2).

		Predicted	
		Positive	**Negative**
Actual	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

TABLE 2.2: Generic Confusion Matrix for binary classification.

Based on Table 2.2, the relevant formulas for calculating the metrics are as following (Branco, Torgo, and Ribeiro, 2015):

- **Precision** represents the positive predictive value:

$$PP_{value} = \frac{TP}{TP + FP} \tag{2.5}$$

- **Recall** or Sensitivity represents the True Positive rate:

$$TP_{rate} = \frac{TP}{TN + FN} \tag{2.6}$$

- **F-score** is a combination of precision and recall. It ranges between 0 (worst) and 1 (best) and is defined accordingly:

$$F = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision} \tag{2.7}$$

For a multi-class classification problem, the confusion matrix is obtained in a similar fashion, but the concepts of *True Positive*, *True Negative*, *False Positive* and *False Negative* are modified to correspond to the nature of a multi-class classification problem and hence these values are not global, but differ depending on the class they are calculated for.

		Predicted		
		$c_0 ... c_{k-1}$	c_k	$c_{k+1} ... c_n$
Actual	$c_0 ... c_{k-1}$	TN	FP	TN
	c_k	FN	TP	FN
	$c_{k+1} ... c_n$	TN	FP	TN

TABLE 2.3: Confusion Matrix for class k in an n-class classification problem.

For optimizing the model's weights during training, the categorical crossentropy loss function was employed. This metric gives an estimation of how distinguishable different discrete probability distributions are from each other and is calculated according to 2.8:

$$loss = -\sum_{i=1}^{m} y_i \cdot log\hat{y}_i \tag{2.8}$$

where $\hat{y}_i$ is the i-th class in the model output, y_i is the corresponding label and m is the total number of classes.

3 Data and Methodology

3.1 Dataset

The data used use for the needs of this research project has been proposed by Gupta et al. (2019) and can be found at https://xview2.org/. This dataset is based on Maxar/ DigitalGlobe's Open Data program and consists of imagery that has been acquired with WorldView and GeoEye satellites. The mean technical specifications of the xBD imagery, as declared in the dataset's metadata, are presented in Table 3.1.

Sensor Resolution	0.66 m
GSD	2.65 m
Off-nadir angle	28.4 degrees
Sun azimuth angle	143.6 degrees
Image Width	1024 pixels
Image Height	1024 pixels

TABLE 3.1: xBD image metadata

The complete xBD dataset refers to multiple disaster categories: volcanic eruption, hurricane, earthquake, fire, flood, monsoon, tornado and tsunami. The dataset is composed by three different file types: 1) VHR satellite imagery (pan-sharpened), 2) JSON files with metadata and labels about the buildings of the region and 3) Building polygons. All three categories contain both pre-and post-disaster data.

The present research is primarily concerned with earthquake-related urban damage, so, for the creation of the predictive models, the earthquake incidents were initially isolated. This led to the collection of 386 images in total, all of which refer to Mexico City 2017 earthquake. As this amount of data is insufficient to train a deep network, data from hurricane incidents were also incorporated and the total amount of images increased to 4432. All images are of the same standard dimensions (1024x1024 pixels) and spatial resolution, thus the pre-processing effort is minimized.

3.1.1 Exploratory Data Analysis

Before designing the methodology to be followed, it is necessary to get familiar with the dataset. The total of 4432 images can be split in two, based on the temporality: pre- and post-event. As mentioned before, the VHR images come with respective labels for the buildings represented within their extent. However, the images referring prior to the disaster, have plain building labels, while the post-disaster building labels are further classified into four levels of damage: 1) *No damage*, 2) *Minor damage*, 3) *Major damage* and 4) *Destroyed*. An example of the imagery overlaid with the building annotation is provided in Figure 3.1.

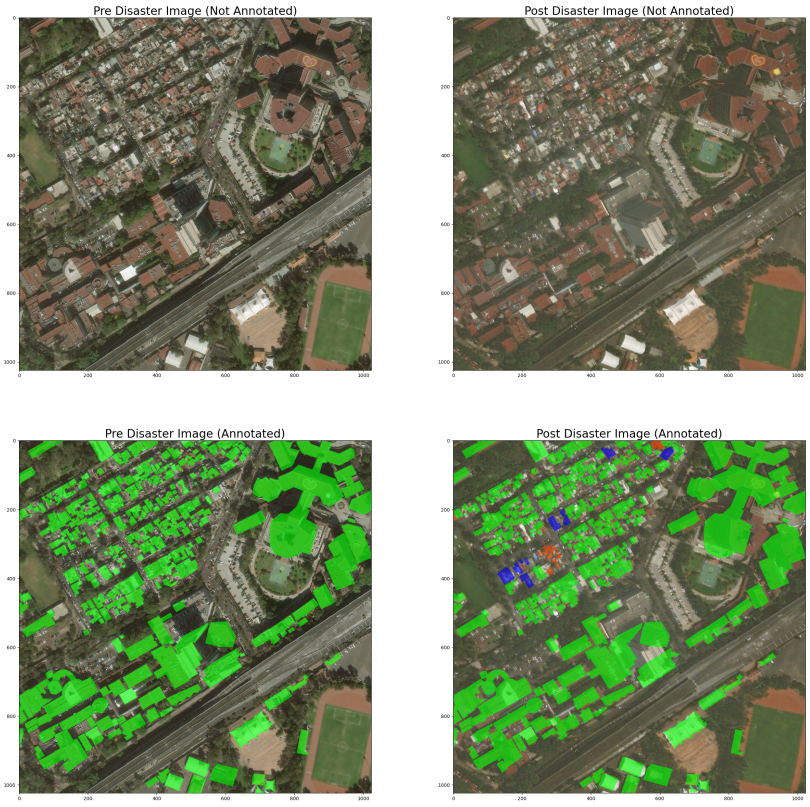

FIGURE 3.1: Top left: Pre-disaster pansharpened image, Bottom left: Pre-disaster image with building annotations, Top right: Post-disaster pansharpened image, Bottom right: Post-disaster image with damaged building annotations. Green stands for *No damage*, blue stands for *Minor damage* and red stands for *Major damage*.

The number of instances for each class is summarized in in Table 3.2.

Class	Number of Instances
No damage	108.157
Minor damage	25.586
Major damage	21.017
Destroyed	5.681

TABLE 3.2: Number of examples per class.

The disaster incidents referred to by the VHR imagery eventually collected from xBD, are presented in Figure 3.2.

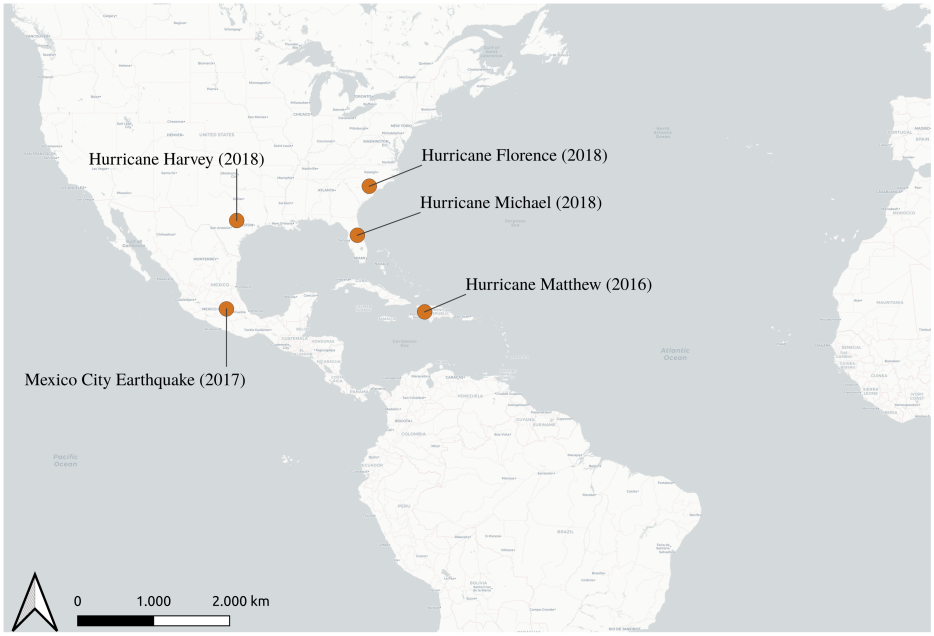

FIGURE 3.2: The disaster incidents in our dataset.

3.2 Methodology

The aim of this thesis is to implement and compare different FSL approaches. For this purpose, four models were developed. In Figure 3.3, we provide an overview of the workflow, from the stage pre-processing the labeled satellite imagery to the stage of producing maps with damage predictions. Given the theoretical fundamentals clarified in Chapter 2, the elaborated methodology is targeted to tackle models

for imbalanced datasets. The following Sections describe in detail the methodological components.

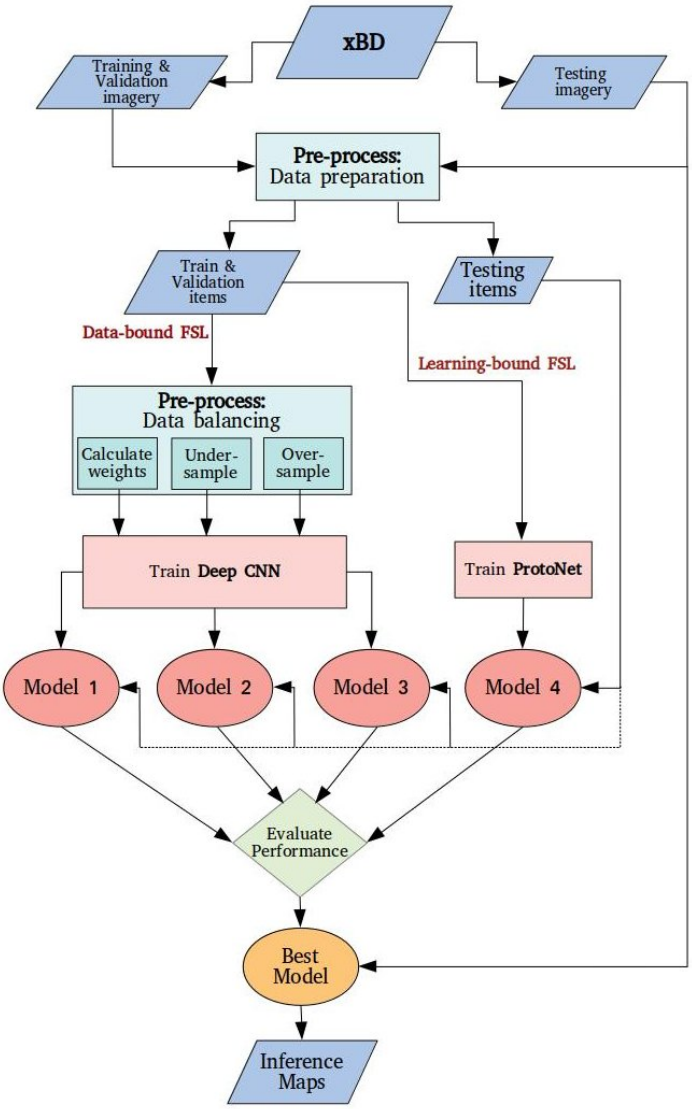

FIGURE 3.3: Methodology overview.

3.2.1 Data preparation

In the context of this study, preprocessing includes the preparation of the data for training and testing the models. Images of dimensions 1024x1024 are not only extensive for our available computational resources, but they also do contain multiple

examples of possibly every class per image. Therefore, all building instances were cropped, as in the example of Figure 3.4, and then mapped to the corresponding label using a .csv file.

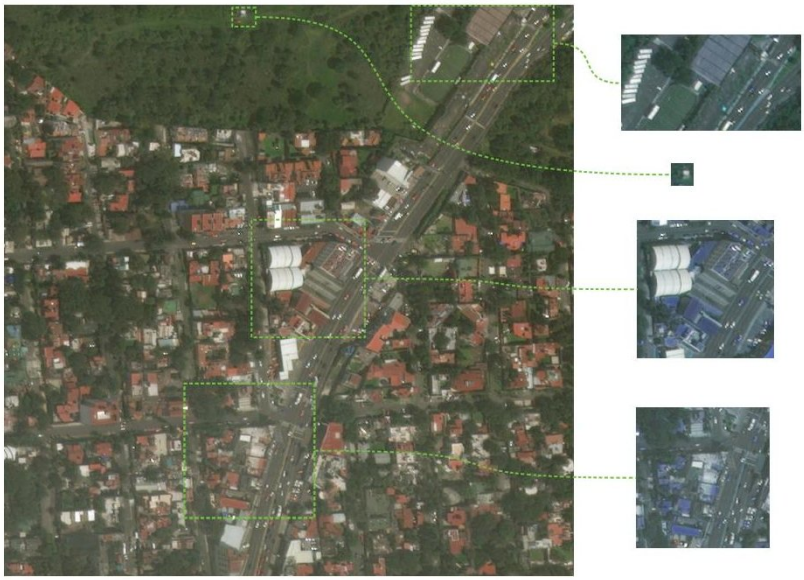

FIGURE 3.4: Isolation of labeled building instances.

The resulting dataset was further split into three parts: *train*, *validation* and *test*. *Validation* and *test* divisions were set to equal the 0.2 of the initial dataset. However, the *test* division was forced to have equal number of representatives for each class, so that the performance evaluation represents fairly the models' predictability. The derived number of examples per class is summarized in Table 3.3.

	Train	Validation	Test
No damage	64.722	21.502	1.000
Minor damage	15.575	5.304	1.000
Major damage	13.116	4.370	1.000
Destroyed	3.487	1.125	1.000

TABLE 3.3: Number of examples per class in *train*, *validation* and *test* subsets.

The subsequent analysis follows two distinct FSL paths: data-bound and learning-bound. All architectures require 3-channel 128x128 images as input. To ensure that the input images comply with the appropriate format, downscaling was enforced where necessary.

3.2.2 Data-bound FSL

As mentioned in Chapter 2, data-bound methods aim to extend the dataset, so that traditional Deep Learning architectures can be leveraged. In this context, three models were developed, with the baseline architecture being the same for all of them.

Baseline Architecture

This architecture is a reproduction of Gupta et al. (2019) baseline, which was released for the complete xBD dataset. For the training procedure, batches of 64 images of dimensions 128x128x3, are fed to a Deep Neural Network that consists of two main components: a shallow CNN and a ResNet50 model. The **shallow CNN** is composed of six alternating layers of Convolutional and Pooling blocks. The **ResNet50** block is initialized with *ImageNet* weights. Finally, the outputs of the two chunks are concatenated and fed to three adjacent fully-connected layers. The output of the last fully-connected layer holds the predicted classes and is compared to the corresponding labels to eventually compute the performance metrics. The Convolutional and the fully-connected layers are ReLu activated. A graphical interpretation of the network's architecture can be observed in Figure 3.5.

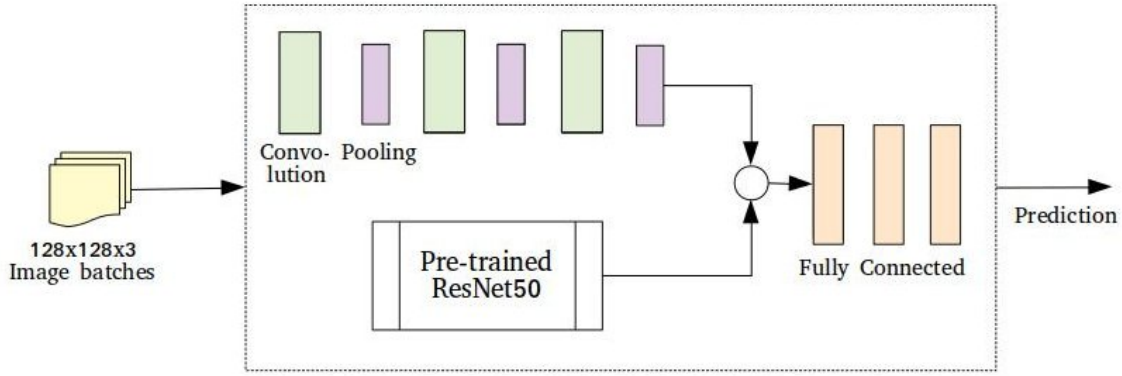

FIGURE 3.5: Deep Neural Network baseline architecture.

Model 1 - Balancing the class weights

The first model we implemented aims to smooth the difference among the classes by balancing the training weights. The class weights are calculated with the Equation 2.1

and the results are demonstrated in Table 3.4.

Class	Weight
no-damage	0.37
minor-damage	1.56
major-damage	1.85
destroyed	6.95

TABLE 3.4: Input class weights for Model 1.

The methodological workflow of the approach is outlined in Figure 3.6. Before training the model, the *train* subset is subjected to Data Augmentation. New images are produced from the initial ones, using vertical flip, dimension shift and rescale. After the training process, the predictive model is used for the calculation of the performance metrics.

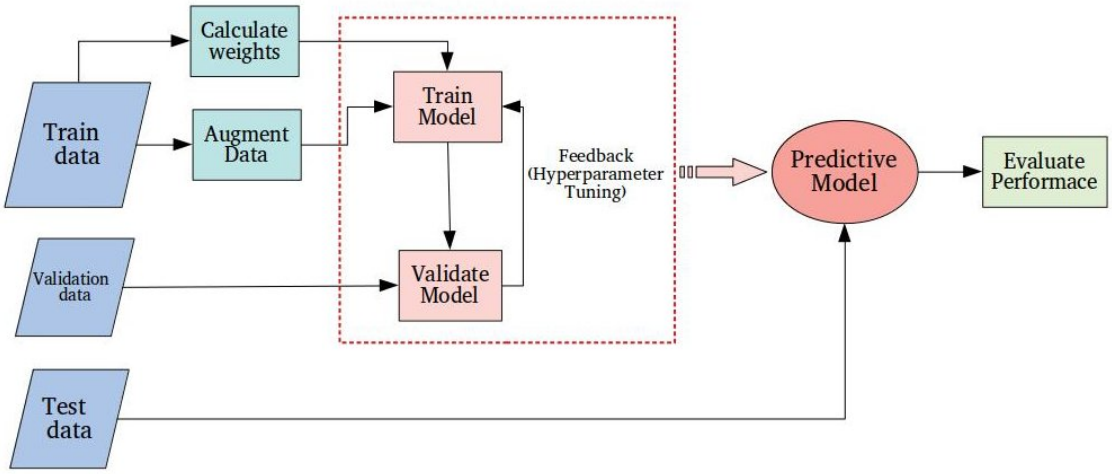

FIGURE 3.6: Methodological Workflow of Architecture 1 - Balancing the class weights.

The learning rate is initially tuned to 0.0001 and the decay is handled by an Adam optimizer, with beta parameters set to $\beta_1 = 0.9$ and $\beta_2 = 0.99$.

Model 2 - Undersampling

The second model considered for damage classification is depicted in Figure 3.7. The primary difference is the way in which we try to overcome the data imbalance. Examples of the dataset presented in Table 3.3 are randomly selected so that each class remains with the same number of representatives for the learning process. The undersampling resulted in 4.600 examples per class, 0.3 of which create the *validation* set. The undersampled *train* dataset underwent data augmentation, too.

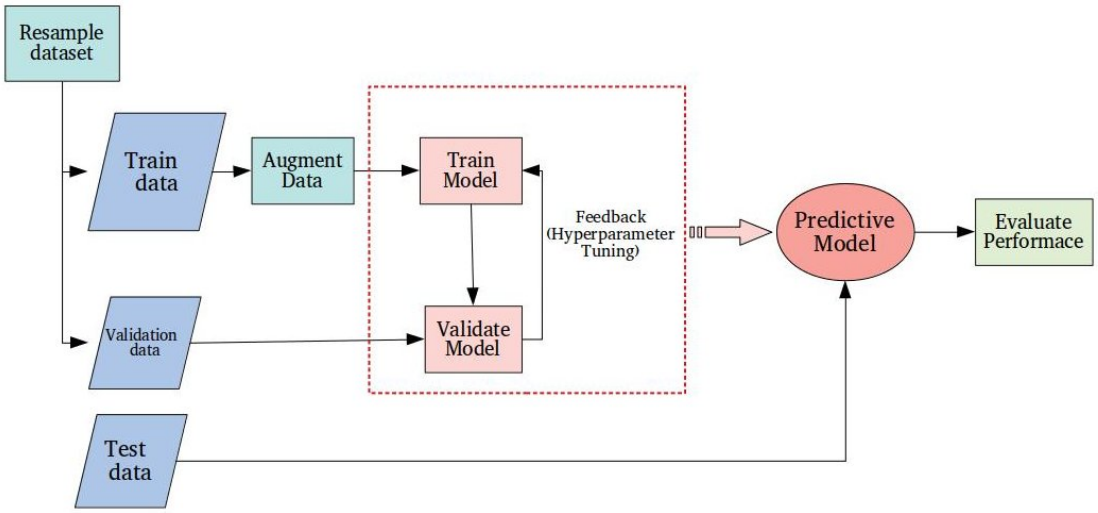

FIGURE 3.7: Methodological Workflow of Architectures 2 and 3 - Resampling. The resampling module can stand either for under- or oversampling.

Since the data are decreased for the creation of this model, the learning rate was also decreased to 0,00001. The Adam optimizer parameters were kept the same ($\beta_1 = 0.9$, $\beta_2 = 0.99$).

Model 3 - Oversampling

In the same manner as undersampling, but from the opposite perspective, oversampling's purpose is to even the classes by creating more examples of the non-majority classes. In this case, the oversampling is handled by applying simple transformations,

namely flipping and rotation, to the existing examples. The methodological workflow is again illustrated in Figure 3.7. The initial learning rate is 0.0001 and the Adam optimizer parameters are set to $\beta_1 = 0.9$ and $\beta_2 = 0.99$.

3.2.3 Learning-bound FSL: Model 4 - Prototypical Networks

The last candidate follows a different FSL strategy. The model was implemented as can be observed in the schematic of Figure 3.8. From the dataset of Table 3.3, 50 examples of each class were selected to support the training process. This creates a 4-way (as the total number of classes), 50-shot (as the number of examples per class) FSL approach.

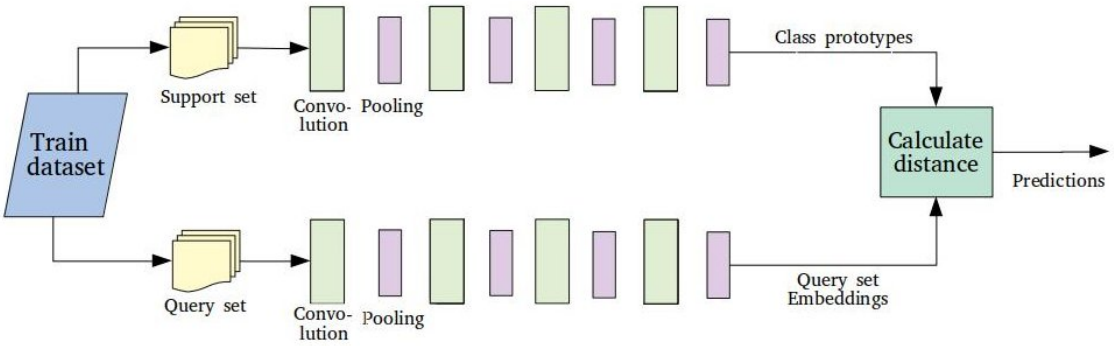

FIGURE 3.8: Methodological Workflow and Network Architecture for Model 4.

In accordance with the previous models, the query and support sets consist of 128x128x3 images. The network has two identical "legs": one for calculating the support set embeddings, namely the class prototypes, and one for calculating the query set embeddings. Each leg is constructed by eight alternating Convolutional and Pooling layers. The Convolutional blocks are ReLu activated. Finally, the Euclidean distance between the query embedding and every class prototype is calculated, in order to compute the class prediction.

The learning rate was set to 0,001 and the decay was handled by an Adam optimizer with beta parameters set to $\beta_1 = 0.9$ and $\beta_2 = 0.99$.

3.3 Tools

The practical implementation of this study is based on open source software components, because apart from being price-efficient, they enable the collaboration among scientific communities around the world. We briefly present the basic tools we used and the role they served for developing the thesis.

- **QGIS** is a GIS application that supports viewing, editing, and analyzing geospatial data. QGIS 3.16 Hannover was used for quickly evaluating the retrieved data or the produced information and for creating demonstrative maps of our work.
- **Python** is a high-level and general-purpose programming language, particularly popular for developing Machine Learning applications. All models were defined, trained and evaluated using Python scripts. The experiments were held in a Python 3.7.8 environment.
- For creating the aforementioned scripts, we exploited the possibilities of several Python libraries, but a special reference to **Keras** and **Tensorflow** needs to be made, as they provided the main API for the elaboration of the analysis. Keras is a model-level library that provides high-level building blocks for developing deep-learning models. Tensorflow is a symbolic tensor-manipulation framework that supports differentiation programming. Keras 2.4.3 and Tensorflow 2.3.0 were installed in our working environment.
- **Miniconda** is a minimalist package management and environment management system. For the creation and maintenance of our Python working environment, Miniconda 4.9.1 was installed.
- **CUDA** is a parallel computing platform and application programming interface model created by Nvidia. Aiming to take advantage of computing power of our system's GPU, CUDA 10.1 was configured accordingly.

All processes were run in a Linux Ubuntu 18.04 machine, equipped with Intel Core i7-7700HQ @ 2.80GHz CPU, Nvidia GeForce GTX 1050 Ti GPU and 16 GB RAM.

4 Results and Discussion

Following the methodological designs described in Chapter 3, this chapter presents the results acquired by the experimental process and carries out a comparative analysis of the models' training duration and their performance on unseen data. The selected as best model is also tested against RGB pan-sharpened satellite images to infer damage assessment maps.

4.1 Learning process

All models were trained until the loss became less than 1% or for a maximum of 60 epochs. The overall accuracy and the loss for the *train* and *validation* datasets were monitored throughout the learning process. The progress of these metrics as a function of the epochs is presented subsequently in the form of diagrams.

Model 1 - Balancing the weights

Figure 4.1 presents the development of the accuracy and loss for the training and validation dataset divisions. The values of the training and validation accuracy seem to range between similar limits. The same behavior is also observed for loss, where the two curves mostly overlap. While accuracy levels fluctuate greatly along the training process, the metric seems to steadily acquire higher values towards the end of the training. In contrast with accuracy, loss drops very soon to low values and ranges around 2 until the end of the training.

The best model for this architecture was obtained after epoch #57, when training and validation accuracies reach their highest values (63%) and losses reach their lowest (1.82 and 1.81 for training and validation respectively).

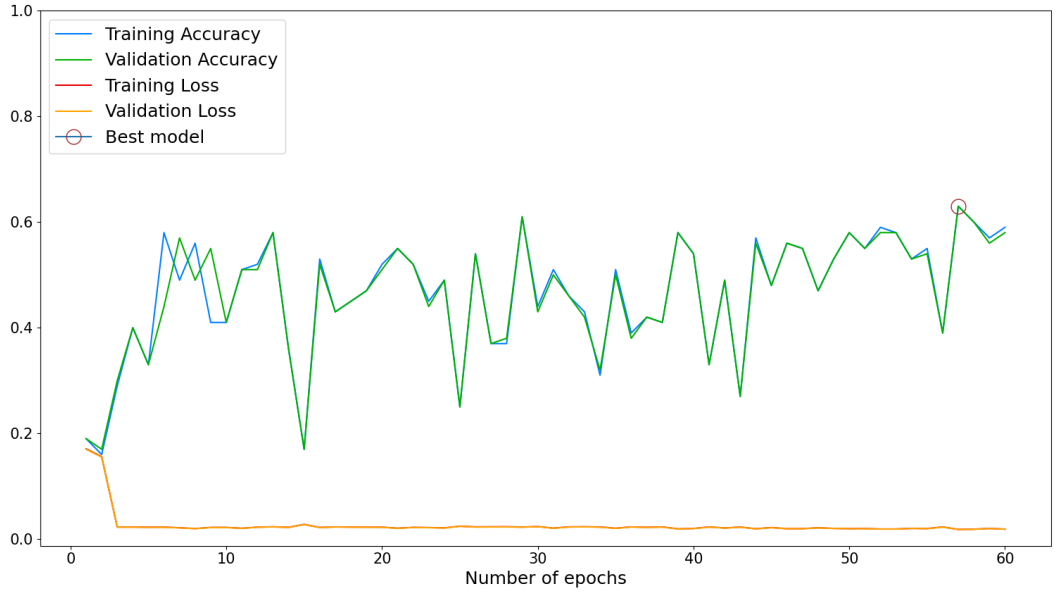

FIGURE 4.1: Accuracy and loss over time for Model 1.

Model 2 - Undersampling

The evolution of the overall accuracy for the *train* and *validation* datasets can be observed in Figure 4.2. The accuracy and the loss difference between the two datasets is small. The accuracy reaches three times a lowest of 0.25, which means that these instances of the model can detect only one single class. The highest recorded accuracy is 0.53 and is reached for both datasets at epoch #55. Training and validation loss start at around 7 and remain at that level until epoch #8, were they both drop below 2 and stay at this level (1.5 - 2) until the end of the training.

Epoch #55 was singled out as the best model of the undersampling strategy, exhibiting training and validation accuracy equal to 53%, training loss equal to 1.62 and validation loss equal to 1.48.

Model 3 - Oversampling

In Figure 4.3 we can inspect how accuracy and loss change over time while training with the oversampled dataset. In contrast with the deep CNN trainings that were analyzed previously, this architecture reaches the highest accuracy and the lowest loss at epoch #40 and then completely collapses. After epoch #40, the overall accuracy for

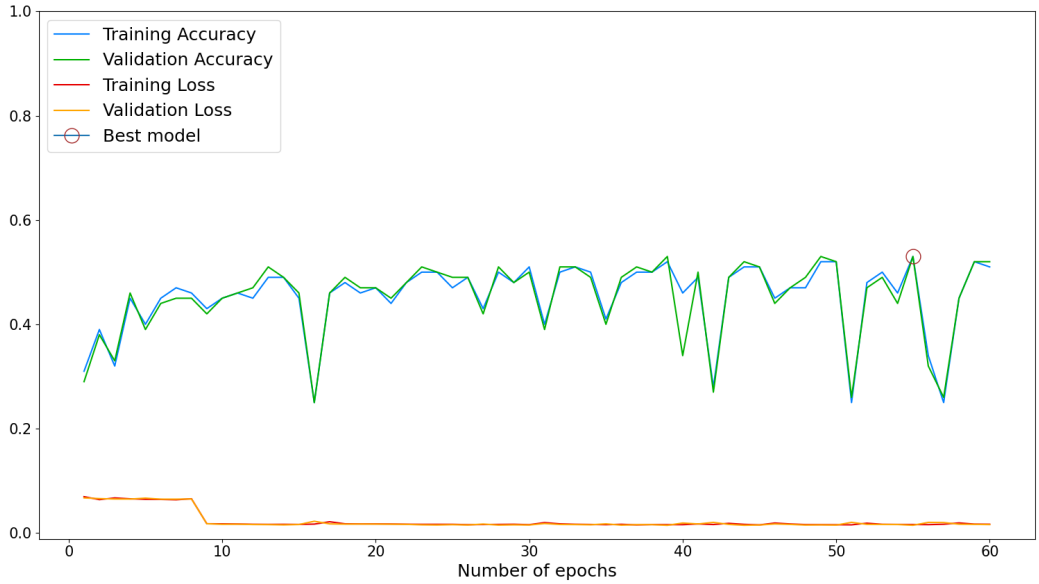

FIGURE 4.2: Accuracy and loss over time for Model 2.

both training and validation is equal to 0.25, indicating that the model attributes all instances to only one class. Moreover, the loss cannot be computed anymore and the function returns NaN.

Consequently, the model of epoch #40 was selected as the best, because it exhibits the highest accuracy (74%) and lowest loss for training (0.95) and validation (0.88) subsets.

Model 4 - ProtoNet

As observed in Figure 4.4, the training accuracy curve has a remarkably different behavior than that of validation. Even though the training accuracy only increases with the number of epochs, the validation accuracy remains practically stable after epoch #20. This architecture reaches higher accuracy levels in the shortest time in comparison with all previous architectures. The loss curves also differ slightly in this case, as training loss, slowly but constantly, decreases, while validation loss stops declining after epoch #20.

The best values for both validation accuracy and loss appear at epoch #21 (62% and 0.88 respectively), so this model is chosen as the best representative for ProtoNet architecture.

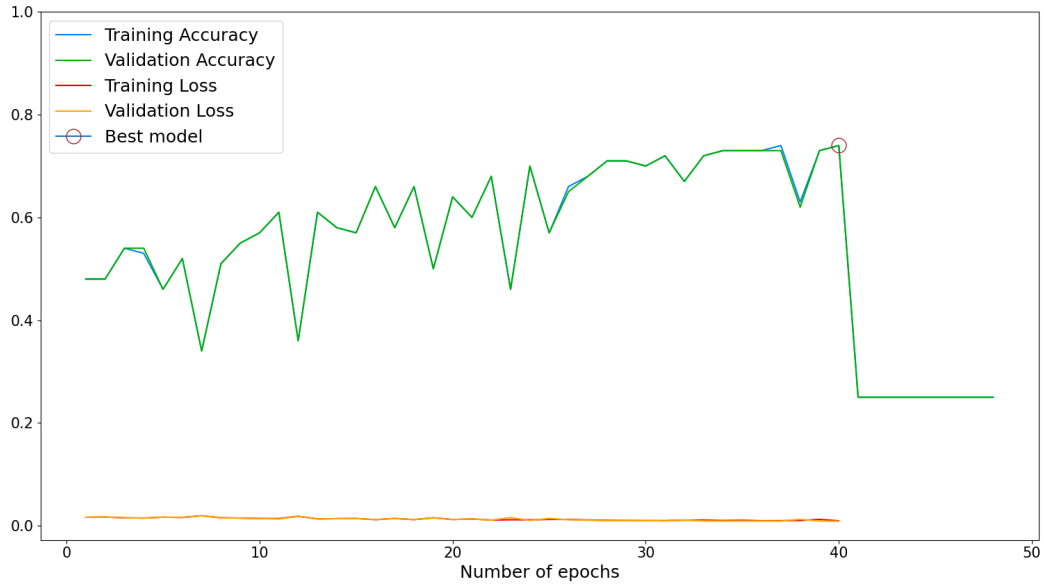

FIGURE 4.3: Accuracy and loss over time for Model 3.

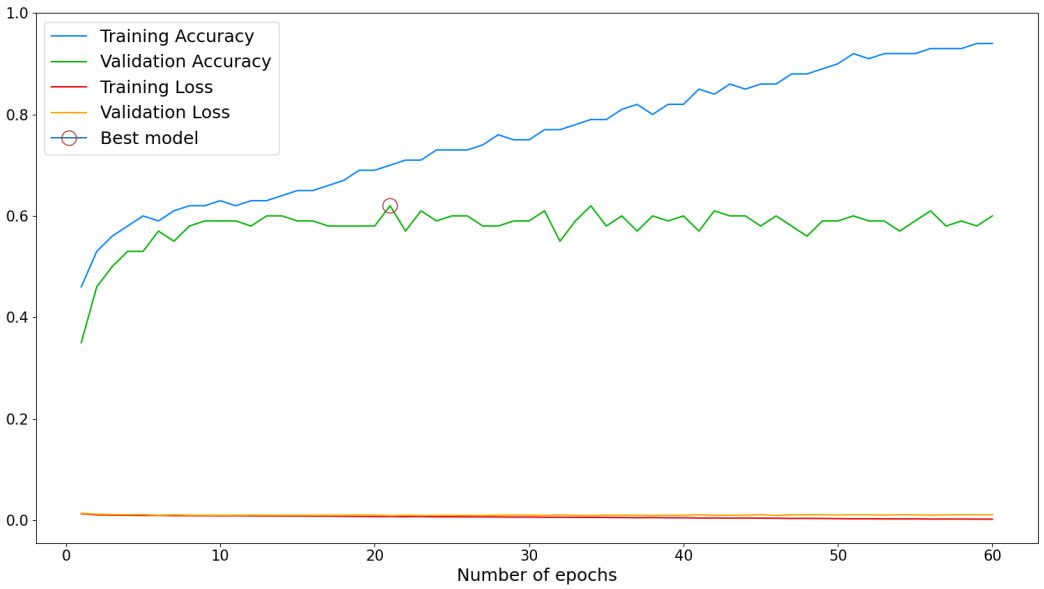

FIGURE 4.4: Accuracy and loss over time for Model 4.

4.1.1 Training time

The choice for the best representatives was based on the overall validation accuracy of every model. A summary of the training time that was required to achieve the best model of each architecture is provided in Table 4.1.

Model Architecture	Epochs	Training time/epoch(mean)	Total training time
Balance weights	57	47 minutes	44 hours, 40 minutes
Undersampling	55	6,5 minutes	6 hours
Oversampling	40	2 hours, 13 minutes	88 hours, 40 minutes
ProtoNets	21	13,5 minutes	4 hours, 45 minutes

TABLE 4.1: Training time variables for the best produced model of each architecture.

Models 1, 2 and 3 use the same baseline, so we can observe how the size of the train dataset affects the time that is required to create one epoch and, eventually, to conclude the learning process. This is expected, as one epoch has to iterate through all data, and thus, more data will demand more time for the same network. In fact, the two variables appear to be almost linearly related (see Figure 4.5). On the other hand, our Prototypical Networks implementation depends a completely different architecture, that apparently needs more time to train an equivalent amount of data.

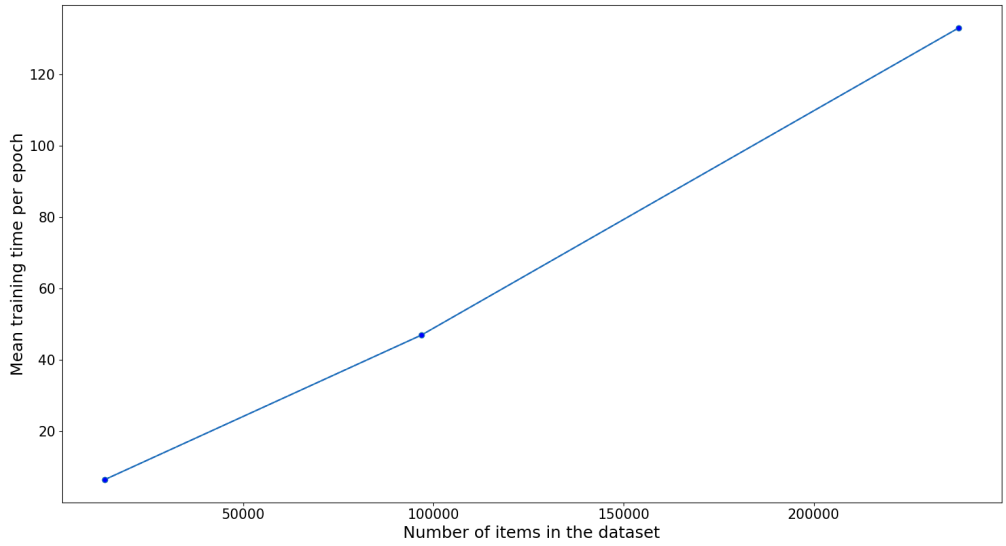

FIGURE 4.5: Mean training time per epoch as the size of the dataset varies.

4.2 Performance Metrics

Based on the observations made on the learning procedure, four models were singled out as exemplars of their corresponding architectures. This section analyzes the details of every model's performance and compares them, so that the most advantageous FSL strategy for the stated problem is selected.

Model 1 - Balancing the weights

The confusion matrix for this approach is provided in Table 4.2. The model scored a value of 0.34 for test accuracy, which is strikingly lower than the training and validation accuracy. This is due to *train* and *validation* subsets being extremely imbalanced, while *test* subset has been designed to have an equal number of examples for every class. As such, a model that favors the majority class but ignores the minority class can still achieve high accuracy numbers, if not tested against a balanced dataset. As seen in the confusion matrix, the minority class is not detected at all. Additionally, class *Minor damage* is the most favored one, attracting examples from all other classes. *No damage* and *Major damage* get less than 50% of their instances correctly classified.

		Predicted			
		No damage	**Minor damage**	**Major damage**	**Destroyed**
Actual	**No damage**	**297**	639	64	0
	Minor damage	253	**695**	52	0
	Major damage	230	393	**377**	0
	Destroyed	119	604	277	**0**

TABLE 4.2: Confusion matrix for Balance weights strategy.

Table 4.3 is derived from Table 4.2. All metrics for *Destroyed* are equal to 0, as no examples of this class are detected. The model demonstrates a high recall value for *Minor damage*, meaning that among the total instances of the class in the dataset, 69% were correctly classified. *Major damage* precision is 0.49, indicating that almost half of the predicted instances are correct predictions. All the other metrics achieved by the model seem very low.

Model 2 - Undersampling

The test accuracy of Model 2 is 0.53, which the same as its training and validation accuracy. The confusion matrix of this approach can be observed in Table 4.4 and the

	Precision	Recall	F-score
No damage	0.33	0.30	0.31
Minor damage	0.30	0.69	0.42
Major damage	0.49	0.38	0.43
Destroyed	0	0	0

TABLE 4.3: Performance metrics for Balance weights strategy.

calculated metrics in Table 4.5. The effect of balancing the dataset that was used for training is immediately apparent, since all classses are detected by this model. *Minor damage* is again the favored class, "stealing" instances especially from its neighboring classes (*No damage* and *Major damage*).

Consequently, the precision for *Minor damage* is the lowest, since the total predictions are many more than the relevant predictions. For the other three classes, precision scores over 50%. On the other hand, recall achieves great values for *Minor damage* and *Destroyed*, resulting in respectively high f-score values. Overall, Model 2 performs much better than Model 1, but stills shows poor prediction capabilities for certain classes.

		Predicted			
		No damage	**Minor damage**	**Major damage**	**Destroyed**
Actual	**No damage**	**228**	652	36	84
	Minor damage	108	**793**	30	69
	Major damage	62	427	**228**	283
	Destroyed	29	128	41	**802**

TABLE 4.4: Confusion matrix for Undersampling strategy.

	Precision	Recall	F-score
No damage	0.53	0.23	0.32
Minor damage	0.40	0.79	0.53
Major damage	0.68	0.23	0.34
Destroyed	0.65	0.80	0.72

TABLE 4.5: Performance metrics for Undersampling strategy.

Model 3 - Oversampling

In Tables 4.6 and 4.7 are listed the performance results of the CNN model that was trained with the oversampled dataset. The test accuracy for Model 3 is 0.63, which,

while much lower than the training and validation accuracies, is the highest test accuracy observed so far. From the confusion matrix (Table 4.6), we can quickly assume that this data-balancing strategy is better than the previous ones, because it allows to detect more than half of the items that belong to the classes *Minor damage*, *Major damage* and *Destroyed*. *No damage*, which is initially the majority class, is discredited, something that is confirmed by its recall in Table 4.7. *Minor damage* mostly attracts the misclassified *No damage* examples, something that lowers its precision, too. Nevertheless, all classes achieve their best performance so far.

		Predicted			
		No damage	**Minor damage**	**Major damage**	**Destroyed**
Actual	**No damage**	**456**	412	33	99
	Minor damage	196	**595**	114	95
	Major damage	83	173	**630**	114
	Destroyed	30	45	142	**783**

TABLE 4.6: Confusion matrix for Oversampling strategy.

	Precision	**Recall**	**F-score**
No damage	0.60	0.46	0.52
Minor damage	0.49	0.59	0.53
Major damage	0.69	0.63	0.66
Destroyed	0.72	0.78	0.75

TABLE 4.7: Performance metrics for Model 3 (Oversampling strategy).

Model 4 - ProtoNet

The selected ProtoNet model scores 0.64 for test accuracy, which is even higher than its validation accuracy (0.62). Table 4.8 represents the confusion matrix of Model 4. As is evident, the model succeeds in identifying more than half of all classes' representatives. Examining more closely the confusion matrix, we can see that the vast majority of the mismatches were assigned to the adjacent classes of the target class. For example, 200 *No damage* polygons were classified as *Minor damage*, 89 as *Major damage* and only 17 as *Destroyed*. Accordingly, *Minor damage* are much more likely to be classified as *No damage* or *Major damage*, rather than *Destroyed*. As this pattern is noticed for all classes, we can conclude that the probability of misclassification is inversely proportional to the distance from the target class prototype.

		Predicted			
		No damage	**Minor damage**	**Major damage**	**Destroyed**
Actual	**No damage**	**694**	200	89	17
	Minor damage	347	**523**	106	24
	Major damage	91	219	**574**	116
	Destroyed	30	11	206	**753**

TABLE 4.8: Confusion matrix for ProtoNet model.

Table 4.9 is calculated using Table 4.8 and conveys an equally competent impression. Precision, recall and f-score exhibit values higher than 50% for all classes, indicating a robust model. Precision is higher than recall for all classes, except for *No damage*. This can be interpreted as "stricter" classification criteria for *Minor damage*, *Major damage* and *Destroyed*, since an instance is classified as such only when there are clear indications features that it indeed belongs to this class.

	Precision	**Recall**	**F-score**
No damage	0.60	0.69	0.64
Minor damage	0.55	0.52	0.54
Major damage	0.59	0.57	0.58
Destroyed	0.83	0.75	0.79

TABLE 4.9: Performance metrics for ProtoNet model.

Comparison

To determine which of the above is the best performing model, a comparative bar chart was plotted. In Figure 4.6, we can examine the precision, recall and f-score of every model for each class separately, as well as the model's average. **Balancing weights** approach is immediately excluded, as it is unable to detect one class and has an overall performance that is the poorest of all candidates. **Undersampling** approach seems relatively efficient in detecting *Destroyed* buildings, but the very low precision of *No damage* and*Minor damage* and the low recall of *No damage* and *Major damage*, make a questionable candidate for this type of problem. **Oversampling** and **ProtoNet** show a comparably adequate overall performance. However, ProtoNet is much more competent in detecting *No damage* buildings and is much faster to train. So, based on the evaluation of the computed performance metrics on the testing dataset, **ProtoNet** is more possible to fulfill the requirements of a structural damage classification problem.

FIGURE 4.6: A comparative view of the performance metrics.

4.3 Inference

To enrich our results with a practical context, three test cases were selected to visualize the model inference in terms of mapping the assessed polygons. For each of these examples, the predictions were collated with the polygon labels and the difference of these two images was calculated. Since the output classes represent a damage gradient, we defined as *misclassification difference from label* the interval between the prediction and the true class of every polygon, to quickly assess how far is the prediction from the reality. For example, a *No damage* building that was falsely identified as *Minor damage*, *Major damage* or *Destroyed* will have a misclassification difference of 1, 2 or 3 respectively. Misclassification difference equal to zero means that the prediction was correct.

Dense urban area, hardly affected

The first test case that was put into action is taken from Mexico 2017 earthquake. It represents a dense urban area that was merely affected by the incident, so almost all polygons are expected to belong to class *No damage* (see Figure 4.7.b). As illustrated in Figure 4.7.a, the model detects almost equally examples from all four classes.

Comparing Figures 4.7.a and 4.7.b, and consulting also 4.7.c, we can see a tendency for missclassification, especially in the top left corner of the region (North-Northwest), where the majority of the instances has been identified as *Major damage* or *Destroyed*. In the rest of the image, most of the buildings are classified as *No damage* or *Minor damage*, with sparse presence of *Major damage* buildings and even fewer *Destroyed*. Likewise, the misclassification differences are higher towards the top and left of the image. Overall, the general impression for the severity of damage for the whole area can be misleading.

Urban area, severely affected

The second test case is a snapshot of tornado Joplin's aftermath, in Joplin, Missouri, USA. In contrast with the first test case, this area is highly affected. According to the ground truth (see Figure 4.8.b), all building labels belong to *Destroyed* or *Major damage* categories, except for four buildings in the lower right corner. Nevertheless, the model classifies instances of all classes across the entire region (see Figure 4.8.a). Moving to the bottom of the image, the structures tend to be classified more as *Major damage*

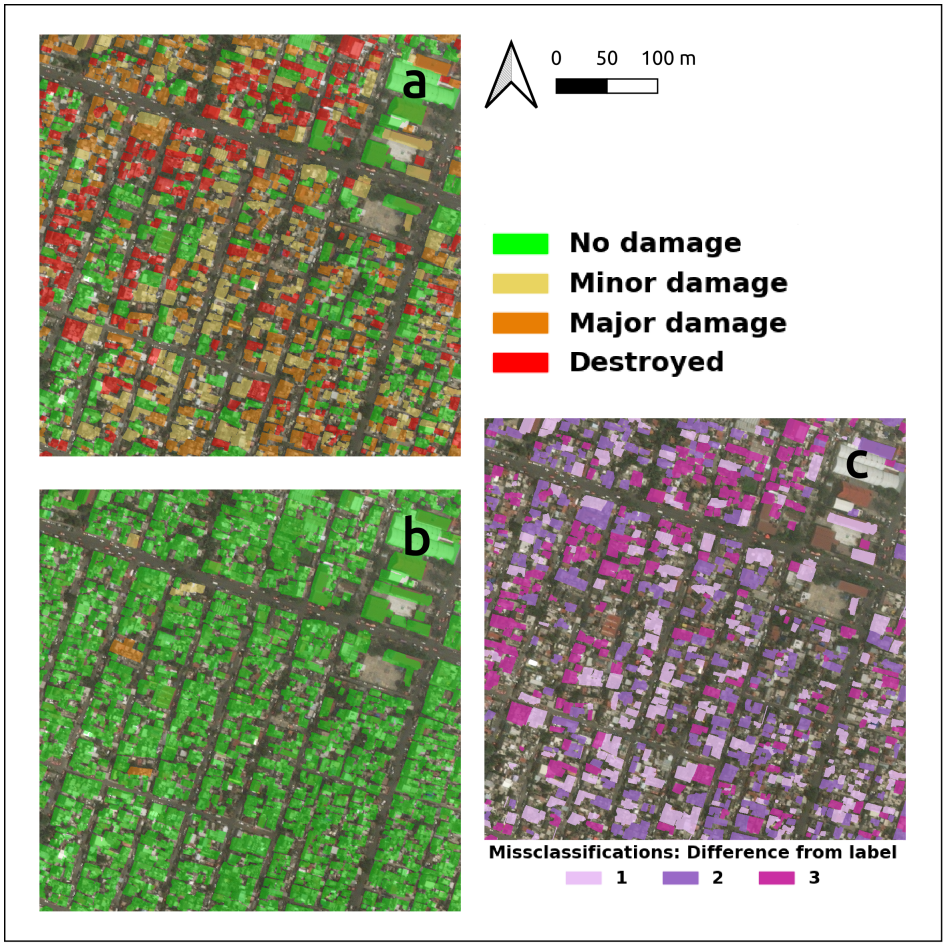

FIGURE 4.7: Application of ProtoNet model on test case from Mexico 2017. Satellite image overlaid with: a. the predictions, b. the labels, c. the difference between the two.

and *Destroyed*, eventually giving the broader look of a region severely attacked by the natural disaster. In a similar fashion, the missclassification differences appeas to be higher in the top half of the image (see Figure 4.8.c).

Coastal area, moderately affected

The last case considers a coastal line from Panama City, Florida, USA after hurricane Michael stroke the area, and contains examples of all four classes (see Figure 4.9.b). Again, we can observe a relatively high rate of missclassifications, as the model misses half of the classified items. The two *Destroyed* buildings in the top section of the image

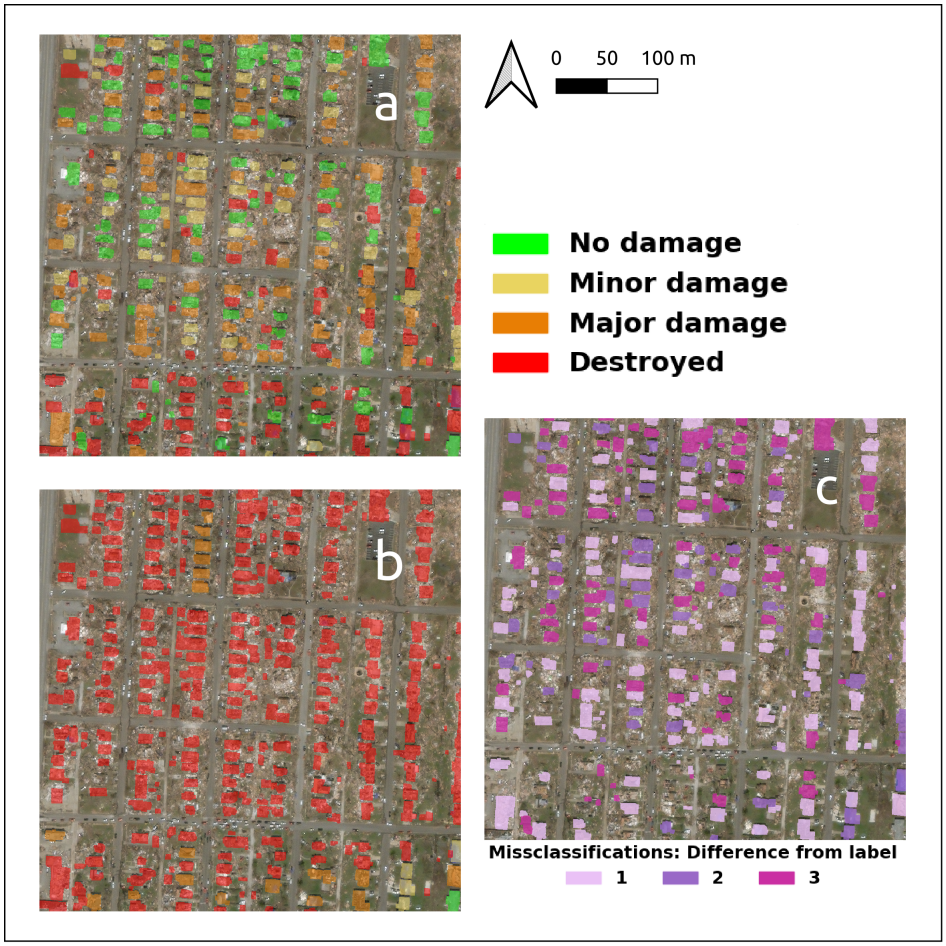

FIGURE 4.8: Application of ProtoNet model on test case from tornado Joplin. Satellite image overlaid with: a. the predictions, b. the labels, c. the difference between the two.

were categorized as *Major damage* and *Minor damage*. The main dissonance between the ground truth in Figure 4.9.b and the predictions in Figure 4.9.a are the three *No damage* structures that were classified as *Destroyed*. Aside from these, the missclassified instances have a distance of 1 or 2 from the correct class, meaning that they were assigned to a neighboring class of the correct one.

As a closing remark for the intuitive interpretation gained by the visual mapping of the predictions compared to ground truth, we can deduce that the tested model has plenty of room for improvement, but is in the right track for tackling the problem of post-earthquake urban damage assessment. The most confounding aspect of the results is when *No damage* buildings are misinterpreted as *Destroyed* and vice versa,

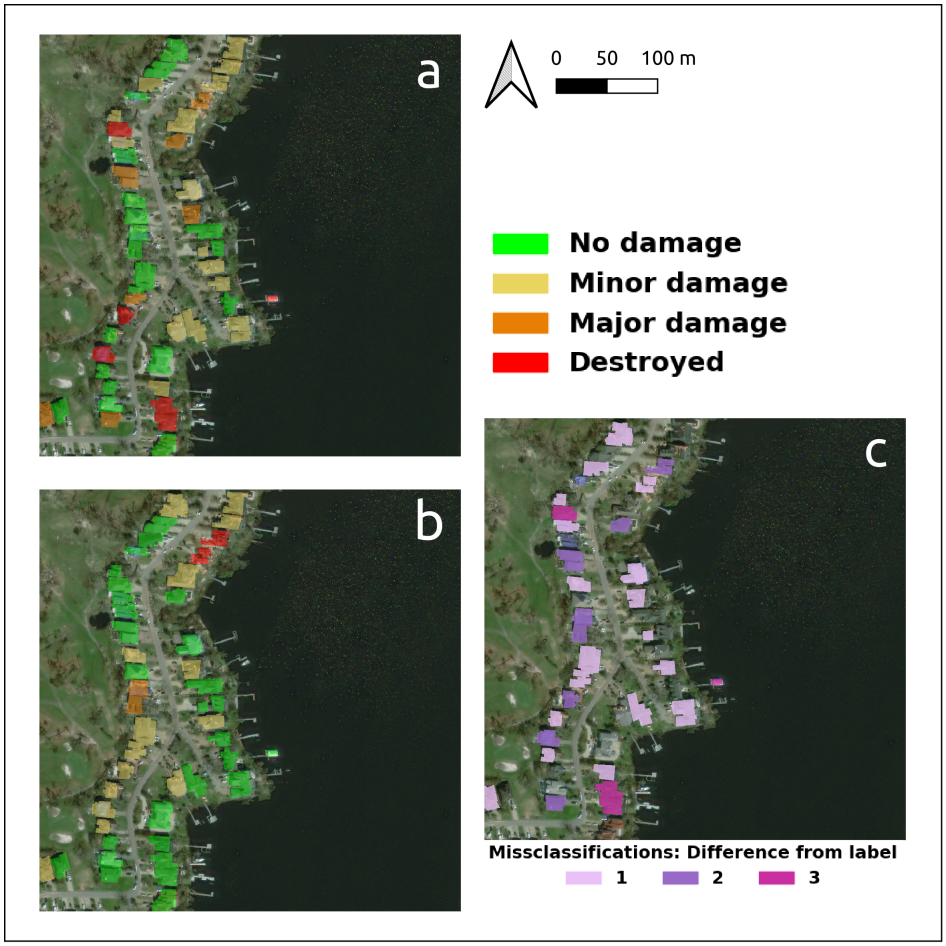

FIGURE 4.9: Application of ProtoNet model on test case from hurricane Michael. Satellite image overlaid with: a. the predictions, b. the labels, c. the difference between the two.

because in a real case scenario it could lead to consuming critical resources and time for assisting the wrong locations. However, such missclassifications seem to be more rare as the area of the polygon increases.

4.3.1 Limitations

While the obtained results from Model 4 seem encouraging, there are certain limitations to the extent a 50-shot metric-learning approach on satellite imagery of this resolution can reach. Randomly picking 50 representatives of each class to train a ProtoNet model may have led to excluding important information carried by the data that were

left out. Additionally, satellite imagery of higher resolution is usually private, and thus, very difficult to acquire.

5 Conclusion

This thesis employed VHR pan-sharpened satellite imagery and Machine Learning in order to identify the level of structural destruction induced by a catastrophic earthquake incident. Aiming to approximate a real case scenario, where the available labeled post-event data are limited and the pre-event data possibly nonexistent, the different explored possibilities tackle data insufficiency and imbalance by implementing FSL strategies. The development of this work, which was defined by the objectives set in Chapter 1, is summarized as following:

- *Review the existing literature and gather applied practices in similar applications.*

  An in depth bibliographic research of the existing literature and state-of-the-art was elaborated in order to gain understanding of how related problems are approached and to familiarize with the necessary theoretical background (Chapter 2). The research showed that the apparent trend for urban damage assessment with Remote Sensing is to use Deep Learning. The data shortage is mostly addressed with data augmentation, but pre-trained models and un-supervised learning have also been put to test. All these approaches are considered FSL, proving the applicability of this Machine Learning family of strategies in relevant problems.

- *Assess the relevant methods and implement the architectures that suit the needs of the thesis.*

  To select the appropriate methodological components, we took into consideration the type of the input imagery (labeled satellite imagery of mean GSD 2.56 meters) and the intended number of output classes (4-level damage scale). Based on the conclusions drawn by the literature review, we confronted the data shortage in two distinct ways: 1) balancing and augmenting the dataset to make it suitable for training a Deep Architecture and 2) metric few-shot learning with Prototypical Networks.

More data from hurricane incidents were incorporated in the analysis as a first step of expanding the dataset with data from related problems. For a balanced Deep CNN training, three different models were created: Model 1 with cost-sensitive learning, Model 2 with undersampling and Model 3 with oversampling. For Model 4, a 50-shot ProtoNet was trained. This process resulted in 4 models, each of them having been trained with a different dataset, in terms of the total number of examples and the class proportions. Nevertheless, all models were evaluated based on the same balanced set of completely unseen data.

- *Measure the performance and the training time of the implemented approaches and provide quantitative and qualitative evaluations.*

  Throughout the learning process, the training and validation loss were monitored to make sure that the model achieves the best possible performance without overfitting. The choice for the best representative of every model relied on its validation accuracy. The four model versions that were singled out, were compared using precision, recall and f-score. A thorough comparison was elaborated, that included the metrics for each class separately as well as the metric average, so that the best approach for solving our research problem could be decided. Model 4, built upon Prototypical Networks, showed the best performance, although Model 3 (data oversampling in the pre-processing stage) exhibited almost equally good results. Consequently, Model 4 was used for creating damage assessment maps, to provide an idea of the model's practical applicability. Model 4 was also proved the fastest to train, since by definition uses the smallest training set, while Model 3 was proved the most time consuming (see Table 4.1).

By fulfilling the above objectives, we reached the answers to our research questions, also stated in Chapter 1:

- *How can the supervised classification of a highly imbalanced dataset be elaborated, where the instances of one class are multiple times higher than the other classes'?*

  The classification of an imbalanced dataset can be solved either by adjusting the weight matrix, in a way trying to normalize the number of examples for every class (cost-sensitive learning), or by selecting the same number of representatives for each class to build the training set. The first method proved to be inefficient in our case, while using a balanced dataset when training image classification models immediately added up to the overall performance. Undersampling may cause loss of decisive information for the classification process, and

thus it is not as considerable as oversampling for training a Deep CNN model. However, for Prototypical Networks, randomly picking a few samples per class proved to be enough for creating the most efficient model.

- *In cases of imbalanced datasets, to what extent are the representatives of the majority and minority classes successfully detected?*

  Taking a closer look, all four approaches have a different impact on each class. It can be argued that such an eminently imbalanced dataset cannot fully support the training of a multi-class predictive model with cost-sensitive learning, since Model 1 is entirely unable to detect the minority class. Surprisingly, the majority class is also overlooked by Model 1. As stated before, undersampling also leads to dismissing valuable information and hence, the resulting Model 2 has a limited predictive ability over certain classes, but performs very well for the minority class. The majority class is again discredited. Oversampling seems to be more competent for creating a Deep CNN-based predictive model, but has a borderline performance for the majority class. Model 4 seems to achieve a robust performance for all classes, being able to predict correctly more than half of their instances.

- *If overlaid with a map, is the visual interpretation of the predictions indicative of the severity of damage suffered by the region?*

  The predictions of Model 4 were overlaid with the satellite imagery and compared with the true polygon labels, to obtain a qualitative impression of how close the output of the model is to the reality. Even though the model is in the right direction for damage assessment, it would not be advisable to base on it an estimation about how gravely was an area affected (Figures 4.7, 4.8, 4.9). Improvements must still be made, so that the possible misinterpretations between *No damage* and *Destroyed* buildings are eliminated. This phenomenon seems less frequent for polygons of larger area.

By accomplishing the objectives and research questions set at the beginning of the study, we also managed to attain the main purpose of this study, which was to implement and evaluate the effect of different FSL methods on imbalanced post-earthquake data. We proved that oversampling is a more efficient balancing method for training Deep Convolutional Neural Networks (CNN) than cost-sensitive learning and undersampling, and we demonstrated the practical applicability of Prototypical Networks in a damage classification problem.

5.1 Future works

Since the obtained results seem encouraging, further research on this subject is recommended. In Section 4.3.1, we saw that the main limitations are introduced by the resolution of the satellite imagery and by randomly picking the samples to represent each class. Although remotely sensed imagery of higher resolution is difficult to obtain, we anticipate that a similar study with satellite or UAV imagery of higher resolution should be pursued. Furthermore, instead of naively picking a few examples from the available data, certain data sampling techniques, such as Near-Miss and Condensed Nearest Neighbor Rule, can be employed to determine the most useful samples to train a model. Furthermore, we strongly believe that Prototypical Networks in the context of urban damage assessment deserve more exploration. The number of shots, the prototyping function and the distance function are parameters to experiment with and that could improve the existing results.

A Source Code and Data

All scripts that were developed for data pre- and post-processing, as well as model training and evaluation, can be found in the following repository:
https://github.com/EftyK/FSL_for_urban_damage.

The CNN baseline architecture for Models 1,2 and 3 can be found at:
https://github.com/DIUx-xView/xView2_baseline

The original xBD dataset can be found at:
https://xview2.org

Bibliography

Albawi, S., T. A. Mohammed, and S. Al-Zawi (2018). "Understanding of a convolutional neural network". In: *Proceedings of 2017 International Conference on Engineering and Technology, ICET 2017*. Vol. 2018-Janua. Institute of Electrical and Electronics Engineers Inc., pp. 1–6. ISBN: 9781538619490. DOI: 10.1109/ICEngTechnol.2017.8308186.

Balamurugan, A. and A. Zakhor (2019). "Online Learning for Indoor Asset Detection". In: *IEEE International Workshop on Machine Learning for Signal Processing, MLSP*. Vol. -2019-Octo. IEEE Computer Society. ISBN: 9781728108247. DOI: 10.1109/MLSP.2019.8918849.

Boccardo, P. and F. G. Tonolo (2015). "Remote sensing role in emergency mapping for disaster response". In: *Engineering Geology for Society and Territory - Volume 5: Urban Geology, Sustainable Planning and Landscape Exploitation*. Springer International Publishing, pp. 17–24. ISBN: 9783319090481. DOI: 10.1007/978-3-319-09048-1_3.

Branco, P., L. Torgo, and R. Ribeiro (2015). "A Survey of Predictive Modelling under Imbalanced Distributions". In: arXiv: 1505.01658. URL: http://arxiv.org/abs/1505.01658.

Choi, W. G. and K. S. Lee (2019). "Conceptual representation for crisis-related tweet classification". In: *Computacion y Sistemas* 23.4, pp. 1523–1531. ISSN: 20079737. DOI: 10.13053/CyS-23-4-3304.

Convolutional Neural Network Tutorial. URL: https://missinglink.ai/guides/convolutional-neural-networks/convolutional-neural-network-tutorial-basic-advanced/ (visited on 01/08/2021).

Cooner, A. J., Y. Shao, and J. B. Campbell (2016). "Detection of urban damage using remote sensing and machine learning algorithms: Revisiting the 2010 Haiti earthquake". In: *Remote Sensing* 8.10. ISSN: 20724292. DOI: 10.3390/rs8100868.

Fannela, D. A. and J. A. Munshi (1998). *Design of Concrete Buildings for Earthquake & Wind Forces According to the 1997 Uniform Building Code*. 1st. Portland Cement Assn. ISBN: 0893121959,9780893121952.

Fernández, A. et al. (2018). *Learning from Imbalanced Data Sets*. Cham: Springer International Publishing. DOI: 10.1007/978-3-319-98074-4. URL: http://link.springer.com/10.1007/978-3-319-98074-4.

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. MIT Press. ISBN: 9780262035613. URL: https://www.deeplearningbook.org/.

Gupta, R. et al. (2019). "xBD: A Dataset for Assessing Building Damage from Satellite Imagery". In: pp. 10–17. arXiv: 1911.09296. URL: http://arxiv.org/abs/1911.09296.

He, K. et al. (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 2016-Decem. IEEE Computer Society, pp. 770–778. ISBN: 9781467388504. DOI: 10.1109/CVPR.2016.90. arXiv: 1512.03385. URL: http://image-net.org/challenges/LSVRC/2015/.

Ji, M., L. Liu, and M. Buchroithner (2018). "Identifying collapsed buildings using post-earthquake satellite imagery and convolutional neural networks: A case study of the 2010 Haiti Earthquake". In: *Remote Sensing* 10.11. ISSN: 20724292. DOI: 10.3390/rs10111689.

Ji, M. et al. (2020). "Discrimination of earthquake-induced building destruction from space using a pretrained CNN model". In: *Applied Sciences (Switzerland)* 10.2. ISSN: 20763417. DOI: 10.3390/app10020602.

Kadam, S. and V. Vaidya (2020). "Review and analysis of zero, one and few shot learning approaches". In: *Advances in Intelligent Systems and Computing*. Vol. 940. Springer Verlag, pp. 100–112. ISBN: 9783030166564. DOI: 10.1007/978-3-030-16657-1_10. URL: https://link.springer.com/chapter/10.1007/978-3-030-16657-1{_}10.

Kakooei, M. and Y. Baleghi (2017). "Fusion of satellite, aircraft, and UAV data for automatic disaster damage assessment". In: *International Journal of Remote Sensing* 38.8-10, pp. 2511–2534. ISSN: 13665901. DOI: 10.1080/01431161.2017.1294780. URL: http://dx.doi.org/10.1080/01431161.2017.1294780.

Kim, P. (2017). "Convolutional Neural Network". In: *MATLAB Deep Learning*. Berkeley, CA: Apress, pp. 121–147. DOI: 10.1007/978-1-4842-2845-6_6. URL: http://link.springer.com/10.1007/978-1-4842-2845-6{_}6.

Li, K. et al. (2020a). "Object detection in optical remote sensing images: A survey and a new benchmark". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 159, pp. 296–307. ISSN: 09242716. DOI: 10.1016/j.isprsjprs.2019.11.023.

Li, Y. et al. (2019). "Building damage detection from post-event aerial imagery using single shot multibox detector". In: *Applied Sciences (Switzerland)* 9.6. ISSN: 20763417. DOI: 10.3390/app9061128.

Li, Y. et al. (2020b). "Unsupervised domain adaptation with self-attention for post-disaster building damage detection". In: *Neurocomputing* 415, pp. 27–39. ISSN: 18728-286. DOI: 10.1016/j.neucom.2020.07.005.

Liu, K. et al. (2019). "Generalized zero-shot learning for action recognition with web-scale video data". In: *World Wide Web* 22.2, pp. 807–824. ISSN: 15731413. DOI: 10.1007/s11280-018-0642-6. arXiv: 1710.07455.

Lollino, G. et al. (2015). "Engineering geology for society and territory – volume 5: Urban geology, sustainable planning and landscape exploitation". In: *Engineering Geology for Society and Territory - Volume 5: Urban Geology, Sustainable Planning and Landscape Exploitation* 5, pp. 1–1400. DOI: 10.1007/978-3-319-09048-1.

Mikołajczyk, A. and M. Grochowski (2018). "Data augmentation for improving deep learning in image classification problem". In: *2018 International Interdisciplinary PhD Workshop, IIPhDW 2018*. Institute of Electrical and Electronics Engineers Inc., pp. 117–122. ISBN: 9781538661437. DOI: 10.1109/IIPHDW.2018.8388338.

O'Shea, K. and R. Nash (2015). "An Introduction to Convolutional Neural Networks". In: arXiv: 1511.08458. URL: http://arxiv.org/abs/1511.08458.

Schmidhuber, J. (2014). "Deep Learning in Neural Networks: An Overview". In: *Neural Networks* 61, pp. 85–117. DOI: 10.1016/j.neunet.2014.09.003. arXiv: 1404.7828. URL: http://arxiv.org/abs/1404.7828http://dx.doi.org/10.1016/j.neunet.2014.09.003.

Sklearn API documentation. URL: https://scikit-learn.org/stable/modules/generated/sklearn.utils.class{_}weight.compute{_}class{_}weight.html (visited on 01/08/2021).

Snell, J., K. S. Twitter, and R. S. Zemel (2017). *Prototypical Networks for Few-shot Learning*. Tech. rep. arXiv: 1703.05175v2.

Taher, R. (2010). *General Recommendations for Improved Building Practices in Earthquake and Hurricane Prone Areas*. Tech. rep. Architecture for Humanity. URL: www.architectureforhumanity.org.

UNDRR & CRED (2019). *An overview of the last 20 years the last 20 years*. Tech. rep. Centre for Research on the Epidemiology of Disasters.

Vanschoren, J. (2018). "Meta-Learning: A Survey". In: *arXiv*. arXiv: 1810.03548. URL: http://arxiv.org/abs/1810.03548.

Vetrivel, A. et al. (2018). "Disaster damage detection through synergistic use of deep learning and 3D point cloud features derived from very high resolution oblique aerial images, and multiple-kernel-learning". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 140, pp. 45–59. ISSN: 09242716. DOI: 10.1016/j.isprsjprs.2017.03.001.

Waldner, F. et al. (2019). "Needle in a haystack: Mapping rare and infrequent crops using satellite imagery and data balancing methods". In: *Remote Sensing of Environment* 233. ISSN: 00344257. DOI: 10.1016/j.rse.2019.111375.

Wang, Y. et al. (2019). "Generalizing from a Few Examples: A Survey on Few-Shot Learning". In: 1.1, pp. 1–34. arXiv: 1904.05046. URL: http://arxiv.org/abs/1904.05046.

Weber, E. and H. Kané (2019). "Building Disaster Damage Assessment in Satellite Imagery with Multi-Temporal Fusion". In: *Remote Sensing* 11, 2765.Ic. DOI: 10.3390/rs11232765. arXiv: 2004.05525. URL: http://arxiv.org/abs/2004.05525.

Xu, J. Z. et al. (2019). "Building Damage Detection in Satellite Imagery Using Convolutional Neural Networks". In: NeurIPS. arXiv: 1910.06444. URL: http://arxiv.org/abs/1910.06444.

Masters Program in **Geospatial Technologies**

Supported by:
Education and Culture
ERASMUS MUNDUS